ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

MASTER'S PROJECT

ETHIOPIC ONLINE HANDWRITING RECOGNITION IN ANDROID-BASED SMARTPHONES

By: Fetiya Beshir

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

JUNE 2010

Addis Ababa University
Office of Graduate Programs
Faculty of Informatics
Department of Computer Science

ETHIOPIC ONLINE HANDWRITING RECOGNITION IN ANDROID-BASED SMARTPHONES

By: Fetiya Beshir

Advisor: Solomon Atnafu (PhD)

APPROVED BY

EXAMINING BOARD:

1. Solomon Atnafu (PhD), Advisor _____
2. _____ _____
3. _____ _____

Acknowledgement

Next to the Almighty Allah, I would like to express my deepest gratitude to my advisor, Dr. Solomon Atnafu, for his helpful ideas, guidance and suggestions he gave me while doing this project.

I would also like to thank individuals who have been showing me their willingness when I needed their help. I thank Daniel Kefale for all his support from the very beginning of this project following up my progress. I would also like to extend my sincere thanks to Menasse Zewdu for his technical assistance providing me the appropriate device that makes this project work more real.

My thanks also goes to Tesfaye Guta, who has reviewed my draft document and provided me valuable comments sacrificing his own precious research time. I thank my friend Amsale Zelalem for giving me the courage and sharing those valuable ideas to accomplish my study. Amsi, thanks for everything you did to me since day one of my stay in this program.

Last but not the least, I would like to offer many thanks to my family for their uncountable contribution. My brothers and sisters, you all have part for who I am today. Mom and Dad, I have no word to express my feeling to thank you. Thank you for your understanding, support and unconditional love you have always been giving me. Long life to you both!

Table of Contents

List of Figures

List of Tables

List of Acronyms

ADT :	Android Development Tools
AVD :	Android Virtual Device
API :	Application Programming Interface
ARM :	Advanced RISC (Reduced Instruction Set Computer) Machine
DDMS	Dalvik Debug Monitor Server
EHWR :	Ethiopic Handwriting
EHWRS :	Ethiopic Handwriting Recognition System
EOHWR :	Ethiopic Online Handwriting Recognition
EOHWRS :	Ethiopic Online Handwriting Recognition System
GUI :	Graphical User Interface
HMM :	Hidden Markov Model
HWRS :	Handwriting Recognition System
IDE :	Integrated Development Environment
IME :	Input Method Editor (Input Method)
OHWR :	Online Handwriting Recognition
OHWRS :	Online Handwriting Recognition System
OS :	Operating System
PC :	Personal Computer
PDA :	Personal Digital Assistant
SDK :	Software Development Kit
SQL:	Structured Query Language
UI :	User Interface
UML :	Unified Modeling Language
WI:	Writer Independent
XML :	Extensible Markup Language

Abstract

A rapid growth in advanced mobile technologies has opened various research interests. Handheld devices like Smartphones are getting a wide range of acceptance due to the full-fledged functionalities they provide. They are being designed to have the ability to access information of any type ubiquitously.

Text entry mechanisms are main concerns in handheld devices as they are the core interaction ways of communication. To best use of all features incorporated in these devices, an efficient text entry mechanism is vital. Considering the aim to achieve the goals of being small in size to hold and easy to use, made to look for alternative text entry techniques rather than sticking to the hardware keyboard which resulted in difficulties to attain the goals set.

Using the touch screen capabilities of handhelds, online handwriting recognition and soft keyboard entries are becoming better options for text entry into these devices considering the inconvenience of hardware keyboards to be used.

In this work, an Ethiopic Online Handwriting Recognition (EOHWR) input method is developed for an Android-based Smartphones where users of the script shall benefit from it by using the developed input method in different applications that exist on the device. In its integration, the work considered previous efforts exerted on the EOHWR.

The work followed a constrained, writer independent approach for its recognition. It developed representation symbols for thirty four basic characters, a numeral and punctuation of the Ethiopic script resembling the natural script and stored their references persistently. Users need to learn the symbols prior to usage. It also puts some restrictions on the orders in which strokes that build a character should be inserted. In its recognition, it followed a template matching technique computing the directional distances between segments of a stroke for each character.

The work incorporates the usage of the non basic characters, numerals and punctuations which are found in the Ethiopic script by making them to be candidate soft key options of their corresponding basic symbol. Thus, whenever recognition takes place for a basic symbol, its non basic character family will be available to be selected. Its usability and accuracy is affirmed getting an average accuracy of about 80 percent.

Keywords: Ethiopic Input Method for Android, Constrained Ethiopic Online Handwriting Recognition System, Writer Independent Ethiopic Online Handwriting Recognition System

CHAPTER ONE

INTRODUCTION

1.1. Overview

The personal computer and the Internet have found revolutionary ways to connect people to entertain and let them exchange information. But none of these is able to reach each person anywhere and anytime like the cell phone does [23].

Growth in demand for advanced mobile devices boasting powerful processors, abundant memory, larger screens and open operating systems has outpaced the handheld devices market for several years. Whether you are an organization looking to mobilize your sales professionals or an individual looking to increase personal productivity, leaving office or laptop behind does not have to mean leaving crucial files and work behind [6].

Smartphone is one of the devices that make this technological era be set in advance containing various features that a typical cell phone does not. In addition to a phone and multimedia services, it enables to send and receive e-mails, view and edit documents of different file format, downloading software and accessing the Web like ordinary computer does.

In [7], Smartphone is described as a large-screen, data-centric, handheld device designed to offer complete phone functions whilst simultaneously functioning as a Personal Digital Assistant (PDA).

Smartphones run operating system (OS) that controls all the programs and documents providing an interface and platform for application developers. Hence, they can be categorized based on the operating system they run. The major operating systems known for Smartphones are BlackBerry, Palm, Windows Mobile (Pocket PC), Symbian and Android. Among the specified operating systems, Android is the newest with an emerging acceptance both by users and manufacturers of handheld devices. It is built for Internet centric devices allowing the device to be extended with capabilities of adding new applications easily. Android is a new, open and comprehensive platform developed by Google to power mobile devices and handsets. The platform comprises of four components: an operating system, a middleware, user interface and applications [4].

Android is a software stack for mobile devices which means a reference to a set of system programs or a set of application programs that form a complete system. This software platform provides a foundation for applications just like a real working. It is a manufacturer spanning platform having a capability to run on every mobile phone devices. Its openness also allows developers to write managed code in the Java language controlling the device via Google developed Java libraries. In general, Android is intended to revolutionize the mobile market by bringing the Internet to the cell phone and allowing its use in the same way as on the personal computer [23].

Currently Android is getting a warm acceptance from the three parties: manufacturers, developers and end users. Major mobile vendors are showing their interest in launching their newly designed Android powered Smartphone products benefiting from its spanning characteristics. Developers are involved in developing numerous applications taking the advantage of a real open Application Programming Interface (API) and its localization capability. Android is also getting a significant acceptance from users of different background as a result of the variety of options they get to use localized applications that are developed to work on their own language.

To best use all the features incorporated in mobile devices, an efficient human-machine interaction technique is important. Text entry method is gaining more and more interest by different subjects due to the frequent usage of applications like text messaging.

In the intention of making PDAs and Smartphones fit for everywhere at anytime usage, their size is becoming as small as possible. This forces the devices to contain a tiny hardware keyboard or none at all.

Currently, rather than sticking to a hardware keyboard, most technologies like Smartphones and PDAs are being designed incorporating multiple functionalities like touch screens through which users can provide entries using a stylus or a finger. Soft keyboard and handwriting are possible input methods for touch-sensitive devices. Handwriting text entry using a stylus is a more convenient input method as it resembles the natural handwriting [5].

According to [8], human eye can read what is written or displayed either in natural handwriting or in printed format. The same work in case the machine does is called handwriting recognition.

Handwriting recognition can be performed in one of the two states: online or offline. In online recognition, the current information is given to the system and recognition is carried out at the same time as the writing but in offline recognition, the recognition is carried out in already written presented data input in the form of bitmap.

As handwriting style varies from user to user, online character recognition systems are characterized based on the number of users they target, the data set size they handle and their ability to recognize users' entries efficiently. The two broad categories of online handwriting recognition systems (OHWRS) are writer dependent and writer independent (WI).

In a writer dependent recognition, the system is trained and optimized to recognize handwriting style of only one individual where as in a writer independent recognition, the system can handle the variations in multiple users' writing styles applying adaptive methods with the ability to learn new writing styles.

Several works on online handwriting recognition for Ethiopic characters have been made based on writer dependent and writer independent states of recognition [1, 2, 3, 10, 11, 12 and 13] some of them focusing for the PDA environment.

To the best of our knowledge, none of the Ethiopic online handwriting recognition systems (EOHWRS) are adopted or developed to work on Smartphones functioning as a text entry technique to each and every applications embedded on the devices. This shows the need for developing an application that can be used as an input method integrating the Ethiopic script handwriting recognition for Android based systems.

1.2. Statement of the Problem

In Ethiopia, the usage of handheld devices has increased significantly in the last couple of years. Most of these devices do not support Ethiopic scripts by all means as they are usually built to support English and some other international languages. None of the mobile phones support Ethiopic online handwriting recognition system as an input method. Integration of OHWR for Ethiopic script is essential for native users to make use of the devices proficiently. The development of such an application can serve the support of different features like editing and composing in e-mail and Messaging services and in saving contacts. Thus, the aim of this project is to facilitate support of these tasks by allowing users to enter, edit, store and send content written in their own local language in the natural way.

1.3. Motivation

Online handwriting recognition has been gaining more interest due to the increasing popularity of handheld computers, digital notebooks and advanced cellular phones. Android based phones are one of the major technological advancements that are expected to gain an enormous acceptance due to several characteristics they have. Their being released as an open source for developers to integrate localized applications of different languages in to them, is an encouraging characteristics. Hence, making use of this benefit to integrate the Ethiopic handwriting recognition system to enter and edit text is the main motive behind this project work.

1.4. Objectives

General Objective

The general objective of this work is to develop and integrate a constrained, writer independent Ethiopic online handwriting recognition input method system to support text entry on popular applications that run in Android based Smartphones.

Specific objective

The specific objectives of this project include

- Reviewing related literatures and exploring thoroughly the models and algorithms developed for writer dependent and writer independent online EHWRS and identify the requirements for the design of WI-EOHWRS for Android platform
- Design a constrained, writer independent EHWRS for Android platform to support Ethiopic handwriting entry.
- Implement the design for writer independent EHWRS.
- Test and evaluate the system integration on applications like messaging and e-mail.

1.5. Scope

Even though there are other text entry techniques for Smartphones, this work is limited to constrained, writer independent text editing based on online handwriting character recognition. It only considers the basic Ethiopic script characters and symbols.

1.6. Methodology

In developing the application for this project, various techniques are used following step by step procedures in order to accomplish the objectives set.

- Reviewing of relevant literatures and related works which are conducted for Amharic and other languages to have a thorough understanding of the subject matter.
- Identifying appropriate design considerations to integrate the EOHWRS.
- Java programming language along with Android platform consisting an emulator and Eclipse with an ADT plugin are the main tools used to build and test the system on the target environment.
- For persistence storage of template features, an SQLite database management is used
- After implementation of the system, the developed system is tested on the target environment

1.7. Document Organization

This document is arranged as follows: chapter two presents general overviews about text entry techniques and handwriting recognition systems focusing on online handwriting recognition. It also presents a brief overview about the Ethiopic script. Chapter three states the reviewed related works carried out on the area and the efforts made to model OHWRS for Ethiopic and other scripts. In chapter four, description on the analysis of the system is presented. Chapter five discusses the design of the system. Implementation of the system is explained in chapter six and chapter seven states the experimental result. The last chapter, chapter eight draws conclusions and recommendation for future work.

CHAPTER TWO

OVERVIEW OF INPUT METHODS, OHWRS and THE ETHIOPIC SCRIPT

In this chapter, different issues that need to be raised to gain an insight into the area to which this work deals are concisely described. Section 2.1 gives possible input methods applied to Smartphones. Section 2.2 presents an overview on handwriting recognition. Categories of online handwriting recognition and major steps followed by them are presented in section 2.3 and 2.4 respectively. The last section of this chapter, section 2.5, gives an overview on the Ethiopic script where this work tries to focus on.

2.1 Input Methods for Handheld Devices

Recent advances in mobile computing made the tasks which are performed with such devices to diverse. Activities which were used to be possible to be carried out only in desktop systems are now emerging to the handheld technology. To make use of the functionalities of these handheld devices, an efficient method of human machine interaction is necessary.

Even though keyboard and pointing devices like mouse have been the primary and efficient text entry methods for desktop systems, such devices are not best options for handhelds due to several factors.

Size is the main constraint that hinders from directly adopting the traditional keyboard into handheld devices. Handheld devices like PDAs and Smartphones are designed targeting to fit into one's pocket. The trade offs in here is that whenever these devices made to be fancy with reduced size incorporating a keyboard, they will be forced to have tiny keys which would result in difficulties to hit them in typing a text. This problem could be worse for keyboards of scripts having large number of characters in their alphabet. This leads the computing technology to focus on other alternative input methods.

Input methods facilitate a way for enabling character entry of supported languages. For scripts with small number of characters in their character set, allocating a dedicated key in the keyboard for each character is possible but character sets like the Ethiopic script, which is relatively large compared to the Latin script, dedicating one key for each character is impossible specially in small handheld devices. Most of these devices are nowadays being

released without any room for hardware keyboard at all. Thus the need for a more established method for input is unavoidable.

Entry modalities have become more varied and getting more attention due to the recent advancements in Smartphones. Speech recognition, handwriting recognition and eye-tracking using image processing on Web cams can be mentioned as alternative options.

Speech based data entry can be used as an alternative input method to handheld devices but it has been observed that it has a computationally intensive nature. Size variations from small to large size of dictionary, isolated speech to continuous speech recognition, vocal characteristics of speakers and environmental conditions like noise and distortion over the channel makes the speech recognition technology challenging to implement[24, 25].

Pen computing is adopted by many handheld devices due to the emergence of touch-sensitive screen in handheld devices. Input methods like virtual keyboards, where soft keys resembling the hardware keyboard are presented on the screen to tap on, and handwriting recognition are greatly in use in these touch screen devices. Handwriting input methods are considered as convenient methods due to their resemblance to the natural handwriting [22].

2.2 Overview of Handwriting Recognition

Handwriting recognition is the task of transforming a language represented in its spatial form of graphical marks into its symbolic representation [15]. Depending on the way and time a recognition takes place, handwriting recognition is divided into two broad categories: offline handwriting recognition and online handwriting recognition.

In [18], handwriting recognition is defined as the ability of a computer or a handheld device to receive and interpret intelligible handwritten sources such as paper documents, photographs, touch-screens and other devices. The image of the written text may be sensed offline from a piece of paper by optical scanning in optical character recognition. Alternatively, the movements of the pen tip may be sensed online, for example by a pen-based computer screen surface or a handheld device like PDA or Smartphones using a stylus or finger.

In offline handwriting recognition, the data obtained for recognition are in static form of text representation lacking order and direction specification of the written text. The input for offline recognition engine is scanned image of a written text of which the writing process is already finished in advance. Offline handwriting recognition is considered as more

challenging problem because of the variations of the written characters styles and symbols lacking information as a function of time.

Online handwriting recognition is the very emerging type of recognition where its usage is dramatically increased due to the technological advancement especially in the handheld devices supporting a touch screen for navigation and data entry. In this type of recognition, the current information containing successive sequences of x and y coordinate pairs of a two dimensional surface as well as the pen pressure information showing the type of motion event being triggered in each coordinate pairs is presented to the recognition engine as a function of time. The pen pressure information can be extracted by observing one of the three motion events triggered on the touch sensitive device: the pen down, pen up or drag events. A pen down event shows the start of a stroke for a character, a pen up event on the other hand shows the end of a stroke which is triggered whenever the user removes the stylus away from the touch sensitive surface and a drag event is termed as an event which is found in between a pen down and a pen up event. These three event information are very helpful in online character recognition since they can precisely describe strokes information of a written character.

Thus, unlike offline handwriting recognition, online recognition engine takes advantage of the temporal information collected from the writing. This helps the online recognizer to know the number of strokes as well as the order in which a character is submitted and performs the recognition process in a real time.

Despite temporal information can be useful in online recognition, variation in writing styles of users is a challenge for recognition. This is a problem even for recognizing a single individual's writings due to the fact that a user might switch styles at different times and conditions rather than adhering to a single format. A recognition system should not be sensitive to minor variations. It should also have the ability to distinguish one character from the other following an efficient algorithm for its implementation.

Whenever these handwriting recognition engines get adopted for Smartphones, they must run on slow processors with a low memory footprint, must have a very high recognition accuracy enabling fast text entry. Moreover, they should allow easy correction [22].

2.3 Categories of Online Handwriting Character Recognition Systems

In designing a handwriting recognition engine, there are some basic features that need to be considered. Writing one's own style brings unevenness in writing units, which is the most difficult part to classify. Writing units reveal number, shape, size, order of stroke, and speed in writing. Variation in the number of strokes, their order, shapes and sizes, tilting angles and similarities among characters from one another are important factors, which are to be considered in the process of recognition [8].

Online character recognition systems are characterized based on the number of users they target, the restrictions they impose on users, the data set size and their ability to recognize users' entries efficiently. In [1, 3, 10, and 11], the taxonomy of online handwriting recognition systems is described from different perspectives. Different OHWRSs follow different approaches for their implementation. Some OHWRSs restrict users to follow well predetermined set of rules in using the recognition engines. Such systems are commonly known as constrained systems. Unconstrained systems on the other hand do not set any rules on the way users should submit handwritten data. Such systems should be optimized to handle variations of writing styles.

OHWRS are further subdivided into two broad categories: writer dependent OHWRS and writer independent OHWRS.

In a writer dependent recognition, the system is trained and optimized to recognize handwriting style of only one individual. The user is needed to provide his writing styles for each character as a training phase and then the system will start recognizing unknown character inputs from the user based on the stored dataset in the training phase. In a writer independent recognition, the system can handle the variations in multiple users' writing styles.

2.4 Steps in Online Handwriting Character Recognition Systems

Though different OHWRS follow different approaches in their implementation, there are common steps which are followed by most [1, 3, 5, 8, 19, 22].

2.4.1. Data collection

Data collection is the first step in the process of OHWR. Online handwriting data is typically a dynamic, digitized representation of the pen trace, containing sequential information about position, velocity, acceleration or pen angles as a function of time.

Online handwriting is usually acquired with a writing pad and a pen. There are two different digitizing technologies which are dominant in OHW data acquisition process. These are: touch sensitive sensors and electromagnetic sensors. Touch sensitive sensors are installed beneath the surface of PDAs and Smartphones and they are able to sense and digitize the contact of any pen like object without requiring a special pen. Whereas electromagnetic sensors are mostly employed in writing tablets and tablet PCs. They work only in combination with a special pen [19].

2.4.2. Preprocessing

Data collected during data collection process is often incomplete, noisy and inconsistent. Limited accuracy of the digitizing process, erratic pen movement and the inaccuracies of pen-down indication are among sources of noise in the online handwriting data [11]. Raw data cannot be used directly in the recognition process due to the artifacts it contains.

Preprocessing techniques are applied to the raw data to end up with a clean, standardized and reduced data removing noises and any unnecessary data points extracted in data collection process. Different recognition systems use a variety of preprocessing techniques depending on the characteristics of the scripts they handle and the methodologies they follow for recognition. Some of the activities handled in preprocessing include:

Extra pen up point elimination: the digitizer is so sensitive because it detects every slight movement of the pen, even when the tip is not quite touching the digitizer but is over the plane. This will cause the co-occurrence of coordinates at a point [8]. This step is useful to ignore data points extracted redundantly while a pen up point event is triggered in the touch screen.

Normalization: refers to the reduction of geometric variance in the handwriting that is typically due to differences in writing style and rendering environment prior to classification and thus to simplify the further recognition process. It includes scaling, translation and rotation activities [19, 31].

Re-sampling: very slow handwriting might generate repetition of coordinates at the same position during the data collection process. Thus to produce a quality data with equally spaced data points, re-sampling is applied. This step facilitates the commencing processing steps of recognition to have better efficiency dealing with reduced length of the data [22].

2.4.3. Segmentation

Segmentation is the process of dividing an entire writing into smaller pieces. Segments can be text lines, words, characters, strokes or sub strokes that occur in different characters [19]. Segmentation is an important task for recognition engines dealing with cursive scripts as it is difficult to distinguish where exactly a character ends and another begins.

2.4.4. Feature Extraction

This is also another step that should be undertaken before recognition or classification process proceeds. Feature is a representation of the on-line handwriting. Feature computation or extraction, is needed to represent the preprocessed input pattern to have a simplified, best representative taking in to account the different constraints accustomed with using the whole preprocessed input pattern.

Feature extraction generates a set of characteristics that shall allow for robust discrimination of patterns of one category against those of others [19].

2.4.5. Training

In training step, the system is trained with certain dataset collected during the data collection process. Characters features together with their respective class labels are specified to the recognizer.

2.4.6. Classification

This is a step of which most of the recognition's engine operations are set. The major task in classification consists of examining features of unknown, newly presented handwritten input pattern against classified saved templates and assigning it to one of the predefined character set class labels. In doing so it needs to apply classification technique. Various classification techniques are applied for online character recognition systems based on the characteristics of different scripts and the approach followed.

According to [15], there are two distinct methods of classification. The first method is formal structural and rule-based and the second method is statistical. In the structural approach, a character shape can be described in abstract fashion. If robust and reliable rules are applied to this method, it will let recognizers to deal with small amount of training data and the number of features used to describe a class of patterns can vary from one class to another. In statistical technique, a character shape needs to be described by a fixed number of features. This approach is based on statistical probability distributions. Artificial Neural Networks and Hidden Markov Models are classification techniques that are found under the statistical approach. Statistical techniques are usually applied for scripts in modeling online cursive handwriting.

2.4.7. Post Processing

Post processing is processing of the output from shape recognition. Language information can increase the accuracy obtained by pure shape recognition. For handwriting input some shape recognizers yield a number of alternatives for each character, often with a measure of confidence for each alternative. A post processor can operate on this information to obtain estimates for larger linguistic units, such as words [26].

2.5. Online Handwriting Recognition for Ethiopic Scripts

Each script has a set of icons, which are known as characters or letters that contain certain basic shapes [15]. The term Ethiopic, is the name given to the script used by Ethiopians and Eritreans to write in the major languages of the region. Amharic, the official language of Ethiopia, and Tigrigna, the official language of Eritrea, are two examples of languages that use Ethiopic script. As the cultural heritage and the script of the working languages of the Federal Government of Ethiopia and the majority of the regional states, the Ethiopic script continues to play an indispensable role in Ethiopia. The script first evolved from the original Ge'ez and it is represented in a 16-bit Unicode. Appendix A presents the Unicode representation of each symbols of the script. Out of the 435 characters in the alphabet, many of them are not practically used more often in day to day communications in the major languages like in Amharic [17, 12].

The Amharic language adapted the classic Ethiopic script for its written form, but a few letters have been derived from existing letters to denote sounds not found in Ge'ez [29]. The script is a syllabic alphabet where consonants are combined with vowels to form a letter. A

tabular format is used to represent the 238 commonly used alphabetic letters of the script in 34 rows and 7 columns.

Each rows are constructed by combining a consonant with ä, u, i, a, e, ə and o respectively. Table 2.1 represents the script's main character set containing characters which are directly taken from Ge'ez, others which are derived from the Ge'ez and one, '**ቨ**', from Latin to represent the 'V' sound in the Latin Script.

Different naming schemes are followed to refer to column members of characters in the tabular representation. The terms *Ge'ez, Ka'ib, Sal'is, Rab'i, Ham'is , Sad'is and Sab'i* are used as group names to letters in the alphabet which are found in columns one up to seven respectively. '*First Order', 'Second Order', 'Third Order', 'Fourth Order', 'Fifth Order', 'Sixth Order'* and *'Seventh Order'* are also used as a naming schemes for each letters based on the position of their column in the alphabet from left to right.

In [1], the term *basic* is used to refer to the core or first column characters and *non-basic* to refer to all subsequent columns in the character set in general. This arrangement is made based on the fact that the shape of the non-basic characters in each row is generally derived from the respective basic character which is found in the same row. In this document as well this naming scheme will be used.

In addition to the 238 letters represented in a tabular format, the Ethiopic script also contains extended letters representing labialized sounds, twenty numerals which are not often used in practice nowadays and some punctuation symbols. The common extended characters of the script are shown in Table 2.2 and the numerals are listed in Table 2.3.

Table 2.1 Main Ethiopic Script character set

Ge'ez	Ka'eb	Sal'is	Rabi'	Ham'is	Sad'is	Sabi
ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ
ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ
ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ

ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ
ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ
ነ	ኑ	ኒ	ና	ኔ	ን	ኖ
ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ
አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ
ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ
ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ
ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ
ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ
ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ
የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ
ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ
ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ
ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ
ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ
ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ
ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ
ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ
ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ
ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ
ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ

Table 2.2 Extended characters of the Ethiopic Script

ሏ	ሟ	ሯ	ሷ	ሿ	ቋ	ቧ
ቯ	ቷ	ቿ	ኋ	ኗ	ኟ	ኧ
ኳ	ዃ	ዟ	ዧ	ዷ	ዯ	ጇ
ጓ	ጧ	ጯ	ጷ	ጿ	ፏ	ፗ

Table 2.3 Ethiopic Numerals

Ethiopic Numeral	፩	፪	፫	፬	፭	፮	፯	፰	፱	፲	፳	፴	፵	፶	፷	፸	፹	፺	፻	፲፻
Arabic Number	1	2	3	4	5	6	7	8	9	10	20	30	40	50	60	70	80	90	100	1000

CHAPTER THREE

RELATED WORKS

Various research works have been done on online handwriting recognition of different languages. Some of them which are done for both the Ethiopic and other scripts are discussed in this chapter.

3.1. Online Handwriting Recognition for non-Ethiopic Scripts

Gareth Loudon et. al. [22], presented two integrated components of handwriting recognition. A recognition engine that supports sentence based handwriting input having a small memory footprint, higher recognition accuracy and fast processing time is stated as the first component and the second component focuses in describing a method that allows very simple editing and correction of recognized characters .

The recognition engine allows users to write a phrase at a time but it puts some restrictions on handwriting styles. Users should write characters in a specific way. They are allowed to write only lower case letters in a hand printed style and can convert to upper case during editing phase. Common punctuations and some gestures are supported in the recognition. Figure 3.1 shows writing styles of supported characters in the work. For its recognition engine, the work used discrete Hidden Markov Models (HMMs) stating their ability for better handling problems of non linear shifting and multiple representations of handwritten characters.

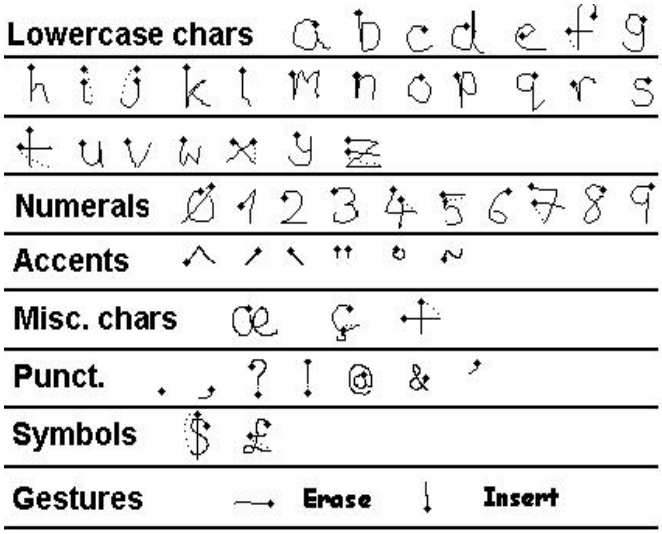


Figure 3.1 Writing styles of supported characters [22]

The work integrated the task of editing and correction with the handwriting input task to make its modification structure resemble the traditional pen paper correction with two options. Users can change a character by writing on the top of it or they can delete words by crossing them out and insert new ones.

H.Tanka et. al. [5], describe Fujitsu's online handwritten character recognition based on two character recognition technologies: hybrid character recognition and bigram-based context processing. Hybrid character recognition is stated by integration of online and offline handwriting recognition algorithms. This integration helps to achieve high performance even when the input pattern is with significantly different stroke numbers, orders and shapes. Bigram-based context processing, which is used for recognition of similar characters that cannot be distinguished by their shapes is also used for discriminating similar characters.

The work also developed an integration of two user adaptation methods named as hybrid adaptation, to improve the recognition performance working with the stored classification dictionary.

In a research work of A. Malaviya. et al. [9], a recognition methodology is described for three scripts by analyzing their individual characteristics. It used a fuzzy pattern description language to store fuzzy feature patterns in the form of fuzzy rules. Finally, it presented integrated multi-script recognition scheme in existing Latin recognition.

In [14], *Frog on hand,* which is an online handwriting recognition system is developed introducing a novel learning approach called cluster generative statistical dynamic time wrapping based on dynamic time wrapping and Hidden Markov Modeling. The system treats writing variations by holistically combining cluster analysis and generative statistical sequence modeling.

In the work of [16], an integration of an unconstraint, cursive online handwritten characters into Smartphones is presented. In the work, a character recognition system called *ResifCar* is introduced. *ResifCar* is based on a hierarchical fuzzy modeling of each character class. It is stated that the modeling approach provides a robust description of the most pertinent primitive shapes that define a character class. Description of the adaptation and the optimization of *ResifCar* for its integration into a smart phone device based on ARM 7 TDMI microprocessor is also specified.

3.2. Online Handwriting Recognition for the Ethiopic Script

Researches in online handwriting recognition have been made recently for Ethiopic character set. The research made by Abnet S. [1] is the first attempt which follows a method where the machine is trained for the handwriting of the users taking into account the usability of handheld devices (PDAs) by a single individual. Hence, it is a writer dependent EOHWRS where rather than storing multiple styles of handwritten data, it requires the user to train the system at first time usage.

In the research, an analysis about determination of shapes for non basic characters from their corresponding basic character for the Ethiopic alphabet characters is made. It states that most of the non basic characters can directly be driven from their respective basic character with some modification. The modification is made by adding a secondary stroke or by deforming their shapes but exceptions are made especially for the sixth and seventh column characters defining their difficult structure to set one rule for them.

In the design of the recognition engine, the 34 basic characters of the Ethiopic script are taken into consideration excluding numerals and extended characters in the characters set. The designed recognition engine consists of data collection, preprocessing, feature extraction and classification modules.

The functionality and effectiveness of algorithms designed in [1] is tested by the work of Abera A.[2] on PDA devices. Some of the algorithms proposed by [1] is modified while its implementation is made by [2] to increase the system's performance.

The research work conducted by Fikru T. [13], is a writer independent OHWRS. In its classification step, the work adopted a Support Vector Machine method for learning from various handwriting samples considering the 238 characters in the character set of the Ethiopic script in its recognition engine.

The research work made by Yonas H.[11], is another effort towards writer independent online handwriting recognition for Ethiopic script. It presented a new simplified Ethiopic character set symbols for characters as well as numerals. Hence, rather than making the machine be trained for multiple handwritings, users are needed to learn the new character set as represented in the simplified script in order to use the handwriting recognition engine. The recognition engine is made to be stroke number, order and direction dependent forcing the user to follow the rules strictly reducing variability in writing styles of each user.

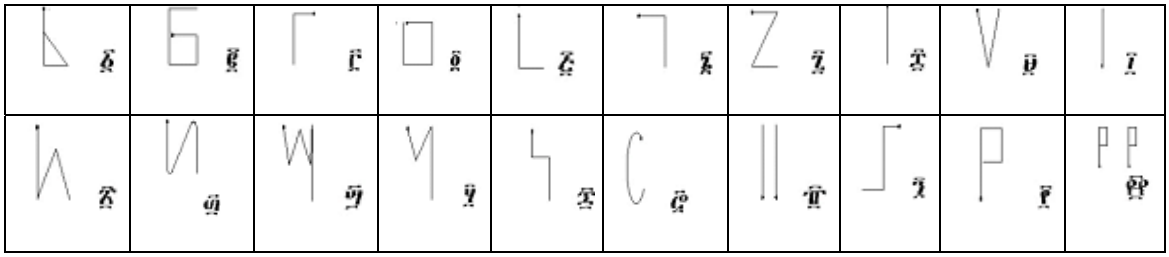
In the design of the simplified script, character groups with same sound in the Ethiopic script are given only one representative symbol for each group letting users to shift modes whenever they want to use these characters. Seventeen of the basic characters are made to directly map to their ordinary character representation in the Ethiopic script while eight of them to have a representation related to their sub strokes or sub components and characters but four characters: ሐ, ቨ, ዘ, and ገ are represented with symbols which totally deviate from the corresponding ordinary script.

For representing the non basic characters, six directional sub strokes are used together with their respective basic character representation. Table 3.1 shows supported characters in the simplified script and Table 3.2 depicts the representation of numerals designed in [11].

Table 3.1 Simplified Ethiopic script [11]

Ge'ez	Ka'ib	Sali's	Rab'i	Hami's	Sadi's	Sab'i
ሀ	→ ሁ	← ሂ	↑ ሃ	> ሄ	< ህ	↓ ሆ
ለ	→ ሉ	← ሊ	↑ ላ	> ሌ	< ል	↓ ሎ
መ	→ ሙ	← ሚ	↑ ማ	> ሜ	< ም	↓ ሞ
ሠ	→ ሡ	← ሢ	↑ ሣ	> ሤ	< ሥ	↓ ሦ
ረ	→ ሩ	← ሪ	↑ ራ	> ሬ	< ር	↓ ሮ
ሸ	→ ሹ	← ሺ	↑ ሻ	> ሼ	< ሽ	↓ ሾ
ቀ	→ ቁ	← ቂ	↑ ቃ	> ቄ	< ቅ	↓ ቆ
በ	→ ቡ	← ቢ	↑ ባ	> ቤ	< ብ	↓ ቦ
ተ	→ ቱ	← ቲ	↑ ታ	> ቴ	< ት	↓ ቶ
ቸ	→ ቹ	← ቺ	↑ ቻ	> ቼ	< ች	↓ ቾ
ነ	→ ኑ	← ኒ	↑ ና	> ኔ	< ን	↓ ኖ
ኘ	→ ኙ	← ኚ	↑ ኛ	> ኜ	< ኝ	↓ ኞ
ከ	→ ኩ	← ኪ	↑ ካ	> ኬ	< ክ	↓ ኮ
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
ዠ	→ ዡ	← ዢ	↑ ዣ	> ዤ	< ዥ	↓ ዦ
የ	→ ዩ	← ዪ	↑ ያ	> ዬ	< ይ	↓ ዮ
ደ	→ ዱ	← ዲ	↑ ዳ	> ዴ	< ድ	↓ ዶ
ጀ	→ ጁ	← ጂ	↑ ጃ	> ጄ	< ጅ	↓ ጆ
ገ	→ ጉ	← ጊ	↑ ጋ	> ጌ	< ግ	↓ ጎ
ጠ	→ ጡ	← ጢ	↑ ጣ	> ጤ	< ጥ	↓ ጦ
ጨ	→ ጩ	← ጪ	↑ ጫ	> ጬ	< ጭ	↓ ጮ
ጸ	→ ጹ	← ጺ	↑ ጻ	> ጼ	< ጽ	↓ ጾ
ፀ	→ ፁ	← ፂ	↑ ፃ	> ፄ	< ፅ	↓ ፆ
ፈ	→ ፉ	← ፊ	↑ ፋ	> ፌ	< ፍ	↓ ፎ
ፐ	→ ፑ	← ፒ	↑ ፓ	> ፔ	< ፕ	↓ ፖ
ቨ	→ ቩ	← ቪ	↑ ቫ	> ቬ	< ቭ	↓ ቮ

Table 3.2 Simplified Ethiopic numerals [11]



Another research made for the Ethiopic script is done by Daniel N. [10], which developed a writer independent, unconstrained OHWR system for the basic characters in the Ethiopic script. For classification of handwritten characters the work adopted Dynamic Time Warping technique stating its flexibility for handwriting styles.

In research work of Daniel K. [3], a writer dependent model for a multi- script (Ethiopic and Latin script) text editing under Palm PDA is presented.

The design created a separate input pad for handwriting entry of the basic 34 characters in the Ethiopic script and key tabs for selecting non basic characters. It aimed at integrating existing Latin recognition system which is based on Graffiti with the Ethiopic script recognizer.

In the work, an experiment was conducted to determine an optimal waiting time for starting a recognition process whenever a handwritten entry is provided. The experiment is conducted by registering consecutive pen up and pen down time instances of characters that indicate the time gap between strokes.

From the experiment conducted, it is realized that an average of 1600 milliseconds is required to write a basic character from the Ethiopic script. From that result it is concluded that for each handwriting character entry, it can be possible to make the recognition process to start its task after waiting the average time extracted from the experiment.

These, aforementioned works and others are efforts which are made to enhance the computing technology hit its target with high performance and ease of use for handheld devices. But in my knowledge no work is done to integrate Ethiopic handwriting recognition in Android-based Smartphones.

This work tries to maximize the performance which could be affected by writing variations placing some constraints. The constraint will force users to follow a particular way while submitting a handwritten input. A user driven input method that presents writing styles for

each of the thirty four basic characters, numerals and punctuation marks having very similar shape and structure to the ordinary Ethiopic script will be employed in the design and implementation of this work.

Moreover, as input methods allow users to supply characters in supported languages, this project will try to develop an input method which can be used as one of the input methods that allows submitting Ethiopic handwritten entry to applications and services that exist in Android-based Smartphones.

CHAPTER FOUR

SYSTEM ANALYSIS

The previous chapter presented the related works reviewed. In this chapter, a brief description about current system and the proposed system as well as functional and non functional requirements are presented. The requirement model which represents the functionality of the system is presented. Based on the requirement model, the analysis model is used for ensuring completeness and consistency of the requirements. For modeling, a standard modeling language, Unified Modeling Language (UML) is followed as an approach.

4.1. Current System

The Android Input Method Framework (IMF) is designed to support a variety of Input methods including soft keyboards, hand-writing recognizers, and hard keyboard translators. But only soft keyboard input method is currently part of the platform [28]. The flexibility of the framework to develop and integrate applications into it, gives a great advantage for developers to implement their own localized input methods (IME) that could be used by their target users. Even though there is no built-in handwriting IME included in the platform, recently third party developers are showing their interest in developing their own handwriting recognition engines for different languages to work on the platform [32, 33].

Currently, as it is stated in the first chapter, there is no EOHWRS that is implemented on Smartphones working in every application as an input method. This work tries to integrate OHWRS for Ethiopic script into the Android platform so that it can be used as an alternative IME in the device. To do so, it will make use of previous researches conducted on the EOHWR as main sources of reference.

4.2. Proposed System

This project work will try to incorporate a text editing facility using Ethiopic Handwriting Recognition as one alternative IME for any text entry applications of Android based devices.

The Ethiopic character set is relatively large compared to the Latin or Arabic scripts containing more than 400 characters. Its' large size as well as the close similarity between characters are challenging problems to have an accurate recognition engine that can distinguish one from the other. To increase the performance as well as usability, the proposed

system is designed to recognize the thirty four basic characters and then incorporate their respective family of non basic characters by setting them as alternatives soft keys to choose from. Whenever a recognition for one of the 34 basic characters is made by the system, a view of soft key buttons of which six of them representing the recognized character's family which are found in the same row in the character set table and other which are related to it from the extended labialized sound character set will be presented to the user as candidates. The Ethiopic numerals can also be used by writing '፩' and then soft key buttons of the nineteen remaining numerals will pop up as candidates. Punctuation symbols of the script will also be selected by writing their representative '**:**', which is usually used by the script users to separate words. In doing so, users will have an option to edit a text for entering non basic characters or symbols by pressing the soft key candidates in the layout.

As recognition accuracy is the main requirement in online handwriting recognition, in its recognition, the system will pursue a constrained, writer independent recognition scheme. This will be helpful to have a general format in user's entry making recognition process efficient. In following such an approach, the system will provide reference symbol for each basic character which best resembles the corresponding natural character in the script. Hence, users need to learn about the representative symbols prior to usage. The proposed system also puts some restrictions on users to write a character with same number of strokes, order and direction as predefined in the reference template.

In implementing the system, simple feature representation of each basic characters designed is persistently stored for reference in the recognition process. Recognition of each user's handwriting input will be made by comparing the entry against the saved template files of basic characters. Once a match is found, the system generates printed Ethiopic character and passes it to the focused field of the currently running application on the device.

If users want to use one of the non basic characters instead of the one which gets recognized, they can press on soft keys presented and then the system puts the pressed non basic character in place of the displayed basic character.

Figure 4.1 depicts the overall block diagram of the EOHWRS IME. The unknown input pattern which is collected as handwritten character input in the form of horizontal and vertical coordinate sequences and pen pressure information (Xt, Yt, and Zt) as a function of time will be passed for recognition then step by step operations will be performed on the input data to produce the desired output. First preprocessing will be handled on the input

pattern then features which represent the input pattern based on the order, direction and number of strokes used will be extracted then a classification step will proceed to match input pattern against the stored reference dataset. Once classification step is finished, the Unicode character representation of the recognized character along with its group in the character set will be computed. Finally the recognized Ethiopic character will be displayed in the edit box of the current application and its group characters will be displayed as soft key candidates.

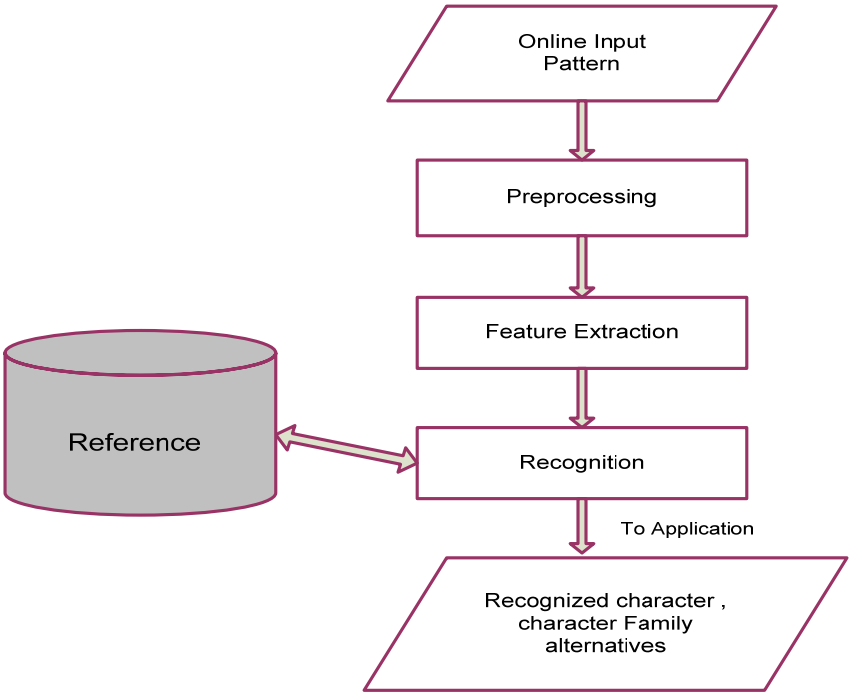


Figure 4.1 Block diagram of the constrained writer independent EOHWRS

4.3. Functional Requirements

The system should provide the following functional requirements:

- The system should allow the user to enter Ethiopic handwritten input for basic characters.
- The system should recognize users' handwritten characters
- The system should display printed Ethiopic character representation of the recognized character in the editable box.
- The system should provide soft key options of non basic characters to be selected.
- The system has to clear the writing canvas after each character entry.
- The system should allow deleting any written character.

4.4. Non functional Requirements

Availability, Ease of use, accuracy and response time are considered as the main non functional requirements which are expected from the proposed system.

Availability

The system should be available for every editable application of the Android based device.

Ease of Use

The system should provide an easy to learn template of the basic characters and its layout should be easy to use.

Accuracy

The system should be precise in recognizing user's entry

Response Time

The system should take a reasonable time to recognize and display printed form of the written input and also to present the generated soft key candidate characters for editing.

4.5. System Models

4.5.1. Use case

Use cases are important starting points which are needed to describe what the system is supposed to do. They are very helpful in mapping requirements to a system and in defining the boundary of the system. They refer to the interactions between a system and users [27].

Use case Identification

'End User', to refer to anybody who uses the Android based Smartphones is identified as an actor who participates and interacts to the system supplying an action. The system produces the desired result based on the user's actions.

Since a constraint, writer independent handwriting recognition approach is followed for this project, the system will be supplied with simplified forms of the script's characters persistently prior to usage. The recognition process will be initiated whenever writing on the screen is performed by an actor. 'Write Character', 'Preprocess' , 'Feature Extract', 'Recognize ', 'Generate non-Basics', 'Select non-Basic' and 'Delete Character' are identified

as use cases for this system. Figure 4.2 depicts the use case diagram of the system with the identified use cases and their interaction with the actor.

The actor will directly initiate three of the use cases: 'Write Character', 'Select non-Basic' and 'Delete Character'. The 'Write Character' use case starts whenever the user starts putting a stylus or his/her finger on the drawing area. Since the system can handle multi stroke inputs, this use case will end if certain timeout is occurred after the user raises the stylus from the drawing canvas.

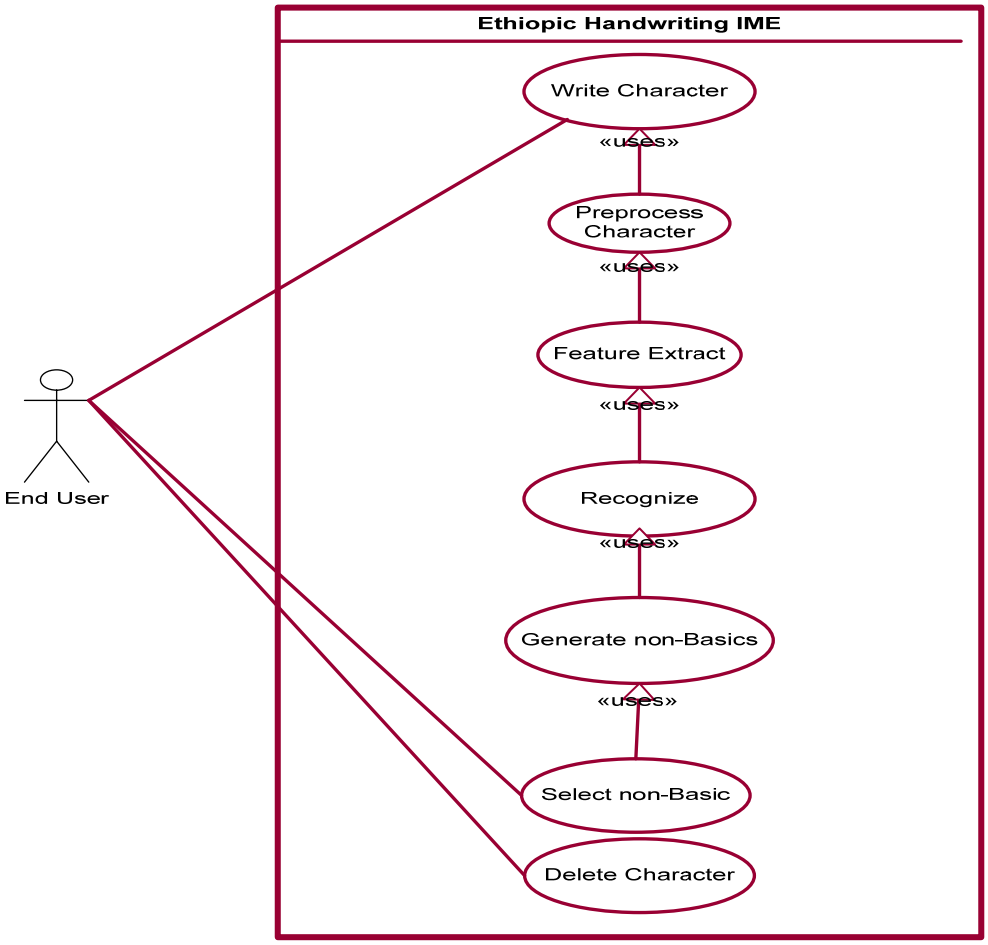


Figure 4.2 Use case Diagram of the System

Whenever the system realizes the user is not contacting to the drawing area for a certain time and the 'Write Character' has ended, it will initiate the recognition to proceed. In addition to the 'Write Character' use case, 'Preprocess' and 'Feature Extract' use cases are also needed to be initiated and execute before the 'Recognize' use case ends. 'Select non-Basic' and 'Delete Character' use cases are initiated by the actor pressing a soft key button on the screen.

Use case Description

A brief description of each of the use cases shown in Figure 4.2 is presented in the subsequent tables from Table 4.1 to Table 4.7.

Table 4.1 Use case description for Write Character

Item	Description
Use case Name	Write Character
Purpose	To write basic characters in the Ethiopic alphabet
Actors	End user
Pre Condition	The Ethiopic input method should be selected as an input method.
Flow of Events	1. The user focuses in edit box of an application. 2. The system presents an interface that allows an actor to perform the handwriting. 3. The user writes on the dedicated area on the interface. 4. The system concurrently draws on the positions where the user is placing his/her stylus on the writing area of the screen. 5. The system preserves the values of (X, Y) coordinate pair points as well as the motion event's action information from the drawn input pattern for latter analysis. 6. The system listens for touch action on the canvas [Alternate A1] 7. The system proceeds in collecting data points for the same stroke.
Post Condition	Drawn character on the drawing panel as well as Sequences of x coordinate, y coordinate and pen pressure values of the unknown drawn input pattern
Alternate A1:	Action-up event is triggered 6.1. The system sets a timer

	6.2. The system listens for Action-down event [Alternate A2] 6.3. The system resets the timer 6.4. The system proceeds writing for the next stroke
Alternate A2:	No Action-down event i.e A timeout is reached before the next Action-down event is carried out on the touch panel. 6.2.1. The system clears the drawing canvas 6.2.1. The system resets the timer. 6.2.2. The use case ends

Table 4.2 Use case description for Preprocess Character

Item	Description
Use case Name	Preprocess Character
Purpose	To eliminate noise and remove unnecessary points on the original input pattern
Actors	End user
Pre Condition	Sequence of x and y coordinates as well as pen pressure of the drawn input
Flow of Events	1. The system manipulates user's input applying noise elimination techniques 2. The system performs extra pen up point elimination process. 3. The system performs transformation and re sampling on the input pattern
Post Condition	Noise free preprocessed input data

Table 4.3 Use case description for Feature Extract

Item	Description
Use case Name	Feature Extract

Purpose	To represent the input pattern in to a representative form for recognition
Actors	End user
Pre Condition	Preprocessed data
Flow of Events	1. The systems tries to find relevant representative sequence coordinates of the input character. 2. The system stores the representative sequences of an input.
Post Condition	Features of input pattern

Table 4.4 Use case description for Recognize use case

Item	Description
Use case Name	Recognize Character
Purpose	To classify to which character class the input which the user submits belongs.
Actor	End user
Pre Condition	Features of Input pattern and saved template reference
Flow of Events	1. The system manipulates the number of strokes of the input pattern.[Alternate A:] 2. The system collects list of character labels having same stroke number as the input pattern from the stored stroke information reference table. 3. The system fetches each feature of characters in the list from the stored feature reference table. 4. The systems tries to match user's input features against each of the fetched character features. 5. The system gives a rank on the matched results. 6. The system returns the top element as its result. 7. The system displays the recognized Ethiopic character in

	the edit box.
Post Condition	Update on edit box.
Alternate flow A:	Input pattern with invalid stroke number is provided 1. The system returns 'NULL' 2. The use case ends.

Table 4.5 Use case description for Generate Non-basics

Item	Description
Use case Name	Generate non-Basic
Purpose	To display non basic character candidates as soft keys
Actor	End user
Pre Condition	Recognized basic character, numeral or punctuation mark
Flow of Events	1. The system looks for characters which are in the same family as the recognized one. 2. The system collects the non-basic characters to be displayed. 3. It allocates the collected characters to each soft key buttons 4. The system displays the buttons on the screen.
Post Condition	Updated buttons Layout

Table 4.6 Use case description for Select non-Basic

Item	Description
Use case Name	Select non-Basic
Purpose	To allow users to use non basic character options
Actor	End user
Pre Condition	Soft keys on the screen

Flow of Events	5. The user presses on one of the soft keys. 6. The system removes the basic character in the edit control box and appends the pressed character in the edit control box 7. Selected character will be part of a text in the edit box control.
Post Condition	Updated edit box

Table 4.7 Use case description for Delete character

Item	Description
Use case Name	Delete Character
Purpose	To allow users delete a character
Actors	End user
Pre Condition	At least one character on the focused edit box.
Flow of Events	The user presses on "ሰርዝ" soft key button 2. The system removes the character which is found just before the cursor's position on the focused edit box. 3. Deleted character will be removed from the edit box.
Post Condition	Updated edit box

4.5.2. Class Diagram

In Object Oriented approach, classes are high level abstractions to specify objects based on their behavior and structure. Objects are instances of classes that are created, modified and destroyed during the execution of the system. Class diagrams can be used to represent the system in terms of objects, classes, attributes and operations [27]. Figure 4.3 shows the class diagram of the system presenting the classes, attributes and operations contained in each classes and relationships between them.

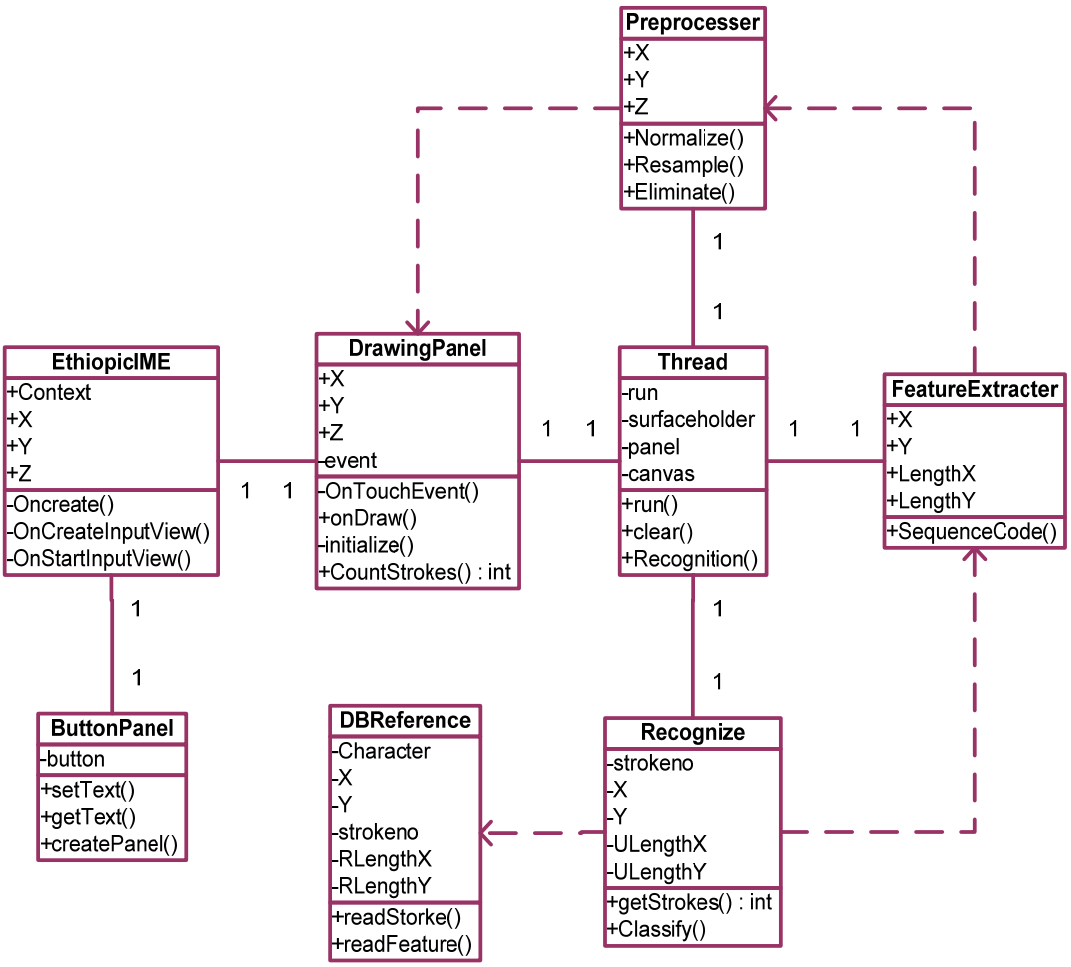


Figure 4.3 Class diagram of the System

The *EthiopicIME* is the core class of the system incorporating several operations of an Input Method. Each instance of the *EthiopicIME* class links to one *DrawingPanel* and a *ButtonsPanel* object in its creation to construct its interface using these objects layout.

The *DrawinPanel* class object, which is responsible for creating and controlling the drawing canvas structure by facilitating visual drawing, clearing and capturing data from a canvas, links to one *Thread* instance to capture each instant. Each *Thread* object on the other side makes a call to *Preprocess*, *FeatureExtract* and *Recognize* objects setting an appropriate time to conduct the recognition process.

The *ButtonsPanel* object is responsible in creating and controlling the contents of the soft key buttons layout.

To accomplish the overall system's task, there exists dependence between objects of classes. The *Preprocess* class depends on the *DrawingPanel* class as its objects always use data extracted by the *DrawingPanel* object. The *FeatureExtract* class on the other hand depends on the *Preprocess* since it needs to work on the preprocessed data. Moreover, there exists dependence for the *Recognize* class. The classifier *Recognize* depends on the *FeatureExtract* and *DBreference* classes to perform matching one against the other.

4.5.3. Sequence Diagram

Sequence diagrams represent flow of logics showing the interactions which take place among objects that participate in the use cases. They are used to formalize the behavior of the system and to visualize the communication among objects. Sequence diagrams describing the behavior of the system in terms of the sequences that are performed in the overall system for each usecases are presented in subsequent figures, from Figure 4.4 to Figure 4.9.

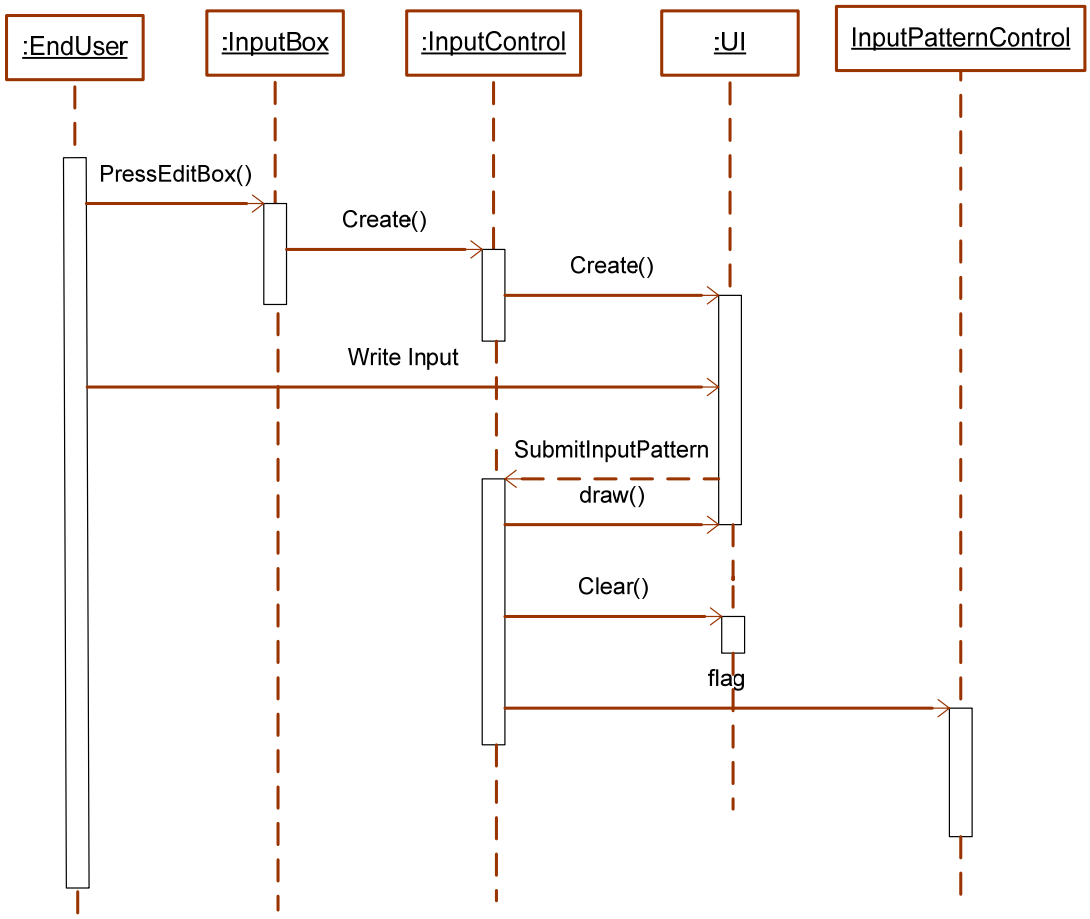


Figure 4.4 Sequence diagram for Write Character use case

As shown in Figure 4.4, the end user initiates the writing process by first clicking on any edit box of an application. The input control creates an interface that allows performing handwritten entry to the system. Once the user interface is presented, the end user can perform the writing process. Whenever the input control finishes collecting the input pattern of a character written by the end user as a sequence of data points from the user interface, it sends a message for the user interface to be cleared to let another character entry. It also sends a confirmation message to the input pattern control, which is responsible for controlling the subsequent operations of recognition, proceed with its tasks.

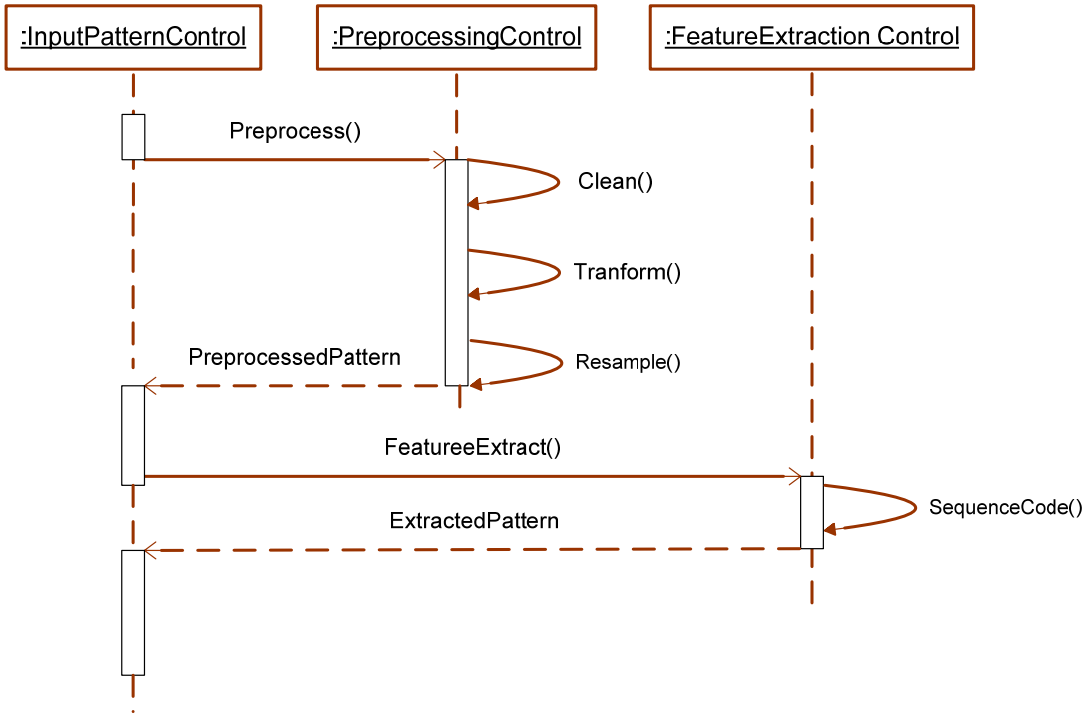


Figure 4.5 Sequence diagram for Preprocess and Feature Extract use cases

Input pattern control manages the collected unknown input pattern during the *Write Character* process for the recognition process to be taken care of. As shown in Figure 4.5, first it passes the collected input pattern to the preprocessing control. The preprocessing control performs all the operations needed and then returns its result back to the input pattern control. The input pattern control then passes the cleaned preprocessed data to the feature extraction control to extract relevant representative data points from it. The feature extraction control makes feature selection operation and returns the result back to the input pattern control.

For recognition, the input pattern control passes the extracted features of the unknown input pattern. The recognition control computes number of strokes of the extracted input feature and reads features of characters having the same number of strokes from the persistently stored templates data via the *ManageReferenceControl* object which is responsible in accessing the database content. Thus, the recognition control performs matching operations on the unknown input feature against each of the collected reference features and ranks the matching result based on the computed distance. Lastly, it sends its result back to the Input pattern control. The Input pattern control passes the result to the Input Control to get the resulting character. The *Input Control* will then send it to the *characterList* to collect the recognized Ethiopic character in the character set. Then the basic character will be passed to the input box where the editing application is being taken care of. This sequence ends whenever the *InputBoxcontrol* gets a command to update its content appending the recognized character. Therefore, the edit box will be updated displaying the appended basic Ethiopic character. Figure 4.6 depicts the sequence of the recognize use case.

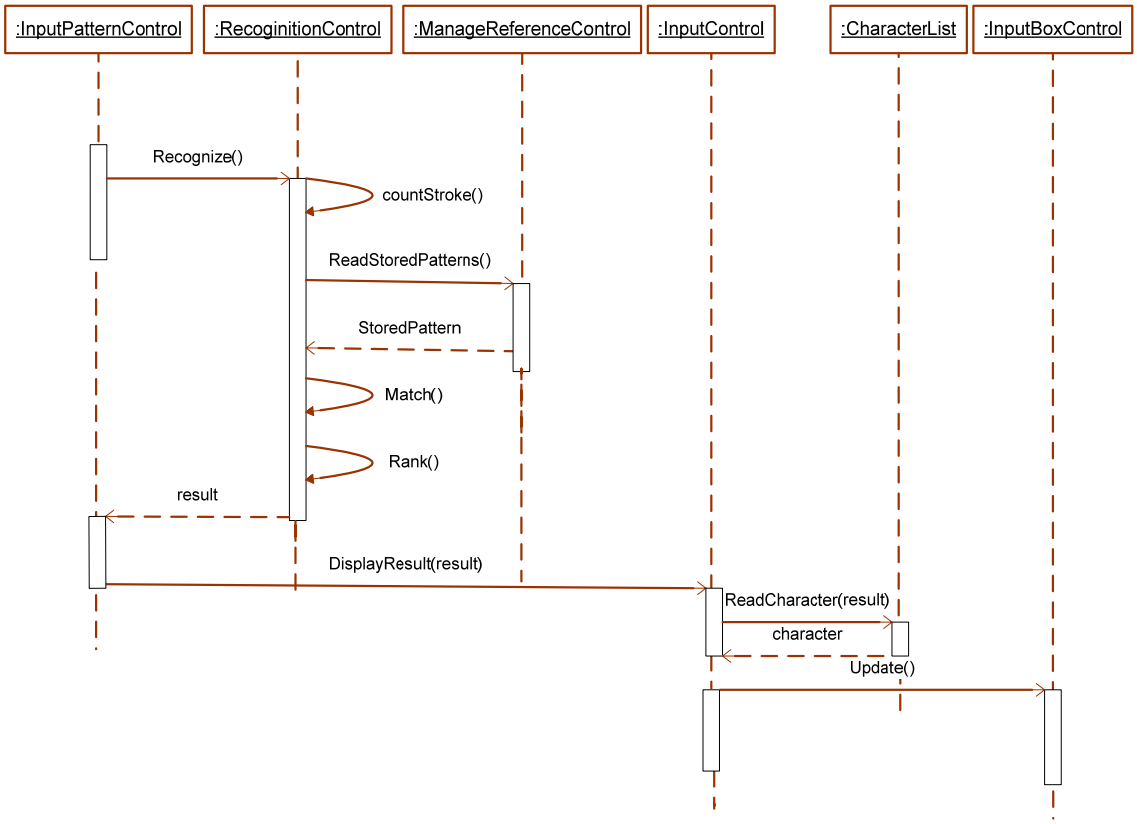


Figure 4.6 Sequence diagram for Recognize use case

The end of a sequence for *Recognize* use case results in the creation of the sequence of *Generate non-Basic*. Whenever the *InputControl* gets a character recognized, it sends a command to the *ButtonsGroup* Control to generate non basic character candidates in the form of pressable soft key buttons. Based on the content of the basic character, which it received from the *InputControl*, the *buttonsGroupControl* performs extraction of character family of the basic character and then updates the contents of its buttons. Lastly, it sends its content to the *User Interface* object to display the updated content. Figure 4.7 shows the sequence for Generate Non-basic use case.

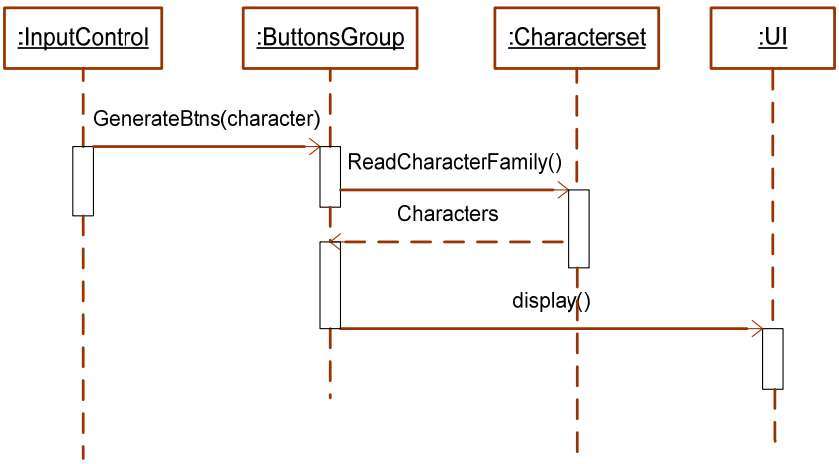


Figure 4.7 Sequence diagram for Generate Non-basics use case

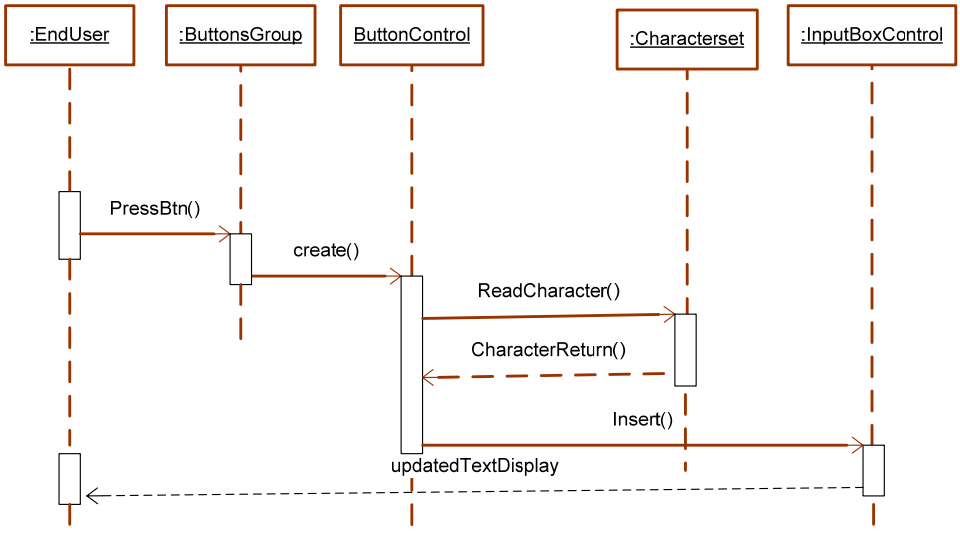


Figure 4.8 Sequence diagram for Select Non-basic use case

As shown in Figure 4.8, the *Select non-Basic* use case is initiated by the end user. Whenever the user triggers a button press in one of the soft key buttons, the *'ButtonsGroup'* object of the *ButtonsPanel* class, which manages the appearance and the contents of soft key buttons on the screen, will create a *ButtonControl* object. The *ButtonControl* object maps the content of the pressed button to a non basic character from the character set and sends the result to the *InputBoxControl* to append it. The *InputBoxControl* then takes the non basic character and updates the content of the edit box by overriding its respective basic character.

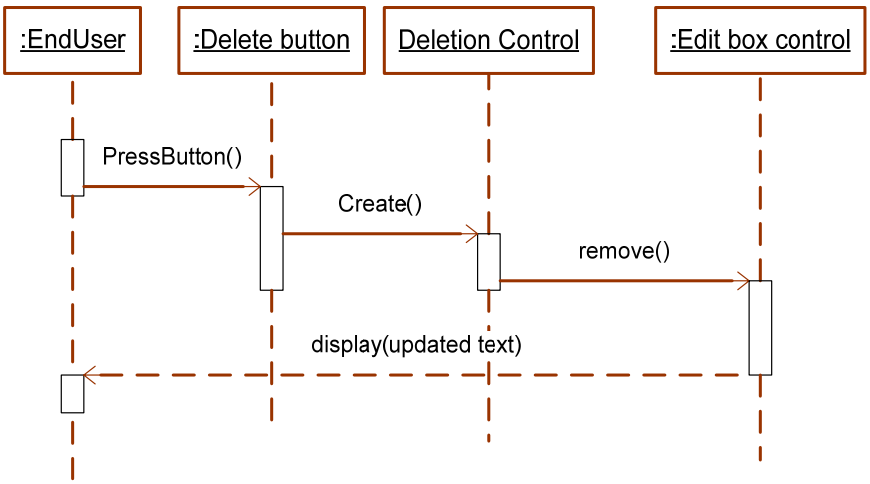


Figure 4.9 Sequence diagram for Delete character use case

If there is one or more characters in the input box, the user can press a button for deletion, "ሕፐፋ". The *'DeletionControl'* will call a removal operation to proceed on a character which is found in the cursor's position. Once the removal operation is performed, the *End user* can see the updated input box. Figure 4.9 shows the sequence for 'Delete Character' use case.

4.5.4. Activity Diagram

An activity diagram describes a system in terms of activities. Activities are states that represent the execution of a set of operations. The completion of these operations triggers a transition to another activity. Activity diagrams can be used to represent control flow, the order in which operations occur and data flow of the objects that are exchanged among operations [27]. The activity diagram for the recognition system is shown in Figure 4.10 describing the set of operations and their order of execution. The activity of the system starts by collecting the input data from a touch sensitive interface. The preprocessing stage will continue by removing the noises from the input data. The representative features of the

preprocessed pattern will then be extracted in the feature extraction operation. The extracted pattern will be matched against the saved template references. After computation of a match, the character representing the best score will be displayed in the edit box. The button layout will also be updated with the resulting character non basic character contents. If the user selects one of the non basic key buttons, the system displays the selected non basic character on the edit box overriding the corresponding basic character.

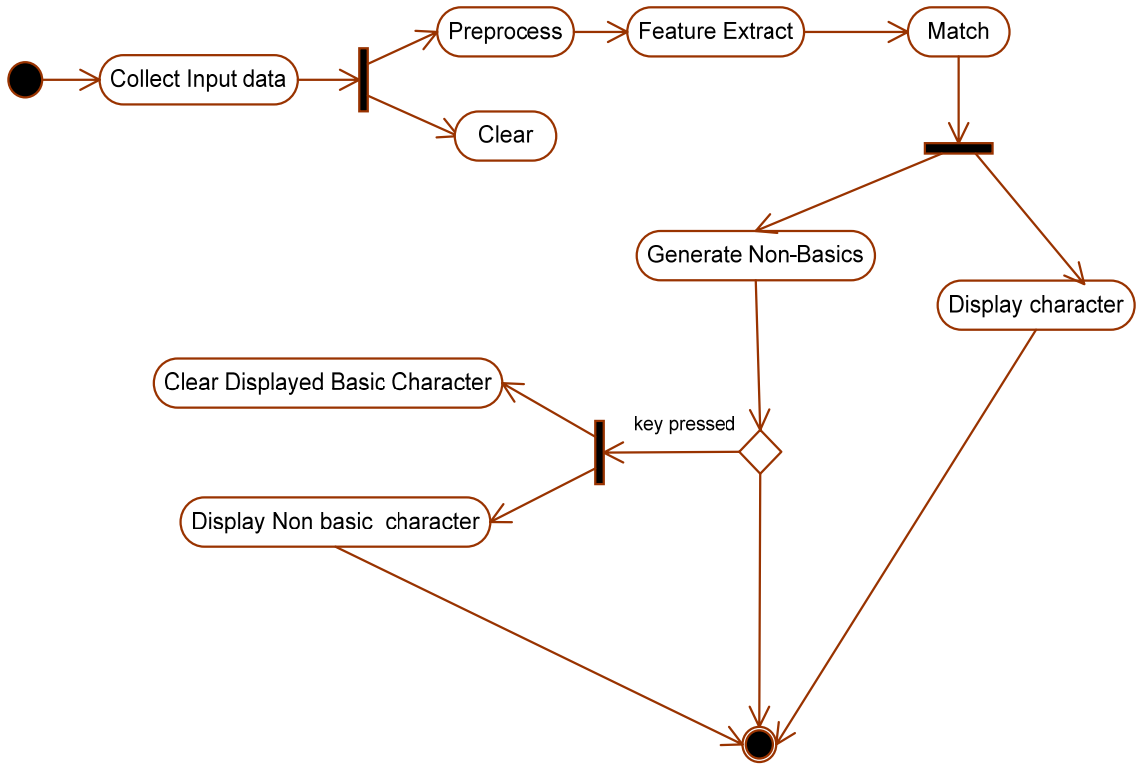


Figure 4.10 Activity Diagram of the System

CHAPTER FIVE

SYSTEM DESIGN

In the previous chapter, the requirements of the system are identified and the analysis models are presented. In this chapter the design of the system, which is the transformation of the analysis model in to design model is described. Identification of the design goals is made in the next section. The system design model is presented in section 5.2 providing a brief description on each subsystems identified for the system followed by a description on the persistent data management. The last section of this chapter, section 5.3 presents a design constructed for the basic Ethiopic script letters which are used for extracting an input for the system.

5.1. Design Goals

For a successful acceptance and usage by target users, the system should set and satisfy some major qualities. Design goals are derived from the non functional requirements [27]. Taking in to account the size, speed and space limitations on handheld devices the design goals are set into the four categorical groups as stated below.

5.1.1. Performance Criteria

Response Time

The system should respond quickly to each user's commands in terms of touches. It has to draw visually on the screen as a user writes on the surface with an un-noticeable time delay.

Throughput

Since the recognition engine is going to be used online, for each written character input, the system should respond in a reasonably short time setting an optimized time that it should wait to start recognition. The system should also return recognized character to an application before another handwriting is performed by the user.

Memory

As space is one of the main constraints in handheld devices, the system should have an optimized space requirement for the application program as well as for the data which is going to be persistently stored.

5.1.2. Dependability Criteria

Availability

The system should be available in every text entry application of the device.

5.1.3. Maintenance Criteria

Extensibility

The system should enable new functionalities like word prediction to be added on it without affecting its current functionality.

Modifiability

The system will be designed in such a way to enable any modifications to its functionalities with minimal effort.

5.1.4. End user Criteria

Usability

Considering screen size and processing speed of handheld devices, the system should have a well designed interface with easy to learn and use components. It should serve as an efficient and comfortable interaction technique for users with few errors if any.

5.2. System Design Model

5.2.1. System Architecture

To have a concise understanding in decomposition of the system into subsystems, the general architecture of the system is presented in hierarchy of layers. Figure 5.1 illustrates architecture of the system. As shown in the figure, Application, Input Method Manager and Input Method are the three high level layers of the system when it works as an IME.

The application layer is a layer that holds system or user application services that users can interact by entering text into it. This layer can be taken as a client to the system.

The input method manager is the one which is responsible in handling interactions between applications and the input method. This layer binds to the input method and manages the visibility of its UI while giving focus to only one application at a time [28].

The input method layer is a component where the Ethiopic handwriting is fully described and what the system handles. It allows users to use Ethiopic handwriting and enter handwritten

text that can be used by the underlying application providing the appropriate UI via the Input method manager. It is this component that will be considered in partitioning the proposed system into subsystems. The Input method component by itself has sub components implementing a particular interaction model. These are Extensible Markup Language (XML), Controller and Model.

The XML component contains an XML file that gives a description to the Android platform about the input method application so that it will get to be executed like any service on the device.

The controller is responsible in listening touch events and key presses made on the interface and handling modifications on the layouts of the interface based on the actions performed.

The model sub layer contains the views which contain the UI controls and the operational classes that perform the data entry, recognition, display and repositories' handlers. This sub layer is responsible in maintaining domain knowledge operating on each of the events generated. It collects the data points from the view's UI and generates the expected result performing different recognition steps. It uses two repositories for referencing features and font data but the two repositories will be stored in different locations. The reference data will be stored in the handwriting system's database's data folder while the font data, which contains a font that can display Ethiopic character based on Unicode format, will be stored in the device's SD card, which is a removable memory.

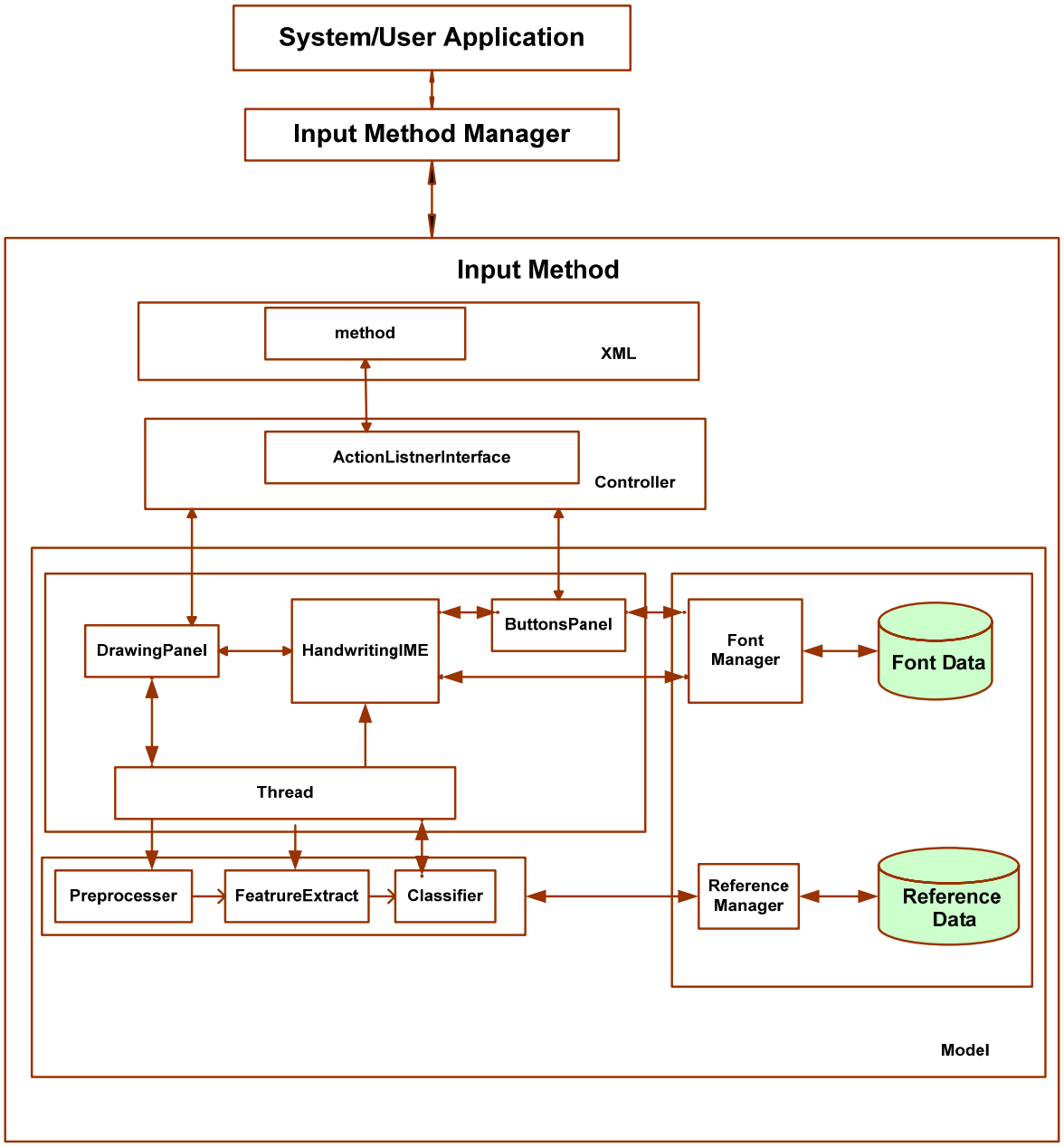


Figure 5.1 System architecture for constrained WI EOHWRS

5.2.2. Subsystem Decomposition

Subsystems are needed for simplifying the complexity of a system. The system is decomposed into five major subsystems: Interaction, Pre-processing, Feature Extraction, Classification and Database subsystems. These subsystems are identified with the aim of achieving strong coherence and loose coupling. Figure 5.2 depicts the system with the identified subsystems and the interaction between them.

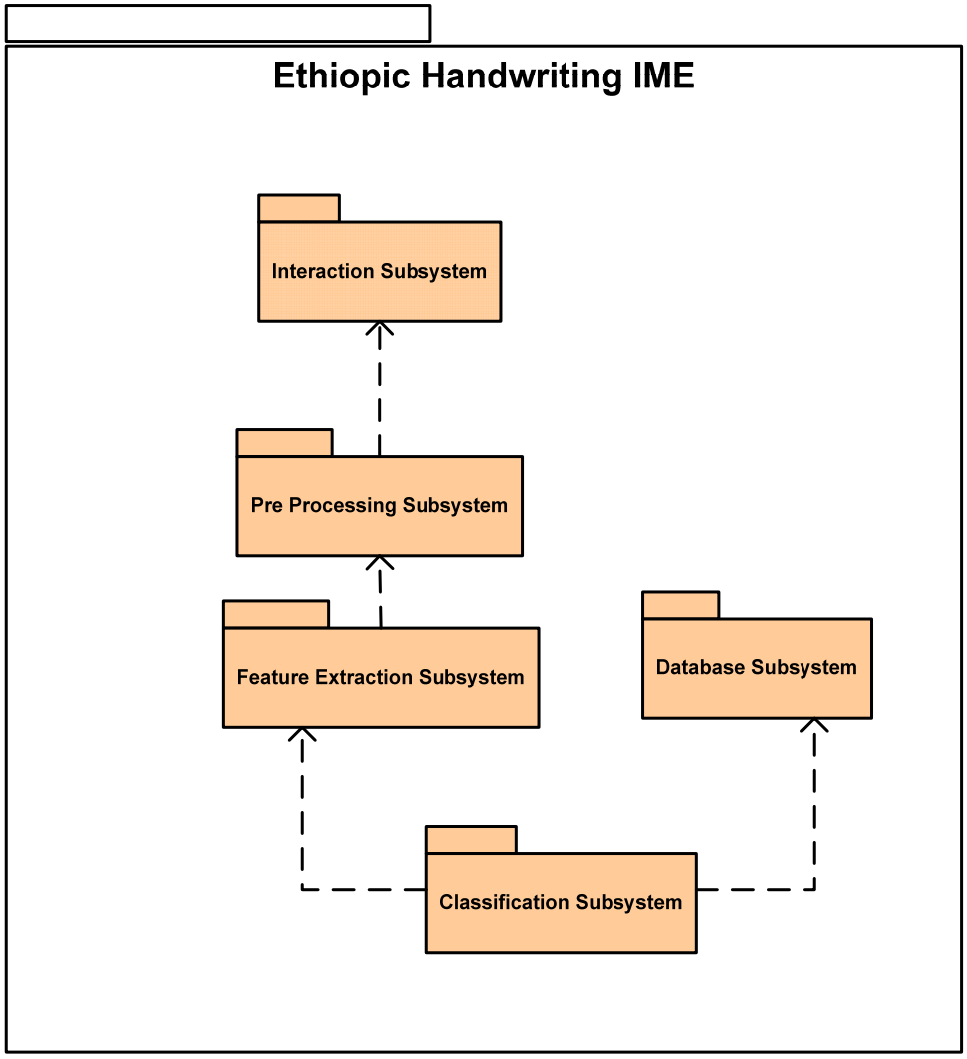


Figure 5.2 Subsystems of the Ethiopic Handwriting IME System and their dependency

5.2.2.1. Interaction Subsystem

This subsystem is the main starting point of the system where data entry and display is handled. It is a subsystem that manages the connection between the system and other applications that use it for editing and displaying text.

It is mainly responsible in creating the layout showing the views of the input method in each application instances and handling the layout. It is also involved in capturing the pen down, drag and pen up events triggered on the screen and drawing the input visually and clearing it when necessary. In addition, it collects the original dataset code sequences of the unknown

input pattern which is comprised of the horizontal and vertical coordinate pairs and their respective pen-pressure information in the form of X, Y and Z respectively.

This subsystem is also responsible in informing the recognition process to start by identifying an appropriate time when the data collection operation of a character is ended.

EthiopicIME, ButtonsPanel, DrawingPanel and *Thread* classes are contained in this subsystem to complete the aforementioned tasks of the subsystem interacting from one to other.

5.2.2.1. Preprocessing Subsystem

This is a subsystem which is mainly concerned in preserving valid data points of the collected data by removing noises and inconsistencies which are introduced during the data collection process of the interaction subsystem. It is also responsible for settling the data collected to have the same format performing transformation on it. In the transformation stage, Size normalization will be done to reduce the complication in the later steps. Preprocess class is contained in this subsystem.

5.2.2.2. Feature Extraction

This subsystem uses the output of the preprocessing subsystem as its input and focuses on selecting representative points of a given sequences of coordinates data to best describe the input pattern so that the commencing stage will be simplified dealing only with the selected features. It provides a sequence of X and Y coordinates values and their respective length that best represent the given input character pattern. Features will be extracted as it is stated in [1] having five different values.

5.2.2.3. Classification

This subsystem is where the recognition algorithm of the system is applied. It accepts the unknown or unclassified feature patterns extracted during the Feature Extraction and then it looks for assigning a class to it. To classify the unknown features into one, it reads features of candidate classes from the stored feature reference templates having the same stroke number as the input feature. It performs a template matching between the unknown input feature segments and each of the candidate reference template features and returns the best matching candidate as its output. This subsystem packages the *Recognize* class in it.

5.2.2.4. Database Subsystem

This subsystem represents the persistent data of the system. It contains representative feature patterns of the known basic (Ge'ez) alphabetic characters which are set based on the design of the script provided in section 5.3 setting a value for each strokes of a character based on its direction. This subsystem contains the *DBReference* class to manage the interaction with the database when ever access to the persistent data store is made by the classification subsystem to read reference template features.

5.2.3. Persistent Data Management

Rapid and efficient data storage and retrieval are essential for a device whose storage capacity is limited by its compact nature. Android provides a lightweight relational database for each applications using SQLite. SQLite database exposes methods to create, delete, execute SQL commands and perform other database management tasks [30].

SQLite Database system will be used for storing persistent template files which represent each character's features in the form of x, y coordinate pairs as well as their respective length measures which are pre set based on the design of the Ethiopic script in section 5.3.

To enhance the computational speed of the recognition, the database structure consists of two tables named as '*FeaturesTable*' and '*StrokesTable'*. The *StrokesTable* is used to list each basic character with its corresponding stroke number while the *FeaturesTable* hold features of each basic character. As a result, whenever an unknown input pattern is presented for recognition, list of letter names having the same number of strokes as the input feature will be extracted from the *StrokesTable* presenting candidate letters that the recognition should consider. Then the classification will start by selecting only features of the candidate letters from the *FeaturesTable* and matching then to the input pattern.

The Strokes table contains two attributes: 'LetterName' and 'StrokeNumber'. Letter Name column will hold the name of each basic character labeled in the form of 'Letter1' to 'Letter34' to represent class labels for characters '  ' to '' respectively, 'Letter35' for Ethiopic numeral 1, 'Letter36' for a space and 'Letter37'for punctuation. Stroke Number column holds the number of strokes allocated to represent the corresponding character in the predefined template as shown in Table 5.4.

Features table on the other hand has five columns: 'Letter Name', X, Y, Length-X and Length-Y columns. Letter Name holds the class labels of each character. X and Y hold the

code sequences of horizontal and vertical coordinates respectively. These coordinate values are computed based on the labeling used in [1] giving representative to segments based on their direction. A value in the range of 1 to 5 is given to each coordinate values showing their direction. Table 5.1 shows the representation used to store the reference features of each character. Based on the direction and order of each stroke to represent a character, Length- X and Length- Y columns hold the weight of each coordinate value reducing duplicate entries in the database. The database table structure of the two tables is shown in Table 5.2 and Table 5.3.

The data that stores the font of the Ethiopic character is also part of the persistent data as Android does not have a built-in Ethiopic font. EBRIMA, which is one of the fonts used to display the Ethiopic script on the screen using its Unicode representation needs to be persistently stored in an SD card.

Table 5.1 Labeling used to represent reference patterns [1]

Condition	Number Code
Beginning of a stroke	1
Increasing	2
Decreasing	3
Constant(nearly constant)	4
End of a stroke	5

Table 5.2 Strokes Table

Attribute	Data Type
LetterName	String, Not Null
StrokeNumber	Int, Not Null

Table 5.3 Features Table

Attribute	Data Type
LetterName	String Not null
X	Int, Not Null
Y	Int, Not Null
LengthX	Int , Not Null
LengthY	Int, Not Null

5.3. Design of Basic Characters of the Ethiopic Script

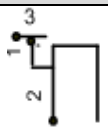
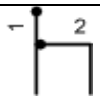
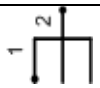
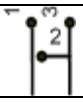
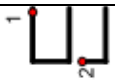
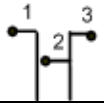
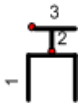
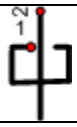
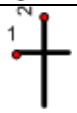
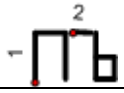
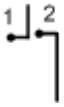
In the simplified script introduced by Yonas H.[11], the design considered both basic and non basic characters of the Ethiopic characters as well as numerals but considerable number of characters are designed to have a structure which varies from the original script to reduce the similarities of characters in the script. In addition, character groups with same sound in the script are given only one representative symbol for each group letting users to shift modes whenever they want to use these characters.

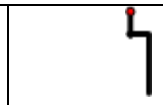
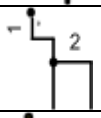
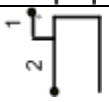
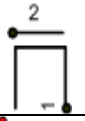
Since only the basic characters are incorporated to be recognized by the handwriting engine of this system, to maximize the usability and efficiency of the system, modification on the basic characters of the Simplified script by [11] is needed. We designed a new template for the 34 basic characters, numerals and punctuations. The template is constructed in such a way to resemble the natural handwriting taking into consideration the most probable ways where writers of the script use to write in a pen paper environment. In designing the symbols about twenty subjects of different background were consulted to assure the usability of the structures.

Even though the number of stroke usage to represent a character may vary from subject to subject, we tried to represent a restrictive, multi–stroke model of characters which will be acceptable and usable by most subjects closely resembling the natural handwriting. The design is made to reduce the complexity to learn a completely different structure and symbols to write characters that are introduced in using a uni-stroke format.

Table 5.4 shows the structure of the new template for basic characters of the Ethiopic alphabet. As it can be seen from the table, maximum of three strokes are used to represent any of the basic characters. In addition to the stroke number assigned for each character in the design, users should also adhere to the order and direction to write a character to have better recognition accuracy. The dot marks in each symbol represents the starting point for a stroke. For the multi-stroke character representations, numbers are used to represent the sequences of strokes that constitute a character should be written.

Table 5.4 Design of basic Ethiopic script characters

Basic printed Character	Designed symbol	Stroke Number	Basic Printed Character	Designed symbol	Stroke Number
ሀ		1	ኸ		3
ለ		2	ወ		2
ሐ		2	ዐ		1
መ		1	ዘ		3
ሠ		2	ዠ		3
ረ		1	የ		1
ሰ		2	ደ		1
ሸ		3	ጀ		3
ቀ		2	ገ		1
በ		1	ጠ		2
ተ		2	ጨ		2
ቸ		3	ጸ		1
ኀ		2	ጸ		2

ሃ		1	ፀ		2
ኻ		2	ፈ		2
ኦ		2	ፐ		2
ኩ		2	ፕ		2
Ethiopic Numeral ፩					1
Punctuation :					2
Space					1

One challenging problem in applying online, multi stroke character recognition system for handheld devices is, knowing the correct time to start the recognition process. For uni-stroke recognition systems, this is not an issue as they can start the recognition process just after a pen-up event, which indicates the end of a stroke, is triggered on the touch sensitive surface. But in a multi-stroke recognition system this is a challenge because of the variations that exist in strokes of characters in the alphabet. This confuses the recognizer to perform its task.

To overcome this problem in the research work of Daniel K. [3], setting a waiting time which is the average time taken from the experimental result was suggested. In such a case, users need to finish writing a character before the limited amount of time is up. If a uni stroke character is inserted in a pretty fast time by a subject, it will result in delay, since the system needs to go to an idle state until the waiting time is up rather than proceeding with the recognition. This average time waiting does not give a full-fledged acceptance by users especially if uni-stroke characters dominate the entry.

For the Ethiopic Handwriting IME of this project, rather than setting a constant time for each characters restricting users to finish writings of each characters within a specified period irrespective of their stroke difference and conversely wait for too long even if a character entry is ended, the system is made to check whether it is the end of a character or not by only computing a delay made by the user after writing a stroke. This leads to have time setting which is only based on the in-between strokes time, regardless of the time taken by users to

finish writing a stroke. This gives flexibility for fast as well as slow writers to benefit from the system based on their behavior. Thus, while writing, whenever a pen up action is triggered, a thread which is made to run in the background starts to listen if another pen down is made by the user before a time out value expires calculating the time differences of the end of the last stroke and the current time. If not, it clears the screen and starts the recognition process. By applying this process the total writing speed and the systems' flexibility will be expected to be maximized. The algorithm is shown below in Figure 5.3.

```
Input:  Pen-up Time // a time which is registered when a pen-up event is triggered showing
                        //end of a stroke
TimeOut  ←  1    // maximum delay allowed before recognition in seconds
T1  ←  Pen-up Time
While (! NextPen-down)      //if input for the next stroke is not started
     T2  ←  getCurrentTime
     If( T1>0 and T2-T1> = TimeOut)   // check if recognition needs to start
          Clear Canvas
          Do Recognition
          T1  ←  0                  // reset timer
          Exit While
     End If
 End While
 If(NextPen-down)
       T1 ←  0
       Continue writing            //continue input for the next stroke of a character
  End If
 Output :  Recognized character
```

Figure 5.3 Algorithm for controlling recognition process

CHAPTER SIX

IMPLEMENTATION

In chapter five, the design of the system is described in detail. In this chapter, the development tools and the environment used to implement the system, whose design is specified in the previous chapter, is described. Section 6.1 presents the tools and the development environment adopted to implement the system on the target Android device and section 6.2 presents the Ethiopic handwriting IME.

6.1. Programming Tools and Development Environment

To implement the system to work on Android devices, a number of tools are required and used. Every application which runs in Android is written in Java programming language. The development environment to implement the system consists of installing Java Development Kit (JDK), Android Software Development Kit (SDK) which contains set of development tools to allow creating software applications, an Eclipse which is a standard, fully featured, free and easy to use Java Integrated Development Environment (IDE) for building applications using Java programming language.

Installation of an Eclipse Android Development Tools (ADT) plugin is also required to enable installation of application files into the Android emulator to test the functionality of an application in the target environment. ADT facilitates various functionalities, it extends the capabilities of Eclipse to quickly set up new Android projects, create an application UI, add components based on the Android Framework API, debug applications using the Android SDK tools and also export applications for distribution [34]. Figure 6.1 shows an Eclipse platform with a plugin embedded to it.

To run the developed java applications under the Eclipse environment in an Android emulator, an Android Virtual Device (AVD) also needs to be created. An AVD defines the system image and device settings used by the emulator. On its creation, it allows specifying a particular Android platform to run on the emulator and hardware options to use.

All the above mentioned tools are installed on a host machine that runs Windows OS to develop the application on it and debug and execute the developed application on the target environment using the emulator. A digitizer tablet with its stylus is used to simulate the pen paper environment for writing the input.

SQLite Database browser which is a SQLite database manager containing a Graphical User Interface (GUI) is used to view and edit the contents of the reference database on the host machine and then the database is pushed to the target environment using its Dalvik Debug Monitor Server (DDMS) perspective which is a debugging tool that provides various services.

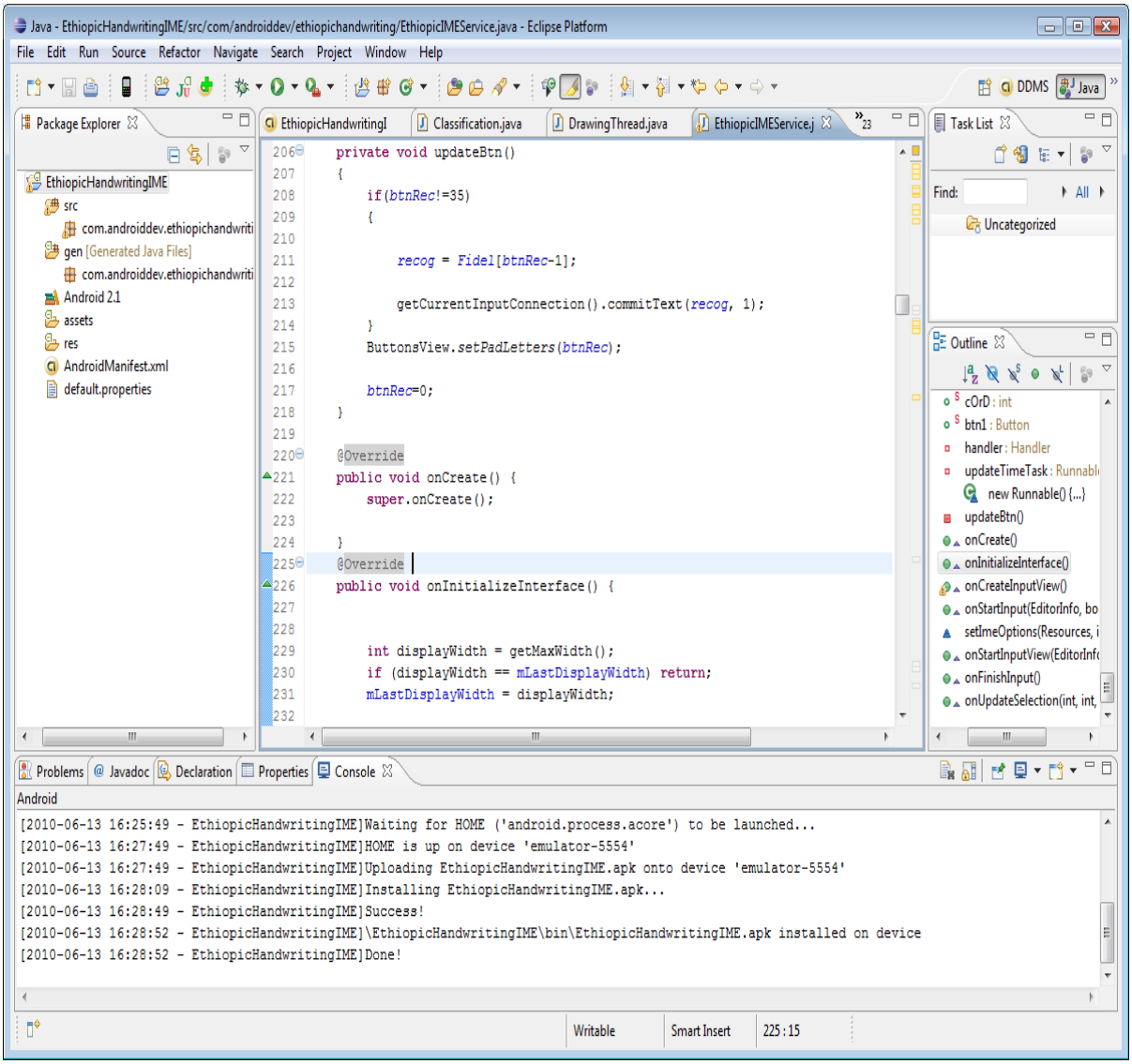


Figure 6.1 Interface of an Eclipse IDE with an ADT plugin

6.2. The Ethiopic Handwriting IME

Since this system is needed to serve as an IME that can be used from any application or service. In creating the IME, Android's *InputMethodService* class which is an API providing the basic implementation of input methods, is extended. This class provides much of the basic implementation for an input method, in terms of managing the state and visibility of the input method and communicating with the currently visible activity [35].

The interface of the system will reside in the bottom of the screen without hiding the appearance of the service or application which is using it for entry. It is designed to have two main components. The first component is the one containing a drawing area where a user can write Ethiopic characters in it. The second component is a layout containing a group of soft keys. The soft keys layout will be visible in the top of the drawing canvas whenever recognition for a basic character, numeral or punctuation is performed. The contents of these soft keys will be labels of non basic characters which are found in the same row as the recognized one. The soft key layout also contains a key to delete character which is found before the cursor's position for editing.

Whenever the system is deployed to the target device, Android will not load it like it does for any other user applications in its 'Home' directory rather it will be available under 'Settings' menu for selection. Thus, before using the Ethiopic Handwriting IME system, it has to be set so that Android can take it as a choice of input method. Android needs a confirmation whether the application is a trusted one or not before letting it to run as an IME. Thus a few steps have to be performed on the device's setting before using the Ethiopic Handwriting IME.

The following screen shot figures show the step by step appearance of the system when it is loaded and configured for the first time.

As shown in Figure 6.2 below, after executing the Ethiopic Handwriting IME application on Android, the configuration starts by opening the system Settings of the device.

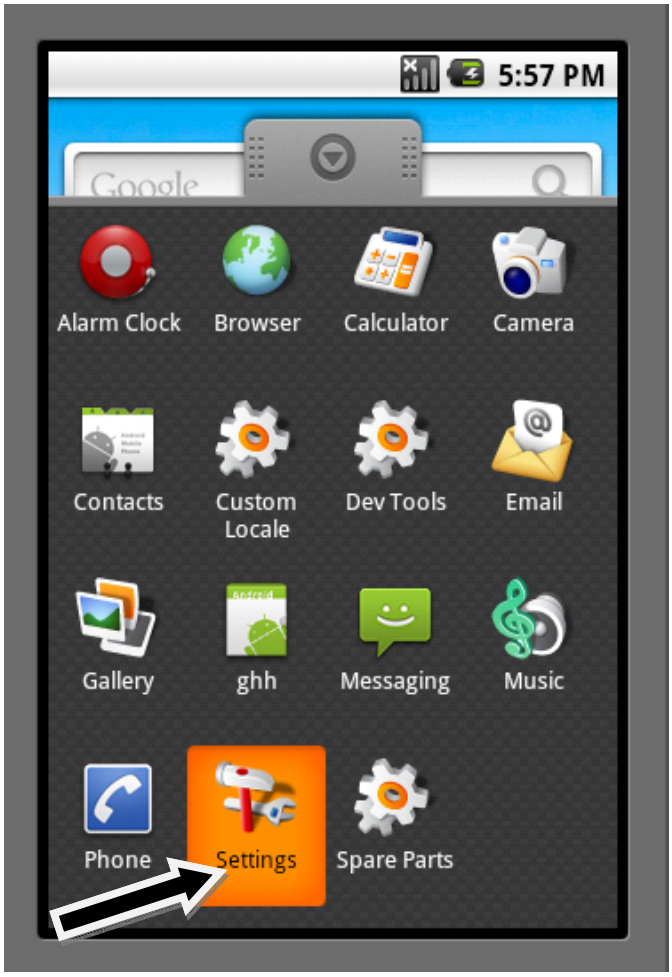


Figure 6.2 Home directory of Android

When the Settings icon is pressed, list of options will come to set. Select "Language & Keyboard" option as shown in Figure 6.3.

List of input methods which are integrated to the device will appear under the Language & Keyboard option. From there the Ethiopic Handwriting IME can be seen as one of the options that can be chosen in the list that contains the available built in input methods in the device as shown in Figure 6.4. The checkbox next to the "Ethiopic Handwriting IME", which is the name given to this system needs to be selected, to set it as an alternative IME on the device and get ready to be usable for Ethiopic input.

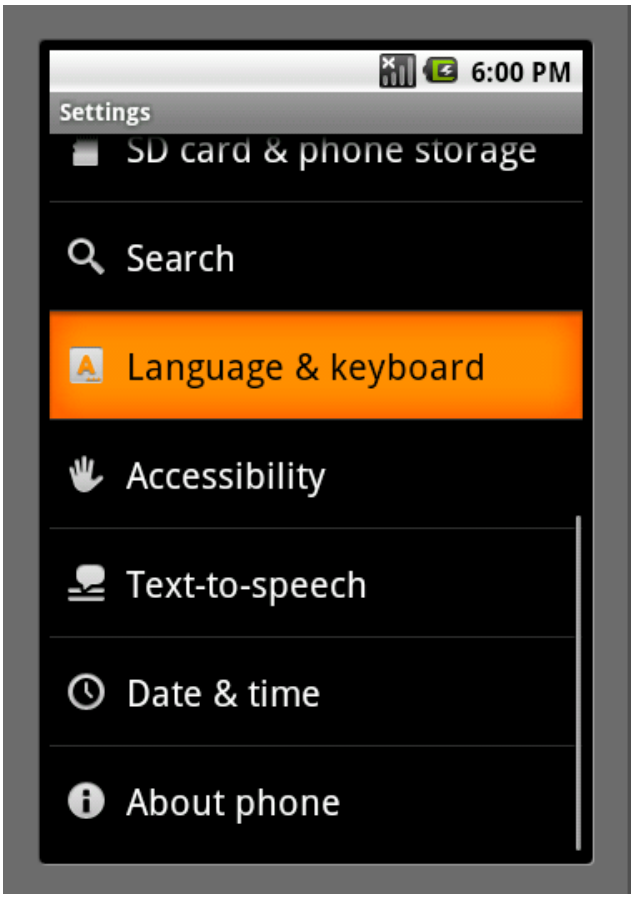


Figure 6.3 Android's Language & Keyboard settings

Figure 6.4 List of Available Input methods

When a security alert message pops up requesting a confirmation of the input method, press on the "Ok" button to apply the changes then the Ethiopic Handwriting IME will be selected in the list. Figures 6.5 and 6.6 depict these steps.

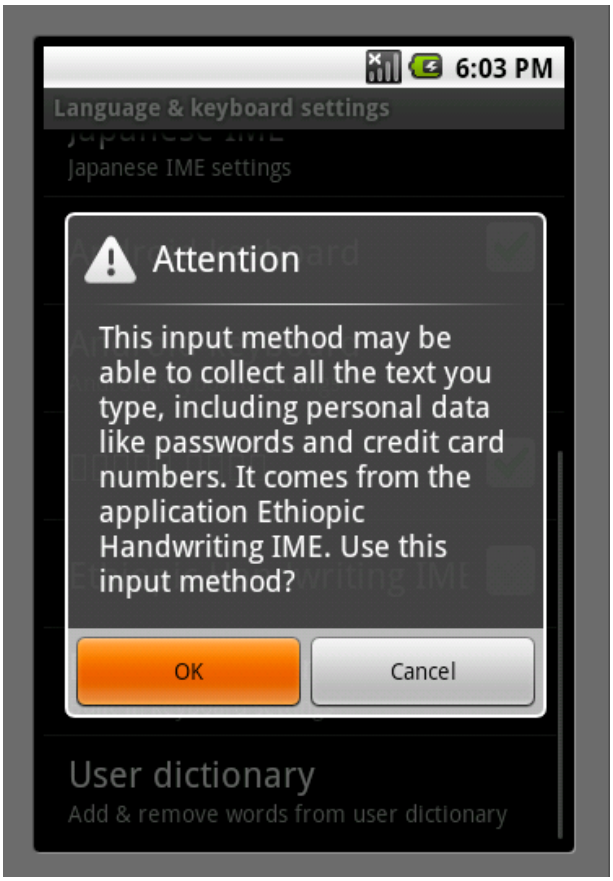


Figure 6.5 Confirmation prompt for an IME

Figure 6.6 Selected Ethiopic IME

Android's input method manager allows only one input method to be used for an application at a time. Thus to start using the Ethiopic Handwriting IME in applications, first it needs to be selected from the available input methods.

Open mail, contact or any other user or system application having a text entry interface on it from the Android device and then press and hold any of the edit boxes until the 'Edit text' menu pops up. Press 'Input method' from the 'Edit text' for input method selection. From the input method menu, select the "Ethiopic Handwriting IME" option to exactly specify to the target device that it has to use it as a primary input method. Figures from Figure 6.7 to Figure 6.10 consequently show the steps used to select an Ethiopic Handwriting input method from the built-in 'Messaging' application of the device.

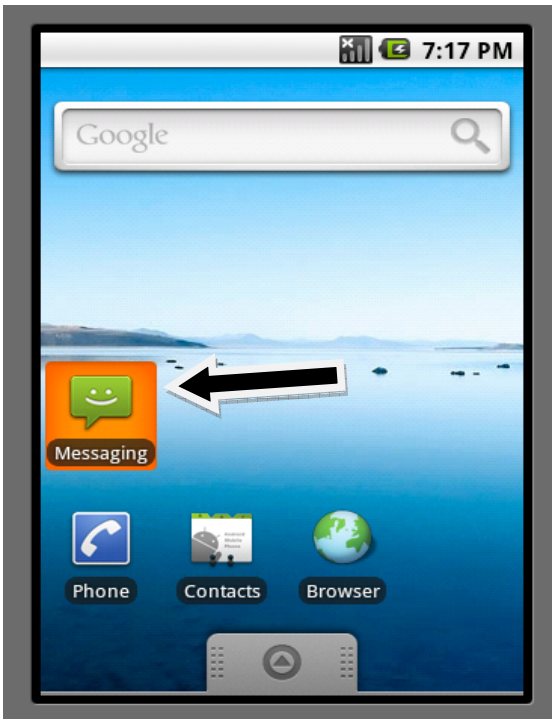


Figure 6.7 The Messaging Application

Figure 6.8 Message Composing Interface

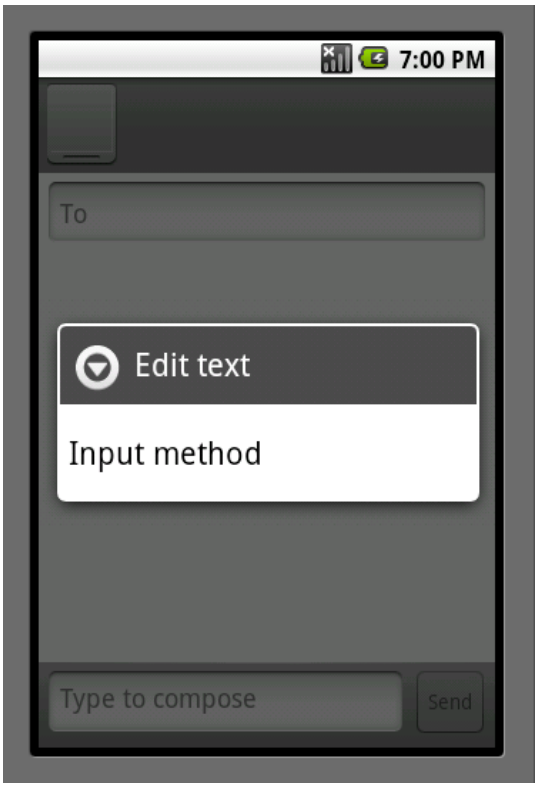


Figure 6.9 Edit text Menu

Figure 6.10 Selected Ethiopic Handwriting IME option

All the above steps need to be done only one time on the device. Once it is set, Android will save the settings. After that in each system as well as user applications that needs text or data to be entered, the selected input method will pop up when the focused Edit text is pressed.

This makes touching or pressing any of the edit boxes to result in the 'Ethiopic Handwriting IME' interface to show up in the bottom of the underlying application's interface on the screen. As shown in Figure 6.11, the bottom level rectangle is an area which is dedicated by the Ethiopic Handwriting IME to extract user's handwriting input.

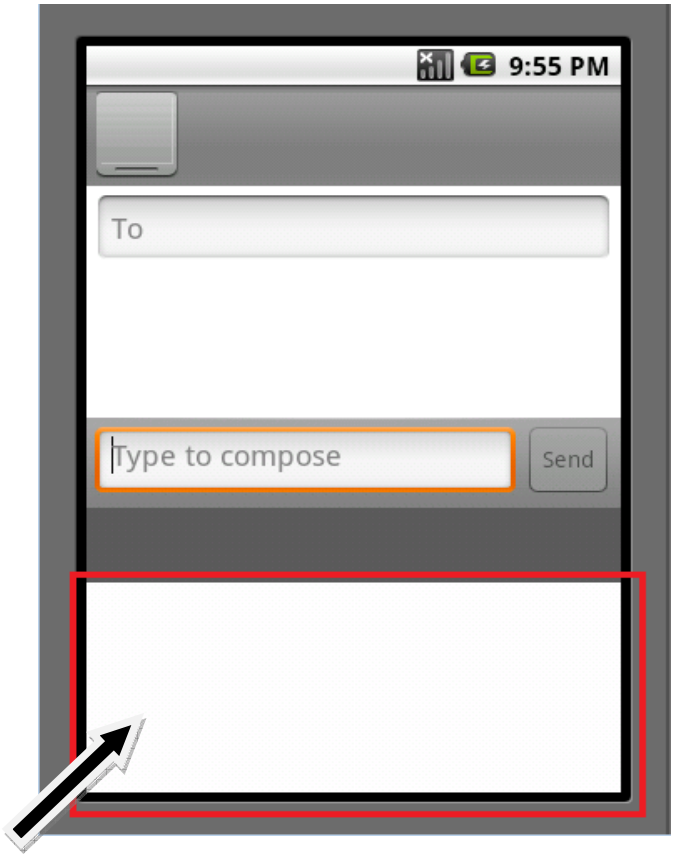


Figure 6.11 Ethiopic Handwriting IME's handwriting canvas

Handwritten input of basic Ethiopic characters can be made by writing on the dedicated writing area. As shown in Figure 6.11. The System draws on the panel visually whenever the user writes on it and then the recognition will start for the written input.

When the recognition completes, the recognized Ethiopic character will be displayed on the focused Edit text control. Figure 6.12 shows a handwritten character '**ፘ**' and 6.13 shows a printed Ethiopic character, '**ፘ**', displayed on the edit control as a result of the recognition operation.

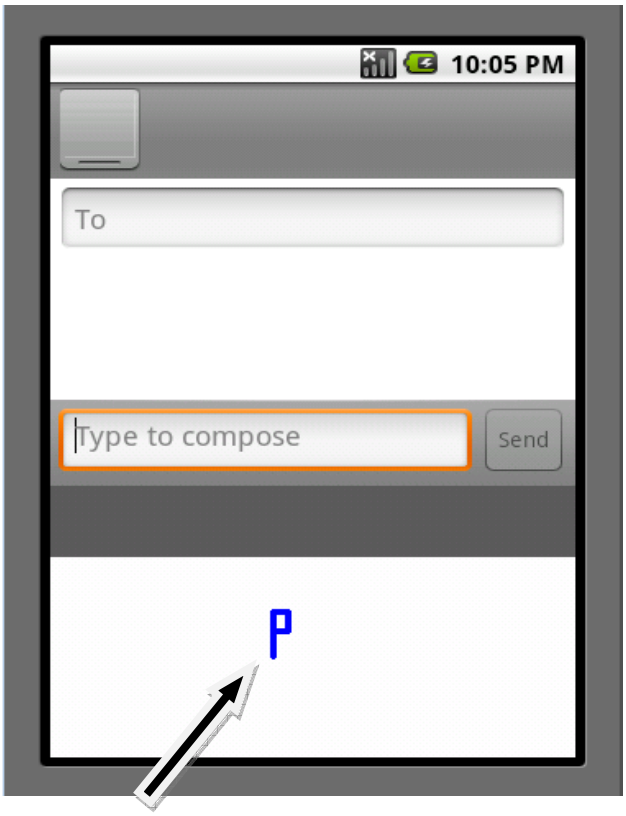


Figure 6.12 Handwriting of Basic character '**ፐ**'

Figure 6.13 Soft key panel for non-basic characters entry

As shown in Figure 6.13, once recognition for a basic character is made, the system clears the writing screen and generates soft key buttons as an option for the user to do editing to insert one of the non basic characters instead of the recognized one.

Figure 6.14 shows an editing made by using both basic and non-basic characters and space to construct a text. If user wants to insert one of the non basic characters, say 'ት', its basic representative, 'ተ', needs to be written first. Then the user can select 'ት' by pressing on the labeled soft key button.

Figure 6.14 Editing text

Punctuation symbols and numerals of the Ethiopic script can also be included in a text by writing their representative ':' and '፩' respectively and choose other symbols from the soft key candidates. Figure 6.15 shows a composed test message which is constructed using characters, punctuation mark, numeral and space.

Figure 6.15 Composed Text

Deletion on any of the characters can be committed by pressing the '**ሰርዝ**' soft key button provided on the screen setting an exact cursor position to delete.

CHAPTER SEVEN

EXPERIMENTAL RESULT

To test the usability and accuracy of the system an experiment was conducted. Ten subjects with age range of 13 to 35 having an educational background from primary to MSc level were selected. Four of the volunteered subjects were women and six of them men.

Digitizing tablet with a stylus is used for the data entry on to the system's drawing area in the Android Emulator which is configured in a host laptop computer that runs Windows Vista OS. The system is also tested by installing it on a real Android device of platform version 2.1.

The subjects were first given the designed template of characters to let them know how to input their handwritten entry of each character to the system following a specific order, direction as well as the predefined number of strokes to use to write each character. Then, once they confirm that they get the structure of the characters, they have been given a chance to practice by writing four times for each character which gives them to enhance their interaction to the system as well as getting use of the tablet. Their fifth time writing is collected and taken as an input to the experiment.

Table 7.1 summarizes the result of the experiment where it is extracted from the fifth trial entry of each character by each subjects. It shows the recognition accuracy for each character as a total of the ten subjects. From the experiment conducted, it is computed that an average of about 80 percent is obtained as a total accuracy rate.

Table 7.1 Accuracy of Ethiopic Handwriting IME system as rated by subjects for each character

Character	**Subjects**										**Total**
	1	2	3	4	5	6	7	8	9	10	Total Recognized (%)
ሀ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
ሰ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
ሐ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
መ	✓	X	✓	✓	✓	✓	✓	✓	✓	✓	90
ሠ	✓	✓	✓	X	✓	✓	✓	X	✓	✓	80
ረ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100

ሰ	X	✓	✓	X	✓	✓	✓	✓	✓	✓	80
ሸ	✓	✓	✓	✓	✓	✓	✓	✓	✓	X	90
ቀ	X	X	X	X	✓	✓	✓	✓	✓	X	50
በ	✓	X	X	✓	✓	✓	✓	X	✓	X	60
ተ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
ቸ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
ኀ	X	✓	✓	X	X	✓	✓	✓	✓	✓	70
ነ	✓	X	✓	X	✓	✓	✓	✓	X	✓	70
ኘ	✓	X	✓	✓	X	✓	✓	X	✓	✓	70
ኸ	X	✓	✓	✓	✓	X	X	✓	✓	X	60
ኰ	✓	✓	X	X	✓	✓	X	✓	✓	✓	70
ኾ	X	X	✓	✓	✓	✓	✓	✓	✓	X	70
ወ	✓	✓	✓	✓	X	✓	✓	✓	X	✓	80
ዐ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
ዘ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
ዟ	✓	✓	✓	✓	✓	✓	✓	✓	X	✓	90
ዩ	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	100
ደ	✓	X	✓	X	✓	✓	✓	✓	✓	✓	80
ጀ	X	✓	✓	✓	X	✓	✓	X	✓	✓	70
ገ	X	✓	X	X	✓	✓	✓	✓	✓	✓	70
ጠ	✓	✓	X	✓	✓	✓	✓	X	✓	✓	80
ጨ	✓	X	✓	X	✓	✓	✓	✓	X	✓	70
ጸ	✓	✓	X	✓	✓	✓	X	✓	X	✓	70
ጸ	X	✓	✓	✓	X	✓	✓	✓	✓	✓	80
ፀ	✓	✓	✓	X	✓	X	✓	✓	✓	✓	80
ፈ	✓	✓	✓	✓	X	✓	✓	✓	✓	✓	90
ፐ	✓	✓	✓	✓	✓	✓	✓	✓	✓	X	90
ቨ	✓	✓	✓	✓	✓	✓	X	✓	✓	✓	90

Almost all of the subjects responded that the assigned symbols are not very far from their way of writing for most of the characters. Moreover, the respondents have given an affirmative reply on the convenience the system's interface and assured the system's usability with its ease of to use.

From the experimental result, it can be concluded that, if users have been given more chance to practice the symbols and input more precise handwriting entries, the system's accuracy could be improved.

CHAPTER EIGHT

CONCLUSION and RECOMMENDATION

Mobile computing is one of the main technological advancements that this technological era provided focusing on anytime and anywhere access to information of any format incorporating several functionalities in small and powerful handheld devices.

Smartphones are getting a higher acceptance due to their ability to give the functionalities of what any typical cell phone can provide, an access to information like e-mail and Internet using wireless networks and also the ability to view and edit documents of different file format that has been used to be possible only through PCs.

Android is a new, real open and comprehensive platform. Its usability is rapidly growing due to the fact that it can span in any smart phone devices creating a fully open environment for user applications to be integrated on it.

Text entry techniques are main concerns in mobile devices as they are the main interaction ways of user-machine communication. Considering the aim to achieve the goal of being small in size for handheld devices, using a hardware keyboard as a text entry technique is not the best option.

The size constraint that restricts these devices to be small, initiates to look for an alternative option as text entry technique rather than adapting a hardware keyboard like in desktops. Thus, most Smartphones are nowadays being released supporting a touch screen that can enable to perform on screen commands integrating soft/virtual keyboard as well as handwriting as input methods. Handwriting recognition is the ability of a machine to recognize handwritten or printed text. It is getting a significant attention on handheld devices being a more convenient input method as it resembles the natural writing in pen-paper environment.

In this work, constrained, writer independent Ethiopic handwriting recognition is integrated to work on Android based Smartphones serving as an input method to support the Ethiopic script which has millions of speakers on each services of the device.

The recognition engine is made to support handwriting entries of the basic characters, numerals and punctuation marks of the script providing the ability to use the non basic

characters and numerals by selecting candidate soft key options that are generated when their respective basic character or numeral gets recognized by the system.

A model that best resembles the original script is designed for basic Ethiopic script characters to improve the performance of the recognition engine without setting difficulties for users to learn the new script.

Directional features of characters and their respective stroke information are persistently stored based on the modeled design so that they can be efficiently used as references whenever the recognition operation is performed for an unknown input pattern.

The work enabled the accessibility and usage of the Ethiopic characters in any applications of the device where a text entry is required. Thus, it contributed in providing a localized application that can be used by the script's users to compose messages and e-mails, save and edit contacts as well as work on saving or editing different files and search for them using their own language.

The experiment conducted has shown that, the work has encouraging recognition accuracy with good usability, getting the feedback from different subjects

Recommendation and Future work

The following features are identified as future works of this project.

- Incorporating a candidate view, where the input method will have a dedicated interface to perform text generation which suggests a list of candidate Ethiopic words from a dictionary of Ethiopic words stored in the device's library and then to run possible combination of letters that the user most likely intended to write. Thus when each time a user writes a character on any service or application, the system will let her/him to commit a word from one of the interpretations without a need to write the whole letters maximizing the interaction process.
- Providing a word level recognition can also be done rather than considering character at a time.
- Refining code and making character symbols of the reference template precise could also be made by using handwriting corpus of the Ethiopic script to improve the accuracy of the recognition system having a well known structure of characters with specified width and length.

References

[1] Abnet Shimeles, "Online Handwriting Recognition for Ethiopic Characters", Master's Thesis, Addis Ababa University, Faculty of Informatics, Department of Computer Science, June 2005.

[2] Abera Abebaw, "Implementation of Online Handwriting Recognition System for Ethiopic Character Set" Masters Project, Addis Ababa University, Faculty of Informatics, Department of Computer Science, May 2007.

[3] Daniel Kefale, "Modeling Multi-Script Text Editing based on Online Handwriting Recognition", Masters Thesis, Addis Ababa University, Faculty of Informatics, Department of Computer Science, June 2008.

[4] Derek Schauland, "What is the Android Operating System?", http://www.wisegeek.com/what-is-the-android-operating-system.html, last accessed February 2010

[5] Hiroshi Tanka, Naomi Iwayama and Katsuhiko Akiyama, "Online Handwriting Recognition Technology and Its Applications", FUJITSU Sci. Tech J. 40,1, pp. 170-178, June 2004.

[6] "Microsoft Word, Excel, PowerPoint and Adobe PDF files on your Android-powered Smartphone", http://www.dataviz.com/products/documentstogo/android, last accessed February 06, 2010

[7] " What is a SmartPhone ? " , http://www.silicon.com/technology/mobile/2006/02/13/analysis-what-is-a-smart-phone-39156391/ last accessed February 07, 2010

[8] Santosh K. C., Cholwich Nattee, "A comprehensive survey on On-line Handwriting Recognition Technology and its Real Application to the Nepalese Natural Handwriting", Kathmandu University Journal of Science, Engineering and Technology, Vol 5. No 1, January, 2009.

[9] Ashutosh Malaviya, Christoph Leja and Liliane Peters, "Multi-Script Handwriting Recognition with FOHDEL" German National Research Center for Information Technology(GMD), Published in the Proceedings of the Biennial Conference of North American Information Processing Society, Berkeley, IEEE, pp. 147-151, 1996.

[10] Daniel Nigussie, "Writer Independent Online Handwriting Recognition for Ethiopic Characters", Masters Thesis, Addis Ababa University, Faculty of Informatics, Department of Computer Science, June 2006.

[11] Yonas Hailu, "Ethiopic online Handwriting Recognition System Using Simplified Ethiopic Script", Masters Thesis, Addis Ababa University, Department of Computer Science, Addis Ababa, Ethiopia, July 2007.

[12] Yaregal Assabie and Josef Bigun, "Online Handwriting Recognition of Ethiopic Script", In Eleventh International Conference on Frontiers in Handwriting Recognition, pp. 153-158, Concordia University, Canada, August 2008.

[13] Fikru Temtem,"Online Ethiopic Handwriting Recognition using Support Vector Machine", Master's Thesis, College of Ethiopian Telecommunications and Information Technology, Department of Information Technology, Addis Ababa, Ethiopia, October 2006.

[14] Claus Bahlmann and Hans Burkhardt, "The Writer Independent Online Handwriting Recognition System frog on hand and Cluster Generative Statistical Dynamic Time Warping", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 26 No 3 , March 2004.

[15] Rejean Plamondon and Sargur N. Srihari, "Onl-Line and Off-Line Handwriting Recognition: A Comprehensive Survey", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 22, No 1, January 2000

[16] E. Anquetil and H. Bouchereau, "Integration of an On-line Handwriting Recognition System in a Smart Phone Device", Proceedings of the 16th IAPR International Conference on Pattern Recognition (ICPR 2002), pp.192.195

[17] Dr. Dawit Bekele, "Report on the Workshop for Ethiopic Standardization", March 2003 Addis Ababa, available at http://www.uneca.org/aisi/docs/ethiopic.doc

[18] Handwriting Recognition, http://en.wikipedia.org/wiki/Handwriting_recognition, last accessed May 2010.

[19] Claus Bahlmann, "Advanced Sequence Classification Techniques applied to Online Handwriting Recognition", PhD thesis, Albert-Ludwigs-Universita't Freiburg , Institut fu'r Informatik, October 2004.

[20] Vuokko Vuori, Matti Aksela, Ram_unas Girdziu_sas, Jorma Laaksonen, Erkki Oja, 'On-line recognition of handwritten characters", Available at http://www.cis.hut.fi/research/reports/biennia100-01/biennial-08.pdf, last accessed June 2010.

[21] S. Jaeger, S. Manke, J. Reichert, A. Waibel, "Online Handwriting Recognition: the NPen++ recognizer", International Journal on Document Analysis and Recognition(IJDAR), Vol 3 PP. 169-180, 2001

[22] Gareth Loudon, Olle Pellijeff, Lizhong-Wei, "A Method for Handwriting Input and Correction on Smartphones", Cyberlab Singapore, Ericson Research, Singapore, Proceedings of the Seventh International Workshop on Frontiers in Handwriting Recognition, Amsterdam, September 2000.

[23] Benjamin Speckmann, "The Android mobile platform", Master's Thesis, Eastern Michigan University Department of Computer Science, April 2008.

[24] Kathleen J. Price and Andrew Sears, "Speech –based Text Entry for Mobile Devices", Interactive Systems Research Center Information Systems Department, Baltimore USA.

[25] I. Scott Mackenzie and Kumiko Tanaka-Ishii, "Text Entry Systems: Mobility, Accessibility, Universality", Morgan Kaufmann Series in Interactive Technologies, Morgan Kaufmann Publishers Inc., San Francisco, CA, 2007.

[26] Charles C.Tappert, Ching Y.Suen, Toru Wakahara, "The State of the Art in On-Line Handwriting Recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No.8, pp. 787-808, August 1990.

[27] Bernd Brugge and Allen H.Dutoit, "Object-Oriented Software Engineering: Conquering Complex and Changing Systems", Carnegie Mellon University School of Computer Science, Pittsburgh, USA, Prentice Hall, 1999.

[28] Android Developers, Input Method Manager, http://developer.android,com/reference/android/view/inputmethod/InputMethodManager.html, Last accessed June 2010.

[29] Ancient Scripts, "Ethiopic", http://www.ancientscripts.com/ethiopic.html, Last accessed June 2010.

[30] Reto Meier, "Professional Android Application Development", Wiley Publishing Inc, Indianapolis, Indiana, 2009.

[31] Jianying Hu, Sok Gek Lim, Michael K. Brown, "Writer Independent on-line handwriting recognition using an HMM approach", Pattern Recognition, Vol 33, No 1, pp.133-147,2000

[32] Chris Davis, "MyScript handwriting recognition for Android", http://androidcommunity.com/myscript-handwriting-recognition-for-android-video-20090611/ , Last accessed May 2010.

[33] Inkmark software LLC, "Handwriting Recognition", http://www.inkmarksoftware.com/, Last accessed May 2010.

[34] Android Developers, "ADT plugin for Eclipse", http://developer.android.com/sdk/eclipse-adt.html, last accessed June 2010.

[35] Android core, "Creating an Input method", http://androidcore.com/android-programming-tutorials/357-creating-an-input-method.html, last accessed June 2010.

[36] "Ge'ez Script: Unicode" , available at http://www.museumstuff.com/learn/topics/Ge'ez_script sub Unicode.htm, last accessed June 2010

Appendix A: Ethiopic Character set and its Unicode representation [36]

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1200	ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ		ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ	ሏ
1210	ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ	ሗ	መ	ሙ	ሚ	ማ	ሜ	ም	ሞ	ሟ
1220	ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ	ሧ	ረ	ሩ	ሪ	ራ	ሬ	ር	ሮ	ሯ
1230	ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ	ሷ	ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ	ሿ
1240	ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ	ቇ	ቈ		ቊ	ቋ	ቌ	ቍ		
1250	ቐ	ቑ	ቒ	ቓ	ቔ	ቕ	ቖ		ቘ		ቚ	ቛ	ቜ	ቝ		
1260	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ	ቧ	ቨ	ቩ	ቪ	ቫ	ቬ	ቭ	ቮ	ቯ
1270	ተ	ቱ	ቲ	ታ	ቴ	ት	ቶ	ቷ	ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ	ቿ
1280	ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ	ኇ	ኈ		ኊ	ኋ	ኌ	ኍ		
1290	ነ	ኑ	ኒ	ና	ኔ	ን	ኖ	ኗ	ኘ	ኙ	ኚ	ኛ	ኜ	ኝ	ኞ	ኟ
12A0	አ	ኡ	ኢ	ኣ	ኤ	እ	ኦ	ኧ	ከ	ኩ	ኪ	ካ	ኬ	ክ	ኮ	ኯ
12B0	ኰ		ኲ	ኳ	ኴ	ኵ			ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ	
12C0	ዀ		ዂ	ዃ	ዄ	ዅ			ወ	ዉ	ዊ	ዋ	ዌ	ው	ዎ	
12D0	ዐ	ዑ	ዒ	ዓ	ዔ	ዕ	ዖ		ዘ	ዙ	ዚ	ዛ	ዜ	ዝ	ዞ	ዟ
12E0	ዠ	ዡ	ዢ	ዣ	ዤ	ዥ	ዦ	ዧ	የ	ዩ	ዪ	ያ	ዬ	ይ	ዮ	
12F0	ደ	ዱ	ዲ	ዳ	ዴ	ድ	ዶ	ዷ	ጀ	ጁ	ጂ	ጃ	ጄ	ጅ	ጆ	ጇ
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1300	ገ	ጉ	ጊ	ጋ	ጌ	ግ	ጎ	ጏ	ጐ	጑	ጒ	ጓ	ጔ	ጕ		
1310	ጘ		ጚ	ጛ	ጜ	ጝ			ጟ	ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
1320	ጧ	ጨ	ጩ	ጪ	ጫ	ጬ	ጭ	ጮ	ጯ	ጰ	ጱ	ጲ	ጳ	ጴ	ጵ	ጶ
1330	ጷ	ጸ	ጹ	ጺ	ጻ	ጼ	ጽ	ጾ	ጿ	ፀ	ፁ	ፂ	ፃ	ፄ	ፅ	ፆ
1340	ፇ	ፈ	ፉ	ፊ	ፋ	ፌ	ፍ	ፎ	ፏ	ፐ	ፑ	ፒ	ፓ	ፔ	ፕ	ፖ
1350	ፗ	ፘ	ፙ	ፚ	፛	፜	፝	፞	፟	፠	፡	።	፣	፤	፥	፦
1360	፧	፨	፩	፪	፫	፬	፭	፮	፯	፰	፱	፲	፳	፴	፵	፶
1370	፷	፸	፹	፺	፻	፼	፽	፾	፿	ᎀ	ᎁ	ᎂ	ᎃ	ᎄ	ᎅ	
1380	ᎆ	ᎇ	ᎈ	ᎉ	ᎊ	ᎋ	ᎌ	ᎍ	ᎎ	ᎏ	᎐	᎑	᎒	᎓	᎔	᎕
1390	᎖	᎗	᎘	᎙	᎚	᎛	᎜	᎝	᎞	᎟						
2D80	ⶀ	ⶁ	ⶂ	ⶃ	ⶄ	ⶅ	ⶆ	ⶇ	ⶈ	ⶉ	ⶊ	ⶋ	ⶌ	ⶍ	ⶎ	ⶏ
2D90	ⶐ	ⶑ	ⶒ	ⶓ	ⶔ	ⶕ	ⶖ									
2DA0	ⶠ	ⶡ	ⶢ	ⶣ	ⶤ	ⶥ	ⶦ	⶧	ⶨ	ⶩ	ⶪ	ⶫ	ⶬ	ⶭ	ⶮ	⶯
2DB0	ⶰ	ⶱ	ⶲ	ⶳ	ⶴ	ⶵ	ⶶ	⶷	ⶸ	ⶹ	ⶺ	ⶻ	ⶼ	ⶽ	ⶾ	⶿
2DC0	ⷀ	ⷁ	ⷂ	ⷃ	ⷄ	ⷅ	ⷆ	⷇	ⷈ	ⷉ	ⷊ	ⷋ	ⷌ	ⷍ	ⷎ	⷏
2DD0	ⷐ	ⷑ	ⷒ	ⷓ	ⷔ	ⷕ	ⷖ	⷗	ⷘ	ⷙ	ⷚ	ⷛ	ⷜ	ⷝ	ⷞ	⷟
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Declaration

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

_____

Fetiya Beshir

This project has been submitted for examination with my approval as an advisor

_____

Solomon Atnafu (PhD)

Addis Ababa, Ethiopia

June 30, 2010