ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
SCHOOL OF INFORMATION STUDIES FOR AFRICA

Text-to-Speech system for *Afaan Oromoo*

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR
THE DEGREE OF MASTER OF SCIENCE IN INFORMATION SCIENCE

BY
Morka Mekonnen

JUNE 2001

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION STUDIES FOR AFRICA

Text-to-Speech System for *Afaan Oromoo*

BY

Morka Mekonnen

Name and Signature of Member of the Examining Board

Ato Getachew Jemaneh, Chairman, Examining Board

Ato Workeshet Lamenew, Advisor

Ato Wakshum Mekonnen, Advisor

Dr. Lars Asker, External Examiner

DEDICATION

Dedicated to my family
Mom, Seble, Melka and Simon, I love you all.

ACKNOWLEDGMENT

I am deeply grateful to my advisors Ato Workeshet Lamenew and Ato Wakshum Mekonnen for their constructive suggestions, review and concern.

My special thanks goes to my relatives Melka Mekonnen and Romanwork Estiphanos who supported me financially during the period of this work. I want also to mention my greatest gratitude to Ato Kebede Hordofa for his co-operation in locating and selecting relevant materials on *Afaan Oromoo* Phonology

I am also indebted to Ato Seuliman Ahmed who helped me print the draft copy of this thesis. Finally I would like to thank W/ro Sara Estiphanos for printing the final draft of this work. I would also to thank my relatives and friends for the encouragement they gave me while I was conducting the research.

Table of Contents

List of Tables

List of Figures

List of Appendices

Abstract

A natural way of communication between humans is through speech. On the contrary human-machine communication has been limited to keying in instructions and receiving answers through text forms. This limitation of human-machine communication is now being solved by the development of Dialogue systems. These systems enable human-computer communication via speech. Besides having natural way of communication.

One component of a Dialogue system is a Text-to-speech synthesizer (TTS). It reads texts aloud. This component gives opportunities to handicapped people to have access to electronic documents. Moreover it can be used for proofreading documents, language education etc.

In this study an attempt is made to address the issue of having textual information in speech forms for the language of *Afaan Oromoo*. In doing so a diphone based text to speech system for *Afaan Oromoo* sample words is presented. The present prototype system consists of two main parts. These are

- an automatic phonetic transcription of the input word and
- a speech synthesis module which synthesizes an utterance by looping (concatenating) the sound equivalents of the phonetic transcription

To test the algorithm, samples of words are selected. The selection was based on previous study result that showed the most frequent words in some *Afaan Oromoo* texts.

Diphones, speech units that cover two sounds and the transition between them, form the basis of the synthesis module. In transcribing the orthography (the writing system) into phonetic units, the *Afaan Oromoo* writing system is found to be well governed to rules. This enabled the transcription to be accurate. On the contrary, success on recognizing the utterance of the transcribed phonetic unit was only limited to 43.33 % for naive listeners and to 83.33% for listeners who heard the utterance at least three times in different days.

Error rates of 37 % can be found in some English Text-to-Speech system. The result of this work is therefore encouraging and open to reaching higher rates of intelligibility with some improvements. Incorporating spectral smoothing techniques that smooth the transition points of diphones can make improvements. Moreover, it is felt that having sound laboratory to record the corpus data for such kinds of work is mandatory.

CHAPTER ONE

INTRODUCTION

1.1 Background to the Study

Computers are fast becoming an ever-present part of our lives, brought on by their rapid increase in performance and decrease in cost. With their increased availability comes our corresponding appetite for information. This phenomenon has led for vast amounts of useful information to be made widely available. As a result people are utilizing the information they get for education, decision-making, etc.

The advent of the information age places increasing demand on technologies to provide universal access. For information to be truly accessible, especially to the technology naive, anytime, anywhere, one must seriously address the problem of user interfaces. A promising solution to this problem is to impart human like capabilities onto machines, so that they can speak and hear, just like the users with whom they need to interact.

One such system is a spoken dialogue system. It is an interactive system (through speech) that operates in a constrained domain. It can range from the fairly basic, in which the system takes the user through a sequence of pre-designed steps (e.g. press, or say three), to those involving complex communication between the system, the user and the underlying application (e.g., How may I help you?). On the contrary conversational systems aim to simulate human conversation by permitting unrestricted interaction between a computer and a human user (Glass, 1999). These systems aim to support spontaneous dialogue. That is dialogue with interruptions, confirmation, clarification, and sentence fragments. But success on this area has

been very limited (Zue, 1997). Nevertheless, the conversational as well as the spoken dialogue technology are important developments that enable casual and naive users to interact with complex computer applications in a natural way using speech.

Although current interactive voice response (IVR), the first kind of a spoken dialogue system, limit users in what they can say, they give a chance to reduce the error rate of speech enabled information retrieval systems that do not use thesaurus for retrieval process. For example, users of speech-based computer systems who do not know exactly what information they require and how to obtain it can be guided by such systems to determine their precise requirements. For this reason it is essential that speech-based computer systems be able to engage in a dialogue with users rather than simply respond to predetermined spoken commands (McTear, n.d).

Spoken dialogue technology has evolved as a method for enabling such human-computer dialogues which take into account two sources of communication difficulty: problems associated with determining what the speaker said due to speech recognition and language understanding errors, and problems in determining exactly what the speaker wants to know or to hear. Indeterminacy in the input can be partially prevented through careful dialogue engineering so that the user's input is restricted to what the system can understand. Limiting the user's input to a constrained set of single words is one way of enhancing understanding, though at the cost of naturalness.

Such a spoken dialogue system involves the integration of a number of components that typically provide the following functionalities (McTear, n.d)

- *speech recognition* - the conversion of an input speech utterance, consisting of a sequence of acoustic-phonetic parameters, into a string of words;

- *language understanding* - the analysis of string of words with the aim of producing a meaning representation for the recognized utterance that can be used by the dialog management component;

- *Dialog management*- the control of the interaction between the system and the user, including the co-ordination of the other components of the system;

- *communication with external system* - for example, with a database system, expert system, or other computer application;

- *response generation* - the specification of the message to be output by the system;

- *speech output* - the use of text-to-speech synthesis or pre-recorded speech to output the system's message.

These components, by interacting with each other, can retrieve the requested information from the external source, and construct the message that is to be sent to the speech output component to be spoken to the user, through the response generator.

Synthetic speech is required when the text is variable or when large amounts of information have to be processed and spoken out. In these cases systems that are able to read aloud the information retrieved are used. But when the text to be read aloud is limited to few words (like time notification systems) the possible words to be read can be digitized and stored for future retrieval of the required sound forms of the words.

There is an extensive literature on the nature of human-human dialogue and communicative interaction that could potentially support the design of spoken dialogue systems. Dialogue, as a multidisciplinary topic, has been studied by linguists, psychologists, philosophers, sociologists, and computer scientists. Some of these studies have provided a basis for spoken

Such one system is a Text-To-Speech synthesizer (here after referred to TTS). It is a system, which synthesizes speech from text using small speech units and extensive linguistic processing (Ince, 1992). Early attempts of this technology assembled clauses by concatenating recorded words. But this approach suffers from the disadvantage that words, not in a database, become unspeakable. Instead, a popular technique today is to store the actual speech segments that contain parts of phonemes (Hallahan, 1996).

A TTS system can be divided into two major subsystems. These are the Natural Language Processing (NLP) component and the Digital Speech Processing (DSP) component. The first subsystem (NLP) is primarily a natural language processing module. It involves the conversion of the input text into a linguistic representation that includes both phonetic and prosodic information. In this subsystem the input text is converted to strings of phonemes by rule or by dictionary lookup (Morton, 1987). The units can then be used to specify what sounds need to be produced while prosodic parameters, judged from the context, are used to specify how they are to be produced. The second subsystem, the DSP, generates speech waveforms as output using as input the linguistic information generated by the natural language processing (NLP) subsystem.

1.2 Statement of the Problem and Justification of the Study

The problems that lead to the requirement of TTS systems are generally universal. Some of these needs, as stated by Dutoit (1996), are:

- Aid to handicapped persons: Visually impaired people for instance, because of their inability to see, need TTS systems to get access to electronic documents. Besides, to get access to printed documents, these people need a combination of optical character reader (OCR) that converts printed texts into electronic form and TTS system that speak the text

aloud. Similarly voice handicaps need such systems because synthetic speech produced in few seconds by TTS systems can be used to remedy their impediments.

- Multimedia, Human-machine communication: The development of high quality TTS systems is a necessary step (as is the enhancement of speech recognizers) towards more complete means of communication between humans and computers.

Besides solving the above problems, a TTS can be used for the following applications (Dutoit, 1996).

- Information retrieval: For queries posed, through a telephone and the user's voice with the help of a speech recognition system, TTS systems are needed to make the textual information available over the telephone. Texts that range from simple messages to huge databases that can hardly be read and stored as digitized speech can be retrieved using a TTS.

- Language education: High Quality TTS synthesis can be coupled with a Computer Aided Learning system, and provide tools to help a child or a student learn correct pronunciation of words.

- Better communication: As Hsia quoted by Mary Whiteside and J. Alan Whiteside (1997) indicate, redundancy in communication is the key to better communication. Redundancy in this context is the transmission of the same or closely related information to the receiver or learner through two sensory channels (usually aural and visual channels). Therefore TTS systems give chance to making errors in communication become minimized.

In view of its application, it is clear that TTS systems must be developed for every language that has a need for it. It is with this understanding that Laine (1998) tried to develop a TTS for

the Amharic language. The conductor of this research, on the other hand, has selected *Afaan Oromoo* as a research topic for a TTS system problem, for the following reasons.

- *Afaan Oromoo* is widely spoken in Ethiopia and in the neighboring countries. For example the 1994 census report of the Central Statistics Office, although it did not include major areas of the east Harerge region, estimated the number of Oromos in Ethiopia to be 18,732,525 (about 32% of the total population of Ethiopia).

- The instruction medium for Primary and Junior Secondary Schools in *Oromyiaa* Regional State is *Afaan Oromoo*. Textbooks as well as reference materials are available in *Afaan Oromoo* for use in the schools. It is also the official language of the region.

- The development of word processing application for *Afaan Oromoo* has become mature. In recent years applications like Oromifaa2000, *Barissa, and Oromoo* thesaurus, are some of the developments that are applied to make the language cope up with the progress in computer applications (Oromosoft homepage). In order to get access to documents prepared using these applications, the visually impaired people need the integration of TTS systems with the applications.

- Proof reading of documents: Lots of documents are prepared by the offices the *Oromiyaa* region. For proof reading these documents, people have to fix their eyes on the document. In an eye's busy environment, and time limitations, the document can be proof read by an integration of TTS systems.

- A TTS system for other languages in the Cushitic family can be developed with less effort

1.3 Objectives

1.3.1 General Objective

The main objective of this research is to develop a TTS system for *Afaan Oromoo* words.

1.3.2 Specific objectives

The specific objectives of the research are to:

- review related literature on phonology of *Afaan Oromoo*;
- compile a list of *Afaan Oromoo* phonemes;
- review the various natural processing component and the digital speech processing techniques used in TTS;
- develop an algorithms for building a prototype TTS for *Afaan Oromoo*.
- test the system on how it performs for selected sample words.

1.4 Methods

1.4.1 Review of Related Literature

Various literatures on the subject and on the language were consulted. Other source of the literature reviewed includes soft copies in CD-ROM and materials on the Internet web pages. Faculty and other appropriate individuals were consulted on the phonology of *Afaan Oromoo.* Moreover articles in journals, books, thesis and other related sources in hard copies were also reviewed.

1.4.2 Program Development Tool

To develop a TTS prototype for *Afaan Oromoo* an algorithms was developed. Visual C++ programming language was used to change the algorithm into implementation. The programming language is chosen for the reasons of:

- It's simplicity to develop interfaces
- It's features for handling string arrays and
- the familiarity of the researcher with the programming language

1.4.3 Testing Techniques.

The most commonly used isolated word tests are the Diagnostic Rhyme Test (DRT), and the Modified Rhyme Test (MRT). In both cases the listener hears a sequence of isolated words, and for each word, must select the word heard from a number of rhyming alternatives given on an answer sheet. Both tests focus upon consonants, because consonants have been generally been found to be more difficult to synthesize than vowels (Klatt 1987). The answer sheet for the MRT lists six responses for each word, each of which differ in an initial or final consonant. The answer sheet for the DRT lists only two options for each word, which differ only in one distinctive feature in an initial consonant. In both cases the words used are monosyllabic, of CVC or CV structures and contain only singleton consonants, not consonant clusters.

The Modified Rhyme Test is sometimes modified to have an Open Rhyme Test. In this test the listener hears words, but is not given choices to identify what he heard (Donovan, 1996).

In *Afaan Oromoo,* although there are monosyllabic words, it is difficult to find such words

that rhyme similarly. Therefore instead of using the Modified Rhyme Test or Diagnostic Rhyme Test, an Open Rhyme Test was used. In this test a person is made to listen to an utterance and is asked to say what he has heard.

1.5 Scope and Limitation of the Study

The TTS algorithm developed for *Afaan Oromoo* text is limited to synthesize normalized words. The prototype TTS system does not read aloud sentences. The development of such systems requires an in depth investigation of prosody rules of the language. In addition the prototype TTS is limited to synthesize only sample words selected and a few more words for which all the diphones exist.

1.6 Organization of the Thesis

The thesis is divided into five chapters. The first chapter comprises background of the study that introduces conversational interfaces and applications in information retrieval system and the rationale for developing the TTS algorithm. The chapter contains statement of the problem, justification, methodology, scope and limitation of the study.

The second chapter is review of speech synthesis systems and the different techniques, methods used to develop synthesized speech. Chapter three entirely deals with *Afaan Oromoo* language in general and its phonology in particular. Different rules of consonants, vowels, consonant clusters, assimilation is discussed in this chapter.

The algorithm developed and the experiment conducted is discussed in chapter four. Conclusions and recommendations constitute the last chapter of the report, chapter five. References and appendixes are appended at the end.

CHAPTER TWO
SPEECH SYNTHESIS

2.1 Introduction

Speech can be synthesized naturally or artificially. A natural speech synthesis is what humans do while communicating. Artificially speech can be synthesized in a number of ways (articulatory, formant, concatenative). Though it is with different degrees, studying the human speech production system helps these approaches to reach to a good model of the human speech production system. In this chapter the three types and the human speech production system are discussed.

2.2 The Human Speech Production System

The speech signal produced by humans is an acoustic sound pressure wave that is transmitted by means of sound generated by the movement of certain of the bodily organs which make up the speech production system (O'Connor, 1973). The figure below shows the main anatomical structures of the human speech production system.

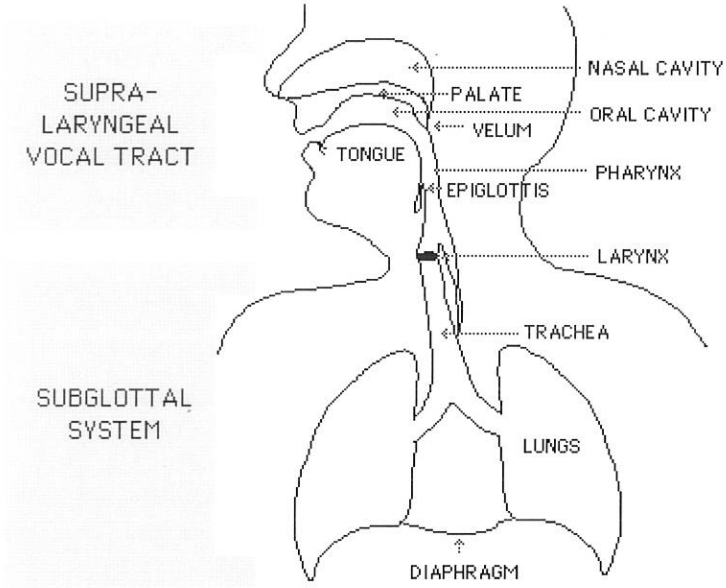


Figure 2.1 The Human Speech Production System
(adopted from Rubin and Bateson, 1998)

The gross components of the human speech production system are the lungs, trachea, larynx, pharyngeal cavity, oral, and nasal cavity. The pharyngeal and oral cavities are usually grouped into one unit referred to as the vocal tract, and the nasal cavity is also termed as the nasal tract. The vocal tract begins at the larynx, and terminates at the lips while the nasal tract begins at the velum and ends at the nostrils (Rabiner and Juang, 1993).

Some of the finer anatomical features involved in speech production include the vocal cords, velum, tongue, teeth, palates, the alveolar ridge, the mouth, and lips. These anatomical components move to different positions to produce various speech sounds and are referred in the literature as articulators (Ladefoged, 1975). Most of the character of a sound is determined by the positions of these articulators in the oral tract.

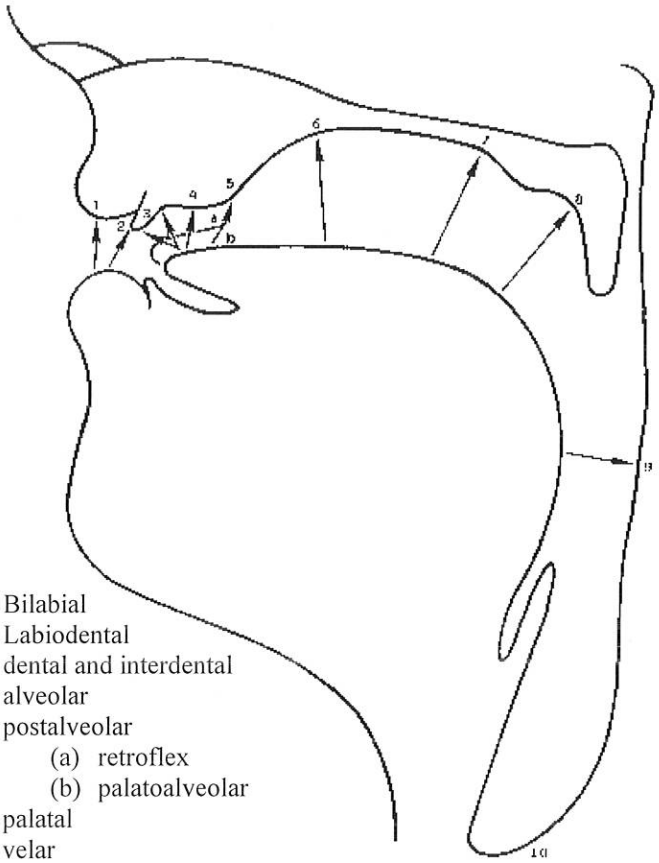
To produce sounds, the oral tract must involve an active articulator, which is raised to form the stricture, as well as a passive articulator towards which the active articulator is raised (Mongham, n.d)

Table 2.1 Major places of articulations
(adopted from Mongham)

Place	Active Articulator	Passive Articulator
Bilabial	Lower lip	Upper lip
Labio-dental	Lower lip	Upper teeth
Dental	Tip of tongue	Upper teeth
Alveolar	Blade of tongue	Alveolar ridge
Retroflex	Tip of tongue	Hard palate
Palatal	Front of tongue	Hard palate
Velar	Middle of tongue	Velum (Soft palate)
Uvular	Back of tongue	Uvula

The number of places along the oral tract where a stricture can be produced is theoretically infinite (how many points are there on a line?), but human languages only distinguish a dozen or so and most languages only use about half of those.

Table 2.1 gives the major places of articulation, and Figure 2.2 mentions a few more:



1. Bilabial
2. Labiodental
3. dental and interdental
4. alveolar
5. postalveolar
   (a) retroflex
   (b) palatoalveolar
6. palatal
7. velar
8. uvular
9. pharyngeal
10. glottal

Figure 2.2 Places of Articulation
(adopted from Mongham)

2.3 Artificial speech production system

Artificial (non-human) method of speech production is a system that automatically transforms arbitrary or unrestricted natural language sentences from its text form into its spoken form. In transforming texts to speech, such systems need to be divided into two major components based on the functions needed of these systems. These components are the natural language processing (NLP) and the digital or speech signal processing (DSP)

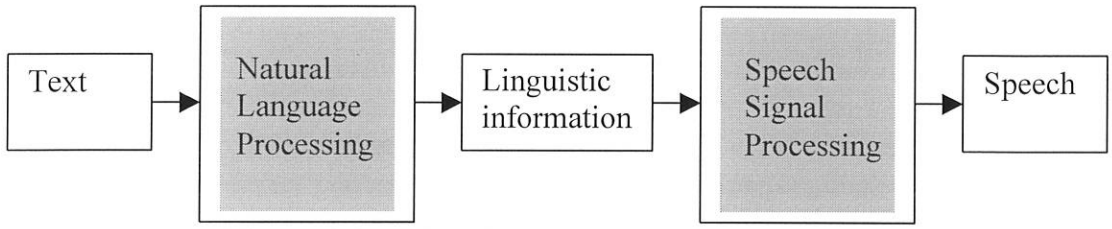


Figure 2.3. Major Components of a TTS System

(adopted from Dutoit,1997)

The output of the first component, which is shown as linguistic information in the figure above, is a phonetic unit and prosodic parameters of the input text. These parameters and units are processed in the second component, which is also referred in the literature as digital speech processing component, to produce the final speech output (Dutoit,1997).

2.3.1 The Natural Language Processing Subsystem

As illustrated in the figure below, the task of converting the input text into a linguistic representation can be further partitioned into two components: The transformation of text in to phonetic units and the conversion of text into prosodic parameters. The phonetic units specify what sounds need to be produced while the prosodic parameters specify how they are to be produced (Ng,1998)

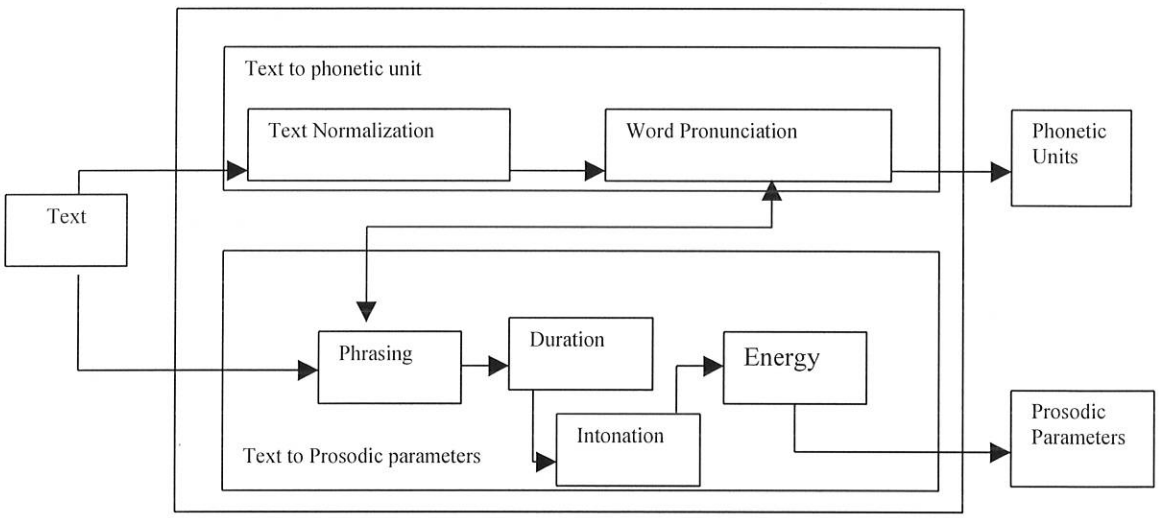


Figure 2.4. Major Components of Natural Language System

2.3.1.1 Text to Phonetic units

The smallest segments of sounds that can be distinguished by their contrast within words are called phonemes. They are the abstract units that form the basis of writing down a language systematically and unambiguously (Ladefoged, 1975). Changes of these units in words can bring a difference in meaning. Therefore to treat sounds that bring meaning differences, the input text has to be converted to these unit of sound. For example the word record can be phonemically transcribed as

- Record /reko:d/-a piece of evidence or information
- Record /riko:d/ -set down in writing

The difference in the orthography (the writing and the spelling system of a language) and phonemic transcription is , as it can seen from the above example, phonemic transcription capture meaning difference. Therefore the input text must be converted first to these units.

In order to convert streams of letters to sounds, the input text is therefore converted to a serious of sound representatives (phonemes). To achieve this objective this component is again subdivided into two components: the text normalization and the word pronunciation modules. Both of these functionalities are discussed below.

2.3.1.1.1 Text Normalization

The goal of the text normalization component is to transform the raw input text stream into a regularized format that can be processed by the rest of the system. This includes breaking up the input into sentences, tokenizing into words, expanding numbers, dealing with abbreviations and possibly tagging words with their part of speech labels to help with later pronunciation and prosody processing (Ng,1998).

Many approaches to text normalization are based on heuristic and rules. For example punctuation, capitalization, and white space can be used as cues for detecting sentence and word boundaries in many languages; numerals and abbreviation expansion can be disambiguated via series of heuristics that examine the surrounding context.

In this component the sentence *The man walked down 56th st.*, for example, should be first changed into *The man walked down fifty sixth street* and then decomposed into its constituent words (Rozak, n.d).

2.3.1.1.2 Word Pronunciation

Once the sequence of words has been specified by the normalization procedure, the next step is to detrmine their pronounciation. The most successful approach todate have been rule based. The simplest approach is to use a set of letter-to-sound rules to map grapheme sequence to phonetic sequences. Ng (1998) argues that in languages such as Spanish where

there is a strong correlation between the orthography and the phonology, this approach works well. On the other hand for languages like English which have more complex relationship between the spelling and the pronounciation, the addition of pronounciation dictionaries containing entries for words which are exceptions to the rules have been effective. The other approach is based on morphological analysis. A word is first decomposed into its morphemes, the minimal meaningful unit of the language such as prefixes, roots, and suffixes; next, morpheme pronounciation are determined using morpheme-to-sound rules and morpheme pronounciation dictionaries; finally the piece are combined using phonological rules (see morphophonemics in chapter three) to get the pronounciation of the whole word. Other methods include determining pronounciation by analogy to known words and disambiguating homographs by using parts of speech. The example given in the previous section, *The man walked down fifty sixth street can* be converted into a sequence of phonemes as " Th-uh m-a-nw-au-l-k-td-ou-nf-ihf-t-ees-ih-k-s-th s-t-r-ee-t" (Rozak, n.d).

2.3.1.2 Text to Prosodic parameters

Prosody is a term used to describe the metrical structure of speech, which includes pauses, the perceived length, stress, and pitch of speech. The physical correlates for the length, stress and pitch are, the duration, energy and fundamental frequency (Ng,1998). The pattern of these parameters as a function of time carries linguistically significant information and is the key to producing natural sounding speech. The task of text to prosodic parameters converting component is, to generate a time varying trajectory of these prosodic parameters.

2.3.1.2.1 Phrasing

A long sentence is typically broken up into phrasal units. These phrases are important for specifying prosodic properties since there are pauses at phrase boundaries. Also the

fundamental frequency and energy are usually reset at the beginning of a new phrase.

The simplest approach to break up a sentence into phrasal units is to use punctuation marks such as commas, semicolons, and periods as indicators of phrase boundary locations. However, a problem arises in long strings of words without punctuation. In these instances, other methods like, keeping a list of words (function words and verbs) that are likely indicators of good places to break, performing a syntactic parse of the sentence to discover the phrase boundaries and clause boundaries are used (Ng,1998).

2.3.1.2.2 Duration

In addition to specifying which phoneme need to be produced, it is also necessary to determine how long to make each phone. There are many factors that influence the duration of a phonetic segment. These include the identity of the phone itself, the characteristics of the neighboring phones, its position within the phrase, etc. Rule based approaches that take into account some of these factors have been developed for predicting segment duration. In one approach, the duration of a segment is determined by successively applying a set of rules; each rule accounting for a particular factor and tries to change the segment duration by a percentage increase or decrease subject to a minimum duration constraint. For example Klatt (1987) specifies a set of rules of which CLAUSE-FINAL LENGTHENING is one. According to this rule, the vowel or syllabic consonant just before a pause is lengthened.

2.3.1.2.3 Intonation

The fundamental frequency or pitch is the frequency of voicing, that is, the frequency at which the vocal folds vibrate (Childers, 2000). The intonation of a sentence is the pattern of pitch changes that occurs. A single tone group is formed by part of a sentence over which a

particular pattern extends. Therefore a short sentence often forms a single tone group, while longer ones are made of two or more (Ladefoged, 1975)

For example the pattern of pitch as given by Ladefoged (1975) over the sentence *the girl gave the money to her father* varies as

the girl gave the money to her father

Intonation variation can bring a difference in meaning. The statement *he ate meat,* for example, when spoken with falling intonation is a statement, and a question when spoken with rising intonation, and with dramatically rising intonation is interpreted as surprise or outrage.

He ate meat - statement

He ate meat - question

He ate meat - surprise

Therefore the goal of the intonation component is to generate fundamental frequency (F_0) contour for the sentence to be synthesized. In this component F_0 rises and falls is predicted through analyzing the sentence location and the syntactic structure.

2.3.1.2.4 Energy

In addition to a fundamental frequency contour, a spoken utterance also has an energy contour. Certain phones are more intense than others, and the ends of phrases are weaker than the beginnings. However, it has been observed that simply using the normal segmental energies inherent in each phone combined in the fundamental frequency contour is sufficient to implicitly specify the energy contour (Ng,1998)

2.3.2 Speech signal processing

Approaches to this task can be categorized into two groups: those that attempt to model the speech production system and those that attempt to model the speech signal. Articulatory synthesis falls into the first category while formant synthesis and concatenative synthesis fall into the second.

In another classification, articulatotry and formant synthesis are categorized in one group as rule based approaches. They are grouped together in this classification, because both approaches specify a set of rules to utter the sound that need to be produced. Concatenative approach is grouped alone in the second classification because, unlike articulatory and formant, it uses parts of actual speech to synthesize an utterance.

2.3.2.1 Rule -Based Approach

In rule-based synthesis, all the perceptually relevant acoustic parameter values are generated by context-sensitive rules formulated on the basis of an analysis of natural speech patterns.

One of the rule based approach, articulatory synthesizers try to generate speech by modeling the human speech production system. According to acoustic theory of speech production, the

human vocal tract can be modeled as an acoustic tube with non-uniform and time-varying cross-section. The acoustic tube can be adjusted to various shapes by adjusting the various parameters. The articulatory parameters are expressed in the form of a vector and specify the positions of the tongue body, tongue tip, jaw, lips, etc (Childers,2000).

While it is possible to express the rules either in terms of articulatory, knowledge of articulation is still at its infancy, and commercially available rule based systems generally model perceptual relevant acoustic values, such as the formant patterns observable in spectrograms.

Formants can be thought as the frequencies at which the vocal cavities resonate during articulation. In this approach, a rule for example, might state that the second formant of the sound /k/ before a front vowel is 2000 Hz, while before a back vowel it is considerably low (Hertz et. al, 1997). Using these parameters, formant synthesizers try to synthesize speech.

Systems developed using this approach include the MITTALK, the JSRU synthesizer, the multilingual INFOVOX system and the INRS system for French (Dutoit,1997).

In an overall evaluation of such systems, Dutoit (1997) argues that because of the large number of parameters (up to sixty), formant synthesizers complicate the analysis stage and tends to produce analysis errors. He adds that in order to cope with the analysis errors intensive trials (taking several years) are commonplace. Moreover, according to him, the synthesis quality achieved up to now in these systems reveals typical buzziness problem.

2.3.2.2 Concatenative synthesis

In concatenative synthesis, speech is produced by retrieving appropriate intervals of stored natural speech. The retrieved stored speech is then stringed together, and then some signal

processing can be added on the result to smooth out the segment transitions and to match the specified prosodic characteristics.

Desirable properties for the set of speech segments include accounting for as many co-articulatory effect as possible, having minimal discontinuities at the concatenation points, and being as few in number as possible. Longer segments are able to capture more co-articulation and have fewer concatenation points than shorter ones. However, the number of different segments grows exponentially with the length off the segment. These conflicting objectives mean that there has to be some tradeoff in selecting the synthesis units.

Many different types of synthesis units have been explored. These include linguistically motivated units such as words, syllables, demi-syllables, diphones, and phones. Other units include sub-phonetic segments corresponding to the states in a trained hidden Markov model (HMM) (Donovan ,1996).

2.4 Units of Concatenative synthesis

Concatenative synthesis can have a number of options as units for implementation. It can have words, syllables, demi-syllables, diphones, phones and sub-phone units as unit of concatenation.

2.4.1 Words

The Oxford Current English Dictionary defines a word as the meaningful component of speech shown with a space on either side of it when written. It is the most obvious synthesis unit to choose, and that most often suggested by people not working in the field. The advantage of using words is that all the within word co-articulation effects are captured in the stored units. Concatenating words is then relatively easy, compared to sub-word synthesis

units, because between words co-articulation is usually weaker than within word co-articulation. However, simply concatenating the waveforms of words recorded in isolation is doomed to failure because a spoken sentence is very different from a sequence of words uttered in isolation. In a sentence, words, are as short as half their duration when spoken in isolation-making concatenated speech painfully slow (Klatt 1987). The sentence stress pattern and rhythm, and intonation which depend on syntactic and semantic factors are disruptively unnatural when words are simply strung together in a concatenation scheme.

A second problem with approaches that attempt to store representations for whole words is that the number of words that can be encountered in free text is extremely large, due in part to the existence of unbounded set of names (e.g. The Social Security Administration in the U.S.A in 1985 estimated that there were over 1.7 million different surnames). In addition general rules permit the formation of larger words by the addition of prefixes and suffixes to the root words. Also new words are coined every day (Klatt,1987).

2.4.2 Syllables

The syllable is a unit containing one and only one vowel either alone or surrounded by consonants in certain number (O'Conner,1973). The use of syllables as synthesis unit represents a halfway stage between words and smaller phone sized units. As with words, the advantage is that the relatively long synthesis unit preserve within unit co-articulation. However, unlike words, the between unit co-articulation is not necessarily weaker than the within co-articulation, and so smoothing across unit boundaries is not simple. For example the word segmentation, when broken down to syllables become

/seg/ /men/ /ta/ /tion/

For the word to be utterable, the syllable units need to be stored. Accordingly to cover all known words of English, up to 10,000 syllables are required (Donovan,1997)

2.4.3 Demi-syllables

Demi-syllables represent another step down in unit size, being the initial and final halves of syllables (Donovan, 1996). The advantage of demi-syllables is the numbers of units required is smaller when compared to higher units, so recording and storing is not difficult (1000 demmi-syllables are required to cover the 10000 syllables for English). The word "segmentation" when represented in terms of demi-syllables is

/se/ /eg/ /ma/ /an/ /t-ey /ey-sh/ /sh-a/ /an/

However Donovan (1997) mentions that when using this unit, co-articulation between syllables can become problematic. This could be the reason why he was only able to mention a single commercial TTS, which used demi-syllable as its unit. The TTS system is called ORATOR TTS.

2.4.4 Phones

The phone is the smallest sound element, which can be segmented. It represents the typical kind of sound and sound nuances of a certain sound. Sounds which are phonetically similar, belong to the same phoneme (Childers, 2000).

Concatenation synthesis using phone-based segments of speech is difficult due to the large amount of contextual variation in the acoustic realization of each phoneme, and the consequent problems in selection appropriate units and ensuring concatenation smoothness (Donovan ,1996).

2.4.5 Diphones

A diphone is roughly the last half of one phone followed by the first half of the next (Dutoit,1997). Diphone units therefore preserve transitions between phones, which are otherwise difficult to produce. The boundaries between diphones during synthesis thus occur in the middle of phones. This tends to result in relatively small concatenation discontinuities because the middle of phones is usually their most spectrally stable regions (Donovan,1996). For these reasons the unit has been used to develop TTS systems for languages like English, French, German, Arabic (Mbrola Homepage) Welsh (Williams, n.d), Scottish Gaelic (Wolters,1997), etc.

The word segmentation in this unit can be represented as

/sil-s//s-e/ /e-g//g-m /m-a/ /a-n//n-t/ /t-ey /ey-sh/ /sh-a/ /a-n//n-sil/

where "sil" represent silence.

2.4.6 Sub-Phone units

There is no as such a representation for sub-phone units. Sub-phone units result from the segmentation of phones. In recent years sub-phone units have started to be applied for the development of speech synthesis. Speech signal when segmented into frames, typically of 10 msec duration, each frame is classified to indicate a sub-phone unit of a phoneme.

The use of sub-phone units for speech synthesis has been accomplished through Hidden Markov Models (HMMs). But, application of HMM to speech synthesis is non-trivial and time taking (Donovan, 1996). It is an expansive field on it's own, and therefore this thesis concentrates on the other units of synthesis.

2.5 Chapter Summary

In this chapter the basic principles of speech synthesis is discussed. In particular, the components of a text to speech system and the various approach of digital speech processing techniques were highlighted. Moreover the units of concatenation in a concatenative speech synthesis have also been discussed. In the following chapter the phonological process of the *Afaan Oromoo* language is discussed.

CHAPTER THREE

THE AFAAN OROMOO PHONOLIGICAL PROCESS

3.1 Overview

Afaan Oromoo is a single common mother tongue for the Oromo people. The language, *Afaan Oromoo* (AO) belongs to the Eastern Cushitic group of languages and is the most extensive of the forty or so Cushitic languages. It is very closely related to Konso, with more than 50% of the words in common, closely related to Somali and distantly related to Afar and Saho (Melba:1988). *Afaan Oromoo* is considered to be one of the five most widely spoken languages from among the approximately 1000 languages of Africa (Gragg, 1982). It is spoken in many parts of Ethiopia, northern Kenya with native speakers living in Somalia and Sudan, too (Lloret: 1997). In fact, Melba (1988) states that *Afaan Oromoo* is a lingua franca in the whole of Ethiopia except for small portion of the northern part. Furthermore it is a language spoken in common by several members of many of the nationalities like Harari, Anuak, Berta, Sidama, Gurage, etc. It is believed that perhaps not less than two million non-Oromo speak the language as a second language.

There is a dialectical variation among *Afaan Oromoo* speakers. According to previous studies in this regard (for example, Gragg (1976)), four major categories can be identified. These are: Western (Wellega, Iluababor, Kaffa and parts of Gojjam), Eastern (Harar, Eastern Showa, and parts of Arsi and Bale), Central (Central Showa, Western Showa and possibly Wollo) and Southern (parts of Arsi, Sidamo and Borena). Although this is the case AO scholars agree on the mutual intelligibility between them (Lloret: 1997).

With regard to the writing system of AO, the Roman (Latin) alphabet has become the official script for AO since the end of 1991 (Tilahun:1994).

Traditionally, the natives, especially the elders refer to the language as *Afaan Oromoo*, while the non-natives designate it as simply *Oromoo*. It is believed that the former is more acceptable since it is indigeneous and the people's self designation (Bender et. al, 1976). Hence *Afaan Oromoo(AO)* is preferred to *Oromoo* in this thesis.

3.2 Phonology of Afaan Oromoo

Phonology is the description of the systems and the patterns of sounds that occur in a language (Ladefoged:1975). The major representatives of sources of the sounds of *Afaan Oromoo*, the vowels and consonants, are shown in the tables below.

3.2.1 Consonants

The following table shows the places of the articulation and the categories of the Afaan Oromoo consonants(Gragg, 1976).

Table 3.1: Afaan Oromoo Constants

		Labial	Alveolar	Palatal	Velar	Glottal
	Voiceless	(p)	T	Ch	k	?
	Voiced	B	D	J	g	
stop	Glotalized	Ph	X	C	q	
	Implosive		Dh			h
	Spirant	F	S	Sh		
continuant	(voiceless)					
	Voiced	(v)	(z)			
	Nasal	M	N	Ny		
resonant	Sonorant		l r			
	Glide	W	Y			

Afaan Oromoo consonants occur single in initial position while intervocalically they may occur single, geminated, or as members of bi-consonantal clusters. In fact consonant gemination is phonemic in Afaan Oromoo (Tilahun:1994).

All consonants except /h/ occur single or geminate (Lloret: 1997).

e.g

harree	'donkey (double r)	*harka*	'hand' (single r)
kalee	'kidney' (single l)	*kallee*	'child's garment' (double l)
butaa	'snatcher' (single t)	*buttaa*	'period of Gada system' (double t)

Some AO words contain /z/ and /p/, but they are clearly loan words. In general loan words which have /z/ tend to be assimilated to /s/ (Occasionally also to ĵ) while /p/ tends to be assimilated to /f/ (occasionally also as /p'/ or /b/). Some loans introduced to the language with /v/ also tend to be assimilated as /f/ (Lloret, 1997).

e.g.

muuza, *muusa*, 'banana'
ajaja 'command' from Amharic azzaza.
Poolisii- foolisii 'police' European origin
Vitaamini, fitaamini, 'vitamin'
Peesaa, *beesee* (Southern) 'money' from Swahili pesa.

The corresponding equivalent pronunciation of English for the consonants of *AO* is given below

Table 3.2 Pronunciation of AO Consonants

(adopted From Mekonnen, 2000)

Afaan Oromoo consonants	*approximately corresponds to English*	*Examples*
B	B	as in book
C	no approx. equivalent	
ch	Ch	as in church
D	D	as in door
dh	no approx. equivalent	
F	F	as in fun
G	G	as in good
H	H	as in hat
J	J	as in jump
K	K	as in king
L	L	as in lamp
M	M	as in man
N	N	as in north
ny	no approx. equivalent	
P	P	as in public
ph	no approx. equivalent	
Q	no approx. equivalent	
R	R	as in root
S	S	as in sand
sh	Sh	as in sharp
T	T	as in table
V	V	as in victory
W	w	as in west
X	No approx. equivalent	
Y	Y	as in yonder
Z	Z	as in zebra

3.2.2 Vowels

AO makes a phonemic distinction between five short and five long vowels which occur either within or at the end of words.

short		long	
i	u	ii	uu
e	o	ee	oo
a		aa	

Although the number of the vowels are many as compared to English, the possible sequences of vowels is limited to identical vowels only. Lloret (1997) argues that Afaan Oromoo words end with vowels. Although her observation is correct for many words, there are some exceptions. For example the word *afaan* 'mouth' end with a consonant n.

eg.

dadhabe	'I am tired'	*dhaaba*	'I plant'
egaa	'then'	*eegaa*	'watch'
dhufii	'come'	*dhuufi*	'fart'
gara	'side'	*gaara*	'hill'

The corresponding equivalent pronunciation of English for the vowels used in A*O* is given below.

Table 3.3 Pronunciation of Afaan Oromoo Vowels
(adopted From Mekonnen, 2000)

Afaan Oromoo Vowels	*Approximately corresponds to English*	*Examples*
A	A	as in but
Aa	a	as in father
E	E	as in bet
ee	e	as in late
I	I	as in sit
Ii	i	as in seat
O	O	as in no
Oo	o	as pope
U	U	as in put
Uu	u	as in pool

Because there is a one to one relationship between the alphabets of the language and the sounds of the letters, the alphabets are considered as phonemes in this work. Therefore the above listed consonants and vowels can be also considered as phonemes.

3.2.3 Consonant Clusters

AO has widely uses consonant clusters in medial positions. All the sequences are bi-consonantal, because the language does not allow sequences of more than two consonants. The syllable break is almost always between the two consonants. These sequences are pronounced with a syllable break between the two consonants in careful speech; but this break is fairly maintained in connected normal speech (Lloret:1997).

Among the dialects, western dialect occasionally show /lb/, /rb/ and /rf/, /br/, /lf/ sequences in free variation with /bl/, /br/ /rb/ and /fl/

e.g.

ablee albee 'knife'
durba dubra 'girl'
durba durfa (Southern Waataa) 'girl'
kolfa kofla 'laughing'

Neither initial nor final consonant clusters are found in AO. Consonant clusters occur only medial positions and the number of consonants forming a cluster is limited to two (Tola:1988)

According to Tola (1988), AO consonant phonemes can be classified into four groups. These groups are

- Consonant phonemes which do not combine with other consonant phonemes as first members to form clusters. These consonant phonemes are

  /ph, t, x, d, dh, ', s, ch, c, j/

- Consonant phonemes which do not combine with other consonant phonemes as second members to form clusters. These consonant phonemes are

  /l, n, r, y/

- Consonant phonemes which do not combine with other consonant phonemes at all. These consonant phonemes are

  /h, w/

- Consonant phonemes which do not combine with other consonant phonemes either as first or second members to form clusters. These consonant phonemes are

  /b, k, q, f, g, s, m, n/

This classification, though lacks precision. For example /n/ which is classified in phonemes which do not combine with other consonant phonemes as second members to form clusters, can be found as a cluster of second member in words like *humna* (to mean force).

3.3 The glottal stop (')

In the writing system of AO, more than two consecutive vowels are not allowed to appear. However, in places where there is a glottal stop (pause), more than two consecutive vowels can appear if separated by the apostrophe (').

Eg.

Ta'uu 'become'
Taa'uu 'to sit'
Danda'uu 'to be able '

3.4 Morphophonemics

The modification of basis and affixes in morphological process are known as morphophonemic change and their study is called morphophonemics (Lehmann:1972). Some of the types of morphological changes in relation to *Afaan Oromoo* are discussed briefly below.

3.4.1 Assimilation

Consonant clusters across morpheme boundaries arise only between the final consonant of a stem and /n/, /t/, or /s/, or between /n/ and a consonant initial stem (Lloret:1997). When these

consonant clusters appear a morphological change called assimilation occurs. Assimilation is the process by which a sound changes to become phonetically more like an adjacent sound (Wardhaugh:1977). The following table summarizes some of the occurrence of assimilation in AO.

Table 3.4. Assimilation process in AO words

(source Lloret, 1997)

Type	Description	Instance
S-Labialization	/s/ becomes /f/ when it is followed by a consonant	ajjees + na → ajjeefna-we kill ciis + ta → ciifta- do you sleep
Nasal total assimilation	/n/ completely assimilates to a preceeding liquid, and to a following liquid or glide	gal + na → galla- we enter moor(a) + ni → moorri- fat hin + latu → hillatu- not giving
Voice assimilation between stops	/t/ assimilates in voicing to a preceeding stop	did + ta → didda- you refuse qab + ta → qabda- do you have
Stop consonant assimilation	Assimilation between a stop and a following /t/ or /n/	fid + na → finna- we bring bit + na → binna- we buy

3.4.2 Deletion

Sounds can also show a process by which one of the sounds deletes the other. For example in Western Oromo /dh/ is systematically dropped before a consonant (Lloret:1997).

e.g

fuudh + ta → *fuuta*- will you take it

fedh + na → *fenna*- we wish

kana + uma → *kanuma*- only this

3.4.3 Ephentisis

Ephentisis is the insertion of a sound in pronunciation (Wardhaugh:1977). A cluster of three consonant medially is impermissible in *Afaan Oromoo*. Because of such constraint the process of ephentisis takes place. A sequence of three consonants can happen as a result of morpheme combination. Ephentisis can also occur to avoid consonant cluster in word final position.

e.g

arg+ne → argne → *argine*- 'we saw'

jibb+ te → jibbte → *jibbite* - 'have you hated'

bor+s → bors → *boris*- 'also tommorow'

3.4.4 Metathesis

Metathesis is a phonological process in which, for variety of reasons, letters, sounds, and even syllables within a word are transposed (Wardhaugh:1977). In AO, the process of metathesis takes place when as a consequence of root and suffix combination, an impermissible sequence of segments would occur. The process of deletion of a vowel leads to metathesis in *Afaan Oromoo*.

e.g.

fr → rf

afur +afa → afuraffaa → afraffa→ arfaffaa - fourth

3.5 Stress and pitch

Tola (1988) states that there are three degrees of stress - primary, secondary, and weak, and two levels of pitch- high (◡) and low(◠) in the Western dialect. The primary stress and the high pitch occurs inseparably on the same syllable and the secondary stress and the low pitch occur on the same syllable in their respective patterns.

pattern one: A word ending in a short vowel has a primary stress and pitch on the penultimate and secondary stress and low pitch on a preceding syllable if there is any.

e.g

barbaade- 'looked for someone'

deeme- 'he went'

nama- 'human being'

pattern two: Any word ending in a consonant or in a long vowel has a primary stress and a high pitch on the ultimate syllable with the secondary stress and low pitch on any preceding syllable

e.g *bishdan*- 'water'

matáa- 'head'

karáa - 'road'

pattern three: In any monosyllabic word, the syllable has a primary stress and a high pitch.

e.g. *shan*- five

yoom- when

The grammatical meanings of words change when the words are uttered with a shift of stress and pitch from one syllable to the other. For example, an infinitive verb form has a primary stress and a high pitch on the ultimate and the penultimate syllable. If it is uttered with secondary stress and a low pitch on the penultimate syllable, and a primary stress and a high pitch on the ultimate, a question is signaled.

e.g.	Infinitive		Question	
	barbaadu	'to look for'	*barbaaduu*	may I look for ?
	gatuu	'to throw'	*gatuu*	may I throw?

3.6 Chapter summary

In this chapter, some of the underlying principles in the writing system and the sounds of *Afaan Oromoo* that are considered important for the development of a Text-to-Speech system are discussed in brief. In general, it can be seen that gemination and elongation of the writing system has given the language to be clear as to the lexical meaning of the word. This advantage will be used as rules for converting the letter sequences to the respective sounds in the next chapter.

CHAPTER FOUR

EXPERIMENTATION

4.1 Introduction

In this chapter, the selection of test and corpus data, the development of a TTS algorithm for *Afaan Oromoo* words and its evaluation are discussed. In addition, the tools considered for development of an *Afaan Oromoo* TTS are briefly pointed out. In developing an algorithm, a unit of synthesis has to be selected. In this work how a unit of synthesis is chosen is also explained. Segmentation of corpus data and its related problems are other points that are incorporated in this chapter.

4.2 Tools considered for AO-TTS

In this study, two tools were identified that could be used for speech synthesis problem. The first one is the Hidden Markov Toolkit (HTK). It is a generic toolkit designed for speech recognition. It has been tested (e.g by Donovan, 1996) and found to be working for speech synthesis too. This toolkit provides documentation only for using the kit to speech recognition system development. In order to adopt the toolkit for a speech synthesis problem, therefore requires more time than the allotted time for this work. The toolkit is therefore not considered for further investigation.

The second tool is the center for spoken language understanding (CSLU) toolkit. It provides tools for the development of speech recognition and synthesis systems (Sutton et. al, n.d). This toolkit is a generic tool for both recognition and synthesis problems of speech. Therefore it was preferred to HTK. However the CSLU toolkit is found to be hardly accessible. It was thus, expensive to download the toolkit at the current rate of the Internet connection in Ethiopia. A minimal installation of this toolkit needs download of 25Mb of data and the

"typical" (recommended) installation 28Mb. A complete download of the toolkit and all of the optional modules and TTS voices is around 88Mb.

4.3 Unit Selection

The commonly used units of synthesis as discussed in chapter two are, words, syllables, demi-syllables, diphones, phones and sub-phone units. For reasons discussed in the same chapter, words, phonemes and sub-phoneme units are not selected as a unit of synthesis for the purpose of this study.

The difference between syllable, demi-syllable and diphone units can be illustrated using the word *gargaarsa* 'help'. It is decomposed into these units. The analysis (by a speech analyzer 1.5) is also given.

- Syllables- g-a-r---------- g-aa-r---------- s-a
- Demi-syllables g-a---a-r---g-aa---aa-r---s-a---a-sil
- Diphones sil-g---g-a---a-r---r-g---g-aa---aa-r---r-s---s-a---a-sil

Demi-syllables, as it can be seen from the above decomposition, are more of like diphones. Their similarity with syllables is that they do not capture the transition between consonant clusters (i.e. there is no sound representatives of r-g, r-s in demi-syllables and syllables). On the other hand, the advantage of demi-syllables as compared to diphones is that, the units are few in number. Therefore it may seem that having fewer concatenation points would give better result. But, the disadvantage by not incorporating consonant transitions shadows this advantage. This disadvantage, because it is the same with that of syllables, is explained in the discussion for syllables.

To make comparison between syllables and diphones an utterance of the word *gargaarsa* ' help', is given in figures 4.1, 4.2, and 4.3. Figure 4.1 shows the word as spoken by a human being. Figure 4.2, and 4.3 show the word when uttered through concatenating units of

syllables and diphones respectively. These figures show the waveform, the magnitude (energy) and the spectrogram of the utterance.

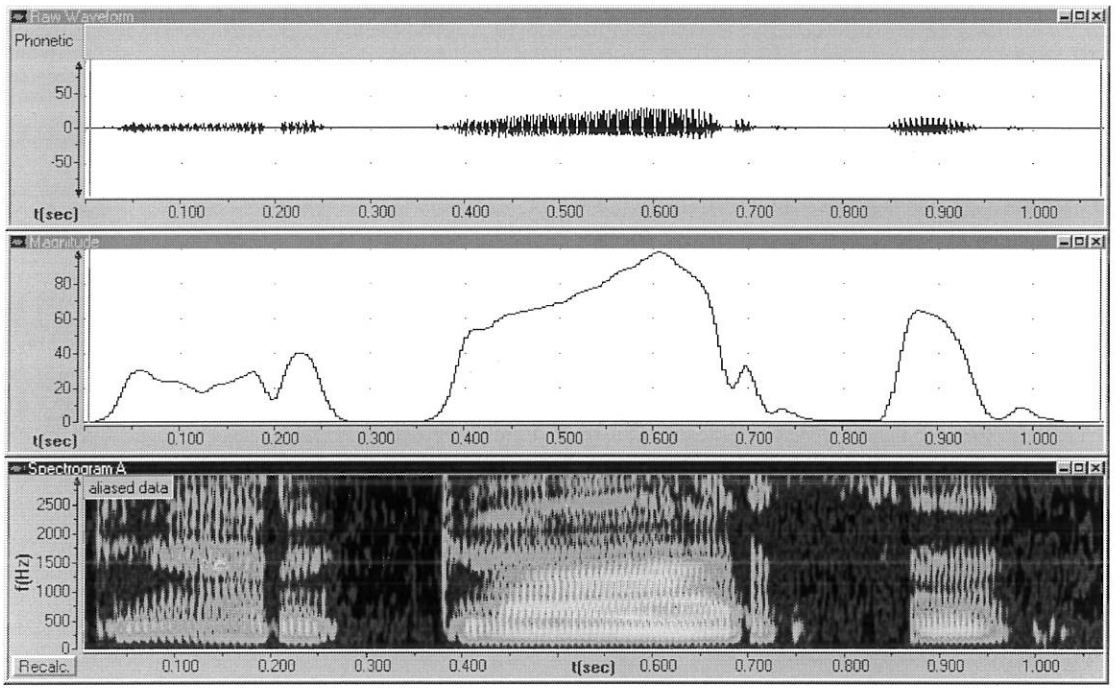


Figure 4.1 *gargaarsa* ' help' as spoken by a human being

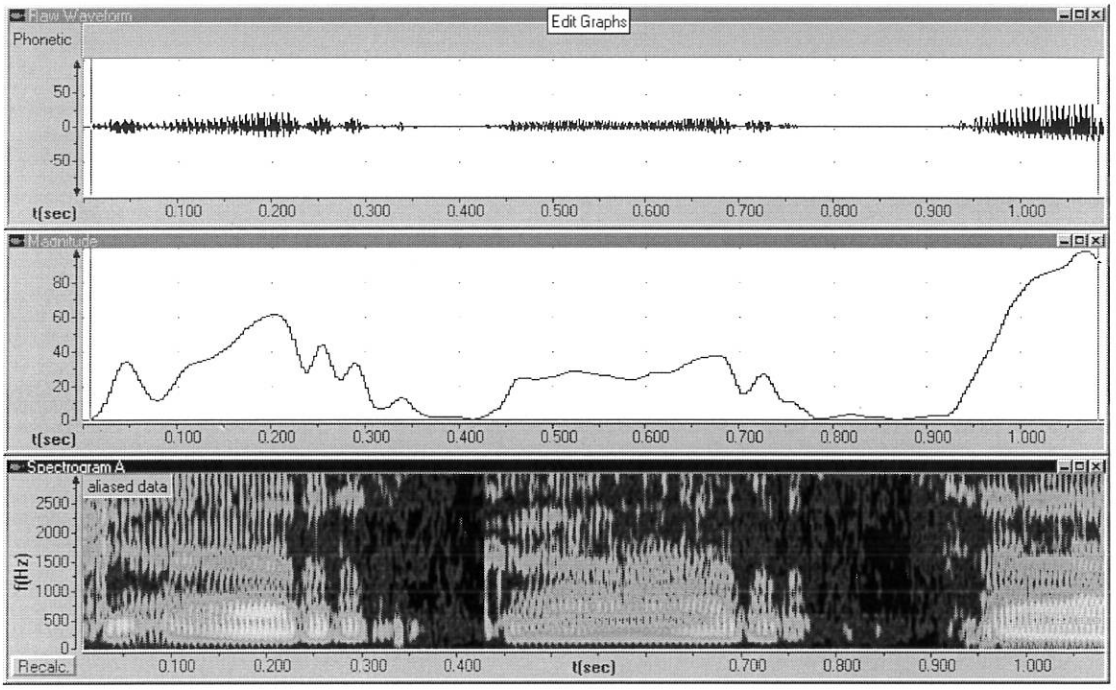


Figure 4.2 the utterance of *gargaarsa* ' help' when syllable are concatenated.

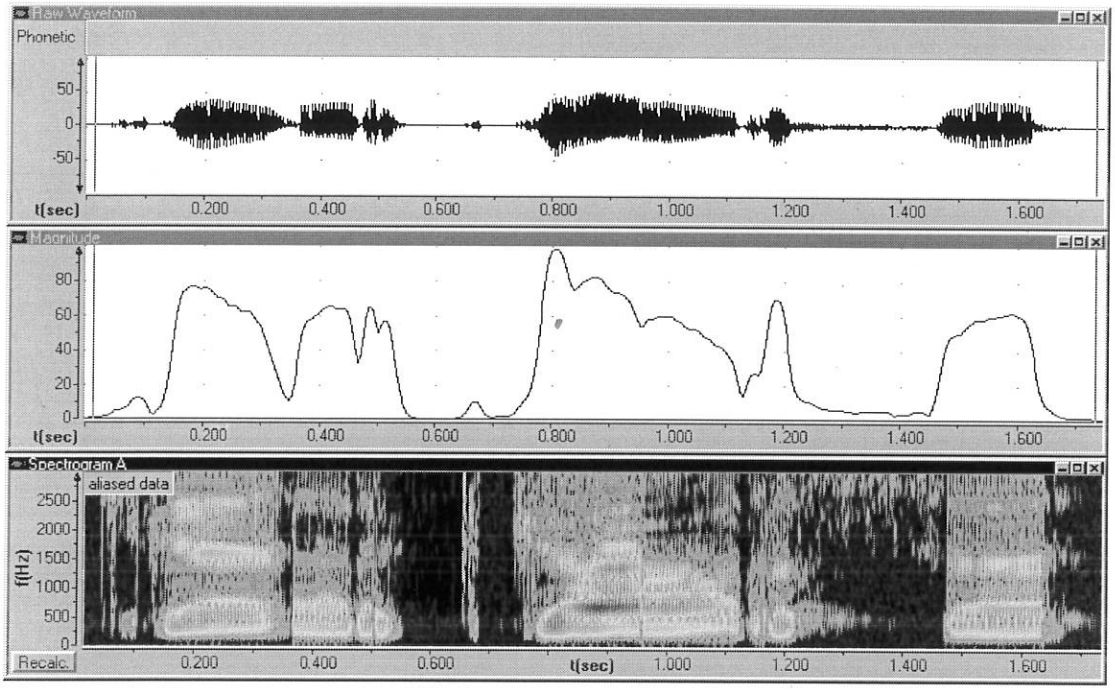


Figure 4.3 the utterance of *gargaarsa* ' help' when diphones are concatenated.

In figure 4.2 the silence area at the transition points between the syllables (the dark space in the spectrograms) covers more area when syllables are concatenated. This makes the utterance of the word *gargaarsa* 'help' an utterance of isolated syllables of *gar gaar sa*. Since demi-syllables are extracted from syllables, this behavior is also manifested when demi-syllables are used.

Moreover, using syllable as a unit can introduce additional problem. If new monosyllabic words are created, additional step is required for the addition of the new monosyllable word to the syllable inventories. For example the word *shan* 'five' is a monosyllabic word. To utter this word using the units of syllables, a syllable based TTS requires the recorded utterance of the word itself. For these reasons syllable can not be considered to be exhaustive in representing a language for a TTS problem.

Diphones on the other hand, as seen from the above figures, show less problems at their concatenation boundaries. This is because the midpoints of phonemes does not change due to

co-articulation effects of other phonemes (Donovan, 1996). These units try to capture every transition of diphone sequences. In addition, as it can be seen from figure 4.3, the dark area is less than from that of syllables. Therefore diphone units where preferred for this study as a unit of synthesis.

4.4 Test data

For the purpose of experimentation (i.e. testing the algorithm developed), selection of test data is necessary. To select sample words for testing, it was preferred to use some kind of criteria to arbitrary selection procedures. A good criterion for a diphone based TTS would be the selection of words that are formed by a combination of the most frequently occurring diphones in the language. But since diphone is a concept found in speech synthesis area, researches on diphones of the language are non-existent. It was therefore decided to use the most frequently occurring words as samples for the test data.

The Thorndike junior illustrated dictionary of English gives information on how often a word is used. But, unlike this dictionary, there is no such information for the words of Afaan Oromoo in dictionaries. The selection of words is, therefore, based on previous works made on the frequency counts of words in specific texts. Mekonnen (2000), in this regard compiled a list of words which are found to be frequent in six texts. From this list, the sample words were selected. The selection is based on that the features of words of Afaan Oromoo like, consonant gemination, vowel elongation, are represented in the selected test data. The feature that lacked to be represented is the representation of words with apostroph ('). By adding diphones that start with pauses to the diphone inventory this feature was made to be incorporated.

The words and the correspondingly required diphones for uttering the test data are shown in the table below.

Table 4.1 The test data and it's diphones

diphone / word	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
gadaa 'democracy'	Sil-g	g-a	a-d	d-aa	aa-sil	-	-	-	
sadarkaa 'grade'	Sil-s	s-a	a-d	d-a	a-r	r-k	k-aa	aa-sil	
wayyaa 'cloth'	Sil-w	w-a	a-yy	yy-aa	aa-sil	-	-	-	
barnoota 'education'	Sil-b	b-a	a-r	r-n	n-oo	oo-t	t-a	a-sil	
barsiisu 'educate'	Sil-b	b-a	a-r	r-s	s-ii	ii-s	s-u	u-sil	
sochii 'movement'	Sil-s	s-o	o-ch	ch-ii	ii-sil	-	-	-	
nama 'person'	Sil-n	n-a	a-m	m-a	a-sil	-	-	-	
waggaa 'year'	Sil-w	w-a	a-gg	gg-aa	aa-sil	-	-	-	
harmee 'mother'	Sil-h	h-a	a-r	r-m	m-ee	-	-	-	
shan 'five'	Sil-sh	Sh-a	a-n	n-sil	-	-	-	-	
yeroo 'time'	Sil-y	y-e	e-r	r-oo	oo-sil	-	-	-	
naannoo 'around'	Sil-n	n-aa	aa-nn	nn-oo	oo-sil	-	-	-	
gargaarsa 'help'	Sil-g	g-a	a-r	r-g	g-aa	aa-r	r-s	s-a	a-sil
yokin 'or'	Sil-y	y-o	o-k	k-i	i-n	n-sil	-	-	
sirna 'order'	Sil-s	s-i	i-r	r-n	n-a	a-sil	-	-	

4.5 Corpus data

After the test data was selected, it was needed to specify the corpus data from which the diphones could be extracted. Although, extracting the diphones from the same test data is possible , it could result in inaccuracy (see 4.6 Segmentation). The corpus data was first selected arbitrarily as long as the word for the corpus data is considered to incorporate a diphone for the test data. For example for the word *sadarkaa* 'grade', the corpus required were set through the following analysis.

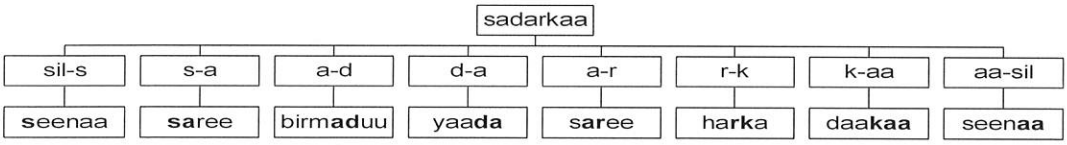


Figure 4.4 The diphones and the corpus data for the word *sadarkaa* 'grade'

Likewise, for all the fifteen words selected as test data, thirty-nine words were selected as corpus data. From these selected corpus data, one was a pseudo-word. A pseudo-word is a word that does not exist in the vocabulary of the language (Wolters, 1997). In this work, to extract the diphones *su* and *ggaa*, the pseudo-word *suggaa* is used.

In general the diphones required for the language can be calculated by categorizing the phonemes as follows.

- Consonants that can be followed by a vowel or any other single consonant

  b, c ,d, f, …, g, j, k,…z numbering 20 (Group a)

- Consonants that can be followed by only a vowel

h, ch, dh, ny, ph, sh, ts numbering 7 (Group b)

- Consonant clusters that can only be followed by a vowel

  bb, cc, dd,…, gg, jj, kk,….zz numbering 20 (Group c)

- All Vowels can be followed by any element from the above three groups

  a, e, i, o, u, aa, ee, ii, oo, uu numbering 10 (Group d)

The vowels can be followed or preceded by a silence (sil) while group a and b can be preceded by a silence. Therefore from this classification, the number of diphones required can be calculated as

- 20 X 10 = 200(Group a X group d) 20 X19 =380 (Group a X (Group a -1))
- 7 X10=70 (Group b X group d)
- 20X10=200 (Group c X Group d)
- 10X 20=200 (Group d X Group c)
- 10 X 7=70 (Group d X group b)
- 10 X 20=200 (Group d X Group a)
- 2 X 10=20 (sil X Group d)
- 1 X 27=27 (Sil X (group a +group b))

The number of diphones required for the language is therefore around 1367. Other rules like that of Tola's (1980) can be used to refine the numbers required.

4.6 Segmentation of the corpus data

Segmentation in this work involves

- Identification of the boundaries of the diphones from the corpus data and
- Copying and saving the waveform between the identified boundaries

To identify the boundaries of the diphones, speech analyzer 1.5 was used.

Segmentation is a painstaking job. Donovan (1997), for example reported that it took three months for two people to construct the diphone inventory for the Center for Speech Technology Research (CSTR) TTS. Therefore to limit the time demand for diphone preparation, this work is limited to test few words.

One option for segmenting the diphones could be to use the test data as a corpus. But segmenting the diphones from the test data may cause errors when the diphones are used for words other than the test data. For example the word *gadaa* 'democracy system of the Oromo people' in its wave form looks like the figure below.

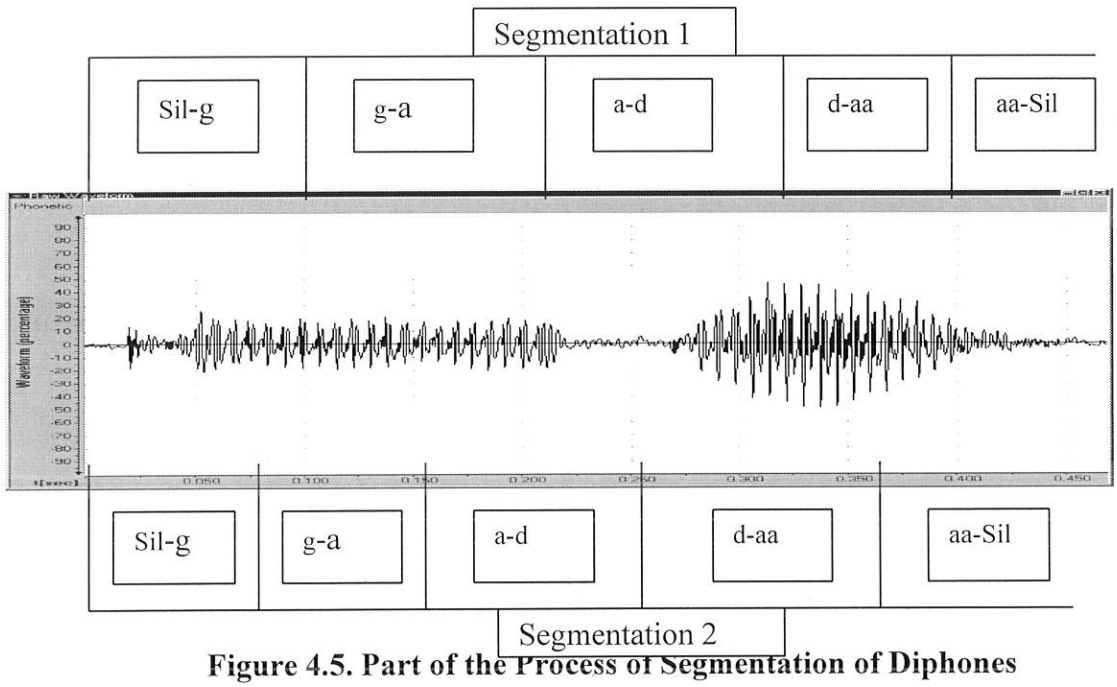


Figure 4.5. Part of the Process of Segmentation of Diphones

In the above figure whether the segmentation is made as segmentation-1 or 2, when the assumed diphones are concatenated, the concatenated utterance will sound like 'gadaa'. Therefore for testing the word *gadaa* the units segmented in either of the above cases work well. But the units can bring problems when used as units of synthesis for other words unless the segmentation is made as accurately as possible. For example if segmentation-2 is the correct segmentation, the diphone 'sil-g' of segmentation-1 represent part of the sound found in the next correct diphone 'g-a' in segmentation-2. Therefore, the diphone 'sil-g' of segmentation-1 when used for words like *gibee* 'name of a river', the diphone will introduce part of the sound of 'g-a' which is not found in the word *gibee*. Therefore, in order to avoid this problem, the diphones are extracted from other words which are found to contain the diphones necessary to synthesize the test word.

To extract the necessary diphones for the test data, the corpus was segmented into the required diphones. This was accomplished through heuristics. i.e the starting and the ending point are identified through repeating trial and error methods until the to be segmented unit is judged to represent the target diphones. During segmentation of the diphones, the chance of inclusion of other sounds is found to be minimal when the diphones are extracted from the beginning or ending parts of a word. For example to extract the diphone *ok* it was found simpler to extract it from the word *okaati* ' rope' rather than *sokoksa* 'rustle'. This is because the co-articulation effect is minimal at boundaries.

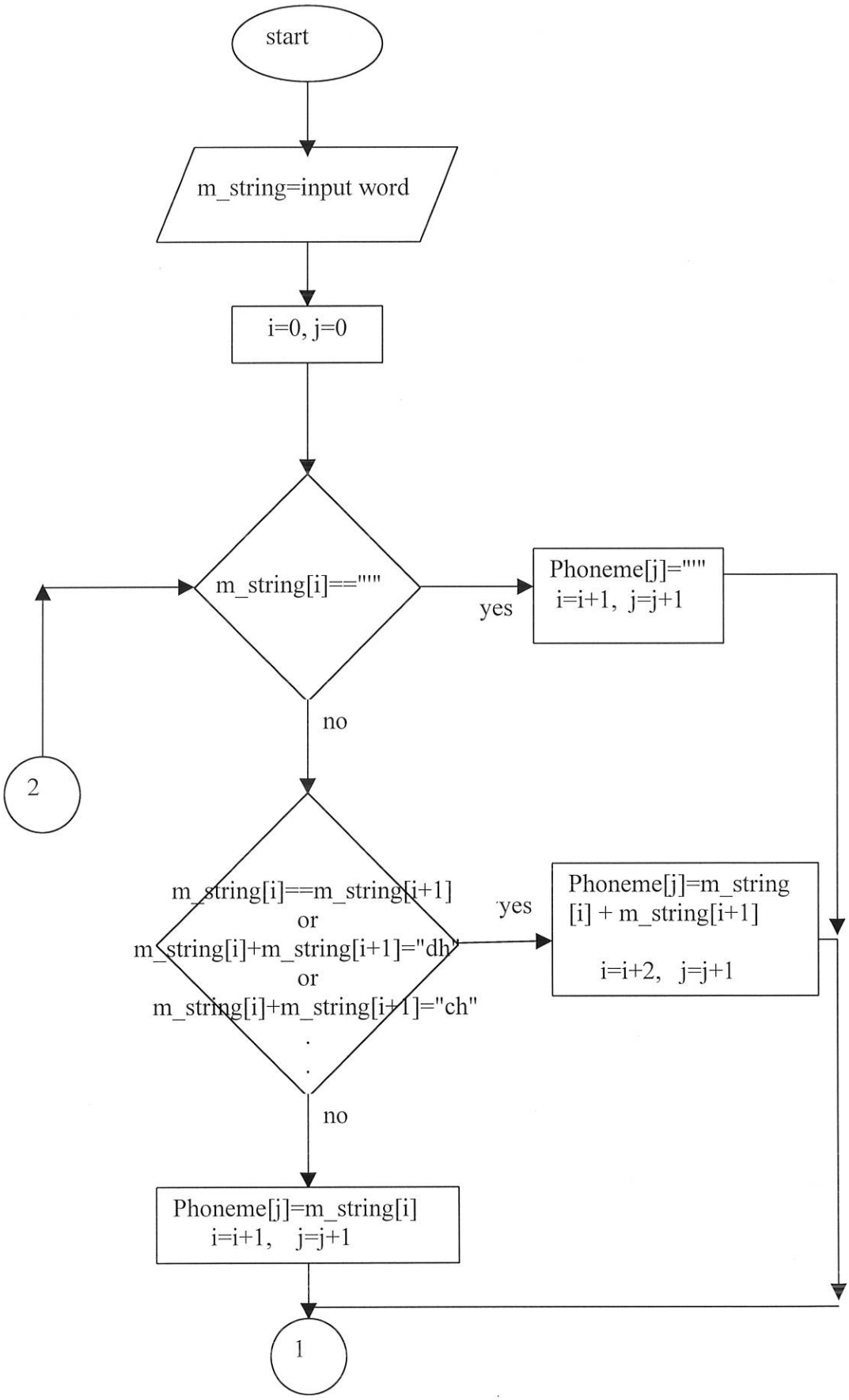
If the diphones extracted for one test data are also required for another, then the diphone segmented for the first test data is used. In this way, to test the fifteen test data sixty three diphones are extracted. Other than the test data words like *sirkaa* ' with you' could be uttered since all the necessary diphones are already extracted for other test words.

4.7 The algorithm

To build a prototype for *Afaan Oromoo* TTS, since the unit selected is diphone, a diphone-based algorithm is developed. The functionalities of the algorithm can be divided into four major categories. The first category converts the input word into a corresponding lowercase word. This category enables the input word to be in any case. The second category parses the input text into a sequence of phonemes. The third category changes the sequences of the phonemes into diphone sequences. The fourth category plays the sounds of the diphones. The algorithm in pseudo code and flowchart is given in figure 4.6 and 4.7 respectively

0. Get Word ,
1. Change the word into small case
2. Set m_string = input word, i=0, j=0
3. Set strnglength= length of word
4. If i< strnglength, go to 5 else go to 10
5. If m_string[i] =="'" go to 8 / apostroph checking*/*
6. If m_string[i]== m_string[i+1] or m_string[i]+m_string[i+1]="dh" or……. go to 9
7. Set Phoneme[j]=m_string[i], j= j+1, i=i+1, go to 4
8. Set Phoneme[j]="'", i=i+1, j= j+1 go to 4
9. Set phoneme[j]= m_string[i]+m_string[i+1], i=i+2 j=j+1 go to 4
10. Set diphone[0]= "0" + phoneme[0] / 0 represents silence*/*
11. Set k=1
12 If k<j go to 13 else go to 14
13. Set diphone[k]=phoneme[k-1]+phoneme[k], k=k+1, go to 12
14. Set diphone[k]=phoneme[k]+"0" / 0 for silence */*
15. Play the sounds of the diphone array

Figure 4.6 The Algorithm for Extracting Phonemes and Diphone


start
m_string=input word
i=0, j=0
m_string[i]==""
yes
Phoneme[j]=""
i=i+1, j=j+1
no
2
m_string[i]==m_string[i+1]
or
m_string[i]+m_string[i+1]="dh"
or
m_string[i]+m_string[i+1]="ch"
.
.
yes
Phoneme[j]=m_string
[i] + m_string[i+1]
i=i+2, j=j+1
no
Phoneme[j]=m_string[i]
i=i+1, j=j+1
1

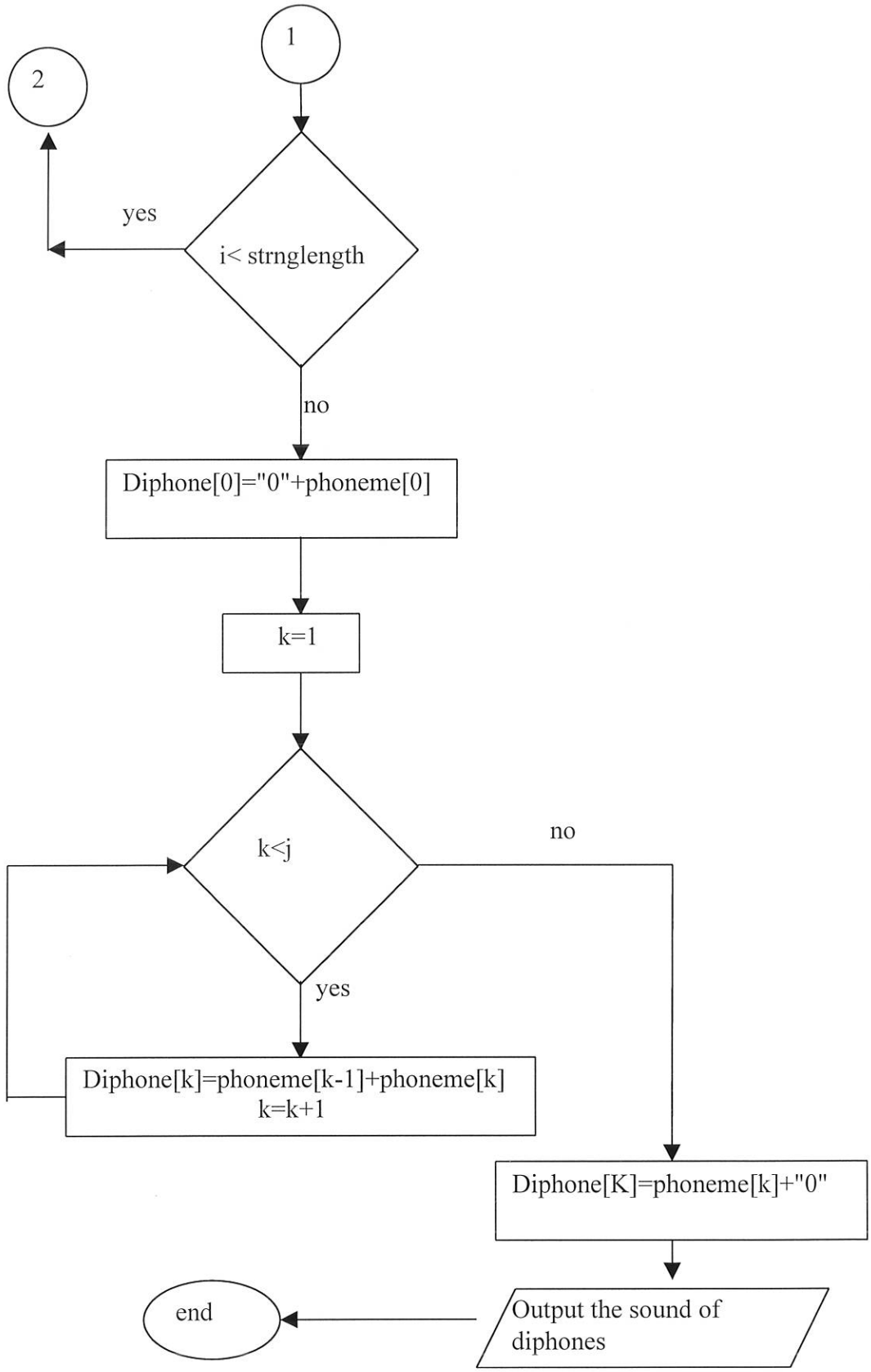


Figure 4.7 Flow Chart of the Algorithm

The input word is accepted and stored into a string variable m_string. The index i is used to access every character in the m_string variable. Two characters or a single character will then form a phoneme depending on whether the single or double character is able to represent a single sound. The phonemes are then stored in CstingArray variable called phoneme. The index j is used to build the array of phonemes. Two consecutive phonemes then will form diphones.

When the algorithm is implemented, in a number of steps, the modules convert the input word into sequences of phonemes, by decomposing the input word into

- A pause (')
- Double alphabets (aa, cc,…)
- Consonant clusters that represent a single sound (dh, ch, ts…)
- Any other alphabet if the above cases do not hold in a single pass

A pause searched is a silence that may be found within a word. In *Afaan Oromoo* as described in chapter three under the subtopic glottal stop, a pause can occur within a word.

This algorithm will finally enable diphones to be constructed from phonemes that are transcribed from the input word automatically. The diphone arrays will then be played to give the utterance of the word.

4.8 Test Result and Discussion

In this report, the correctly recognized words are counted to evaluate the TTS algorithm. Using an Open Rhyme Test, the test was made on two groups of listeners. Type-1 and Type-2.

Type-1 people were those who heard the utterance of the words by the TTS for the first time and Type-2 people where those who heard the utterance at least for three times before the test on different days. During testing if the listener wanted the utterance to be repeated, one more chance was given. If the listener is again not sure the utterance was considered as not recognized.

Using an Open Rhyme Test, approximately 43.33% of the test data are recognized by the people who heard the utterance of the words by the TTS for the first time (Type 1 people). But people who heard the TTS perform for three or more times (type 2 people) recognized 83.33 % of the test data. It was seen that words with few number of syllables were recognized easily when compared to words with more number of syllables. For example the words *shan* 'five', *sochii* 'movement' were consistently recognized. The table below summarizes the result of the test. Recognized words are shown with a mark √, while unrecognized words are marked as X.

Table 4.2 Test results on the test data

Word	Type 1 Persons				Type 2 Persons	
	A	B	C	D	E	F
gadaa 'democracy'	X	X	X	X	X	X
Sadarkaa 'grade'	X	X	√	X	√	√
Wayyaa 'cloth'	√	√	X	√	√	√
barnoota 'education'	√	X	X	X	√	X
barsiisu 'educate'	X	X	X	X	√	√
sochii 'movement'	√	√	√	X	√	√
nama 'person'	√	√	√	√	√	√
Waggaa 'year'	√	√	√	√	√	√
harmee 'mother'	√	√	X	X	X	X
shan 'five'	√	√	√	√	√	√
yeroo 'time'	X	√	X	√	√	√
naannoo 'around'	X	X	X	√	√	√
gargaarsa 'help'	X	X	X	X	√	√
Yokin 'or'	X	√	X	X	√	√
sirna 'order'	√	X	X	X	√	√

As there are many concatenation points for long words, the chance of having segmentation errors in these words increases. This is one of the reasons why long words are not recognized as accurately as the short ones.

Another source of confusion was noise. The listeners sometimes were not able to recognize the test words because of the lack of clear human sound from the TTS. This was a result of the inclusion of noise and sound bursts that came from recording problems.

Spectrally, also, there is a big gap between the similarity of a concatenated word and its natural form. The following diagrams, for example can be compared to see the difference

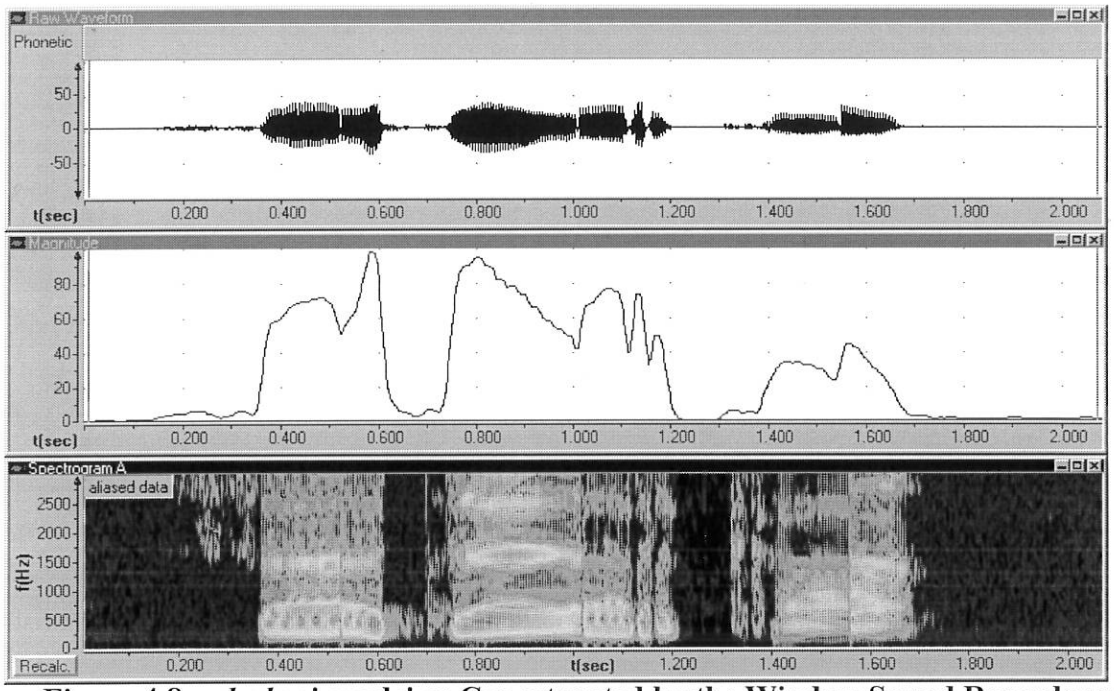


Figure.4.8 *sadarkaa*' grade' as Concatenated by the Window Sound Recorder using diphone units

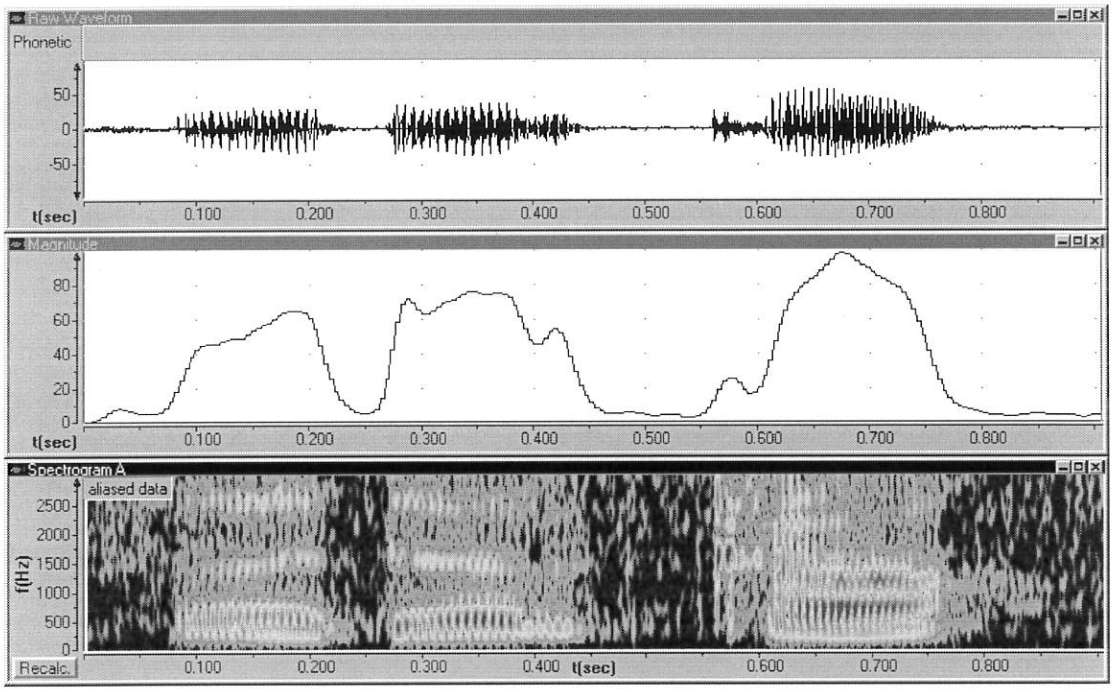


Figure 4.9 *sadarkaa* 'grade' in its Natural Form

There is a big difference in the energy contour of the word *sadarkaa* ' grade' in its natural form and concatenated waveform. This is possibly a result of the energy difference from

which the diphones were extracted. The other difference is the duration of the segments of the word *sadark*aa 'grade'. The word as spoken in the concatenated form takes longer duration time (approximately 150% of its natural form).

4.9 Chapter Summary

In this chapter, why the diphone units are chosen as a unit of synthesis for the *Afaan Oromoo* TTS has been explained. The type of the test conducted on the algorithm and its result were also given. Segmentation is another area that was highlighted in this chapter. Moreover, with the help of various figures that involve comparison of the output of the utterance from the TTS system and the utterance in its natural form is presented.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Making information universally accessible is a vision of information specialists. In this regard, to bridge the gap of the information need of the visually impaired people, a way of presenting information through speech has been and is being a research topic. A Text-To-Speech system is one development of this field. It is a system that tries to reads aloud any text as if it was spoken by somebody.

In view of this an attempt has been made in this study to develop a Text-to-Speech system for *Afaan Oromoo* words. The major issues for building a TTS system for the *Afaan Oromoo* language have been raised. Prosody matters of the language are briefly stated in chapter three for completeness purpose.

The writing system of the language is seen to be good for developing rules that relate alphabets or sequences of its alphabet to sounds. This was possible because seeing the orthography one does not need to use contexts to guess the pronunciation of words.

On the other hand, as seen from the test results, it was not easy to get high rate of intelligibility results. Burst of sounds, elongation of concatenated utterances, lack of smooth transitions between diphones, and noise that originate from recording problems are the major sources of errors that caused low rates of intelligibility for people of type-1. But once people became familiar to the utterance of the words by the TTS, the intelligibility rate was observed to increase. But to conclude that this is a result of acquaintance with the TTS system is

impossible. This is so because the test data used are few in number that, one could simply associate the utterance type with the word. Nevertheless, the result pauses a question whether becoming used to the TTS system increases the number of recognized words.

In relation to other TTS systems of English, the result achieved during the time allotted for this work is encouraging. The TTS systems shown in the table below are results that probably took years for the development of the systems.

Table 5.1 modified rhyme test results for various TTS systems (adopted from Donovan, 1996).

System	Error rate %
Natural speech	0.53
DECtalk 1.8 Paul	3.25
DECtalk 1.8 Betty	5.72
Prose 3.0	5.72
MITalk-79	7.0
Amiga	12.25
Infovox SA 101	12.5
TSI-Proto 1	17.25
Smoothtalker	27.22
Vortrax Typ'n Talk	27.44
Echo	37.56

5.2 Recommendation

As this work is an initial attempt to build a Text-To-Speech system for *Afaan Oromoo*, there is room for improvement. Lack of smooth transitions, clear human sound, and elongation problems of the TTS system can be solved by using

- *Spectral smoothening technique to smooth the concatenation boundaries of the diphones*. Errors that can arise from segmentation errors are usually smoothened by spectral smoothening techniques like LPC (Linear predictive coding) or other spectral smoothening algorithms. The linear predictive coding theory, for example, states that each speech sample can be approximately predicted as a linear combination of previous samples of speech (Donovan, 1996). Therefore in order to have smooth transition between diphones, spectral smoothning techniques should be used.

- *Recording corpus data in a sound laboratory*: Recording corpus data in sound laboratories has the advantage of avoiding any source of noise while recording the corpus data. In this work, having a good quality sound was one problem. Therefore using sound laboratories will give the chance to record good quality sounds.

- *Using continuous speech to record the corpus data:* Isolated word utterances are very slow as compared to their counterparts in a continuous speech. Although this work is based on isolated word speech synthesis, the elongation of word utterances that arise because of the use of the word in isolation in speech, adds more elongation period while using diphone units to synthesize the words. Therefore, even if, the words for testing purpose are isolated words, the corpus should be taken from continuous speech.

- *Using more numbers of test data:* Using more number of test data could actually show that becoming used to the utterance of the TTS has effect on the recognition test. If when using more number and similar test data, the intelligibility result for type 2 persons is still high, then familiarity with the TTS system can have an effect in the intelligibility result.

In an extension work of this study, a TTS system that includes text normalization and prosody generating components can be added. To handle prosody issues researches on parts of speech of the language has to be made.

References

Bender, M. L., Bowen, J. D., Cooper, R. L. and Ferguson, C. L. (1976). Two Cushitic languages. In M.L Bender et al. (Eds), *Language in Ethiopia*, (pp. 135- 148), London, Oxford University Press.

Central Statistics Office (1994). *The 1994 Population and Housing Census of Ethiopia, Results at Country Level, Vol. II.* Analytical Report, Addis Ababa.

Childers, D.G (2000). *Speech Processing and Synthesis Toolboxes*. New York: John Wiley & Sons, Inc.

Donovan, R. Edward. (1996). *TRAINABLE SPEECH SYNTHESIS.* PHD. Dissertation. Cambridge. Cambridge University. Available at URL: http:// citesser.nj.nec.com/donovan96trainable.html

Dutoit T.(1996). *A Short Introduction to Text-to-Speech Synthesis*. Available at URL: http://www.tcts.fpms.ac.be/synthesis/introtts.html

Dutoit T.(1996). *A Short Introduction to Text-to-Speech Synthesis*. Available at URL: http://www.tcts.fpms.ac.be/synthesis/introtts.html

Glass, R. James (1999). *Challenges For Spoken Dialogue Systems*. Available at URL: http://www.sls.lcs.mit.edu/sls/publications/1999/asru99-jrg.pdf

Gragg G. (1976). Oromo of Wellegga. In M. L. Beder (Eds.), *The Non-Semitic Languages of Ethiopia,* (pp. 166- 195). Michigan: African Studies Center, Michigan State University & Illinois University.

Gragg G. (1982). *Oromo Dictionary*. Published by African Studies Center: Michigan State University

Hallahan, W. William (1996). *DECtalk Software: Text-to-Speech Technology and Implementation.* Available at URL: http://www.digital.com/info/DTJK01.

Hertz, R. Susan, Young, J. Rebecca, and Hoskins, R. Steven (2000*). Space, Speed, Quality, And Flexibility: Advantages of Rule-Based Speech Synthesis*. Available at URL: http://www.eloq.com/tts.htm#space, speed quality, and flexibility.

Ince, A. Nejat (1992). Overview of Voice Communication and Speech Processing. In A. Nejat Ince(Ed.). *Digital Speech Processing: Speech Coding, Synthesis and Recognition.* Boston: Kluwer Acadamic Publishers.

Klatt, D. H. (1987). Review of text-to-speech conversion for English. *Journal of he Acoustical Society of America*, vol. 82, pp. 737-793.

Ladefoged, Peter (1975). *A Course in Phonetics*. New York: Harcourt Brace Jovanovich, Inc.

Laine Berhane (1998). *Text To Speech Synthesis of the Amharic Language*. M.Sc Thesis, Addis Ababa: Addis Ababa University.

Lehmann, W. P. (1972) *Descriptive Linguistics: An Introduction.* New York: Random House

Lloret, M. R (1997). Oromo Phonolgy. In Alan S. Kaye (Ed). *Phonologies of Asia and Africa.* pp 493-519

Mbrola. URL: http://tcts.fpms.ac.be/synthesis/mbrola.html

McTear, F. Michael(n.d). *Spoken Dialog Systems: Enabling Conversational user interface*. URL:http://www.infj.ulst.ac.uk/~cbdg23/survey/spoken_dialog_technology.html

Mekonnen, W. (2000). *Development of a Stemming Algorithm for Afaan Oromoo* Text. M.Sc. Thesis, Addis Ababa: Addis Ababa University.

Melba, G. (1988). *The Oromoo People and Oromia*. Khartoum, Sudan.

Mongham, Alex (n.d). *ACL1 Principles of Phonetics*. Available at URL: http://www.compapp.dcu.ie/~alex/CA162/PHONETICS/Processes.html

Morton, K. (1987). *The British Telecom Research Labratories Text-to-Speech Synthesis System*. URL: http://www. Essex.ac.uk/speech/archive/dbt//bt-1.html

Ng, Kenny (1998). *Survey of Data-Driven Approaches to Speech Synthesis*. Available at URL: http:// www.sls.lcs.mit.edu./kng/papers/areaexam.pdf.

O' Conner, D. J. (1973). *PHONETICS*. Middlesex: Penguin Books Ltd.

Oromosoft. URL: http://www.oromosoft.com

Rabiner, L. and Juang, B. Hwang (1993). *FUNDAMENTALS OF SPEECH RECOGNITION*. New Jersey: Prentice Hall Inc.

Rozak, Mike (n.d). *Talk to your computer and Have it answer back with the Microsoft Speech API.* MSDN: CD-ROM

Rubin, P. and Bateson, E. (1998). *Talking Heads: Speech Production.* Available at: URL: http://www.haskins.yale.edu/haskins/HEADS/MMSP/intro.html

Sutton, S., Cole, R., Villiers, J., Schalkwyk, J., Vermeulen, P., Macon, M., Yan, Y., Kaiser, E., Rundle, B., Shobaki, K., Hosom, P., Kain, A., Wouters, J., Massaro, D., Cohen, M. (n.d). *UNIVERSAL SPEECH TOOLS: THE CSLU TOOLKIT.* Available at; URL: http//www.cse.ogi.edu/cslu

Tilahun G. (1994). Afaan Oromoo, *Journal of Oromoo Studies.* URL: http://www.sas.upenn.edu/African.studies/Hornet/Afaan_Oromoo_19777.html.

Wardhaugh, R. (1977). *Introduction to Linguistics.* New York: McGraw-Hill

Williams, B. (n.d). *Diphone synthesis for Welsh.* Available at URL: http://www. Citeseer.nj.nec.com/51239.html.

Wolters, M.(1997). *Diphone-Based Text-to-Speech system for Scottish Gaelic.* PHD thesis. Available at: URL: http://www. Citeseer.nj.nec.com/309369.html.

Zue, V.(1997). Conversational Interfaces: Advances and Challenges. Available at URL: http://www.sls.lcs.mit.edu/sls/publications/1997/eurospeech97_zuekeynote.pdf

Appendix A

Header file of the implementation of the prototype

```
// TTSAO3Dlg.h : header file
//

#if
!defined(AFX_TTSAO3DLG_H__D7E69B63_22E4_11D5_8566_E9E5DE9E9041__INCLU
DED_)
#define
AFX_TTSAO3DLG_H__D7E69B63_22E4_11D5_8566_E9E5DE9E9041__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

/////////////////////////////////////////////////////////////////////////////
// CTTSAO3Dlg dialog

class CTTSAO3Dlg : public CDialog
{
// Construction
public:
        CTTSAO3Dlg(CWnd* pParent = NULL);    // standard constructor

// Dialog Data
        //{{AFX_DATA(CTTSAO3Dlg)
        enum { IDD = IDD_TTSAO3_DIALOG };
        CString         m_string;
        CString filename;
        CStringArray phoneme;
        CStringArray diphone;
```

```
        int j;
        int strglen;
        //}}AFX_DATA

        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CTTSAO3Dlg)
        protected:
        virtual void DoDataExchange(CDataExchange* pDX);      // DDX/DDV support
        //}}AFX_VIRTUAL

// Implementation
protected:
        HICON m_hIcon;

        // Generated message map functions
        //{{AFX_MSG(CTTSAO3Dlg)
        virtual BOOL OnInitDialog();
        afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
        afx_msg void OnPaint();
        afx_msg HCURSOR OnQueryDragIcon();
        afx_msg void OnparseTophoneme();
        afx_msg void ChangeCase();
        afx_msg void OnparseTodiphone();
        afx_msg void Ondhubadhuu();
        afx_msg void Ongadaa();
        afx_msg void Onsadarkaa();
        afx_msg void Onwayyaa();
        afx_msg void Onbarnoota();
        afx_msg void Onbarsiisu();
        afx_msg void Onsochii();
        afx_msg void Onnama();
        afx_msg void Onwaggaa();
        afx_msg void Onharmee();
```

```
        afx_msg void Onshan();
        afx_msg void Onyeroo();
        afx_msg void Onnaannoo();
        afx_msg void Ongargaarsa();
        afx_msg void Onyokin();
        afx_msg void Onsirna();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif                                                                    //
!defined(AFX_TTSAO3DLG_H__D7E69B63_22E4_11D5_8566_E9E5DE9E9041__INCLUDED_)
```

Appendix B

CPP file of the implemetation of the prototype

```
// TTSAO3Dlg.cpp : implementation file
//

#include "stdafx.h"
#include "TTSAO3.h"
#include "TTSAO3Dlg.h"
#include "mmsystem.h"
#include "afxcoll.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
        CAboutDlg();

// Dialog Data
        //{{AFX_DATA(CAboutDlg)
        enum { IDD = IDD_ABOUTBOX };
        //}}AFX_DATA

        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CAboutDlg)
        protected:
```

```
        virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
        //}}AFX_VIRTUAL

// Implementation
protected:
        //{{AFX_MSG(CAboutDlg)
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
        //{{AFX_DATA_INIT(CAboutDlg)
        //}}AFX_DATA_INIT
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);
        //{{AFX_DATA_MAP(CAboutDlg)
        //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
        //{{AFX_MSG_MAP(CAboutDlg)
                // No message handlers
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CTTSAO3Dlg dialog

CTTSAO3Dlg::CTTSAO3Dlg(CWnd* pParent /*=NULL*/)
```

```
    : CDialog(CTTSAO3Dlg::IDD, pParent)
{
    //{{AFX_DATA_INIT(CTTSAO3Dlg)
    m_string = _T("");
        //}}AFX_DATA_INIT
    // Note that LoadIcon does not require a subsequent DestroyIcon in Win32
    m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}

void CTTSAO3Dlg::DoDataExchange(CDataExchange* pDX)
{
    CDialog::DoDataExchange(pDX);
    //{{AFX_DATA_MAP(CTTSAO3Dlg)
    DDX_Text(pDX, IDm_string, m_string);
    DDV_MaxChars(pDX, m_string, 20);

    //}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CTTSAO3Dlg, CDialog)
    //{{AFX_MSG_MAP(CTTSAO3Dlg)
    ON_WM_SYSCOMMAND()
    ON_WM_PAINT()
    ON_WM_QUERYDRAGICON()
    ON_BN_CLICKED(IDdhubadhuu, Ondhubadhuu)
    ON_BN_CLICKED(IDC_gadaa, Ongadaa)
    ON_BN_CLICKED(IDC_sadarkaa, Onsadarkaa)
    ON_BN_CLICKED(IDC_wayyaa, Onwayyaa)
    ON_BN_CLICKED(IDC_barnoota, Onbarnoota)
    ON_BN_CLICKED(IDC_barsiisu, Onbarsiisu)
    ON_BN_CLICKED(IDC_sochii, Onsochii)
    ON_BN_CLICKED(IDC_nama, Onnama)
    ON_BN_CLICKED(IDC_waggaa, Onwaggaa)
```

```
        ON_BN_CLICKED(IDC_harmee, Onharmee)
        ON_BN_CLICKED(IDC_shan, Onshan)
        ON_BN_CLICKED(IDC_yeroo, Onyeroo)
        ON_BN_CLICKED(IDC_naannoo, Onnaannoo)
        ON_BN_CLICKED(IDC_gargaarsa, Ongargaarsa)
        ON_BN_CLICKED(IDC_yokin, Onyokin)
        ON_BN_CLICKED(IDC_sirna, Onsirna)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CTTSAO3Dlg message handlers

BOOL CTTSAO3Dlg::OnInitDialog()
{
        CDialog::OnInitDialog();

        // Add "About..." menu item to system menu.

        // IDM_ABOUTBOX must be in the system command range.
        ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
        ASSERT(IDM_ABOUTBOX < 0xF000);

        CMenu* pSysMenu = GetSystemMenu(FALSE);
        if (pSysMenu != NULL)
        {
                CString strAboutMenu;
                strAboutMenu.LoadString(IDS_ABOUTBOX);
                if (!strAboutMenu.IsEmpty())
                {
                        pSysMenu->AppendMenu(MF_SEPARATOR);
                        pSysMenu->AppendMenu(MF_STRING,          IDM_ABOUTBOX,
strAboutMenu);
```

```
            }
        }

        // Set the icon for this dialog.  The framework does this automatically
        //  when the application's main window is not a dialog
        SetIcon(m_hIcon, TRUE);                 // Set big icon
        SetIcon(m_hIcon, FALSE);            // Set small icon

        // TODO: Add extra initialization here

        return TRUE;  // return TRUE  unless you set the focus to a control
}

void CTTSAO3Dlg::OnSysCommand(UINT nID, LPARAM lParam)
{
        if ((nID & 0xFFF0) == IDM_ABOUTBOX)
        {
                CAboutDlg dlgAbout;
                dlgAbout.DoModal();
        }
        else
        {
                CDialog::OnSysCommand(nID, lParam);
        }
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CTTSAO3Dlg::OnPaint()
{
        if (IsIconic())
```

```
        {
                CPaintDC dc(this); // device context for painting

                SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);

                // Center icon in client rectangle
                int cxIcon = GetSystemMetrics(SM_CXICON);
                int cyIcon = GetSystemMetrics(SM_CYICON);
                CRect rect;
                GetClientRect(&rect);
                int x = (rect.Width() - cxIcon + 1) / 2;
                int y = (rect.Height() - cyIcon + 1) / 2;

                // Draw the icon
                dc.DrawIcon(x, y, m_hIcon);
        }
        else
        {
                CDialog::OnPaint();
        }
}

// The system calls this to obtain the cursor to display while the user drags
//  the minimized window.
HCURSOR CTTSAO3Dlg::OnQueryDragIcon()
{
        return (HCURSOR) m_hIcon;
}

 void CTTSAO3Dlg::ChangeCase() // changes the case of the input word to small case
{
        //TODO: Add your control notification handler code here
        m_string.MakeLower();
```

```
        strglen=m_string.GetLength(); //strglen holds the length of the input word
}

void CTTSAO3Dlg::OnparseTophoneme() /*decomposes the input word into phonemes */
{
        //TODO: Add your control notification handler code here
        int i=0;
        j=0;
        m_string.Insert(strglen,'0');// insers '0' to identify the end point of the input //word
        phoneme.RemoveAll();// to delete all phoneme array data
        phoneme.SetSize(15);// memory allocation
        do
        {
                if (CString(m_string[i])=="")// checking for the presence of silence
                {
                        phoneme[j]='0';
                        i=i+1;
                }
                else if ((m_string[i] == m_string[i+1])||(CString(m_string[i])+m_string[i+1]=="ch")||(CString(m_string[i])+m_string[i+1]=="dh")||(CString(m_string[i])+m_string[i+1]=="ny")||(CString(m_string[i])+m_string[i+1]=="ph")||(CString(m_string[i])+m_string[i+1]=="sh")||(CString(m_string[i])+m_string[i+1]=="ts")) // checking for double consonsnts that represent a sound
                {
                        phoneme[j]= CString(m_string[i])+ m_string[i+1];
                        i=i+2;
                }
                else
                {
                 phoneme[j]=m_string[i];
                 i=i+1;
                }
          j=j+1;
```

```
        }
        while (i<strglen);
}
void CTTSAO3Dlg::OnparseTodiphone() // constructs diphones from phonemes
{
        // TODO: Add your control notification handler code here
        int i;
        diphone.RemoveAll();
        diphone.SetSize(15);
        diphone[0]='0' + phoneme[0];
        for (i=1; i<j; i++)
                diphone[i]= phoneme[i-1]+phoneme[i];
        diphone[i]= phoneme[i-1]+'0';
}



void CTTSAO3Dlg::Ondhubadhuu()
{
        // TODO: Add your control notification handler code here
        UpdateData(TRUE);
        CTTSAO3Dlg::ChangeCase();
        CTTSAO3Dlg::OnparseTophoneme();
        CTTSAO3Dlg::OnparseTodiphone();
        for (int i=0; i<=j; i++)
        {
                filename="c:\\morka\\"+diphone[i]+".wav";
                sndPlaySound(filename, SND_SYNC);
        }
}

void CTTSAO3Dlg::Ongadaa()
{
```

```
	// TODO: Add your control notification handler code here
	m_string="gadaa";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onsadarkaa()
{
	// TODO: Add your control notification handler code here
	m_string="sadarkaa";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onwayyaa()
{
	// TODO: Add your control notification handler code here
	m_string="wayyaa";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onbarnoota()
{
	// TODO: Add your control notification handler code here
	m_string="barnoota";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onbarsiisu()
{
	// TODO: Add your control notification handler code here
	m_string="barsiisu";
	UpdateData(FALSE);
}
```

```
void CTTSAO3Dlg::Onsochii()
{
	// TODO: Add your control notification handler code here
	m_string="sochii";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onnama()
{
	// TODO: Add your control notification handler code here
	m_string="nama";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onwaggaa()
{
	// TODO: Add your control notification handler code here
	m_string="waggaa";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onharmee()
{
	// TODO: Add your control notification handler code here
	m_string="harmee";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onshan()
{
	// TODO: Add your control notification handler code here
	m_string="shan";
	UpdateData(FALSE);
```

```
}

void CTTSAO3Dlg::Onyeroo()
{
	// TODO: Add your control notification handler code here
	m_string="yeroo";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onnaannoo()
{
	// TODO: Add your control notification handler code here
	m_string="naannoo";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Ongargaarsa()
{
	// TODO: Add your control notification handler code here
	m_string="gargaarsa";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onyokin()
{
	// TODO: Add your control notification handler code here
	m_string="yokin";
	UpdateData(FALSE);
}

void CTTSAO3Dlg::Onsirna()
{
	// TODO: Add your control notification handler code here
```

```
        m_string="sirna";
        UpdateData(FALSE);
}
```

Appendix C

Corpus data

Word	Diphones extracted				
achii	ch-ii				
afaan	n-sil				
amanamee	a-m	a-n	m-ee		
arsii	r-s				
baarneexa	r-n				
basii	s-ii				
bayyaan	a-yy				
birmaduu	Sil-b	r-m	m-a	a-d	
biyyaa	yy-aa				
ciisee	ii-s				
daakaa	d-aa	k-aa			
daannisa	aa-nn				
ergaa	e-r	r-g			
geba	Sil-g	b-a	a-sil		
gocha	o-ch				
goota	oo-t				
harka	h-a	r-k			
herrega	Sil-h	g-a			
hinjiru	i-n	u-sil			
irbaata	i-r	t-a			
jaarsa	aa-r				
kiyyoo	k-i				
niitii	Sil-n	ii-sil			
noonnoo	n-oo				
okaati	o-k				
rooba	r-oo				
saggoo	a-gg				
saree	s-a	a-r	ee-sil		
seenaa	Sil-s	n-aa	aa-sil		

shakka	Sh-a				
shinoo	Sil-sh	oo-sil			
sigigaacha	s-i	g-aa			
sodaa	s-o				
suggaa	s-u	gg-aa			
wannoo	w-a	nn-oo			
wiirtuu	Sil-w				
yaada	Sil-y	d-a			
yemmuu	y-e				
yona	y-o	n-a			

DECLARATION

This thesis is my original work and has not been submitted for a degree in any other university.

Morka Mekonnen Wolde

June 2001

This thesis has been submitted for examination with our approval as university advisors

Workeshet Lamenew

Wakshum Mekonnen