Design of an Advanced and Fluent Sesotho Text-to-Speech System through Intonation

Prepared by: Lehlohonolo Mohasi

Supervised by: Professor Daniel Mashao

University of Cape Town
Department of Electrical Engineering
May 2006

This dissertation is submitted to the University of Cape Town in fulfillment of the academic requirements for the Degree of Master of Science in Engineering.

Declaration

I declare that this dissertation is my own work. It is being submitted for the degree of Master of Science in Engineering at the University of Cape Town. To my knowledge, it has not been submitted before for any degree or examination in any other university.

Signature of Author . . . . . Signed by candidate Signature removed . . . . . .

Lehlohonolo Mohasi

May 2006

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Daniel Mashao, for the assistance and guidance he provided throughout the execution of this project. I have acquired many skills under his supervision. I am also grateful to my fellow friends and colleagues in the STAR and CRG research groups. I had a wonderful time with you guys and I am going to miss those moments of laughter and fun we had. My thanks also go to another STAR, Francois Rousseau, for his unselfish assistance and allowing me to use his questionnaire. I am just as grateful to my family for their relentless support and encouragement throughout my studies. I would not be where I am today if it were not for them. My special thanks also go to my special and true friend, Mxolisi Zimu, for all the encouragement and patience during this time. Most important of all, I am grateful to God Almighty, who has been holding my hand all the time. He deserves all the glory, honor and praise.

Abstract

Speech output quality of text-to-speech (TTS) systems has dramatically improved in the past decade. Text-to-speech systems that are based on limited domain techniques produce speech which sounds natural or is close to a human voice. A limitation of these systems is that they can only synthesize words in their database. On the other hand, open vocabulary systems are flexible in that they can synthesize not only words in their database, but words out of their vocabulary as well. In order to get an advanced system that incorporates the two criteria, i.e., naturalness and flexibility, a hybrid system of both techniques is necessary. Even though the hybrid system gives natural speech output and is flexible, it has a weakness of not producing fluent speech. This is due to glitches or discontinuities, which are sudden irregularities in the speech.

This project is based on an advanced TTS system built by Francois Rousseau. His system is considered advanced as it incorporates the following features: understandability, flexibility, pleasantness, and naturalness. Rousseau's system is used as a baseline on which we intend to add fluency and improve naturalness of the system.

The main objective of this thesis is to implement techniques that can mask the discontinuities of two Sesotho text-to-speech hybrid systems so that the speech output sounds more natural and fluent. This was implemented by using intonation modeling techniques, duration and fundamental frequency (F0), on unit selection hybrid systems. The pitch contours of the original and generated modules were compared for improvement. Subjective listening tests were carried out to assess the speech output quality of the old systems (with no prosody) and the new systems. Evaluation was based on the Mean Opinion Score (MOS) questionnaire. Overall, the new systems gave an average MOS score of 3.7. From these results, it was concluded that intonation implementation did improve the naturalness and fluency of the speech output, though not to a desired level. Therefore, for future work, it is recommended that more studies should be done on phone duration analysis and modeling of Sesotho. An investigation should also be carried out on stress characteristics of the Sesotho language and energy differences between phones.

Table of Contents

List of Tables

List of Figures

List of Acronyms and Abbreviations

ASCII - American Standard Code for Information Interchange
CART – Classification And Regression Tree
CC – consonant consonant
CHATR – Generic speech synthesis system developed at ATR (Advanced Telecommunications Research)
CMU – Carnegie Mellon University
CPU – Central Processing Unit
DTW – Dynamic Time Warping
EGG – electroglottograph
ESTDIR – Edinburgh Speech Tools directory
F0 – Fundamental Frequency
FESTVOXDIR - Festvox directory
HiF0 – high fundamental frequency
HLT – Human Language Technologies
HMM – Hidden Markov Model
HTK – HMM Toolkit
HTML – Hyper Text Markup Language
Hz – Hertz
IP – (big) intonational phrase
ip – (small) intonational phrase
IPA – International Phonetic Alphabet
LPC – Linear Predictive Coding
LR – Linear Regression
LS – Sesotho of Lesotho
MBROLA – Multiband resynthesis overlap add
MELCEP – Mel Cepstral
MFCC – Mel Frequency Cepstral Coefficients
MOS – Mean Opinion Score
NIST – National Institute of Standards and Technology
NN – Neural Networks
OALD – Oxford Advanced Learner's Dictionary
OOV – out of vocabulary
PDA – pitch determination algorithm
POS – part of speech
POSLEX – Part of speech lexicons
PSOLA - Pitch Synchronous OverLap and Add
RMSE – root mean squared error
SAS – South African Sesotho

SIOD – Scheme in One Defun
SLH – Strict Layer Hypothesis
SMS – Short Message Service
ToBI – Tone and Break Index
TTS – text-to-speech
UCT – University of Cape Town
VC – vowel consonant
VOCODER – Voice Coder
VV – vowel vowel

Chapter 1

1. Introduction

Much research has been put in text-to-speech technology [4, 9, 13, 19, and 45] in order to improve voice quality of text-to-speech (TTS) systems. Different techniques have been used with concatenative synthesis being the most popular. Concatenative synthesis strings together segments of recorded speech and gives the most natural sounding speech. However, natural variation in speech and automated techniques for segmenting waveforms sometimes result in audible glitches in the output, detracting from the naturalness. There is still therefore, an ongoing research on different techniques [14, 48, and 50] to tackle the problem.

A stage has been reached where intonation is becoming the main obstacle to achieving natural sounding synthesized speech. This thesis looks at ways of improving naturalness and fluency of a Sesotho advanced TTS system. The system is a hybrid of two unit selection synthesis methods, a limited domain synthesis and an open vocabulary synthesis. The ultimate goal is to smooth or mask audible glitches of the output speech through intonation. This involves getting control of the intonation aspects and modifying them to the desired level of our system. The main aspects of intonation are pitch (F0), rhythm and stress; they give speech the naturalness of a human speaker and also enhance comprehension.

1.1 A brief overview of Text-to-Speech Synthesis

Research in speech technology has produced text-to-speech synthesizers with very high intelligibility but the sound quality and naturalness still remain a major problem. However, the quality of most TTS products has reached an adequate level for several applications. With more investigation going into TTS technology and aspects affecting the speech output, near-perfect solution will be reached.

The text-to-speech synthesis procedure consists of two main phases: text analysis and generation of speech waveforms. In text analysis, input text is transcribed into a phonetic or some other linguistic representation, whereas in speech waveform generation, an acoustic output is produced from this phonetic and prosodic information. These two phases are usually known as high- and low-level synthesis respectively. A simplified version of the procedure is presented in Figure 1-1. The input text might be, for example, data from a word processor, standard ASCII

from e-mail, a mobile text-message, or scanned text from a newspaper. A character string is then preprocessed and analyzed into phonetic representation which is usually a string of phonemes with some additional information for correct intonation, duration, and stress. Speech sound is finally generated with the low-level synthesizer by the information obtained from the high-level synthesizer.

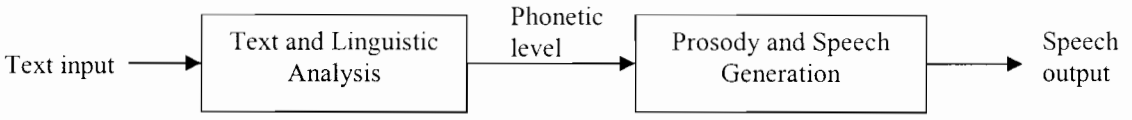


Figure 1-1: **A simple text-to-speech synthesis procedure [3]**

The most popular and simplest way to produce synthetic speech is to play long pre-recorded samples of natural speech, such as single words or sentences. This concatenation method produces a natural-sounding speech output. Due to its limited vocabulary, this synthesis technique is usually used for making announcements. However, a database of all words and common names in the world cannot be created. Therefore, for unrestricted text-to-speech, shorter pieces of speech signal such as syllables, phonemes, diphones, or even shorter segments are the best option to use.

Another method to produce synthetic speech is formant synthesis which is based on the source-filter-model of speech production depicted in Figure 1-2. The method models only a sound source and formant frequencies, not any physical characteristics of the vocal tract [3]. The excitation signal could be either voiced with fundamental frequency (F0) or unvoiced noise. A mixed excitation of these two may also be used for voiced consonants and some aspiration sounds. The excitation is then gained and filtered with a vocal tract filter which is constructed of resonators similar to the formants of natural speech.

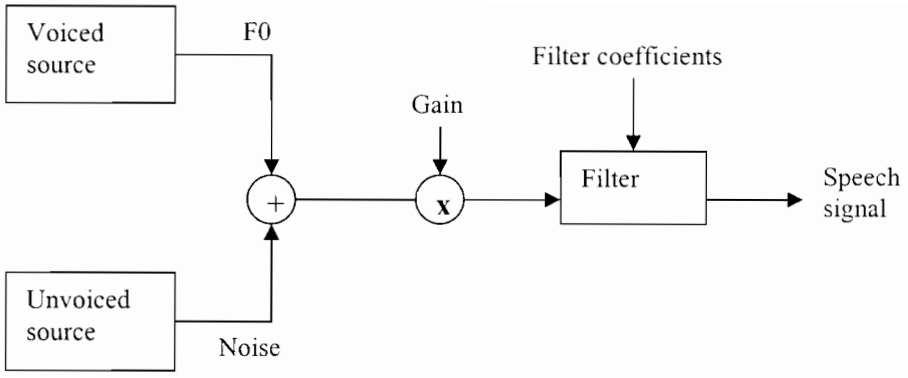


Figure 1-2: Source-filter model of speech [3]

In theory, the most accurate method to generate artificial speech is to model a human speech production system directly [60]. This method, called articulatory synthesis, typically involves models of human articulators and vocal cords. The articulators are usually modeled with a set of area functions of small tube sections. The vocal cord model is used to generate an appropriate excitation signal, which may be for example, a two-mass model with two vertically moving masses [68]. Articulatory synthesis holds a promise of high-quality synthesized speech, but due to its complexity, the potential has not been realized yet.

Each of the synthesis techniques has its advantages and disadvantages. With concatenative and formant synthesis techniques, some promising results have been achieved, but also articulatory synthesis may arise as a potential method in the future. Different synthesis techniques and their algorithms are discussed more closely in Chapter 2.

1.1.1 Background to Investigation of Text-to-Speech

Remarkable advances in text-to-speech methodologies such as unit selection techniques and signal processing algorithms have greatly contributed to improving the overall speech quality of text-to-speech systems. In spite of the improvements, the fact remains that the naturalness and expressiveness of synthetic speech is, in general, far from recorded speech [14]. Therefore, pre-recorded speech prompts are still used in various speech applications, particularly when their speech output messages can be covered with relatively limited vocabularies, such as voice response systems via telephone, car navigation systems, speech dialogue systems for robots, etc.

At the same time, even in such applications, text-to-speech is commonly used to support variable phrases or out-of-domain words. Simple concatenations of synthetic and recorded speech do not, however, provide the best solution in the overall speech quality. Synthetic speech when mixed with recorded speech is sometimes heard as being even worse than when used independently, because the synthetic speech is directly compared with its ideal version, the recorded speech. Therefore, some strategies are needed to maximize the overall speech quality when mixing synthetic and recorded speech.

Given this background, there have been several studies [including 12, 14, 18, 46, 58] trying to optimize the use of recorded speech in a generic text-to-speech framework in order to improve the naturalness of the overall speech quality. In previous studies, even combined with text-to-speech technology, the improvement focus is still limited mainly to fixed phrases or sentences, which correspond to recorded speech portions. In other words, the synthetic speech portions are still treated as almost independent and standard text-to-speech sections, and the effect of interactions with the recorded speech is not seriously considered, particularly as it affects intonation. However, actual intonation patterns of individual speech utterances are affected by degrees of emphasis or emotion and by conveyed intentions of the speaker, as well as by fluctuations of utterances themselves. Considering these various behaviors observed in actual utterances, the fixed intonation of synthetic speech predicted using only linguistic factors might not always be appropriate for combination with really expressive recorded speech.

1.1.2 The Role of Intonation in Speech Synthesis

Intonation plays a significant role in human speech communication process. Intonation is realized mainly by varying fundamental frequency (F0), and the perceptual correlation of F0 is pitch. A human hearing system is very sensitive to variations of pitch, and this provides a physical foundation of effective intonation perception.

From the speaker's point of view, intonation is used to convey pragmatic information, emotion, etc. Syntactically similar sentences with different intonation patterns can convey dramatically distinct information. Sun [25] indicates that from the listener's point of view, intonation plays an important role in: (1) segmenting utterances; (2) resolving syntactic ambiguity; (3) serving as a continuity guide in noisy environments; and (4) providing cues to the state of the speaker. In addition to the above functions, pitch contours also carry lexical meaning in tone languages, such as Sesotho.

As intonation is such an important factor in the human speech communication process, it is thus crucial to understand how it fits in speech technology applications such as speech synthesis and speech recognition. In speech synthesis, the naturalness of intonation directly affects the overall

quality of synthetic speech. Intonation modeling has thus been one of the central areas in speech synthesis research [46, 47, 52, 54, 57, and 58]. Figure 1-3 illustrates the schematic architecture of a speech synthesis system, in which intonation prediction is an intermediate step between text analysis and waveform generation. Intonation is also valuable in speech recognition systems as shown in Figure 1-4. In speech recognition, F0 can be combined with spectral information to identify individual speech units, such as a phoneme, word, etc. Intonation can also be applied at later stages. Certain prosodic events, such as pitch accent and prosodic boundaries, can be predicted from acoustic utterances using features like F0, duration, and energy [25]. A speech recognition system can use these symbolic prosodic markers to identify some syntactic, semantic, or emotional information.

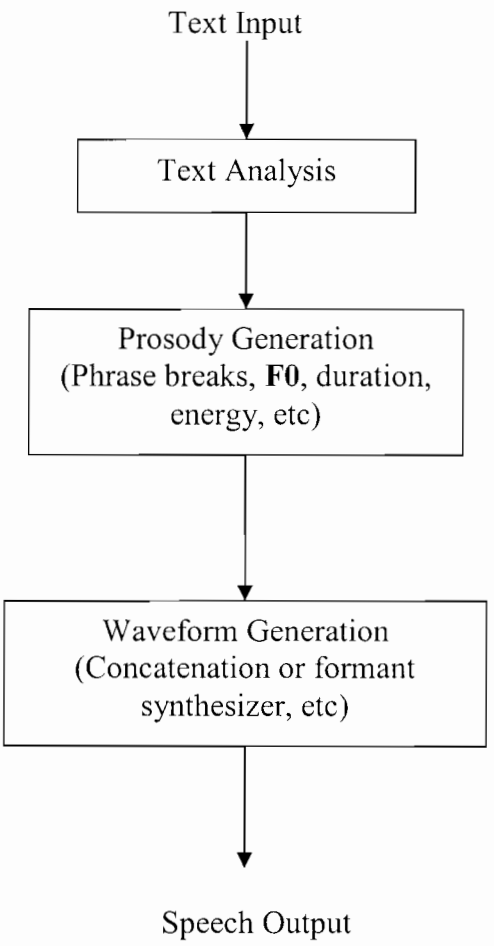


Figure 1-3: Simplified block diagram for a speech synthesis system [25]

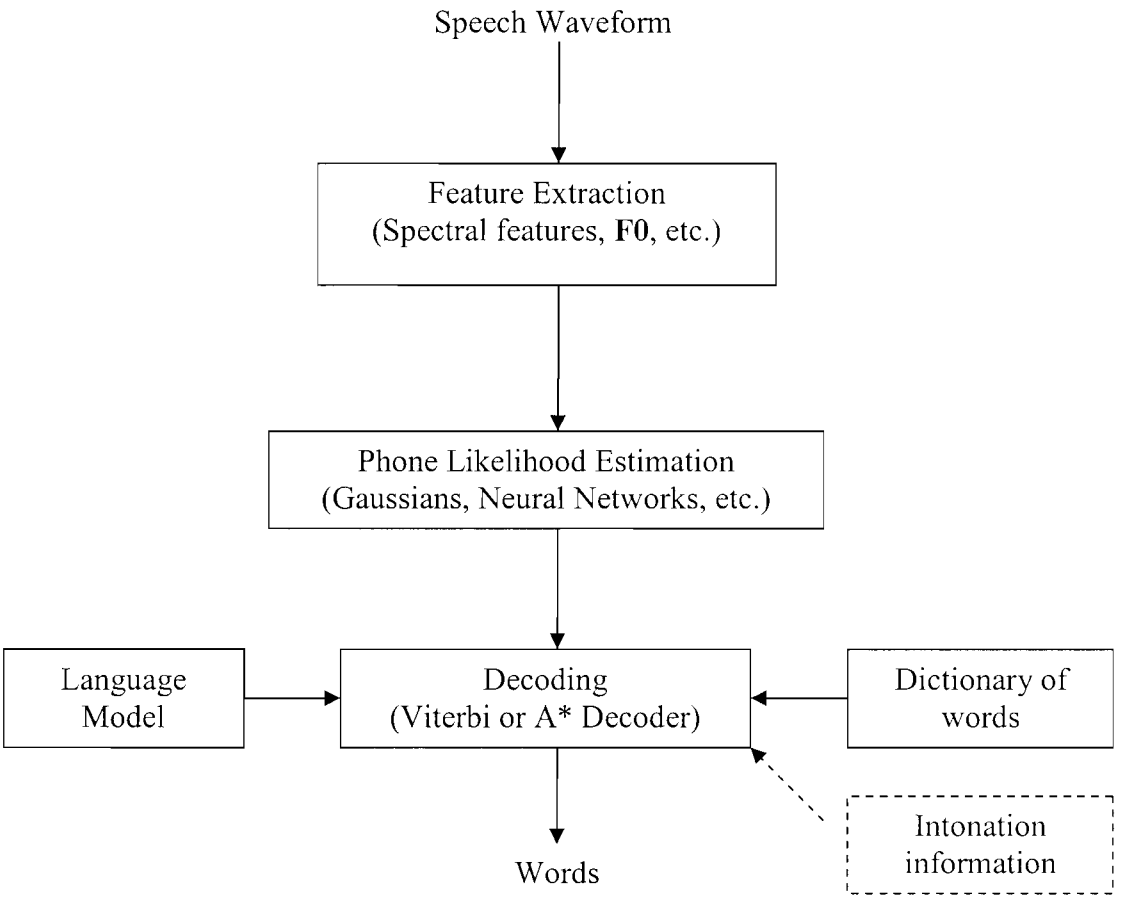


Figure 1-4: Simplified block diagram for a speech recognition system [25]

1.2 Overview of the Sesotho Language

This section introduces the language of study in this thesis, Sesotho. The background of the language and its phonetic structure are discussed.

1.2.1 Background on Sesotho Language

Sesotho, also known as Southern Sotho, Sotho, or Sesuthu, is a language spoken in Southern Africa. Sesotho is generally classified as a Bantu language, belonging to the Niger-Congo language family [6]. It is most closely related to two other languages in the Sotho language group, Setswana and Northern Sotho (Sesotho sa Leboa). Sesotho is one of the eleven official languages of South Africa, and one of the two official languages of Lesotho. According to the 2001 census, approximately 3.5 million people in South Africa speak Sesotho as their first

language. In Lesotho, Sesotho is spoken by close to 85% of the population (1993 census) [6]. Figure 1-5 (modification of graph taken from [73]) depicts the areas where Sesotho language population is dominant in both Lesotho and South Africa. In South Africa, Sesotho is more dominant in the Free State province. Smaller percentages of Sesotho speakers are found in Eastern Cape, Gauteng, and Mpumalanga provinces.

Figure 1-5: **Sesotho language in Lesotho and South Africa**

Sesotho is one of the first African languages to be reduced to writing [67]. The written form for Sesotho was introduced by missionaries, Casalis and Arbousset of the Paris Evangelical Mission who arrived in Lesotho in 1833. According to [67], the written form was originally based on the Tlokwa dialect, which is now mostly based on the Kwena and Fokeng dialects.

Even though Sesotho spoken in Lesotho (LS) and South Africa (SAS) has the same dialect, the two countries use slightly different orthographies for the language. For instance, South African Sesotho uses the semi-vowels **w** and **y** for the vowels ***o*** and ***e*** and **di** and **du** for ***li*** and ***lu***. More examples are shown in Table 1-1. In carrying out the design for this project, the two writing systems were taken into consideration when compiling data for recording.

Table 1-1: Writing systems for Sesotho language in Lesotho and South Africa

LS	SAS	Example	English Meaning
ch	tjh	se**ch**aba vs se**tj**haba	nation
kh	kg	**kh**omo vs **kg**omo	cow
ts'	tsh	**ts'**epo vs **tsh**epo	hope

1.2.2 Motivation for Sesotho TTS system

The digital divide, especially in the third-world countries, contributes to people having fear of using any technology-related designs. One reason for this could be that most people are illiterate. Literacy, or lack of it, plays an important part in the lack of motivation to try new things. For instance, people who cannot read properly are not going to be very anxious to search for information which is mostly presented as written text. According to Underwood [71], the literacy rate in South Africa, based on the population aged fifteen and above, is approximately 85%. Underwood further explains that this figure might not be revealing big differences between rural and metropolitan areas. The study made by South Africa Ministry for Welfare and Population Development in 1997 [71] showed rates of illiteracy to vary between 27% in metropolitan areas to 50% in rural areas.

The other reason could be due to not being proficient in English. One would have to understand English in order to make use of any technological system as most of them operate in English. In addition, ability to speak a language does not guarantee high reading proficiency in the same language. This is of major concern as not everyone will benefit from the technology due to their lack of language understanding. It is therefore, essential that technology designers meet the needs of the disadvantaged by introducing systems that everyone will be able to use with their limited literacy skills.

The aim of integrating Sesotho as one of the official languages into the text-to-speech technology therefore, is to accommodate people who speak the language and eliminate the fear of using the technology. In this way, an initiative is being made to take into account diverse language groupings within the country. As depicted in Figure 1-6 (based on information obtained from [72]), eleven languages are recognized in South Africa, but their use is unevenly distributed. As this figure shows, isiZulu has the highest population of 25%, followed by isiXhosa at 18%. English and Sesotho have the same population of percentage of 8%. Recognition of all languages will enable people to use, for example, governmental services where they can communicate with a machine in their language on the services they require. The machine will also reply in the language they prefer and understand. This way, people will be encouraged to use such services regardless of their computer- or language literacy.

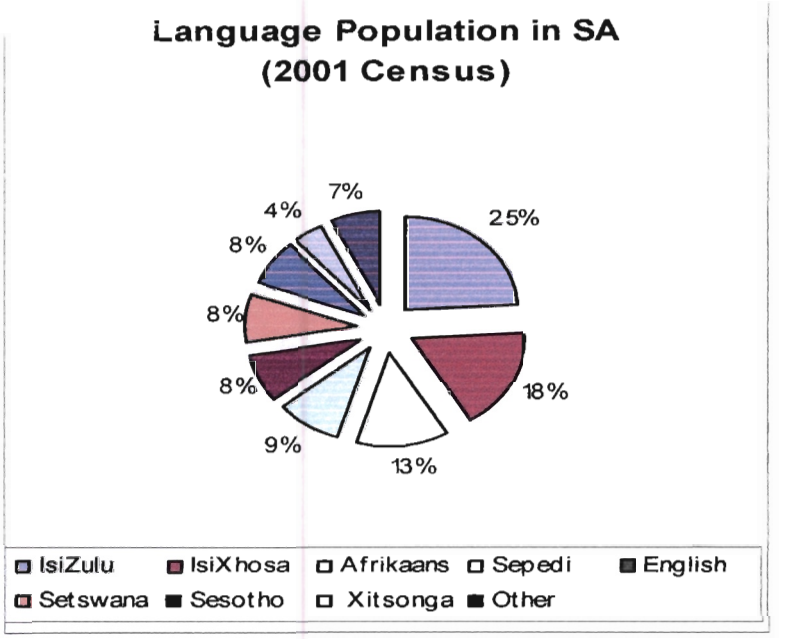


Figure 1-6: **Language population in South Africa**

1.2.3 Intonation in Sesotho

The prosodic feature, intonation, is sometimes closely related to tone. A tonal language, therefore, is one that makes a particular use of pitch as an element of speech. Sesotho is one such language. In Sesotho, tone is used in two particular ways. First, it is used to differentiate or distinguish meanings of words that are spelt alike but pronounced differently. Tone is then said to be significant. There are only two tonal values or tonemes in Sesotho and they contrast with each other. They are a high (-) and a low (_) toneme. Each of these may have non-significant varieties of a raised, lowered, level or falling type. A toneme is high only if it is higher than its neighbor(s) or low only if it is lower than its neighbor(s). It is the relative, and not the absolute pitch of tonemes that is significant. In other words, what is important is "*the relationship of the pitch on one syllable to that of a neighboring syllable, and not its relation to an absolute scale*" [8].

Tones in Sesotho can be categorized as characteristic, semantic and grammatical.

Characteristic tone - This is an inherent tone for the syllables of each Sesotho word.

E.g. Katse [- _] *cat*
Ntja [_ -] *dog*

Semantic tone – This tone distinguishes between words which have the same syllables/phonemes but have different meanings depending on the tone used.

E.g. Ho aka [_--] *to kiss*
Ho aka [___] *to lie*

Joang [_-] *grass*
Joang [- _] *how?*

Grammatical tone – This is when two similar sounding phrases have two very different meanings mainly due to a difference in tone or one or more words or concords.

E.g. Ke ngoana oa hao [_ - _ _ - _] *I am your child*
Ke ngoana oa hao [- - _ _ - _] *He/she is your child*

O mobe [_ _ -] *You are ugly*
O mobe [- _ -] *He/she is bad*

It should be noted that when grammatical tone is used, the tone of the significant word influences the relative pitch of the rest of the phrase, although the tones of other words remain intact.

1.2.4 Sesotho Phonetic Structure

In most languages the written text does not correspond to its pronunciation therefore, in order to describe correct pronunciation, some kind of synthetic presentation is needed. Every language has a different phonetic alphabet and a different set of possible phonemes and their combinations. The number of phonetic symbols is between 20 and 60 in each language [3, 60]. A set of phonemes can be defined as the minimum number of symbols needed to describe every possible word in a language. In Sesotho, there are about 50 phonemes [5]. Due to complexity and different kinds of definitions, the number of phonemes in most languages cannot be defined exactly.

A phonetic alphabet is usually divided in two main categories, vowels and consonants. Vowels are always voiced sounds and they are produced with the vocal cords in vibration, while consonants may be either voiced or unvoiced. Vowels have considerably higher amplitude than consonants and they are also more stable and easier to analyze and describe acoustically. Because consonants involve very rapid changes, they are more difficult to synthesize properly. Articulatory phonetics in Sesotho are described below.

1.2.4.1 Articulatory Phonetics in Sesotho

There is no clear agreement between authors of Sesotho books about the number of vowels in Sesotho. According to Gama [8], Sesotho has eleven vowel sounds or phones. They are grouped into seven vowel phonemes, i.e. "*a group of sounds consisting of one main member together with others which take its place in certain sound-groups.*" The phonemes are ***a***, ***i***, ***u***, two kinds of ***e*** and ***o*** each. On the other hand, Zuzu as referenced by [8] and Paroz [15] say there are nine vowels in Sesotho: ***a***, ***i***, ***u***, three kinds of ***e***, and three kinds of ***o***. This report is based on seven phonemes. These vowels can be divided into different categories depending on how they are formulated: front/back position of tongue, wideness/roundness of the constriction position, place of tongue (high or low), and how open or closed the mouth is during articulation. Sesotho vowels and their categorization, based on the International Phonetic Alphabet (IPA) system, are summarized in Figure 1-7.

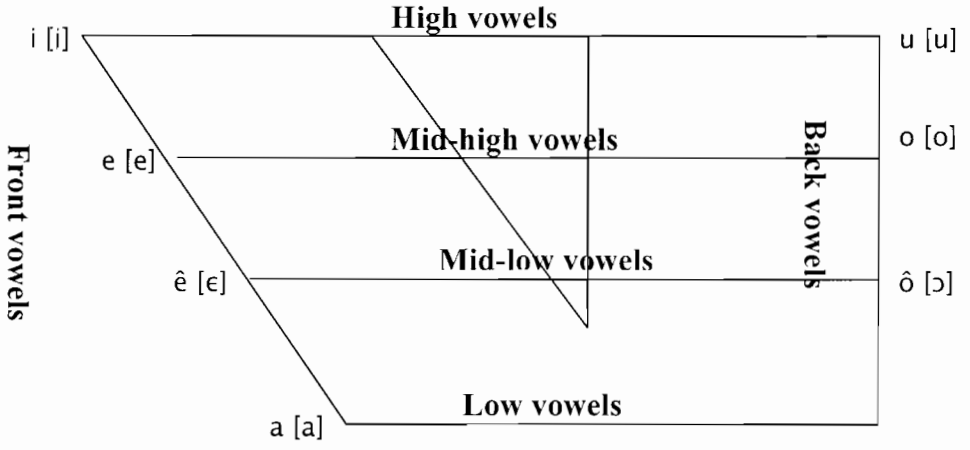


Figure 1-7: **Classification of Sesotho vowels [5]**

Sesotho consonants can be divided into the following categories: place of articulation and manner of articulation, as shown in Figure 1-8.

Figure 1-8 shows 43 phonetic symbols of Sesotho consonants. For this thesis, 31 consonants which accommodate both LS and SAS writing systems are taken into consideration. The reason for this is that as the writing system of the language continually changes, some symbols are replaced or dropped out. Also, some of the symbols cannot be written using today's system. For instance, the consonant **tš** is now written as ***ts'***, (or ***tsh*** in SAS). As mentioned earlier, the phonemic structure of Sesotho, like other languages, is not constant.

Manner of Articulation				**Place of Articulation**												
Type of consonant	Manner of airstream release	Channel of airstream release	Nature of airstream	Bilabial	Labio-dental	Labio-prepalatal	Apico-dental	Apico-alveolar	Apico-prepalatal	Lamino-prepalatl	Lateral	Medio-palatal	Dorso-palatal	Dorso-velar	Uvular	Glottal
Stop	Plosive	M	VI	p				t						k		
		M	A	p^h				t^h						k^h		
		L	VI								tl					
		L	A								tl^h					
		M	V	b				d								
	Affricative	M	VI			pʃ		tʃ				kx^h				
		M	A			$pʃ^h$		$tʃ^h$								
		M	V			bʒ		dʒ								
	Click	M	VI				ǀ		ǃ							
		N and O	V				ŋǀ		ŋǃ							
			VI/A						$ǃ^h$		ǁ					
		N and O	V								ŋǁ					
Continuant	Trill	M	V					r								
	Fricative	M	VI		f	fʃ		s		ʃ				x		
		M	V/VI							ʒ					R	ɦ/h
		L	VI								ɬ					
	Approximant	M	V	w									j			
		L	V								l					
	Nasal	N	V	m				n				ɲ		ŋ		

M = Medial
L = Lateral
N = Nasal
O = Oral

VI= Voiceless
V= Voiced
A=Aspirated

Figure 1-8: The consonants of Sesotho [5]

1.3 Problem Statement

As it has been mentioned earlier, research in speech synthesis has led to natural sounding systems which are used mostly for announcements. Even though these systems sound natural, their limitation is that they can only synthesize words in their database. It is not easy to collect all the words into the database, especially with new terms emerging every day. The solution is to use an open domain system which is flexible. By flexible we mean a system which can synthesize all text, irrespective of whether the word is in the database or not. In order to have a text-to-speech system that is both natural-sounding and flexible, Rousseau [4] built a hybrid system which meets the requirements of an advanced text-to-speech system. His system is a hybrid of a limited domain system together with an open domain system. An advanced system features understandability, flexibility, pleasantness, and naturalness. Even though this system works, its drawback is that the speech output is not fluent. This is due to glitches caused by an alternation of two voice systems as they synthesize input text.

This project therefore, looks at smoothing these discontinuities to improve naturalness and fluency of the hybrid system. This is done by applying intonation modeling techniques on the open domain system to get it to be as natural-sounding as possible, much like the limited domain speech output. With this method working, the glitches heard between an interchange of the two voices will be greatly reduced.

1.4 Objectives of the Report

The aim of this project is to improve naturalness and fluency of an advanced text-to-speech system for Sesotho. This will be done by investigating ways of masking the glitches that are heard when the advanced text-to-speech system is reading out text. The following points were considered in the strategic development of this work:

- Study and understand the general concepts in text-to-speech technology.
- Investigate and implement a popular TTS technique, concatenative speech synthesis.
- Build a Sesotho TTS synthesizer using limited domain, open domain and diphone systems.
- Implement a hybrid text-to-speech system for Sesotho. The procedure followed is the same as that by Rousseau [4] for his advanced text-to-speech system for Afrikaans. This system is a hybrid system of two unit selection voices, limited domain and open domain.

- Test and evaluate the speech output of the hybrid system.
- Investigate a way of masking the glitches and getting the tones of the two voices to be as close to each other as possible. This will be done by asking the following questions:
    - What parameters do we need to change? Is it intonation, tone, pitch, F0, duration?
    - Which parameters do we have control of?
- Look at the pitch marks and F0 contours of the two voices and analyze their spectrograms.
- Perform intonation modeling of the features duration and F0 of the open domain system.
- Carry out listening tests to compare the original advanced system with the 'fluent' advanced system.
- From the results obtained, conclusions will be drawn and recommendations for further action made.

1.5 Contribution to Knowledge

This thesis is of great essence in both speech synthesis and Human Language Technologies (HLT). In the field of speech synthesis, this thesis plays a part in contributing knowledge on how to design an advanced TTS system for any language. It also provides a good overview of the Festival speech synthesis system as a tool for building a synthetic voice for different languages. This tool can also be used for intonation modeling to enhance naturalness and fluency of the speech output.

HLT makes it easier for people to interact with machines. This can benefit a wide range of people – "from illiterate farmers in remote villages who want to obtain relevant medical information over a cellular phone, to scientists in state-of-the-art laboratories who want to focus on problem-solving with computers" [59]. In following the procedure used in this thesis for other languages, more people who are not conversant in English will be accommodated in the technology world. In a developing country such as South Africa, this design methodology can be extended to all South African official languages.

Having multilingual TTS systems will be beneficial in information access as well. For instance, text-to-speech can be used in telephone-based systems, say, government services, where they can request information which will be relayed back to them as it is read off government website.

This support for language diversity will enable everyone to have access to information in a convenient manner, regardless of the language one speaks.

1.6 Limitations and Scope of Investigation

This project was carried out within a scope which contributed to the following limitations:

- The recording studio and recording equipment were not professional. The recording environment was a laboratory which had 13 computers. The noise from these computers contributed in not getting noise-free recordings. The microphone used was a close-talking one which was head-mounted.
- The recordings were done on a standard PC (CPU speed 2.40 GHz) by the author, who is not a professional speaker.
- For intonation modeling, no accent modeling was done. The prosodic features which were modeled were duration and pitch (F0). The other prosodic features were not investigated in detail.
- An electroglottograph (EGG) was not used in the recording and pitchmarking of the speech signals. The EGG gives almost perfect pitchmark moments.

One other limitation is the time taken to do manual labeling. Manual labeling was necessary as the automatic labeling technique, dynamic time warping, provided by Festival, though fast, was not perfect. Manual labeling is laborious and time-consuming, especially for large databases. Labeling is a crucial step in the voice building process since it labels the positions of the speech units to be used for synthesis. Labeling determines the quality of the speech output of a TTS system; incorrect labeling will result in a wrong region of the speech units being used during synthesis.

Although there are different techniques in speech synthesis, the focus of this thesis is limited to one popular technique, concatenative synthesis. This method has two aspects which fall under it, diphone concatenative synthesis and unit selection synthesis. Both aspects were investigated and applied in the design of an advanced Sesotho TTS system. For intonation, focus falls on duration and F0 modeling.

The whole design is implemented using Festival speech synthesis engine which is written in C and Scheme programming languages.

1.7 Thesis Outline

This report describes a way of improving naturalness and fluency of the Sesotho TTS system by investigating and applying intonation. The system is an advanced text-to-speech system in that it meets TTS requirements which are naturalness, flexibility, pleasantness and understandability. Though all these requirements are met, naturalness and fluency are degraded by glitches heard when the system alternates between two voices it uses to read out text. The two voices are the unit selection systems, limited domain and open domain. Hence, focus is on controlling the intonation (pitch/F0) of the least natural-sounding voice; open domain in our case. The aim is to improve the open domain naturalness and get it to be at the same level as that of the limited domain, if not close.

The remainder of this report is outlined as follows.

In Chapter 2, history and development of speech synthesis are discussed. One of the issues tackled is how speech is produced. In order to understand speech synthesis, knowledge of speech production is essential. The chapter further describes different speech synthesis techniques which have been researched and their implementation. Speech synthesis has many applications and some of them are given in this chapter.

Chapter 3 introduces the Festival speech synthesis system and the requirements necessary for installation and compilation of Festival. The architecture of the Festival system is discussed where the utterance structure is the main object in this aspect. Of the speech synthesis techniques discussed in Chapter 2, Festival concentrates on the concatenative synthesis technique. The different categories of concatenative synthesis and their implementation in Festival are explored. The same methods were also implemented in carrying out this project.

Chapter 4 focuses on intonation and different aspects related to it. These include prosody, which is the superset under which intonation falls, or fundamental frequency, F0. Different intonation modules are discussed, some of which are used in Festival system. Fundamental frequency generation can be done by various techniques and these are also discussed. The chapter also looks at the research that has been done by different people in F0 modeling.

The design of the two hybrid Sesotho TTS systems is portrayed in Chapter 5. The four synthetic voices are built here: diphone system, limited domain and two kinds of open domain synthesis systems. The baseline system is a hybrid system which uses the limited domain and open

domain voices. The second hybrid system incorporates implementation of intonation in order to get a more natural sounding and fluent speech output.

The results obtained, their analysis and evaluation are given in Chapter 6. Chapter 7 discusses the conclusions drawn based on these results, and the recommendations made for future work.

1.8 Summary

This chapter has given a brief overview of the text-to-speech synthesis and its background. Sesotho language has also been investigated, the focus being on its background, motivation for implementing a Sesotho TTS system, and its phonetic structure. Furthermore, the chapter discussed the role on intonation in this thesis and in the Sesotho language. The research objectives of the thesis and its contribution to knowledge were explored. Focus was also directed to a scope under which the thesis was carried out and limitations encountered in the whole process. The chapter concluded by providing the outline of the report as a whole.

Chapter 2

2. History and Development of Speech Synthesis

The purpose of this chapter is to look at the history of text-to-speech synthesis technology and how it came to be – a dream for man to be able to communicate with machines. In order for text-to-speech technology to be implemented in an effective manner, the human speech production system has to be thoroughly understood. Therefore, in this chapter, the speech production system is investigated and how speech is produced. Three techniques have been researched in the field of speech synthesis and these are explored in detail. Finally, situations where text-to-speech technology can be applied are surveyed.

2.1 History

Human beings have dreamt of producing synthetic speech for centuries. First attempts to produce human speech by machine were made in the second half of the 18th century [1]. In 1773, Professor Christian G. Kratzenstein managed to produce vowels using resonance tubes connected to organ pipes. He constructed acoustic resonators similar to a human vocal apparatus to produce them artificially. The basic structure of the resonators is shown in Figure 2-1. Around the same time, Wolfgang von Kempelen was working on constructing a speaking machine. He is known as the first experimental phonetician and in his book, "*Mechanismus der menschlichen Sprache nebst Beschreibung einer sprechenden Maschine* " (1791), he included a detailed description of his speaking machine so that others could reconstruct it and make it more perfect [1]. Von Kempelen's machine was the first that not only produced some speech sounds, but also whole words and short sentences.

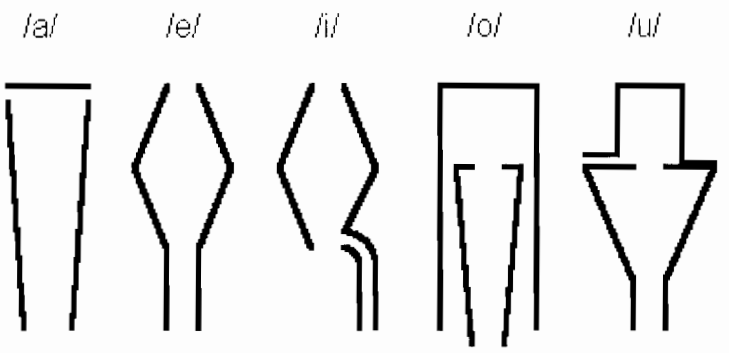


Figure 2-1: Kratzentein's resonators [1]

The essential parts of the machine were a pressure chamber for the lungs, a vibrating reed to act as vocal cords, and a leather tube for the vocal tract action. By manipulating the shape of the leather tube, he could produce different vowel sounds. Consonants were simulated by four separate constricted passages and controlled by fingers. For plosive sounds he also employed a model of a vocal tract that included a hinged tongue and movable lips [16].

In about mid 1800's Charles Wheatstone constructed a version of von Kempelen's speaking machine which as illustrated in Figure 2-2. It was a bit more complicated and could produce vowels and most consonant sounds. Some sound combinations and even full words were also possible to produce. Vowels were produced with a vibrating reed and all passages were closed. Resonances were affected by deforming the leather resonator like that in von Kempelen's machine. Consonants, including nasals, were produced with turbulent flow through a suitable passage with reed-off.

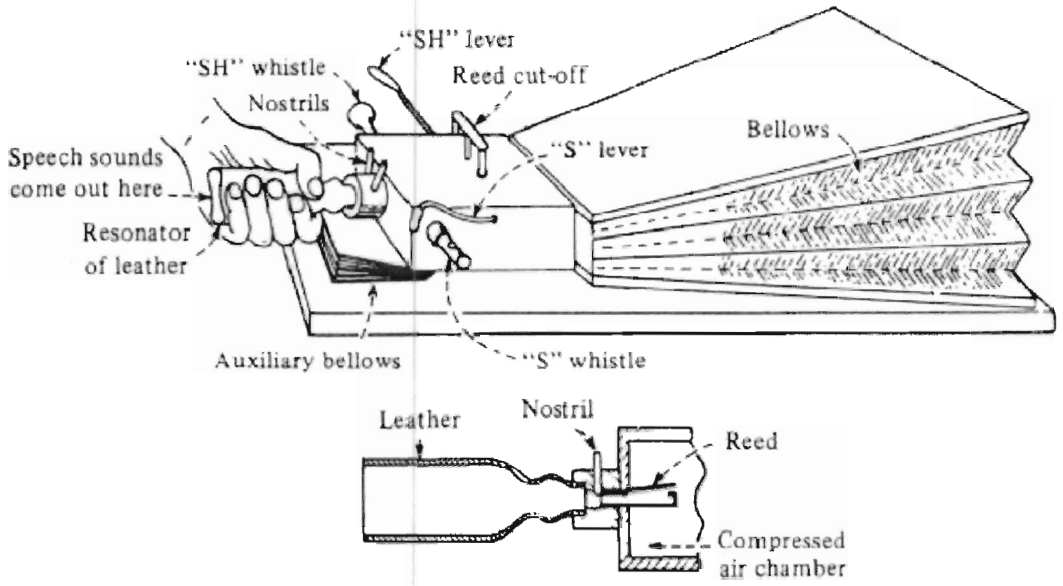


Figure 2-2: Wheatstone's reconstruction of von Kempelen's speaking machine [1]

More machines depicting von Kempelen's were constructed in the 19th century, but none were fundamental innovations in the field of speech synthesis. However, Joseph Faber devised a machine whose speech production mechanism included a tongue model and a pharyngeal cavity whose shape could be controlled. It was also suited for the synthesis of singing [1]. Its bellows was operated via a pedal or via a keyboard.

In the late 1800's, Alexander Graham Bell with his father, inspired by Wheatstone's speaking machine, constructed the same kind of speaking machine. Bell also used his dog [16] to produce speech-like sounds by modifying its vocal tract by use of his hands.

In another attempt, the American, R. R. Riesz constructed a device with a vocal tract shape that was close to the natural one. The model is shown in Figure 2-3.

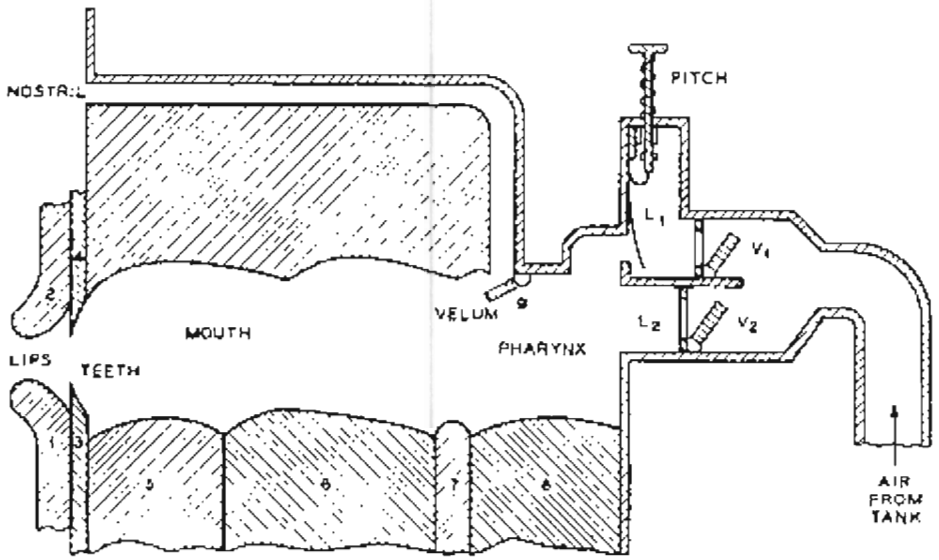


Figure 2-3: The mechanical model of speech production built by Riesz [1]

At the beginning of the 20th century, speech synthesis could be done by electrical means. The first such device, the Voder, was developed by Homer Dudley. However, this device needed long training hours for it to work successfully. Kempelen developed his device in parallel with his investigation of the human speech production mechanism. On the other hand, Dudley's device was based on his VOCODER (Voice Coder), whose purpose was to reduce the bandwidth necessary for telephonic transmission of speech, so that a larger number of telephone calls could be transmitted over a given line [1].

Since 1970, speech synthesis has been developed through use of computer technology. This technology is useful in developing text-to-speech systems faster, though not necessarily the best in terms of quality of speech output.

2.2 Speech Production System

Knowledge of how speech is produced, articulatory phonetics and related terminology are essential in order to understand how speech synthesis works. This section discusses the basic theory of these issues.

Human speech is produced by vocal organs as depicted in Figure 2-4. The lungs, together with the diaphragm, are the main source of energy. As described by [16], when speaking, air flow is

forced through the glottis between the vocal cords and the larynx to the three main cavities of the vocal tract, the pharynx, and the oral and nasal cavities. From the oral and nasal cavities, the air flow exits through the nose and mouth, respectively. The glottis is the most important sound source in the vocal system. The vocal cords may act in several different ways during speech. The most important function is to modulate the air flow by rapidly opening and closing, causing a buzzing sound from which vowels and voiced consonants are produced. The fundamental frequency of vibration depends on the mass and tension and is about 110 Hz, 200 Hz, and 300 Hz with men, women, and children, respectively [16]. With stop consonants the vocal cords may act suddenly from a completely closed position, in which they cut the air flow completely, to totally open position producing a light cough or a glottal stop. On the other hand, with unvoiced consonants, they may be completely open. An intermediate position may also occur with, for instance, phonemes like /h/.

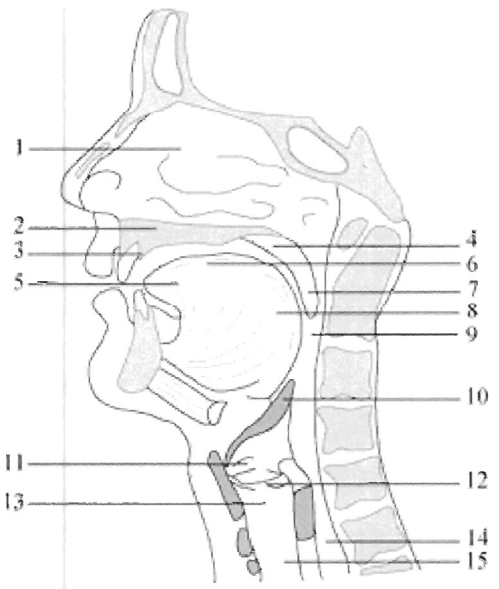


Figure 2-4: **The human vocal organs [16]**

(1) Nasal cavity, (2) Hard palate, (3) Alveoral ridge, (4) Soft palate (Velum), (5) Tip of the tongue (Apex), (6) Dorsum, (7) Uvula, (8) Radix, (9) Pharynx, (10) Epiglottis, (11) False vocal cords, (12) Vocal cords, (13) Larynx, (14) Esophagus, and (15) Trachea.

The description below of how human vocal organs are connected and function is taken from [16]. The pharynx connects the larynx to the oral cavity. It has almost fixed dimensions, but its length may be changed slightly by raising or lowering the larynx at one end and the soft palate at the other end. The soft palate also isolates or connects the route from the nasal cavity to the pharynx. At the bottom of the pharynx are the epiglottis and false vocal cords to prevent food

from reaching the larynx and to isolate the esophagus acoustically from the vocal tract. The epiglottis, the false vocal cords and the vocal cords are closed during swallowing and open during normal breathing.

The oral cavity is one of the most important parts of the vocal tract. Its size, shape and acoustics can be varied by movements of the palate, the tongue, the lips, the cheeks and the teeth. The tongue is very flexile, the tip and the edges can be moved independently and the entire tongue can move forward, backward, up and down. The lips control the size and shape of the mouth opening through which speech sound is radiated. Unlike the oral cavity, the nasal cavity has fixed dimensions and shape. Its length is about 12cm and volume 60cm^3. The air stream to the nasal cavity is controlled by the soft palate.

2.2.1 Classification and Description of Sesotho Speech Sounds

The speech sounds of Sesotho, like most languages, can be divided into two groups: vowels and consonants.

The vowels of Sesotho are characterized by the fact that their air stream [5]:

- is pulmonic;
- becomes voiced when moving through the vocal bands;
- moves relatively unimpeded through the speech channel;
- moves along the oral passage;
- moves over the middle of the tongue.

On the other hand, the consonants are characterized by an air stream that [5]:

- is either pulmonically or laryngeally or orally initiated and
- may be voiced or voiceless;
- is either stopped or continued;
- may be released orally, nasally or both orally and nasally;
- may be released over the middle of the tongue (i.e. medially) or over the sides of the tongue (i.e. laterally).

Besides these general characteristics, the individual vowels and consonants have features or characteristics distinguishing each of them from the other consonants and vowels of Sesotho.

Table 2-1 below shows a phonetic symbol representation for Sesotho vowels as per IPA system. In order to distinguish ordinary symbols from phonetic symbols, the IPA decided to write phonetic script (symbols) between square brackets as shown in the table.

Table 2-1: The IPA table depicts the relative frontness and backness of low and high vowels [5]

Tongue height	Ordinary symbols	Phonetic symbols
High vowels	i u	[i] [u]
Mid-high vowels	e o	[e] [o]
Mid-low vowels	ê ô	[ɛ] [ɔ]
Low vowels	a	[a]

Consonants can best be studied when classified according to their place and manner of articulation. *Place of articulation* refers to an interaction between active and passive articulators. It is usually taken for granted that an active articulator lies opposite a passive articulator with which it interacts. The term *manner of articulation* refers to:

- the nature of the air stream;
- the channel along which the air stream is released;
- the manner in which the air stream is released.

Just as with vowels, the IPA also decided on a simple and unambiguous set of characters to represent consonant sounds or phones. In the case of Sesotho, the symbols used in phonetic script correspond for the most part with those used in ordinary writing. The phonetic representation is shown in Figure 1-8 (Chapter 1).

2.2.1.1 The Syllable

There are three types of syllables in Sesotho [8]. They are (a) the type that consists of a vowel only, i.e. V syllable; (b) the type that consists of a consonant plus a vowel, i.e. CV syllable, and (c) the type that consists of a syllabic consonant, i.e. C syllable. The four nasal consonants (/n/, /m/, /ng/, /ny/) and /l/ may each occur syllabically, i.e. as a syllable. Of the three types of syllables two end in a vowel. For this reason syllables in Sesotho as well as in the other Bantu languages are said to be open. The syllable is a carrier of tone in Sesotho.

2.3 Speech Synthesis Techniques

Speech can be synthesized using three different techniques, namely:

- Articulatory synthesis, which attempts to model a human speech production system directly.
- Formant synthesis, which models pole frequencies of speech signal or transfer function of a vocal tract based on source-filter model.

- Concatenative synthesis, which uses different lengths of prerecorded samples derived from natural speech.

2.3.1 Articulatory Synthesis

Articulatory synthesis models the human articulatory system. When speaking, the vocal tract muscles cause articulators to move and change shape of the vocal tract which causes different sounds. The articulatory method has a potential of producing high-quality synthesis because it tries to model the human speech organs directly. The challenge though, is that it is a difficult method to implement and the computational load is also considerably higher than other methods. It has, therefore, not received much attention from speech synthesis researchers and it has not achieved the same level of success yet.

Advantages of articulatory synthesis are that the vocal tract models allow accurate modeling of transients due to abrupt area changes, whereas formant synthesis models only spectral behavior. The articulatory synthesis is quite rarely used in present systems, but since the analysis methods are developing fast and the computational resources are increasing rapidly [1], it might be a potential synthesis method in the future.

2.3.2 Formant Synthesis

Formant synthesis does not use any human speech samples at runtime. This method models pole frequencies of speech signal or transfer function of a vocal tract based on a source-filter model [16]. Parameters such as fundamental frequency, voicing, and noise levels are varied over time to create a waveform of artificial speech. It also provides infinite number of sounds and this makes it the most flexible synthesis method.

Even though formant synthesis systems sound robotic, they do have advantages. First, these systems are intelligible, even at high speeds. This ability gives them an advantage over concatenative synthesis systems. Formant synthesis systems do not have the acoustic glitches that are often audible in concatenative systems. Second, formant synthesizers are often smaller programs than concatenative systems because they do not have a database of speech samples [24]. They can thus be used in embedded computing situations where memory space and processor power are often scarce. Last, because formant-based systems have total control over all aspects of the output speech, a wide variety of prosody or intonation can be output, conveying not just questions and statements, but a variety of emotions and tones of voice.

2.3.3 Concatenative Synthesis

Concatenative synthesis is the stringing together of segments of recorded speech. Generally, concatenative synthesis gives the most natural sounding synthesized speech. However, natural variation in speech and automated techniques for segmenting the waveforms sometimes result in audible glitches in the output, detracting from the naturalness.

One of the most important aspects in concatenative synthesis is to find correct unit length [16]. The selection is usually a trade-off between longer and shorter units. With longer units, high naturalness, less concatenation points and good control of coarticulation are achieved, but the amount of required units and memory is increased. With shorter units, less memory is needed, but sample collecting and labeling procedures become more difficult and complex. In present systems, units used are usually words, syllables, demisyllables, phonemes, diphones, and sometimes triphones.

Concatenative synthesis has some weaknesses compared to other methods.

- Distortion from discontinuities in concatenation points, which can be reduced using diphones or some special methods for smoothing signal [16].
- Memory requirements are usually very high, especially when long concatenation units are used, such as syllables or words.
- Data collecting and labeling of speech samples is usually time-consuming.

There are three types of concatenative synthesis:

Unit selection synthesis uses large speech databases which are more than one hour of recorded speech. During database creation, each recorded utterance is segmented into some or all of the following: individual phones, syllables, morphemes, words, phrases, and sentences. The division into segments can be done using a number of techniques, like clustering, using a specially modified speech recognizer, or by hand, using visual representations such as the waveform and spectrogram [24]. An index of the units in the speech database is then created based on the segmentation and acoustic parameters like the fundamental frequency (pitch). At runtime, the desired target utterance is created by determining the best chain of candidate units from the database (unit selection). This technique gives the greatest naturalness due to the fact that it does not apply digital signal processing techniques to the recorded speech, which often makes recorded speech sound less natural. According to [24], an output from the best unit selection systems is often indistinguishable from real human voices, especially in contexts for which a TTS system has been tuned. However, maximum naturalness often requires unit selection speech databases to be very large, in some systems ranging into the gigabytes of recorded data and numbering into the dozens of hours of recorded speech.

Diphone synthesis uses a minimal speech database containing sound-to-sound transitions occurring in a given language. In diphone synthesis, only one example of each diphone is contained in the speech database. Diphones are defined to extend a central point of a steady state part of a phone to a central point of the following one, so they contain transitions between adjacent phones. That means that the concatenation point will be in the steadiest region of the signal, which reduces the distortion from concatenation points. At runtime, the target prosody of a sentence is superimposed on these minimal units by means of digital signal processing techniques such as Linear Predictive Coding (LPC), PSOLA, or MBROLA [24]. The quality of the resulting speech is generally not as good as that from unit selection but more natural-sounding than the output of formant synthesizers. Diphone synthesis suffers from the sonic glitches of concatenative synthesis and sounds robotic. This has led to its use in commercial applications to decline, though it is still being used in research.

Limited domain synthesis concatenates pre-recorded words and phrases to create complete utterances. The naturalness of these systems can potentially be very high because a variety of sentence types is limited and closely matches the prosody and intonation of the original recordings. However, because these systems are limited by the words and phrases in its database, they are not general-purpose. They can only synthesize the combinations of words and phrases they have been pre-programmed with. Limited domain synthesis systems used in applications where the variety of texts the system will output is limited to a particular domain, for instance, in flight schedule announcements or weather reports.

Of the three speech synthesis techniques mentioned, the formant and concatenative methods are the most commonly used in present synthesis systems. The formant synthesis was dominant for a long time, but today the concatenative method is becoming more and more popular [16]. The articulatory method is still too complicated for high quality implementations, but may arise as a potential method in the future.

2.4 Speech Synthesis Applications

The applications of synthetic speech are expanding and the quality of text-to-speech systems is improving. Speech synthesis systems are also becoming more affordable for ordinary users, which makes these systems more suitable for everyday use [16]. There are many applications to which synthetic speech can be put to use. Some of these applications are discussed below.

Unified Messaging

Unified messaging is one of the most compelling applications of TTS today [19]. It enables storing voice mail, email, and fax within one unified mailbox that is accessible via standard email clients, over the web, and over the phone. When accessed over a standard telephone, email and fax messages need to be rendered in voice by using TTS.

Reading of Web Pages

Selective reading of web pages is carried out by a screen reader, which is a piece of software that runs alongside other programs, capturing whatever they display on the screen. The great advantage of a screen reader is that it works with standard application software, so it is not necessary to develop a talking word processor and a talking spreadsheet [20]. This application is also imperative to visually impaired people. Through this application, one is able to access the web and get information they request delivered to them through a text-to-speech system, which renders the information by voice. Examples of services offered include e-mail reading, weather reports, and alerting customers of stock exchange prices. A major advantage of this application is that one does not need to have internet connection as the message can be relayed over the phone.

Augmentative Communication

This is the use of computer-based technology to facilitate personal communication. A person who cannot speak for one reason or another may use a device through which they can specify utterances to be communicated across. The medium of communication here is synthetic speech, since the voice is being replaced. The most celebrated user of this application is Professor Stephen Hawking.

Applications for the Blind

Text-to-speech systems can also be used in the reading and communication aids for the blind. This application replaces audio books into which the content of the book was read into audio tape.

Applications for the Deaf and Vocally Handicapped

Synthesized speech gives the deaf and vocally handicapped an opportunity to communicate with people who do not understand sign language. Use of a talking head improves the quality of the communication situation more because visual information is the most important with the deaf and dumb. A speech synthesis system may also be used with communication over a telephone line. An example of this is the use of a Teldem system through which deaf people can communicate

with hearing people over long distances. This system makes use of both speech synthesis and speech recognition technologies.

Educational Applications

Speech synthesis combined with proper computer software can offer unsupervised training for teaching dyslexic people how to read and write. This speech synthesizer can be used to teach 24 hours a day and 365 days a year. It can be programmed for special tasks like spelling and teaching pronunciation for different languages. It can also be used with interactive educational applications.

A speech synthesizer with a word processor is also a helpful aid for proof reading. Many users find it easier to detect grammatical and stylistic problems when listening than reading. Normal misspellings are also easier to detect.

Other Applications

In principle, speech synthesis may be used in all kinds of human-machine interactions. For example, in warning and alarm systems synthesized speech may be used to give more accurate information of the current situation [3]. Using speech instead of warning lights or buzzers gives an opportunity to reach a warning signal for instance, from a different room. Speech synthesis may also be used to receive some desktop messages from a computer, such as printer activity or received e-mail. It can also be used in telephone enquiry systems where one can call in and request some information and feedback is given in a form of a synthetic voice. Text messages (sms) in mobile phones can also be read out through the use of synthetic speech. Talking mobile phones increase usability considerably. For example, they are essential for visually impaired users and in situations where it is difficult or dangerous to reach the output from mobile phone, for instance, when driving a car.

During the last few decades, communication aids have been developed from talking calculators to modern three-dimensional audiovisual applications [16]. The application field for speech synthesis is becoming wider all the time which also brings more funds into research and development areas.

2.5 Summary

This chapter has given a background history of speech synthesis in detail. A speech production system has also been discussed and the classification of Sesotho speech sounds. The chapter also looked at three speech synthesis techniques, their implementation, their advantages and disadvantages. Finally, some of the applications in which speech synthesis plays a major role were discussed.

Chapter 3

3. The Festival Speech Synthesis System

This chapter focuses on the main resource tool used in carrying out the project: the Festival Speech Synthesis System. Festival is a popular and freely available speech synthesis engine. The requirements necessary for proper use of the engine are discussed. Understanding of this engine's architecture is also of importance so that the user can maneuver around the system and make changes according to their needs. Having learnt about different text-to-speech techniques in the previous chapter, this chapter expands more on one particular technique - concatenative synthesis method. This is the method used in Festival development, which is also implemented in this project. The chapter concludes by describing how intonation fits into the Festival system.

3.1 Introduction to Festival

The tool used in carrying out this project is the Festival speech synthesis system. Festival is a speech synthesis engine which is available as free software on [22]. Festival is a complete TTS system in that it attempts to take arbitrary text input and synthesize it. This tool is useful as it is easy to configure and offers the ability to add new external modules. It offers a general framework for building speech synthesis systems as well as including examples of various modules.

The basic Festival system has the following major modules available: text pre-processing, lexical lookup, prosodic assignment and waveform synthesis. Text processing involves conversion of input characters into a list of tokens that will be understood by the synthesis system. For lexical lookup, the pronunciation of each word is obtained from a letter to sound rule system. Prosodic assignment chucks utterances into prosodic phrases, while waveform synthesis is a method for generating a waveform.

Festival is a multi-lingual based synthesis system [22] with English as the most advanced language. More languages are being released by other groups using this tool. A complete set of tools and documentation for building new voices are available through Carnegie Mellon's FestVox project [27]. The system is written in C++ and uses the Edinburgh Speech Tools Library for low level architecture and has a Scheme (SIOD) based command interpreter for control.

The Festival speech synthesis system is designed for different levels of users. The first group is people who simply want high quality speech from arbitrary text with minimum effort applied. The second group is those who are developing language systems and wish to include synthesis output. In this case, a certain amount of customization is desired, such as different voices, specific phrasing, dialog types, etc. The third group is those who want to develop and test new synthesis methods.

One of the advantages of using Festival as a synthesis engine is that it offers the user the ability to test and improve their desired part in the whole system. Without systems like Festival, one would need to spend significant effort building the whole system, or adapting an existing one before they could start working on their improvements. Finally, another aspect which makes Festival more efficient to use is that it can be embedded in other packages that require speech output.

3.2 Requirements

This section identifies the basic requirements necessary for use of the Festival Speech Synthesis System.

3.2.1 Hardware/Software Requirements

Festival runs on both Windows and Linux operating systems, though much of the testing by its developers [10] was done under Linux. For this thesis, the Linux operating system was used. A full distribution of Festival and Edinburgh Speech Tools software programs are required to get maximum usage of the text-to-speech system. Festival 1.9.5 (full version), Edinburgh Speech Tools 1.2.1 and all the Festvox voices were used in carrying out this project. Edinburgh Speech Tools lies at the heart of Festival and its library is obtainable at [21] and Festival software is available at [22].

In brief, the following source packages are necessary for installation and running of Festival [10]:

- festival-1.4.3.tar.gz – Festival Speech Synthesis System source.
- speech_tools-1.2.3.tar.gz – The Edinburgh Speech Tools Library.
- festlex_NAME.tar.gz – The lexicon distribution. The lexicons have varying distribution policies, but are all free except OALD, which is only free for non-commercial use.
- festvox_NAME.tar.gz – A collection of speech databases (with varying distribution policies).

- Festdoc_1.4.3.tar.gz – Full postscript, information and HTML documentation for Festival and Speech Tools.

In addition to Festival specific sources, the following are also necessary:

- A C++ compiler – According to the latest Festival manual [10], the tested systems are:
  - Sun Sparc Solaris 2.5, 2.5.1, 2.6, 2.7: GCC 2.7.2, egcs 1.1.1, egcs 1.1.2, GCC 2.95.1
  - Sun Sparc SunOS 4.1.3: GCC 2.7.2
  - FreeBSD for Intel 3.x and 4.x GCC 2.95.1, GCC 3.0
  - Linux for Intel (RedHat 4.1/5.0/5.1/5.2/6.0): GCC 2.7.2, GCC 2.7.2/egcs 1.0.2, egcs 1.1.1, egcs 1.1.2, GCC 2.95.[123], GCC "2.96", GCC 3.0
  - Windows NT 4.0: GCC 2.7.2 plus egcs (from Cygnus GNU win32 b19), Visual C++ PRO v5.0, Visual C++ v6.0.
- GNU make – This is the one that has been tested by Festival system developers.
- Audio hardware – A number of audio systems are supported.

Festival also offers a simple test suite which requires less number of installation packages. These are the three basic voices and their respective lexicons:

- festlex_CMU.tar.gz
- festles_OALD.tar.gz
- festlex_POSLEX.tar.gz
- festvox_don.tar.gz
- festvox_kedlpc16k.tar.gz
- festvox_rablpc16k.tar.gz

Database recording needs both an ideal recording environment and recording equipment. A nearly soundproof studio, with minimum background noise is the best option for a recording studio. A high quality sound card and a close-talking microphone are ideal for getting a near perfect speech input during recording. A machine with a high computational speed is also necessary for quick processing.

3.2.2 Building a Voice in a New Language

The following list is a basic check list of the core areas which need to be taken care of in building a voice for a new language. Areas to be defined are [9]:

- phone set
- token processing rules (numbers, etc)
- prosodic phrasing method

- word pronunciation (lexicon and/or letter-to-sound rules)
- intonation (accents and F0 contour)
- durations
- waveform synthesizer

Once all necessary tools have been installed and compiled, Festival can be used for synthesizing text using available modules.

3.3 Architecture

3.3.1 Festival Utterance Structure

An utterance is the basic object for synthesis in Festival. An utterance is a sentence that is to be rendered as speech. In many cases, this 'sentence' does not conform to the standard linguistic syntactic form of a sentence. In general, the process of text to speech is to take an utterance which contains a simple string of characters and convert it step by step; filling out the utterance structure with more information until a waveform is built that says what the text contains [10].

The processes involved in conversion are, in general, as follows:

Tokenization	Conversion of characters into a list of tokens.
Token identification	Identification of general types for the tokens.
Token to word	Conversion of each token to zero or more words, expanding numbers, abbreviations, etc.
Part of speech	Identifying the syntactic part of speech for the words.
Prosodic phrasing	Chucking utterances into prosodic phrases.
Lexical lookup	Finding the pronunciation of each word from a lexicon/letter to sound rule system including phonetic and syllables structure.
Intonational accents	Assigning intonation accents to appropriate syllables.
Assign duration	Assigning duration to each phone in the utterance.

Generate F0 contour Generating tune based on accents, etc.

Render waveform Rendering waveform from phones, duration and F0 target values. This may take several steps including unit selection (diphones or other sized units), imposition of desired prosody (duration and F0) and waveform reconstruction.

An utterance structure consists of a set of items which may be part of one or more relations. *Items* represent words and phones, though they may also be used to represent less concrete objects like noun phrases, and nodes in metrical trees. An item contains a set of features (name and value). *Relations* are simple lists of items or trees of items. For example, the *Word* relation is a simple list of items each of which represents a word in the utterance. Those words will also be in other relations, such as the *SylStructure* relation where the word will be the top of a tree structure containing its syllables and segments.

The architecture is fully general and new items and relations may be defined at run time, such that new modules may use any relations they wish. A set of relations used in Festival is as follows [10]:

Token A list of trees. This is first formed as a list of tokens found in a character text string. Each root's daughters are the *Word*'s that the token is related to.

Word A list of words. These items may appear as daughters (leaf nodes) of the *Token* relation or as leafs of the *Syntax* relation.

Phrase A list of trees. This is a list of phrase roots whose daughters are the *Word*'s within those phrases.

Syntax A single tree. This, if the probabilistic parser is called, is a syntactic binary branching tree over the members of the *Word* relation.

SylStructure A list of trees. This links the *Word*, *Syllable* and *Segment* relations. Each *Word* is the root of a tree whose immediate daughters are its syllables and their daughters in turn as its segments.

Syllable A list of segments (phones). Each member (except silences) will be leaf nodes in the *SylStructure* relation. These may also be in the *Target* relation linking them to F0 target points.

IntEvent A list of intonation events (accents and boundaries). These are related to syllables through the *Intonation* relation as leafs on that relation. Thus their parent in the *Intonation* relation is the syllable these events are attached to.

Intonation A list of trees relating syllables to intonation events. Roots of the trees in *Intonation* are *Syllables* and their daughters are IntEvents.

Wave A single item with a feature called wave whose value is the generated waveform.

3.3.2 Utterance Types

Utterance types define which modules are to be applied to an utterance. The function *defUttType* defines which modules are to be applied to an utterance of that type. When the function *utt.synth* is called, it applies this list of modules to an utterance before waveform synthesis is called.

During the application of the function *utt.synth,* there are three hooks applied. This allows additional control of the synthesis process. *before_synth_hooks* is applied before any modules are applied. *after_analysis_hooks* is applied at the start of Wave_Synth when all text, linguistic and prosodic processing have been done. *after_synth_hooks* is applied after all modules have been applied. These are useful for altering the volume of a voice that happens to be quieter than others.

3.3.3 Modules

The basic synthesis process in Festival is viewed as applying a set of modules to an utterance [10]. Each module accesses various relations and items and potentially generates new features, items and relations. Thus, as the modules are applied, the utterance structure is filled in with more and more relations until ultimately the waveform is generated.

The modules are executed according to utterance type. For most text-to-speech cases, this is defined to be of type Tokens. The function *utt.synth* simply looks up an utterance's type and then looks up the definition of the defined synthesis process for that type and applies the named modules. Synthesis type may be defined using the function *defUttType*.

In general, the modules named in the type definitions are general and actually allow further selection of more specific modules within them. The basic modules used for text-to-speech have the following basic functions [9]:

Token_POS Basic token identification, used for homograph disambiguation.

Token Apply the token to word rules building the *Word* relation.

POS A standard part of speech tagger.

Phrasify Build the *Phrase* relation using the specified method.

Word Lexical look up building the *Syllable* and *Segment* relations and the *SylStructure* relate these together.

Pauses Prediction of pauses, inserting silence into the *Segment* relation, through a choice of different prediction mechanisms.

Intonation Prediction of accents and boundaries, building the *IntEvent* relation and the *Intonation* relation that links *IntEvent*s to syllables. This can easily be parameterized for most practical intonation theories.

PostLex Post lexicon rules that can modify segments based on their context. This is used for vowel reduction, contractions, etc.

Duration Prediction of durations of segments.

Int_Targets The second part of intonation. This creates the *Target* relation representing the desired F0 contour.

Wave_Synth A general function that in turn calls an appropriate method to actually generate the waveform.

3.4 TTS Synthesis Techniques in Festival

There are different techniques used for TTS synthesis, the most popular being concatenative synthesis. Concatenative synthesis is composed of diphone and unit selection synthesis systems. Festival speech synthesis is based on concatenative synthesis. This technique and its use in Festival development are explored in this section.

In concatenative systems, speech units can be either fixed-size diphones or variable length units such as syllables and phones. The latter approach is known as unit selection, since a large

speech corpus containing more than one instance of a unit is recorded and variable length units are selected based on some estimated objective measure to optimize the synthetic speech quality.

3.4.1 Diphone Concatenative Synthesis

A diphone is a unit of speech starting in a stable part of a phone and ending in a stable part of the following one. The basic idea behind building diphone databases is to explicitly list all possible phone-phone transitions in a language [9]. Unlike generalized unit selection where multiple occurrences of phones may exist with various distinguishing features, in a diphone database only one occurrence of each diphone is recorded. This makes selection much easier but also makes for a large laborious collection task.

In general, the number of diphones in a language is the square of the number of phones. However, in natural human languages, there are phonotactic constraints – some phone-phone pairs, even whole classes of phones-phone combinations, may not occur at all. These gaps are common in the world's languages. Humans are capable of generating the so-called non-existent diphones.

Diphone synthesis makes a fixed choice about which units exist, and in circumstances where something else is required, a mapping is necessary. When humans are given a context where an unusual phone is desired, they will (often) attempt to produce it even though it falls outside their basic phonetic vocabulary. The articulatory system is flexible enough control to produce (or attempt to produce) unfamiliar phones, as humans all share the same underlying physical structures. Concatenative synthesizers, however, have a fixed inventory, and cannot reasonably be made to produce anything outside their pre-defined vocabulary. Formant and articulatory synthesizers have the advantage here.

3.4.2 Unit Selection Synthesis

Unit selection is a selection of a speech unit which can be anything from a whole phrase down to the smallest speech unit available. Usually, there is more than one example of the unit and some mechanism is used to select between them during run-time [9]. Unit selection has emerged as a promising methodology to solve the problems with the fixed-size unit inventory synthesis, e.g. diphone synthesis.

Two forms of unit selection which have been implemented in the TTS design of this project will be discussed. These are limited domain and unit selection (open domain). Limited domain is vocabulary specific while open domain can synthesize any text given to it, regardless of whether the words are in the database or not.

3.4.2.1 Limited Domain Synthesis

Limited domain systems are applications whose speech output is constrained. These systems make use of only targeted words and phrases for the application in question. This could be, for instance, a weather report system, or an airline flight schedule announcements. These systems can only output speech of words which are in their vocabulary in a natural and intelligible voice. Any out-of-vocabulary (OOV) words or phrases cannot be synthesized.

In limited domain synthesis systems, it is much easier to build a unit selection type synthesizer where there is a much smaller and controlled number of units. Unlike diphone systems which sound robotic, limited domain systems sound natural.

In designing a database for limited domain, at least two or more words should be repeated in the vocabulary. The utterances selected should also maximize the bigram coverage. All this is done to enhance the quality of the synthesized speech output.

One other important point to consider is the phrasing of the prompts. The phrasing has to be explicitly marked and have the speaker follow the phrasing as it allows for much better joins in unit selection. The speaker should also give prosody that is as consistent as they possibly can.

In unit selection, automatic labeling can be done by any one of the alignment techniques; dynamic time warping (DTW) or hidden markov models (HMMs). DTW works well even though it is not as robust for general speech as it is for carefully articulated nonsense words in a diphone system. This technique does not allow for alternative pronunciations which are common [9]. HMMs modeling is a speech recognition technique based on Baum-Welch training. This technique is the most consistent in labeling.

The basic cluster unit selection code available in Festival uses segments as the size of a unit. However, the acoustic distance measure used in cluster uses significant portions of the previous segment. Thus, the cluster unit selection effectively selects diphones from the database.

The type of units the cluster selection code uses is based on the segment name, by default. In a case of limited domain synthesis, constraining this further gives both better and faster synthesis. A default unit type used in limited domain is ***segment_word***. That is, the segment plus the word the segment comes from. This does not mean use of word concatenation for the synthesis. Selection is still based on phone units, but the phones are differentiated depending on the word they come from. This process both cuts down the search and improves synthesis quality.

As mentioned earlier, limited domain systems can only synthesize words which are in their vocabulary or their database. For words which are not in their database, the system usually uses a default back-up voice which is a robotic-sounding diphone voice. This voice, which is the explicitly defined **closest_voice**, is used when the limited domain synthesis fails. This back-up voice will synthesize not just the OOV (out-of-vocabulary) word(s); it will synthesize the whole utterance which has these OOV words. This is not what is usually required and thus, a hybrid system which will use an alternative voice for OOV words only is a better solution.

The limited domain synthesis technique is slower than the diphone synthesizer. This is due to the fact that the cost of finding the optimal join in selected units is higher for limited domain synthesis. Also, this synthesizer technique requires more memory than diphones as the cepstrum parameters for the whole database are required at run time, in addition to the full waveforms [9].

3.4.2.2 Open Domain Unit Selection

Open domain synthesis is also based on cluster unit selection. The idea here is to take a database of general speech and try to cluster each phone type into groups of acoustically similar units based on the (non-acoustic) information available at synthesis time. These units are phonetic context, prosodic features (F0 and duration) and higher level features such as stressing, word position, and accents. The actual features used may be changed and experimented with as can the definition of acoustic distance between the units in a cluster [9].

The content of a speech corpus has a major effect on speech quality. During unit selection process, finding units that best match the target specification is more probable if sufficient prosodic and acoustic variability for the phones can be represented in the corpus. The speech quality is severely degraded when appropriate units cannot be found and significant prosodic modifications are performed.

The work in this thesis builds on CHATR selection algorithm. This cluster algorithm pre-builds CART trees to select an appropriate cluster of candidate phones. In this way, the algorithm avoids the computationally expensive function of calculating target costs (through linear regression) at selection time. As the clusters are built directly from the acoustic scores and target features, a target estimation function is not required, removing the need to calculate weights for each feature. This cluster method uses more generalized features in clustering and uses a different acoustic cost function. A group of candidates is selected and from this, the best overall path selection found through each set of candidates for each target phone.

Basic processes involved in building a waveform synthesizer for the clustering algorithm are as follows:

- Collect a database of general speech;
- Build utterance structures for the database;
- Build coefficients for acoustic distances, typically some form of cepstrum plus F0, or some pitch synchronous analysis (e.g. LPC);
- Build distances tables, pre-calculating the acoustic distance between each unit of the same phone type;
- Dump selection features (phone context, prosodic, positional and whatever) for each unit type;
- Build cluster trees using 'wagon' with the features and acoustic distances dumped by the previous two stages;
- And finally, build the voice description itself.

These processes are explained in detail in Chapter 5 (Implementation).

3.5 Intonation in Festival

One difficulty often faced by intonation researchers is how to extract *suitable* F0 values from acoustic utterances [9]. Due to the nature of intonation research, an accurate pitch determination algorithm (PDA) is probably not enough. According to Sun [25], intonation research often requires a continuous and smooth F0 contour, namely "intonation contour". The abrupt jumps resulting from effects like glottalization, which are not excluded in this case, are meaningless in intonation. Sun further states that most intonation research focuses on the pitch movement patterns on a larger domain. Also, in some cases, these variations may not be intended by the speaker to convey intonation related information. Conventional approaches usually try to eliminate these local pitch jumps by using methods such as smoothing, dynamic programming, EGG signal comparison, and manual correction, etc. Smoothing is probably the most widely used, particularly median smoothing, which is quite effective in removing local outliers. In this thesis, the intention is to mask the glitches by intonation application.

Festival supports different modules of intonation which will be discussed in the next chapter. The implementation on this thesis is based on CART/LR (Cart And Regression Tree/Linear Regression) module for intonation modeling.

3.6 Summary

This chapter gave an introduction to a Festival speech synthesis system as one of the commonly used TTS engines. It then explored requirements necessary for installation and compilation of Festival. The architecture of Festival, which is mostly based on the utterance structure, was also discussed. Implementation of Festival is dependent on concatenative synthesis method and its sub-types. These methods, together with their implementation, were surveyed. The chapter concluded by looking at intonation and how it is implored in Festival.

Chapter 4

4. Intonation

Intonation is an ambiguous word whose meaning has different explanations depending on an individual's perspective of the word. This chapter tackles different meanings before dwelling on one explanation which is explored in this thesis. The chapter then proceeds to inspect the prosodic structure, an aspect which intonation is part of. Different intonation modules are examined and how fundamental frequency (F0) is generated and predicted as an intonation feature. The chapter concludes by investigating the techniques in statistical modeling.

4.1 Intonation and Meaning

There are various terms associated with intonation, the common ones being prosody, supra-segmental, pitch accent and tune. These terms are usually used interchangeably with intonation. According to Clark [28], these features are related to intonation in the following manner:

- Intonation is considered to be that part of an acoustic speech signal that cannot be accounted from the segmental structure of an utterance alone. This component of speech is known as supra-segmental.
- On the other hand, prosody describes the way in which something is being spoken. In this case, intonation differs from prosody in that it is a quantifiable entity.
- Pitch accent is when emphasis is put more on some words than others in a spoken utterance. Interestingly, intonation is how the pitch pattern or fundamental frequency changes during speech [16].

Of all the above terms, prosody and pitch are those frequently taken to mean intonation, or used interchangeably with this entity. Intonation is sometimes used to refer to pitch patterns brought about by prosody. Other people refer to both pitch patterns and the underlying prosody [28].

As it can be seen from these associations, it is not easy to differentiate these entities completely apart. The reason for this is that the properties they describe are "often very difficult to tease apart from the segmental structure." In this thesis, intonation is defined as being equivalent to fundamental frequency, F0, or the pitch pattern.

The next section investigates prosody, which is considered as the main cover under which supra-segmental features fall, intonation included.

4.2 Prosody

Finding correct intonation, stress, and duration from written text is probably the most challenging problem for years to come [16]. These features together are called prosodic or supra-segmental features. The prosody of continuous speech depends on many separate aspects, such as the meaning of a sentence, the speaker characteristics and emotions. Prosodic dependencies are shown in Figure 4-1. However, written text usually contains very little information of these features and some of them change dynamically during speech. With some specific control characters, this information may be given to a speech synthesizer.

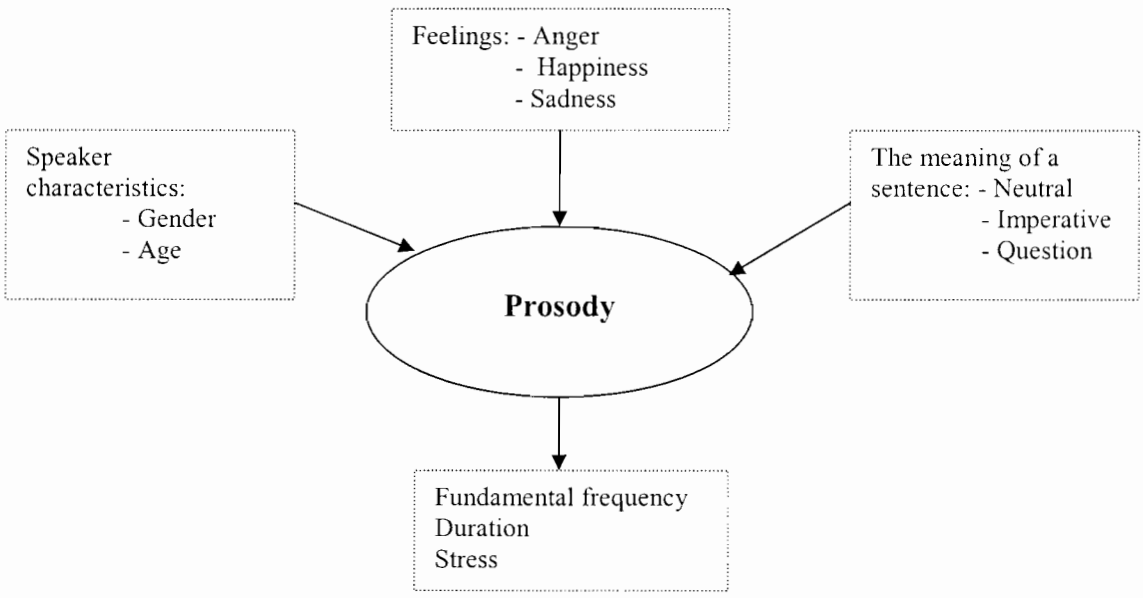


Figure 4-1: Prosodic dependencies [16]

4.2.1 Prosody as High-Level synthesis

Prosodic features consist of pitch, duration, and stress over time. A good control of these can assist in the modeling of other features in speech, though accurate modeling would be difficult [16]. Prosodic features can be divided into several levels such as syllable, word, or phrase level. For example, at word level, vowels are more intense than consonants. At phrase level, correct prosody is more difficult to produce than at the word level.

The pitch pattern or fundamental frequency over a sentence in natural speech is a combination of many factors [16]. The pitch contour depends on the meaning of the sentence. For example, in normal speech, the pitch slightly decreases toward the end of the sentence and in a question, the pitch pattern will rise at the end of the sentence. The pitch contour is also affected by gender, physical and emotional state, and attitude of the speaker.

Duration can be explored at several levels; these include phoneme duration, sentence level timing, speaking rate, and rhythm. The segmental (phoneme) duration is determined by a set of rules to determine correct timing. These rules are based on the neighboring phonemes. At sentence level, speech rate, rhythm, and correct placing of pauses for correct phrase boundaries are important.

Emotions of a speaker can affect speech in various ways. If these are implemented properly in synthesized speech, the quality of the speech output can be increased considerably. However, this implementation is not easy in TTS systems because written text does not give information on emotions. Nevertheless, this kind of information may be provided to a synthesizer with some specific control characters or character strings [16]. The users of speech synthesizers may also need to express their feelings in "real-time". For example, deaf people cannot express their feelings when communicating with speech synthesizers through a telephone line.

Looking at some basic emotional states that affect voice characteristics, the voice parameters affected by emotions can be categorized into three types [16]:

- Voice quality which contains largely constant voice characteristics over the spoken utterance, such as loudness and breathiness. For example, an angry voice is breathy, loud and has a tense articulation with abrupt changes while a sad voice is very quiet with a decreased articulation precision.
- Pitch contour and its dynamic changes carry important emotional information, both in the general form for the whole sentence and in small fluctuations at word and phonemic levels. The most important pitch features are the general level, the dynamic range, changes in overall shape, content words, stressed phonemes, emphatic stress, and clause boundaries.
- Time characteristics contain general rhythm, speech rate, lengthening and shortening of stressed syllables, length of content words, and the duration and placing of pauses.

4.2.2 Prosodic Structure

Usually, the goal in the study of intonation is to find and describe an accent of a given type in a given place. In speech synthesis, though, this is not enough. An accent should be able to be recreated such that it can be placed with respect to its prosodic context in a given speaker's range. Some insight, therefore, of what prosody is, its underlying prosodic structure and how it relates to pitch range is essential.

Prosodic structure is used to account for high level patterns in intonation, such as the difference in pitch range between two phrases of the same utterance [28]. There are different theories regarding the representation of the prosodic structure and two of them are explained below.

4.2.2.1 Hierarchical Structure

The hierarchical structure deals mainly with two questions [28]:

- To what size chunks of speech do tune and pitch range apply?
- How do these chunks relate to each other?

Intonational phrase (IP) refers to these chunks of speech, but there are different ideas concerning how big such a chunk is and what it should be called. According to Ladd, as referenced by Clark [28], definitions such as phonemic clause, macro-segment, tone group, breath group and intonational phrase, all mean one and the same thing. He further summarizes these definitions as the largest phonological chunks into which utterances are divided. "*They extend from one phonetically definable boundary to the next, have a specifiable intonational structure, and are phonological units which are assumed to relate to syntactic or discourse level structure.*" However, Ladd points out that there is a problem with the definition of boundaries: "*the general assumption is that the domains over which phonological structure is specified are defined by the audible phonetic boundaries which occur in the speech stream. If something is structurally an IP then it is assumed to have boundaries, and if something has boundaries it is assumed to be an IP, leading to a somewhat circular definition.*" Two such IP levels of phrasing are generally assumed. These are proposed in various ways, but basically they consist of big intonation phrases (IPs) containing little intonation phrases (***ip***s) [28]: e.g. single and double bar boundaries, major and minor tone groups and intermediate phrases and intonational phrases. A general definition is that the little ***ip*** has a nuclear structure, that it contains one primary stressed unit, and the big IP consists of little ***ip***s and has an audible break associated with it.

4.2.2.2 The Strict Layer Hypothesis (SLH)

It is generally assumed that prosodic structure is non-recursive [28]. This means that any given level of structure has to be made up only of units of the level below it. This idea, though, is questionable as shown in the following example:

My brother, who was ill, is out of hospital. (1)

As revealed by Clark in his thesis [28], Cooper & Sorensen (1981) found that declination is interrupted by the parenthetical and continues after it as if it was not there. This suggests the structure:

[My brother [who was ill]$_{MP}$ is out of hospital]$_{MP}$ (2)

rather than:

[My brother] [who was ill]$_{MP}$ [is out of hospital]$_{MP}$ (3)

(MP is a major phrase which is set off by audible prosodic breaks.)

is more applicable to the SLH structure.

This remains an open question of whether prosodic structure is really non-recursive.

4.3 Intonation Modules

This section discusses different modules of intonation, most of which are applicable in Festival. The section begins by focusing on accent and boundary tones which are the two main types of intonation event. Different modules in intonation are then investigated.

4.3.1 Accent/Boundary Placement

Accent and boundary tones are used to refer to two main types of intonation event. For most languages, the prediction of position of the accents and boundaries can be done as an independent process from F0 contour generation itself [18]. In general, intonation is split between accent placement and F0 generation. Accent position influences durations and an F0 contour cannot be generated without knowing the durations of the segments the contour is to be generated over.

Accents also align with the segmental material of an utterance in a particular way. This alignment is based on the *sonorant rhyme* (or *s-rhyme*) which is defined as [28] "being the part of the syllable from first non-initial sonorant through to the end of the last sonorant. This unusual definition of the s-rhyme gives an idea of the level of detail at which the syllabic structure needs to be examined, suggesting that the relationship between pitch event and syllabic structure is not necessarily a simple one."

Although accent and boundary types have been identified for various languages and dialects, a computational mechanism for generating an F0 contour from an accent specification often has not yet been specified. As a result, this thesis does not implement accent modeling, even though durations depend on accent.

4.3.2 Default Intonation

This is the simplest form of intonation which offers the modules Intonation_Default and Intonation_Targets_Default. The first module actually does nothing, while the second one simply creates a target at the start of the utterance, and one at the end. The default target values are 130 Hz and 110 Hz respectively.

4.3.3 Simple Intonation

This module uses the CART tree to predict if each syllable is accented or not. A predicted value of NONE means no accent is generated by the corresponding *Int_Targets_Simple* function. Any other predicted value will cause a 'hat' accent to be put on that syllable [10].

The function *Int-Targets_Simple* uses the parameters ***f0_mean*** and ***f0_std.*** ***f0_mean*** gives the mean fundamental frequency for a speaker (default 110 Hz) and ***f0_std*** is the standard deviation of the speaker (default 25 Hz). The second value is used to determine the amount of variation to be put in the generated targets.

4.3.4 Tree Intonation

This module uses two different CART trees to predict accents and endnotes. Although at the time of writing, this module was used for an implementation of the ToBI intonation (described below) labeling system, it could be used for many different types on intonation system. The target module for this method uses Linear Regression (LR) model to predict start, mid-vowel and end targets for each syllable using arbitrary specified features.

4.3.5 General Intonation

Due to a vast number of intonation theories that predict F0 by rule, this module aids the external specification of such rules for a wide class of intonation theories [10]. This is designed to be multi-lingual and offers a quick way to port often pre-existing rules into Festival without writing new C++ code.

The accent prediction part uses the same mechanisms as the Simple intonation method described earlier. It uses a decision tree for accent prediction which, in the variable *int_accent_cart_tree*, is used on each syllable to predict an IntEvent. The target part calls a

specified Scheme function which returns a list of target points for a syllable. In this way, arbitrary tests may be done to produce the target points.

4.3.6 Tilt Intonation

Tilt intonation is a model orientated towards speech technology. It is a descriptive model which provides a parameterized representation of change in pitch related to intonation events and takes a bottom-up approach [2, 23]. It therefore makes very few assumptions about the underlying intonation theory. The model assumes that pitch events occur in a linear fashion at given times and have distinct starts and ends. It is appropriate to describe an intonation theory based around peaks and troughs (pitch movements) or rises and falls (pitch targets) [28].

A tilt labeling of an utterance consists of a set of pitch events: accent and boundary labels which are identified with syllables. Each event is characterized by a set of five independent parameters which describe pitch movement:

- **Amplitude -** Amplitude of an event.
- **Duration -** Duration of an event.
- **Tilt -** A measure of the shape on the interval [-1:1]. -1 is a pure rise, 0 is a rise-fall and +1 is a pure fall.
- **Position -** A measure of the F0 position relative to a baseline (usually 0 Hz).
- **Time -** A measure of time position of an event.

The amplitude and duration parameters can be extracted directly from a labeled intonation contour. The time and position parameters are flexible in what they are measured relative to. Time is usually measured relative to the start of the vowel that the pitch event is assigned to, while position is measured with relative to the distance from the start of the utterance.

The tilt model uses a simple set of labels to identify accents and boundaries, **a** and **b** respectively. **a** signifies all types of pitch accent and **b** signifies boundary. The boundary, **b**, can be distinguished by **rb** and **fb** which are rising and falling boundaries in respective order.

4.3.7 ToBI

ToBI (Tone and Break Index) is arguably the most popular representation of F0 contours, among both the linguists and speech synthesis community [2]. The ToBI system was originally proposed for American English and is based on the work by Pierrehumbert [2, 23]. Speech synthesis from the ToBI framework is implemented in the Festival Speech Synthesis System.

The labeling scheme consists of [23]:

- 6 discrete intonation accents types: H*, !H, L*, !L, L*+H, and L+H* (H = high tone and L = low tone).
- 2 phrase accent types: H- and L-
- 4 boundary tones: L-L%, L-H%, H-L% and H-H%
- 4 break levels: 1, 2, 3, and 4. This tier is used to mark breaks on a 0 to 4 scale (0 to 6 in an extended version) which measure the strength of association between adjacent words. The break indices with values of 3 and above relate to prosodic boundaries. The 3 is a `-' boundary and 4 to a `%' boundary. In the extended version of the system, 5 is used for `%' boundaries stronger than those marked 4, such as those with particularly long pauses, and 6 is used to signify end of an utterance.
- A HiF0 marker which marks the highest F0 peak in each intonational phrase.
- A miscellaneous tier is used for marking hesitations, disfluencies, non-speech and the like.

ToBI is a framework for developing community-wide conventions for transcribing the intonation and prosodic structure of spoken utterances in a language variety [29]. This framework though, has a weakness in achieving consistency required for training statistical models. The inconsistency is brought about by ambiguity in the tonal tier and the break index tier. The number of different types of accents in Pierrehumbert's theory and the similarity between certain accents, or at least the similarity in accents when realized in speech, presents a problem [28]. Making a distinction between accent shapes is not always easy. The F0 traces seen do not always exhibit clear accent shapes as expected, so it is not always obvious what type an accent is or even if there is an accent at all.

Although systems of classifying the F0 contours such as ToBI are popular, they are language and dialect dependent. Moreover, it is difficult to reliably and automatically classify F0 contours [2].

4.4 F0 Generation and Prediction in Festival

This section discusses how fundamental frequency is generated and predicted in Festival.

4.4.1 F0 Generation

There are various theories for F0 generation, each with various advantages and disadvantages. Different methods have been researched for multiple languages.

Fujisaki [61] built an F0 contour from parameters per prosodic phrase and obtained good results for simple declarative (Japanese) sentences. Taylor [69] has a more general system which can automatically label contours with parameters and generate contours from these parameters for a wide range of intonation tunes.

The task of generating a contour from accents is much better defined than generating the accent placement or their types. Festival supports a number of methods which allow generation of target F0 points. These target points are later interpolated to form an F0 contour. Many theories make strong claims about the form of interpolation [9]. The simplest model adds a fixed declining line. This is useful when intonation is being ignored in order to test other parts of the synthesis process.

There are three basic F0 generation modules available in Festival and these are: by general rule, by linear regression/CART, and by Tilt.

4.4.1.1 F0 by Rule

In this method, target points are created for each syllable in an utterance. The idea follows closely a generalization of the implementation of ToBI type accents, where n-points are predicted for each accent. These accents form a baseline and place target F0 points above and below that line based on accent type and position in phrase. The baseline is defined to decline over the phrase reflecting the general declination of F0 over type.

A simple idea behind this general method is that a Lisp function is called for each syllable in the utterance. That Lisp function returns a list of target F0 points that lie within that syllable. Thus, the generality of this method actually lies in the fact that it simply allows a user to program anything they want [9].

4.4.1.2 F0 by Linear Regression

The idea is to find an appropriate F0 target value for each syllable based on available features by training from data. A set of features are collected for each syllable and a linear regression model is used to model three points on each syllable. The technique produces reasonable synthesis and requires less analysis of the intonation models that would be required to write a rule system using the general F0 target method described in the previous section.

However, though this technique is simpler, there are a number of intonational phenomena which it cannot capture. For instance, it cannot capture multiple accents on syllables and, it also cannot get accents placement with respect to the vowel. The previous technique allows specification of structure but without explicit training from data, while this technique imposes almost no structure but depends solely on data [9].

The advantage of the linear regression method is that very little knowledge is needed about the intonation of the language under study.

4.4.1.3 Tilt Modeling

In Tilt modeling, a tilt parameterization of a natural F0 contour is automatically derived from a waveform and a labeling of accent placements (a simple **a** for accents and **b** for boundaries).

For each **a** in a labeling, four continuous parameters are found: height, duration, peak position with respect to vowel start, and tilt. Prediction models are then generated to predict these parameters which capture the dimensions of an F0 contour.

4.4.2 F0 Extraction

The fundamental frequency is the basic tune in speech, ranging between about 90 Hz and 120 Hz and about 140 Hz to 280 Hz in females [23]. In general, an F0 starts higher at the beginning of a sentence and decreases towards the end of the sentence.

A typical F0 contour looks like Figure 4-2 below.

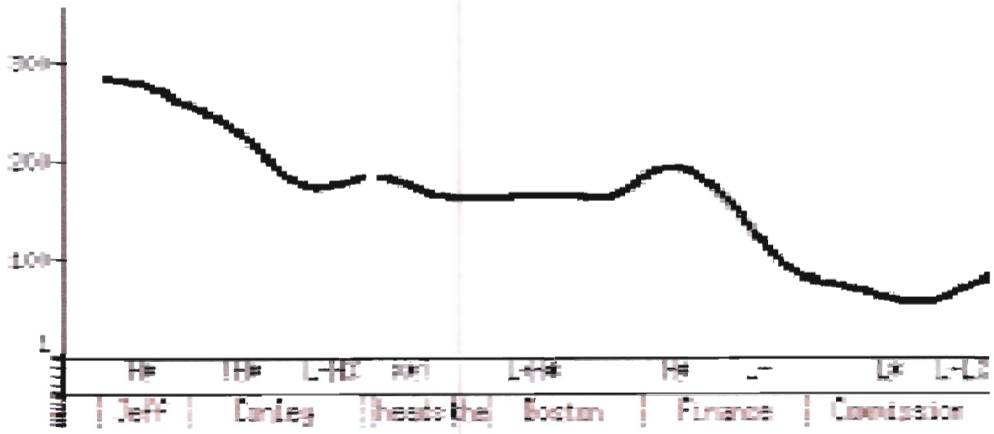


Figure 4-2: A typical F0 contour shape (with ToBI labeling) [23]

Figure 4-2 shows an F0 contour for an utterance "Jeff Conley heads the Boston Finance Commission".

An F0 contour is not usually as smooth as depicted in Figure 4-2. First, there are only pitch periods during voiced speech, second certain segments (particularly obstruents such stops), can locally affect the pitch so that the underlying natural pitch may actually look like Figure 4-3 below.

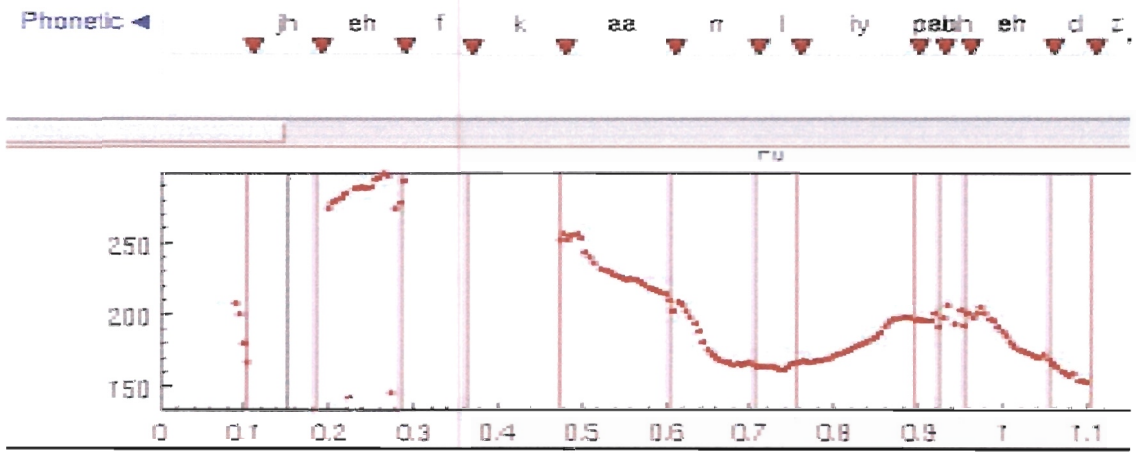


Figure 4-3: An F0 contour with obstruents [23]

From Figure 4-3, there is no F0 during the unvoiced phones, /f k/. Also, there is a variable F0 value around the phone /hh/ of the word "heads" which, although might be considered to be unvoiced, has pitch periods near it.

It is useful to have a clear representation of F0 when modeling it. The best tool to use in order to obtain perfect results is an electroglottograph (EGG); extraction from raw wave signals is not as accurate.

F0 extraction has two basic uses: getting the whole contour and signal processing. In order to get the whole contour, smoothing is required to fill unvoiced sections (by interpolation) and to smooth out segmental perturbations. In this case what is required is a track of values at fixed intervals which give the approximate F0 values throughout the utterance.

In signal processing, each pitch period is identified exactly. Many signal processing techniques that can modify pitch and duration independently work on the pitch periods of a signal [23]. It is easier to get an estimate of the pitch over a section of waveform than to reliably get the pitch period properly identified. As a result, F0 modeling is often done from pitch contours extracted from a waveform. However, it is harder to perform a clean signal processing without an EGG signal, even though it is desirable not to rely on the extra piece of information.

4.4.3 Research on F0 Modeling

There is an ongoing research on intonation and F0 modeling, and this section mentions some of the people involved in this research.

The Fujisaki model [61] is a bottom up data-driven model. It was originally developed for Japanese, though it also has a German model as implicated by Moebius [62]. In the Fujisaki model, intonational phrases are represented by a phrase command and an accent command, and a number of accent commands may exist in each phrase [25]. Here, a command consists of amplitude and a position. The model is a critically damped impulse module where the F0 decays over time based on the energy inserted by phrase and accent commands.

Moehler [63] offers a model that uses both the nothings of data-driven as in Tilt and the Fujisaki model, but also the advantages of a phonology model like ToBI. His parameterization of F0 is closely similar to Tilt, though he also clusters different types of intonation contours by vector quantization.

Ross [64] uses a dynamical model to represent F0 and has good results on the Boston University FM Radio corpus.

Taylor [65, 66] developed Tilt/RFC (Rise/Fall/Connection) models for speech synthesis and ease of automatic labeling. In this scheme, linguistically relevant parts of the F0 contour are called intonation events and they are linked to a syllable from the text [2]. The intonation events are described by the rise and fall components parameterized by the amplitude and duration. The intervals between the intonation events are called connections and are again parameterized by the amplitude and duration. The Tilt model reduces the RFC parameters to a single dimensionless tilt parameter that describes the F0 shape of an event, as well as the event amplitude and duration parameters.

In the synthesis stage, the tilt parameters can be easily predicted and unpacked into the RFC representation, which can be used to synthesize the F0 contour. Taylor [66] argues in favor of continuous representations such as the Tilt model compared to the more traditional discrete representations such as ToBI. In comparison with the Fujisaki model, Taylor [66] points out that the phrase component of the Fujisaki model restricts the global F0 contour to a gentle fall over the prosodic phrase. This is too constrained for some English utterances where the F0 contour can gradually rise [2].

4.5 Statistical Modeling Techniques

A statistical basis for a system can achieve reliability and consistency with a certain level of accuracy which is considered a safe compromise. This is in comparison to rules which may perform particularly well in many situations, but fail in a small number of unpredictable circumstances.

Building a statistical model involves training on a corpus of data, where the model "learns" an association between an input which is generally a parameter vector representing an entity, and an output which is the entity itself. Once trained, the model can generate a suitable entity from a given input vector.

In speech synthesis, techniques used in building statistical models are neural networks (NN), linear regression (LR), and classification and regression trees (CART). In this thesis, we pay particular attention to LR and CART models. Neural networks are not discussed as they are not a popular approach in intonation generation.

4.5.1 Linear Regression

Linear regression models assume that a predicted variable (p) can be modeled as the sum of a set of weighted real-valued factors.

$$p = w_0 + w_1 f_1 + w_2 f_2 + w_3 f_3 + \ldots + w_n f_n \qquad (4)$$

The factors (f_i) represent parameterized properties of the data, and the weights (w_i) are trained using a stepwise least squares linear regression technique.

Often subsets of mutually exclusive binary valued factors are used. This partitions the model into parts where only a subset of weights contributes towards the predicted variable in a given context. For example, w_1 and w_2 may contribute to unaccented syllables.

A simple example of a linear regression model is shown in Figure 4-4, taken from [28]. Here F0 is predicted by the presence of an accent or a boundary and the number of syllables from the start of an utterance. An f_1 component causes a peak on accents, and an f_2 component causes a dip at a boundary. If a syllable has no pitch event associated with it, then both f_1 and f_2 are zero valued so their associated weights make no contribution to the predicted F0. f_3 is always positive and causes the F0 to decline through the utterance.

pitch F0 = 50 + 15f_1 + -5f_2 + -0.5f_3
f_1 Syllable is accented {0,1}
f_2 Syllable is a boundary {0,1}
f_3 Number of syllable from start of Utterance [1,n]

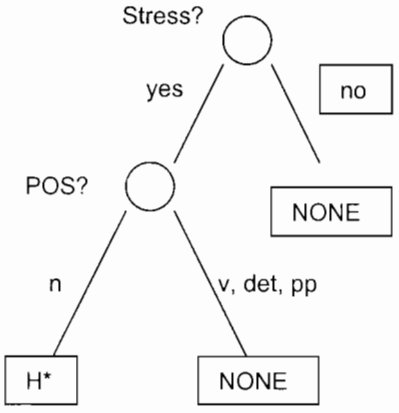
Syl	Event	$pitch_{F0}$
2	None	49
6	Accent	62
8	None	46
12	Boundary	39

Figure 4-4: **An example of a simple linear regression model**

In Figure 4-4 above, the linear model on the left is used to calculate pitch targets for a selection of syllables from a hypothetical utterance of twelve syllables, where there is an accent on the sixth syllable and a boundary after the last syllable.

4.5.2 Classification And Regression Trees

A Classification And Regression Tree (CART) model is a binary decision tree which is widely used for intonation predictions. It is the preferred method in the Festival speech synthesis system. A CART tree can be grown or trained on actual data. When a feature vector is fed into a trained CART tree, a vector is tested at each node against a criterion such as phoneme type or position of the segment within the phrase. The outcome of the test is either yes or no and the data is dropped down the appropriate branches until a terminal node is reached or no further branching can occur because of missing data. At this point, a prediction is made based on the number of training data clustered in that node. In the training phase, the decision for splitting the node is made locally, so trees generated by CART are not globally optimal. The strength of CART lies in its ability to classify data with '*no clear notion of similarity (or metric)*' [70]. It accepts a mixture of numerical data as well as non-metric data (like phoneme class). The popularity of CART in intonation research stems from the ease of interpreting the decision process and the possibility of incorporating heuristics into the system.



Word:	sat	mat
POS:	v	n
Stress:	yes	yes
Punc:	None	.
Accent Class:	NONE	*H**

Example: "The cat sat on the mat."

Figure 4-5: **An example of a CART prediction model [28]**

The leaves of the decision tree specify the class a target case falls into, and each node of the tree contains a binary decision based on one of the recorded parameters for the case. Figure 4-5 shows a simple CART intonation prediction model. The sample sentence is "The cat sat on the mat." Each syllable is parameterized by the features: POS (part of speech), Stress and Punc (following punctuation). The tree classifies accent class for each syllable. Parameter values for the words 'sat' and 'mat' are shown. 'Mat' receives an accent but 'sat' does not. A point to note is that the tree does not necessarily use all of the parameters in its decision making process. It may also use different parameters for different circumstances depending on which branches are taken.

In this example, the Punc feature is not used at all and the POS feature is only used on the left branch of the tree. The example tree here was constructed by hand by Clark [28], but trees in general are constructed by a data driven training process. For classification, this works by repeatedly partitioning the data into subsets of classes, where each split adds a node in the tree until all classes are accounted for individually at the leaf nodes. In reality, the leaves of the tree tend to list probabilities of a case being in each class, and the class with the highest probability is chosen to classify an unknown case. In the example above, the restricted set of classes representing accent type is `H*' and `NONE'.

For classification, the partitioning is performed to find the division which best partitions the data so that cases of the same class are placed in the same subset. For regression, the partitioning is done to minimize the error in the predicted variable. The value of the predicted variable is the mean value of the class a case is classified as. If Figure 4-5 were being used for regression, each class would be accompanied by a mean and standard deviation which are calculated from individual members of that class in training.

CART models have an advantage over linear regression models in that they are non-linear and as a result, they can handle non-linear data better. This, though, does not mean they do not have weaknesses. CART models do not guarantee the best possible solution, just a local maximum, and a classifier may well classify a proportion of its training data incorrectly. Their efficiency however, usually makes up for these drawbacks.

4.6 Summary

This chapter has shown how intonation can have different meanings based on the other prosodic features which are sometimes used interchangeably with it. Prosody and its structure were investigated and the different modules in intonation. The chapter further looked into the F0 generation and prediction as carried out by Festival system, and the development of some ongoing research on intonation. The last aspects discussed were the popular statistical modeling techniques in speech synthesis: linear regression and CART models.

Chapter 5

5. Implementation

Now that we have got some insight of what text-to-speech synthesis is, its techniques, and its relation to intonation, we look at how to implement these techniques using the resource tool, the Festival system. This chapter describes the processes involved in designing two hybrid Sesotho TTS systems – one as a baseline system and the other with intonation implementation. Intonation is implemented as a way of masking the glitches in the advanced hybrid system thus leading to a more natural sounding and fluent system. This was achieved by building voices based on unit selection synthesis and bringing the systems together to form a hybrid system. In building the voices, the procedure followed was the one in "Building Voices in the Festival Speech Synthesis System" [9].

5.1 Baseline System - Design of Sesotho Advanced TTS System1

The first Sesotho TTS system built is a hybrid system of limited domain and open domain systems. The procedure is basically the same as that by Rousseau [4] even though the limited domain/diphone combination was not implemented. The main idea is to get a hybrid system with the best qualities - naturalness, flexibility, pleasantness, intelligibility and fluency. The reason for not building a limited domain/diphone hybrid is that, for this thesis, the feature of most importance is naturalness, which this hybrid system does not possess. The diphone voice sounds robotic. The Sesotho diphone voice was built as it would be needed for synthesis prompting in building the other unit selection voices.

5.1.1 Generic Method in Building Voices

Even though the voices built in this thesis are different, they have some steps which can be regarded as general. The steps below give a procedure of the common steps in building synthetic voices. These steps will be explained in detail in the respective voice building processes if there are changes to be done.

It must be noted that by convention in building voices in Festival, a voice name consists of an institution name, a language identifier, and a speaker identifier. For instance, for this thesis, the institution name is **UCT** (University of Cape Town), the language is **Ses**otho, and the speaker, who is the author, is Le**hlo**honolo. Therefore, following this convention, for a diphone voice, the voice will be ***uct_ses_hlo_diphone***. This convention has been followed, though not for all voices and the reason for this will be given in the respective voice-building procedure.

The general procedure for building a voice is given in the following steps:

- Setting the environment variables

  The first step is to set the environment variables, FESTVOXDIR and ESTDIR, to point to their respective directories. These are the directories into which Festvox and Edinburgh Speech Tools are installed. This is done by the commands (in the case of this project):

```
export FESTVOXDIR=/home/hlonie/festvox
export ESTDIR=/usr/
```

  A name is then selected which will contain the directory for voice building. For instance, for an open domain (cluster units) voice the directory will be ***uct_ses_hlo_clunits.*** The template files for voice building are then copied in by the command (after changing into this directory):

```
$FESTVOXDIR/src/unitsel/setup_clunits uct ses hlo
```

  This command will differ depending on the voice being built, but the concept is the same.

- Designing the database and prompts

  In designing prompts for a limited domain voice, a section of the vocabulary was taken from the opening part of the speech given by the Prime Minister of Lesotho. This section had some of the words repeated three or four times in order to improve the quality of the output speech. The speech was taken from [31]. The database consisted of 400 utterances and was used for all voices. More utterances were obtained from various sources, namely, a Sesotho drama book [26], a Sesotho book on idiomatic expressions [32], the national anthem of Lesotho and the rest were randomly made up by the author. The bigram coverage was maximized by using different word-word pairings in the database.

The utterances were first put in a prompt file with an utterance number and the prompt in double quotes as required by Festival thus:

(time0001 "lesotho fatshe la bo-ntata rona.")
(time0002 "hara mafatshe le letle ke lona.")
...

These prompts were put in **'etc/time.data'** file.

As required by Festival, each utterance line was enclosed in braces, just like Festival commands. The braces contained two items, an identifier for the utterance based on the choice of domain and a number, and the utterance itself in double quotes.

- Customizing the front end

  In building the unit selection voices (limited domain and open domain), the "closest voice" has to be changed before synthesizing the prompts. This is where the diphone voice comes in because the "closest voice" is the voice that is used for synthesis of the prompts. In our case, we need a Sesotho diphone voice as the prompts will be synthesized using appropriate Sesotho phones which were generated during the building process of the Sesotho diphone voice. If the synthesizing voice is left as default, (***voice_kal_diphone***), an English phone system will be used for synthesis. It is only after this change has been done that the prompts can be generated.

- Recording the prompts

  According to [9], the best way to record the prompts is to use a professional speaker in a professional recording studio using dual channel and a high quality head-mounted microphone. In our case, the recordings were done by the author, who is a native Sesotho speaker from Lesotho. The author is not a professional speaker, neither is the room in which the recordings were done. The recordings were done with a standard PC using a standard sound blaster type 16bit audio card and a head-mounted microphone. The test utterances were 16 bit linear speech files with a sampling frequency of 16 kHz.

- Labelling the prompts

  This is the part where the recorded prompts are aligned against the synthesized prompts. This is done automatically by using the aligning technique, dynamic time warping (DTW). Dynamic time warping is a technique applied in automatic speech recognition to cope with different speaking speeds. In general, it is a method that allows finding an optimal match between two given sequences (e.g. time series) with certain restrictions, i.e. the

sequences are "warped" non-linearly to match each other [33]. This sequence alignment method is often used in the context of hidden Markov models.

Although this technique works, it is worth checking if the prompts are properly labeled. The labeling is checked using ***wavesurfer***, and the prompts which are not properly aligned are manually labeled. Wavesurfer is an open source tool for sound visualization and manipulation. It is a general tool for analyzing, synthesizing, and manipulating speech.

- Building utterances

  Once all the prompts have been properly labeled, an utterance structure for the spoken utterances is constructed. This is done by combining the basic structure from the synthesized prompts and the actual times from the automatically labeled ones. This is done by the command:

  ```
  festival -b festvox/build_ldom.scm '(build_utts
  "etc/time.data")'
  ```

- Extracting pitchmarks

  Getting good pitchmarks is important to the quality of the speech synthesis output. The best form of pitchmark extraction is through the use of an electroglottograph (EGG) signal, which is exact. The EGG records electrical activity in the glottis during speech, which makes it easier to get the pitch moments, and so they can be more precisely found [9]. In our case, we did not have an EGG signal and therefore we opted for extracting the pitchmarks from the waveform itself. In comparison to an EGG signal, extraction from the waveform is computationally more expensive.

  We used a Speech Tools program 'pitchmark' for extraction. This method uses parameters to specify the range of the pitch periods. These periods were modified to best match the speaker's range (-min and –max), a female speaker in our case. These were set to 0.0033 (303 Hz) and 0.0055 (181 Hz). The filter parameters were left as 0.01.

  There are a number of parameters to the 'pitchmark' program for which values must be selected. The speech is first low-pass filtered to remove unwanted high frequencies (**-lx_lf** parameter) then high-pass filtered to remove unwanted low-frequency noise (**-lx_hf** parameter). Pitch marking is then performed and finally a post-processing step to remove suspicious pitchmarks (e.g. too close together) and fill in unvoiced regions. The **-min**, **-max** and **-def** parameters are for the post-processing step.

Table 5-1: Pitchmark parameters [34]

parameter	what it means	value
-lx_lf	cutoff frequency for low-pass in Hz	the highest value you expect for F0, e.g. 175
-lx_hf	cutoff frequency for high-pass filter in Hz	the lowest value you expect for F0, e.g. 80
-min	minimum pitch period in seconds	1/F0, e.g. 0.0057 (i.e. F0 of 175Hz)
-max	maximum pitch period in seconds	1/F0, e.g. 0.012 (i.e. F0 of 80Hz)
-def	F0 for filling in unvoiced regions	0.01 - do not change

The correct procedure for tuning parameters for pitchmarking is as given by [34]:

- Remove the ***-fill*** option to 'pitchmark' in the ***bin/make_pm_wave*** script. This turns off the filling in of false (100 Hz) pitchmarks in unvoiced regions and helps in making one see more easily whether the pitchmarking is working well.
- **Repeat**
    - Set the parameters to pitchmark in the ***bin/make_pm_wave*** script, as described in the Table 5-1.
    - Carry out the commands:
        - `bin/make_pm_wave wav/*.wav`
        - `bin/make_pm_fix pm/*.pm`
        - `bin/make_pmlab_pm pm/*.pm`

    The pitchmarks are now in label file format in ***pm_lab/***. These can be examined with wavesurfer.
- **Until** the pitchmarks are to satisfaction.
- Replace the ***-fill*** option to pitchmark in the ***bin/make_pm_wave*** script.
- Run the commands:
    - `bin/make_pm_wave wav/*.wav`
    - `bin/make_pm_fix pm/*.pm`

The last command is a post-processing stage which moves the pitchmarks to the nearest peak.

Once the pitchmarks have been satisfactorily extracted, the pitch synchronous MELCEP (Mel Cepstral) parameterization of the speech used to build the cluster synthesizer (unit selection voices) is generated. From this, a cluster unit based synthesizer is built from the utterances.

The last stage in building a voice is the testing stage to see if the voice built works as expected.

With the general steps given, the procedure for building each voice is detailed below. The first voice built was the diphone voice as this would be useful in synthesising the prompts for the prospective unit selection synthesis voices.

5.1.2 Building a Diphone Voice

The basic idea behind building a diphone database is to explicitly list all possible phone-phone transitions in a language [9]. The phone-phone transitions for our system were made from five vowels and 31 consonants in Sesotho language. The voice was built as follows:

- Setting up environment variables and the directory structure

  The environment variables, FESTVOXDIR and ESTDIR were set and the directory, ***uct_ses_hlo_diphone*** was set up for holding the diphone voice.

- Defining a diphone list

  The diphone database is a collection of nonsense words that iterate through all the possible combinations of the vowel vowel (VV), vowel consonant (VC) and consonant consonant (CC) combinations of Sesotho language. Festival's Scheme interpreter was used to generate the list. The idea here is that the diphone will come from a middle syllable of the nonsense word so it is fully articulated and minimize the articulatory effects at the start and end of the word.

  For best results, an attempt was made in pronouncing the words with consistent vocal effort, and with as little prosodic variation as possible.

  As mentioned earlier, the directory to build the voice was named according to the identifiers institution, language, and speaker. For this system, the directory name is **uct_ses_hlo**. 'uct' is the institution name, 'ses' is for Sesotho language and 'hlo' is for the speaker, Lehlohonolo. It is important to follow this naming procedure for a directory because this directory name is used when Festival searches for available voices.

  The diphone database was constructed in a particular format to correlate with the tools and scripts included in Festival. Each line contained a file id, and a diphone name. The file id was used in the filename for the waveform, label file, and any other parameter files associated with the nonsense word. An example of the Sesotho diphone database is shown below. The Sesotho database consisted of 362 nonsense words:

  (ses_0001 "pau t a b a b a pau" ("b-a" "a-b"))
  (ses_0002 "pau t a bj a bj a pau" ("bj-a" "a-bj"))

(ses_0003 "pau t a ch a ch a pau" ("ch-a" "a-ch"))
...

- Synthesizing prompts

  Prompts were generated from the diphone database which produces prompts and phone label files that will be used by the alignment tool during labeling.

- Recording the diphones

  Recordings were done by the author in a lab which had 13 computers. A head-mounted microphone was used.

- Labeling the diphones

  The idea behind labeling is to take the prompt and the spoken form and derive Mel-scale Cepstral parameterizations (and their deltas) of the files. This was done by using the dynamic time warping (DTW) alignment tool. During the alignment, a DTW algorithm is issued to find the best alignment between these two sets of features. Then the prompt label file is used to index through the alignment to give a label file for the spoken nonsense word [9]. All this is done by the script **festvox/src/diphones/make_labs** which processes a set of prompts and their recorded form generating a set of label files, to the best of its ability.

  After the labeling of nonsense words was complete, a diphone index was built. The index identifies which diphone comes from which files, and from where [9]. This was done automatically from the label files using a Festival script **festvox/src/diphones/make_diph_index**. This script takes the diphone list and finds the occurrence of each diphone in the label files, and builds an index. The index consists of a simple header, followed by a single line for each diphone: the diphone name, the file id, start time, mid-point (i.e. the phone boundary) and end time. The times are given in seconds. An example of the Sesotho index is:

  EST_File index
  DataType ascii
  NumEntries 617
  EST_Header_End
  pau-pau ses_0362 1.1725 1.225 1.6125
  ntl-u ses_0361 0.82 0.87 0.92
  u-ntl ses_0361 0.7125 0.77 0.82
  ntlh-u ses_0360 0.815 0.87 0.935
  ...

- Extracting the pitchmarks

  Festival supports residual excited Linear-Predictive Coding (LPC) resynthesis, which is a pitch synchronous technique. This means that it requires information about where pitch periods occur in an acoustic signal.

  The Edinburgh Speech Tools program, 'pitchmark', was used for pitchmark extraction from the waveform. The procedure followed was that mentioned in Section 5.1.1 above. This program though, is not fully automatic and the parameters were modified to fit the speaker's range. This is also useful in improving the overall speech output. The aim is to produce a single mark at the peak of each pitch period and "fake" periods during unvoiced regions.

- Building LPC parameters

  Residual LPC is the only publicly distributed signal processing method in Festival [9]. Thus, LPC parameters and LPC residual files were extracted for each file in the diphone database. This was done pitch-synchronously.

- Power normalization

  Power normalization is useful for getting equivalent power for the waveforms. Differing power in the nonsense words occurs if not enough care has been taken in the recording. The idea is to find the power of vowels in each nonsense word, then find the mean power for each vowel overall files. Then, for each file, find the average factor difference for each actual vowel with the mean for that vowel and scale the waveform according to that value.

- Testing

  The voice was then tested to check if it performed the correct synthesis.

5.1.3 Building a Limited Domain Voice

Limited domain synthesis systems have a limited vocabulary and as such their speech output is constrained. These systems are usually built for certain applications where specific words and phrases are used. For our Sesotho system, there was no real target application. Of the first 400 utterances in the database, 36 were taken from the speech given by Lesotho Prime Minister [31] which were greetings targeted at his audience.

In Festival, a simple script is provided for setting up the basic directory structure and copying in some default parameter files. The Festvox distribution includes all the setup for the time domain.

- Setting environment variables and a directory structure

  Environment variables were set as in Section 5.1.1. After setting the environment variables and creating a directory to hold the voice, the directory structure was set up for a limited domain voice. The directory holding the voice is ***uct_time_hlo3_ldom***.

- Designing the prompts

  Prompts were taken from random sources as mentioned in Section 5.1.1. They were put in the prompt file with an utterance number and the prompt in double quotes like this:

  (time0001 "lesotho fatshe la bo-ntata rona.")
  ...

  These prompts were put in **etc/time.data** and they were built by the command:

  ```
  festival -b festvox/build_ldom.scm '(build_prompts
  "etc/time.data")'
  ```

- Recording the prompts

  Recording was done by the same speaker as in the diphone voice, with the same recording equipment and environment.

- Labeling the prompts

  An alignment tool used for labeling was dynamic time warping, as used in the diphone voice above. The labels were then checked and manually corrected using wavesurfer.

- Building utterance structures for recorded utterances

  Once all the prompts had been properly labeled, an utterance structure for the spoken utterances was constructed. This was done by combining the basic structure from the synthesized prompts and the actual times from the automatically labeled ones.

- Extracting pitchmarks and building LPC coefficients

  The same pitchmarking extraction procedure was followed as mentioned in Section 5.1.1. The pitchmarks were then moved to the nearest peak and normalized.

- Building a cluster unit based synthesizer from the utterances

  A cluster unit based synthesizer was built from the utterances.

- Testing and tuning

  The system was finally tested. This synthesiser can only say the phrases that it has phones for. This means it can only synthesise words that are covered in its database, otherwise it will revert to a default voice which is a diphone voice.

5.1.4 Building an Open Domain Voice 1

The procedure in building an open domain (cluster unit) voice is almost the same as that for building the limited domain voice. The idea is to take a database of general speech and try to cluster each phone type into groups of acoustically similar units. These units are based on the information available at synthesis time, such as phonetic context, prosodic features (F0 and duration) and higher level features such as stressing, word position, and accents.

The work builds on the results of the CHATR selection algorithm. This cluster algorithm pre-builds CART trees to select the appropriate cluster of candidate phones. In this way, it avoids the computationally expensive function of calculating target costs (through linear regression) at selection time. As the clusters are built directly from the acoustic scores and target features, a target estimation function is not required, removing the need to calculate weights for each feature. This cluster method uses more generalized features in clustering and uses a different acoustic cost function. A group of candidates is selected and the best overall selection is sought by finding the best path through each set of candidates for each target phone.

The procedure is as follows:

- Setting up a directory structure

  The environment variables were set and a directory for holding the voice created. The directory structure was then constructed for a unit selection voice, this being ***uct_ses_test_clunits***.

- Designing and recording the prompts

  The same database of 400 utterances as that used in the limited domain voice was used. It was ensured that the lexicon (which contains the letter-to-sound rules) and the phoneset (which contains the list of Sesotho phones and their features) lists used here were the same as that for the diphone voice. The prompts were recorded by the same speaker under similar circumstances.

- Labeling the prompts

  The prompts in the database were phonetically aligned by a speech recognition tool, dynamic time warping tool. The phone boundaries were manually corrected using wavesurfer. A cluster unit utterance structure was then built from the labeled prompts.

- Extracting pitchmarks

  The same procedure for extracting pitchmarks was followed as for other voices.

- Making cepstrum parameter files (MFCCs)

  In order to cluster similar units in a database, their acoustic representation was built. This was achieved through the use of Mel cepstrum plus delta Mel cepstrum plus F0.

  The script used for generating these parameters is in **festvox/src/unitsel/make_mcep**. The main loop generates the cepstrum parameters and the F0 and then combines them into a single file with F0 as parameter 0. This format is assumed for the later acoustic measures though the number of cepstrum/delta cepstrum parameters may be changed if desired.

  When pitch synchronous parameters were built, the clunits module automatically put the local F0 value in coefficient 0 at load time which was appropriate from LPC coefficients. The script in **festvox/src/general/make_lpc** was used to generate the parameters.

  An advantage of using LPC coefficients is that they are usually required for LPC resynthesis, thus allowing less information about the database to be required at run time.

- Building the clusters

  Building of clusters was mostly done automatically. This was done through a file which contains the basic code for building a cluster model for a database that has utterance structures and acoustic parameters. In this script, distance tables were built, features were dumped, and cluster trees built. The steps in carrying out all these procedures are explained below.

  The first stage was to load in all the utterances in the database, sort them into segment type and give them with individual names.

Acoustic parameters were then loaded and distance tables generated. The acoustic distance between each segment of the same type was calculated and saved in the distance table.

The next stage was to dump the features that were used to index the clusters. The clusters are defined with respect to the acoustic distance between each unit in the cluster, but they are indexed by these features. These features are those which will be available at text-to-speech time when no acoustic information is available. Thus, they include things like phonetic and prosodic context rather than spectral information. The named features may be over general allowing the decision tree building program wagon to decide which of these features actually does have an acoustic distinction in the units.

The CART tree builder, wagon, was then used to find the relationship between these features and the acoustic distances. It finds out questions about which features best minimize the acoustic distance between the units in that class.

The final stage in building a cluster model is to collect the generated trees into a single file and dumping the unit catalogue, i.e. the list of unit names and their files and position in them.

- The newly built voice was finally tested for correct synthesis.

5.1.5 Building an Open Domain Voice 2

Building the second open domain voice follows the same procedure as that for building the first one. The only differences are the setting of environment variables and the type of alignment tool used for labelling.

The first step of setting the environment variables in this case includes setting four variables: FESTVOXDIR, ESTDIR, SphinxTrain and Sphinx speech recognition system. The latter two systems were developed at the Carnegie Mellon University and must be installed before they can be used. There are other accessible training systems, HTK being the most famous. The reason for using SphinxTrain is because Festival developers are more familiar with this training system. SphinxTrain has been reliably used to label hundreds of databases in many different languages [9].

Sphinx2 is a real-time speech recognition system made available under a free software license and the source is available at [35]. The version used in carrying out this project is sphinx2-

0.4.tar.gz. SphinxTrain is a set of programs and scripts that allow the building of acoustic models for Sphinx2 (and Sphinx3). SphinxTrain can also be downloaded from [35].

For automatic labelling, an alternative method to DTW was used - SphinxTrain labelling using HMM models. Here full acoustic HMM models are trained on the recorded data.

After setting all the appropriate environment variables for open domain 2 (Odom2), the prompts were designed and recorded as in open domain 1 (Odom1). Utterance files were then built through the command:

```
festival -b festvox/build_clunits.scm '(build_prompts
"etc/text.data")'
```

This generated label files in **prompt-lab/** and waveform files in **prompt-wav/** which technically were not needed for this labelling process. Utterances were saved in **prompt-utt/**.

- Training stage

  After the prompt utterances were generated, the SphinxTrain directory **st/** was setup. All processing and output files were done within that directory. The file into which the labels were converted back into the voice's own phone set was put in **lab/** directory.

  The script **./bin/sphinxtrain** does the work of converting the Festvox database into a form suitable for SphinxTrain. In all there are 6 steps: setup, building files, converting waveforms, the training itself, alignment and conversion of label files back into Festvox format. The training stage consists of 11 parts and takes the longest time.

  The first step was to set up the sub-directory **st/** where the training would take place.

  ```
  ./bin/sphinxtrain setup
  ```

  The training database name for this voice is '***uct_ses_sphinx_clunits***'.

  The next stage was to convert the database prompt list into a transcription file suitable for SphinxTrain. All of the generated files were put in **st/etc/**.

  The next stage was to generate MFCC's for SphinxTrain which must be in a different format from the MFCC's used in Festvox. SphinxTrain only supports raw headered files, and NIST header files, so the waveform files in **wav/** were copied into the **st/wav/** directory converting them to NIST headers.

The training consists of eleven stages which were run automatically.

- Module 0 checks basic files for training.
- Module 1 builds vector quantisation parameters.
- Module 2 builds context-independent phone models. This runs Baum-Welch over the data building context-independent HMM phone models. This runs for several passes until convergence (somewhere between 4 and 15 passes).
- Module 3 makes the untied model definition.
- Module 4 builds context dependent models.
- Module 5a builds trees for asking questions for tied-states.
- Module 5b builds trees. One for each state in each HMM.
- Module 6 prunes trees.
- Module 7 retrains context dependent models with tied states.
- Module 8 deletes interpolation.
- Module 9 converts the generated models to Sphinx2 format.

All the above stages were run together with the command

```
./bin/sphinxtrain train
```

Once trained, these models were used to align the labels against the recorded prompts.

The final stage was to take the output from the alignment and convert the labels back into their Festvox format.

Phone labels in **prompt-lab**/ were then merged into the original utterances with the command:

```
festival -b festvox/build_clunits.scm '(build_utts
"etc/time.data")'
```

The steps from pitchmark extraction to testing were followed as in building Odom1 voice.

5.1.6 Building Advanced Sesotho TTS Systems

Two hybrid systems were built using a combination of three of the voices we had built, limited domain and the two open domain systems. The reason for using these voices and not a diphone one is because a diphone voice sounds robotic. The requirements for an advanced TTS system are naturalness, pleasantness, understandability and flexibility. We believe that we can get all these from a combination of limited domain and each of the open domain voices. Even though a

diphone synthesis system is flexible, we do not need its robotic-sounding voice. Rousseau [4] in his work also confirmed that a TTS system using a combination of limited domain and a diphone voices gave the worst performance, in terms of naturalness, compared to other voice combinations.

To build the hybrid systems, a python script was written (modification of that by Rousseau [4]) which helped in calling the two voices of the system of interest to read out text. The first system was a hybrid system of a limited domain voice and the first open domain voice (Odom1), which was named System 1a. The second was a hybrid of limited domain voice and the second open domain voice (Odom2), which was called System 1b. The reason for building the two hybrid system is for comparison purposes of the two automatic labeling tools.

The hybrid system uses the two voices in question which alternate when the system synthesizes text. The voice alternation depends on whether the word encountered is in the database or out of the database. For out-of-vocabulary (OOV) words or those words not in the database, the voice used is open domain. Limited domain voice is used only for words in the database.

5.2 Implementation of Intonation - Design of Sesotho Advanced TTS System 2

Even though the first Sesotho TTS hybrid systems above meet all the requirements of an advanced TTS system, more work still needs to be done on them to make them sound more natural and fluent. The systems above can read any Sesotho text using the two voices, but the voices have different tones which are audible as glitches during synthesis. The glitches reduce the quality of the output speech on the aspect of both naturalness and fluency. The systems thus, are not really pleasant to listen to, especially for long periods. In this case, the limited domain voice sounds more natural than the open domain voices. We decided as such, to smooth or mask the glitches heard between the switching of the voices. As this is an issue of intonation, the first task was to compare the pitch or F0 contour of the voices separately. We then worked on getting intonation of the open domain voice to the level of the limited domain and form a seamless and fluent concatenation system from the voices.

5.2.1 Building Prosodic Models

The procedure used to build the prosody models was done following the steps in [9]. This process is almost the same as in building unit selection voices in Section 5.1.1 but the main objective here is to build duration and F0 models. The prosody models were built for the two open domain systems, Odom1 and Odom2. Therefore, the process of building the prosody

models closely followed that of each respective system. A set of prompts was designed which was to cover the prosody that we wished to model. The data were then recorded and labeled and the models were built from the utterances built from the natural speech.

The basic procedure is as follows:

- Setting the environment variables
  The environment variables, ESTDIR and FESTVOXDIR, were set up.

- Creating a directory to hold the model
  The voice directories were ***uct_ses_protest_clunits*** and ***uct_ses_protest2_clunits*** for Odom1 and Odom2 respectively.

- Setting up directories and scripts for building prosody models
  The setup procedure for the directory structure was the same as that for the open domain structure. Extra directories and scripts needed to build prosody models were then set up with the command:

```
$FESTVOXDIR/src/prosody/setup_prosody
```

- Designing the database
  The database used here was the same database as the one we had in the open domain synthesis systems – Odom1 and Odom2. These are the synthesis systems whose prosodic features we wish to change. The database files from the open domain synthesis were then copied into '***etc/***' directory.

- Synthesizing prompts for recording
  Synthesis of the prompts is done for two reasons. The first reason is for use in auto-labeling, where the synthesized prompts are aligned using DTW or Sphinx against what the speaker actually says. The second reason is construction of Festival utterance structures for each utterance in the database with natural durations and F0.

  The line setting the "closest" voice was changed to that used in building the diphone voice, this being '***voice_uct_ses_hlo_diphone***'. This is the voice that was used to synthesize the prompts. It was also ensured that the lexicon, duration data and phoneset lists match those of the open domain systems. It was only after these changes that the prompts were synthesized.

- Recording the prompts

  The recording circumstances were the same as for the other voices. The only difference here is that the script for prompting the utterances was modified so that it did not play the prompts. According to [9], playing the prompts confuses the speaker.

- Phonetically labeling the prompts

  For the model that is being built for Odom1, dynamic time warping was used as a tool for labeling the prompts. For the model for Odom2, SphinxTrain labeling was implemented.

- Extracting pitchmarks and F0 contour

  The pitchmark extraction procedure is the same as detailed in Section 5.1.1. The additional step to do was to copy the new pitchmark parameters in the file **bin/make_pm_wave** to the file **bin/make_f0_pm**.

  The second part was the construction of an F0 contour which was built from the extracted pitch marks. Unvoiced speech sections were assigned an F0 contour by interpolation from the voiced section around it, and the result was smoothed. The label files were used to define which parts of the signal were silence and which were speech.

  The variable SILENCE in **bin/make_f0_pm** was modified to 'pau', which is the symbol used for silence in our phoneset. The smoothed F0 was then extracted by the command:

```
bin/make_f0_pm wav/*.wav
```

- Building utterance structures

  With the labels and F0 created, we rebuilt the utterances structures by synthesizing the prompts and merging in the values from the natural durations and F0 from the naturally spoken utterances. The command run was:

```
festival -b festvox/build_ldom.scm '(build_utts
"etc/time.data")'
```

5.2.2 Duration Modelling

A duration model was built for both systems using a CART tree to predict zscore values for phones. Zscores (number of standard deviations from the mean) have often been used in duration modeling as they allow a certain amount of normalization over different phones [9].

- Setting environment variables

  The environment variables to be set are the ESTDIR and FESTVOXDIR if these were not set before.

- Making necessary changes to the variables in the **bin/make_dur_model** file.

  The variables that were changed are:

  SILENCENAME = 'pau' This is the silence name used in the phoneset of the unit selection systems, Odom1 and Odom2.

  VOICENAME - This was changed to the name of the voice in which the model would be used, in our case being the open domain systems. We built a duration model for each voice. For Odom1, the voice name was changed to '(***voice_uct_ses_test2_clunits***)' and for Odom2 it was changed to '(***voice_uct_ses_sphinx2_clunits***)'.

  MODELNAME - The model name was given the first three arguments of each respective voice systems, ***uct_ses_test2*** and ***uct_ses_sphinx2***.

- Extracting means and standard deviations of phone durations

  This is where mean durations and standard deviation of each phone are calculated. A Festival script used for this is ***durmeanstd***.

- Extracting features for prediction

  The next stage was extracting features from which durations could be predicted. The extraction process takes each phoneme and dumps the named feature values for that phone into a file. This uses the standard festival script ***dumpfeats*** to do this.

- Removing silence phones

  In this step all the features were saved in one file and the silence phones removed. These feature files were concatenated into a single file which was then split (90/10) into training and test sets. The training set was further split for use in the training phase. All silence phones were also removed from the training and test sets. The reason is that silences at the start and end of the utterances are not part of the speech content, and can thus skew the results in the training set.

- Building a feature description file

  A feature description file was built. A huge list of numbers in the **dur.desc** file was replaced with floats as appropriate.

- Building a regression model to predict durations

  The key factor in building the model is the "stop" value. This is the number of examples that must exist before a split is searched for. We used the default value which is 50.

- Testing the results on data not used in the training

  The testing was done by using the ***wagon_test*** command.

- Constructing a Scheme file and incorporating into the respective system where the model will be used in.

  The final stage was to package the model into a scheme file that can be used with a voice. This scheme file contains means and standard deviations (so that predicted values can be converted back into seconds) and the prediction tree itself. The duration models were packaged for ***uct_ses_test2_clunits*** and ***uct_ses_sphinx2_clunits***.

5.2.3 F0 Modelling

In carrying out this modeling technique, no accents were dealt with.

- Setting environment variables

  The environment variables to be set are the ESTDIR and FESTVOXDIR if these were not set before.

- Making necessary changes to the variables SILENCENAME, VOICENAME and MODELNAME in the **bin/make_f0_model** file.

  The variables that were changed are the same ones as done in duration modeling.

- Extracting features for prediction

  The next stage was extracting features from which F0 could be predicted. The extraction process takes each phoneme and dumps the named feature values for that phone into a file. This uses the standard festival script ***dumpfeats*** to do this.

- Removing silence phones

  The same procedure was followed as in duration modeling.

- Building feature description files
  A feature description file, **f0.desc**, was built. A huge list of numbers in the file was replaced with floats as appropriate.

- Building a regression model to predict F0 at start, mid and end of syllable
  This part builds a CART tree that predicts F0 at the beginning, mid-vowel and end of syllables. This step was split into three sections for building and testing the F0 model for a syllable at the start, the mid and the end of the vowel.

- Constructing a Scheme file and incorporating into the respective system where the model will be used in.
  The models were packaged into scheme files that would be used with their respective voices.

5.2.4 Building Advanced Sesotho TTS Systems - Part 2

Two hybrid systems were built as was done in Section 5.1.6, the only difference being that the unit selection systems had intonation modelling. The new hybrid systems were called System 2a (from Ldom and ***uct_ses_test2_clunits*** (Test2)) and System 2b (from Ldom and ***uct_ses_sphinx2_clunits*** (Sphinx2)).

5.3 Summary

This chapter has given detailed information on the procedure followed in building the synthetic voice systems and the hybrid systems. It has also shown how intonation modelling techniques, duration and F0, were applied on the unit selection voices as a form of improving the naturalness and fluency of the speech output.

Chapter 6

6. Analysis and Evaluation of Results

This chapter gives the results obtained from running tests on the voice systems built in the previous chapter. Analysis and evaluation of the results are also discussed.

6.1 Testing Procedure

Different text-to-speech systems were built and evaluated. The systems evaluated were the different voices built under concatenative synthesis, these being:

- A diphone voice.
- A limited domain voice (Ldom).
- Two open domain voice systems (Odom1 and Odom2) which use two different automatic labeling methods.
- Two hybrid systems of limited domain and open domain systems:
    - Ldom + Odom1 = System 1a
    - Ldom + Odom2 = System 1b
- Open domain systems on which intonation modeling had been applied: Protest1 and Protest2. These two systems are the same as the ones above, the only difference being that duration and F0 modeling has been implemented in this case.
- The hybrid system of limited domain and open domain systems with prosody modeling.
    - Ldom + Test2 = System 2a
    - Ldom + Sphinx2 = System 2b

In evaluating the systems, two kinds of testing procedure were followed. The first form of test carried out was the comparison of the pitch (F0) contours of the systems built. The reason for this was to check if the contour shapes of the systems were in agreement with the speech output of their respective systems.

The second procedure was to perform subjective listening tests in order to evaluate the quality of the voice produced by the systems built. This test required listeners to rank the voice quality using a Mean Opinion Score (MOS) sheet. MOS tests are commonly used for both evaluating the effectiveness of speech coding algorithms and assessing the quality of synthesized speech [17]. The listening tests were done in two groups: for the synthetic voices (6) and for the hybrid

systems (4). The features that were tested on the synthetic voice systems were: intelligibility, naturalness and pleasantness. For the hybrid systems, the features tested were understandability, naturalness and fluency. These features are explained below. These features were taken from [36] and [4]. The MOS score sheet used was a modified form of that used by Rousseau [4]. Permission was obtained from Rousseau to use his sheet for evaluation of our systems.

Understandability/Intelligibility – This feature tests if the subject can understand the message without any effort at all. For instance, does one need to listen to the sample more than once, just for understanding?

Pleasantness – This is based on whether the system being tested can be listened to for long periods of time without getting tired. Is the system pleasant to listen to?

Naturalness – The speech output could sound more like a human voice or sound robotic. Does the system sound close to a human voice?

Fluency – Each hybrid system is a combination of two voice systems. Thus, the speech output is two voices reading text in alternation. The concatenation points to look for are the ones where the voices alternate. Can the subject differentiate between the two voices? Are the concatenation points very audible? Does the speech output flow?

The evaluation set for voice systems contained 25 sentences, 10 of which were long sentences (translated to Sesotho) taken from the South African Government Services website [37], 10 were short sentences made up by the author, and the last 5 were taken from the database. The reason for choosing utterances taken from the Government website is that this is one area where our Sesotho TTS system can be applicable. Individuals who speak Sesotho in South Africa (or elsewhere) can access the information on this website regarding any of the services offered by the government. The last 5 sentences were not in context, i.e. the words of each utterance were jumbled up to test if the listeners' perception was controlled by sentence context.

For the hybrid systems, 7 sentences were tested; 4 long sentences and 3 short sentences. These sentences were different from the ones used for the listening tests for synthetic voice systems.

There were 20 subjects and each one listened to the sentences using headphones. The subjects were all fluent in Sesotho, though not necessarily their mother-tongue. Of all 20, one person was from Botswana, 6 were from South Africa, and the rest were from Lesotho. The sentences were

sampled at 16 kHz. The task for the listeners was to rate each sentence on a scale of 1-5, where 1 is very poor and 5 is excellent. The subjects were first familiarized with the TTS system by listening to some sample utterances of varying quality.

Of all the subjects doing the tests, only one listener was familiar with TTS systems. This one person was used as a control to see if his listening would be affected by his knowledge of the TTS systems.

The samples of the MOS questionnaires for individual and hybrid systems can be found in Appendix A and Appendix B respectively.

6.2 Voices

The synthetic voices which were tested are diphone (Diphone), limited domain (Ldom), and the unit selection voices (Odom1 and Odom2). The results are given in this section.

6.2.1 Pitch Contour

The results shown here are for the voice systems with no prosody (Diphone, Ldom, Odom1 and Odom2). The sample text tested for all voices is "le be le khotso.", which is one of the utterances in the database (time0015).

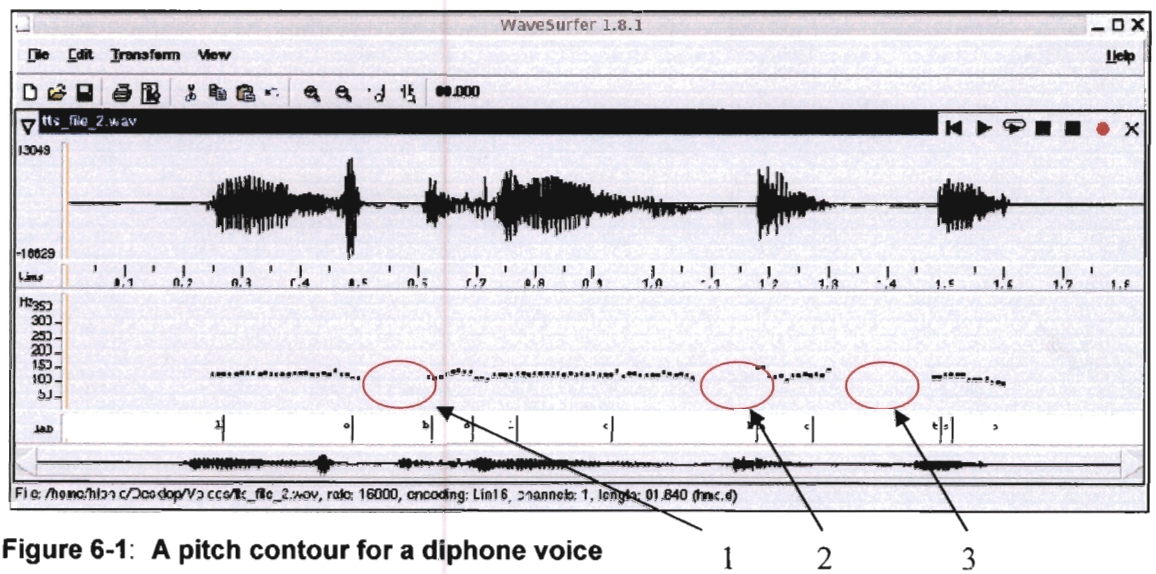


Figure 6-1: A pitch contour for a diphone voice

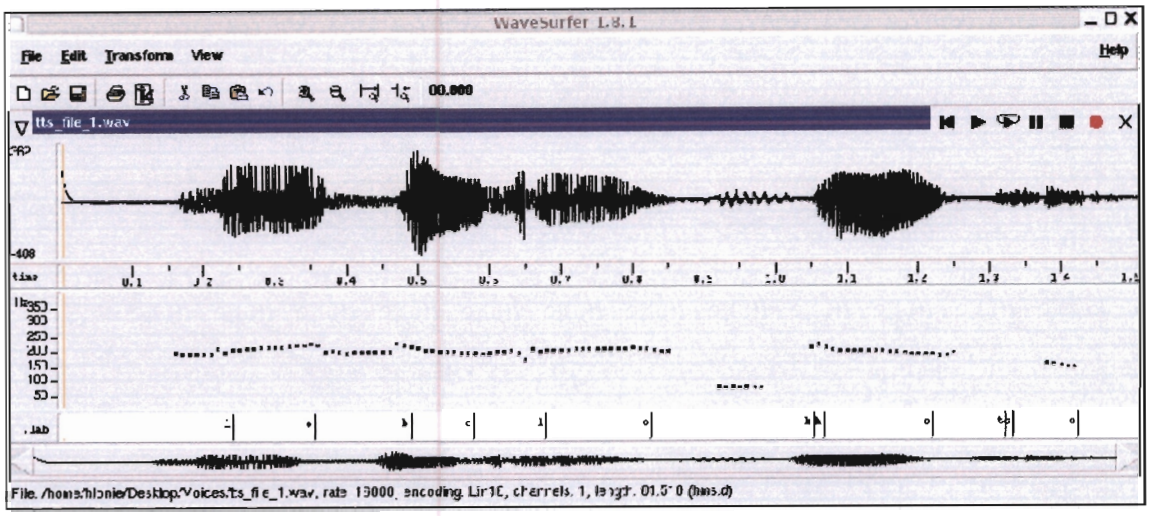


Figure 6-2: A pitch contour for a limited domain voice (Ldom)

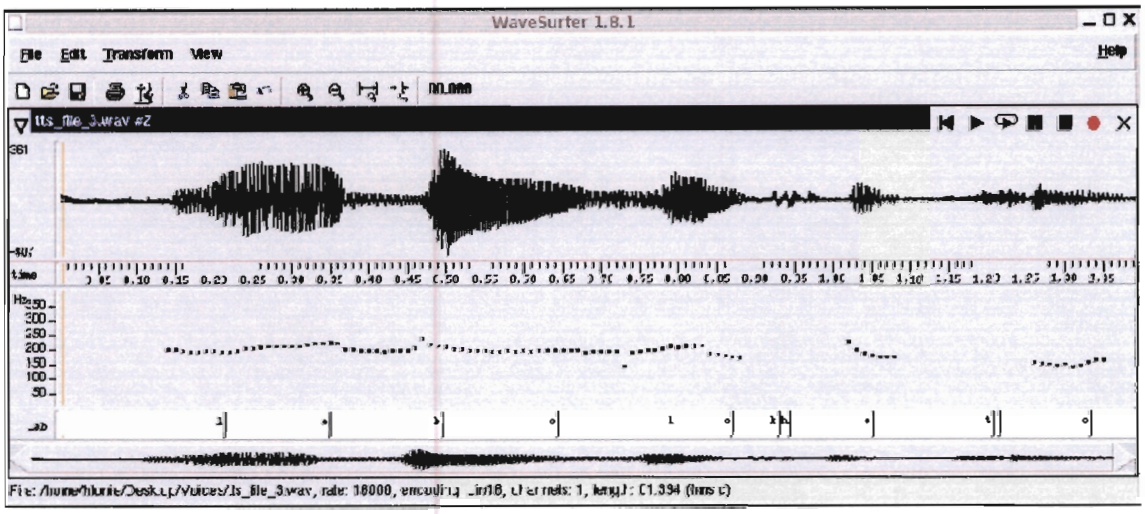


Figure 6-3: A pitch contour for Odom 1

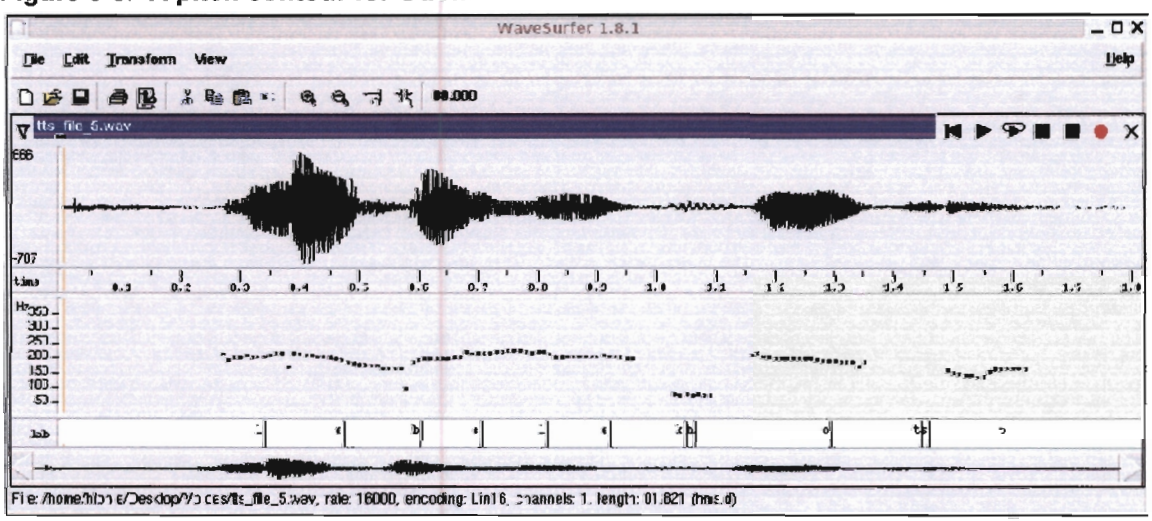


Figure 6-4: A pitch contour for Odom 2

6.2.1.1 Pitch Contour Analysis

For F0 contour comparison of the four voices, we will use the diphone as our reference. Even though the sample sentence, "le be le khotso" has four words with three slight pauses in between them, the real 'pauses' by the waveforms do not correspond to this. There is no visible (waveform-wise) pause between the words "be le". This means this is treated like one word. The waveform 'pauses' are at encircled sections (in red) as shown in Figure 6-1 of the diphone voice. We will use these 'pauses' in our analysis and name them Unvoiced1, 2 and 3 respectively.

The diphone voice has no F0 at the unvoiced sections and has very minimal noise at these sections. On the other hand, the limited domain voice (Ldom) has a more continuous F0 contour, even at some unvoiced sections; the pitch contour is very low at Unvoiced 2 (caused by the noise in the microphone when breathing) and there is none on Unvoiced 3. Odom1 has a contour shape very close to that of the limited domain voice except that at Unvoiced 2 and 3, there is no pitch. Odom2 has a variable F0 which goes down at Unvoiced 1, lower on Unvoiced 2 and none on Unvoiced 3. Overall, the three unit selection voices (Ldom, Odom1 and Odom2) have their contours at a higher frequency than that of a diphone voice.

6.2.2 Listening Tests

MOS score results obtained from the listening tests are tabulated below. Their evaluation and analysis are given in Section 6.4.

Table 6-1: Long Sentences for Individual Voices

System	Understandability	Naturalness	Pleasantness	Overall
Diphone	2.5	2.5	2.9	2.7
Ldom	4.7	4.9	4.6	4.7
Odom1	4.1	3.9	4.1	3.8
Odom2	3.8	4.4	3.7	3.6

Table 6-2: Short Sentences for Individual Voices

System	Understandability	Naturalness	Pleasantness	Overall
Diphone	2.6	2.9	2.6	2.4
Ldom	4.5	4.8	4.6	4.6
Odom1	4.3	4.9	4.1	4.0
Odom2	4.1	4.4	3.8	3.7

Table 6-3: Out-of-context Sentences for Individual Voices

System	Understandability	Naturalness	Pleasantness	Overall
Diphone	2.2	2.5	2.4	2.3
Ldom	4.3	4.4	4.5	4.3
Odom1	3.4	3.7	3.4	3.4
Odom2	3.6	4.4	3.4	3.5

6.3 Intonation Tests

The results depicted in this section are those obtained after prosodic modeling of the open domain systems, Odom1 and Odom2. The new voice systems from these are Protest1 and Protest2 respectively.

6.3.1 Pitch Contour

These are the results for the two voices on which intonation was implemented. The sample test sentence was "le be le khotso". Prosody voice for Odom1 is Protest1 and that for Odom2 is Protest2. These were modeled into Test2 and Sphinx2 voices.

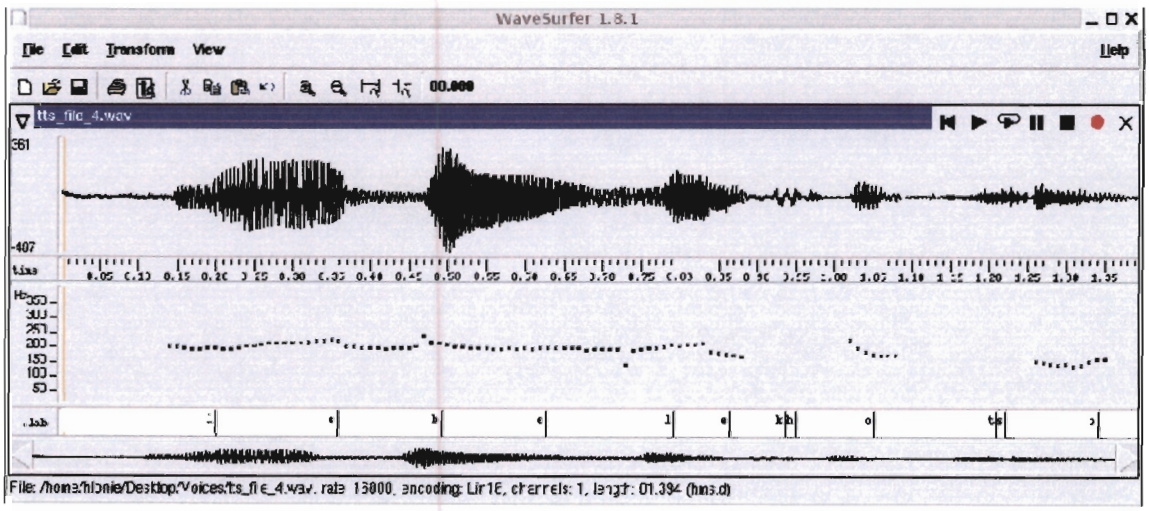


Figure 6-5: A pitch contour for Protest1

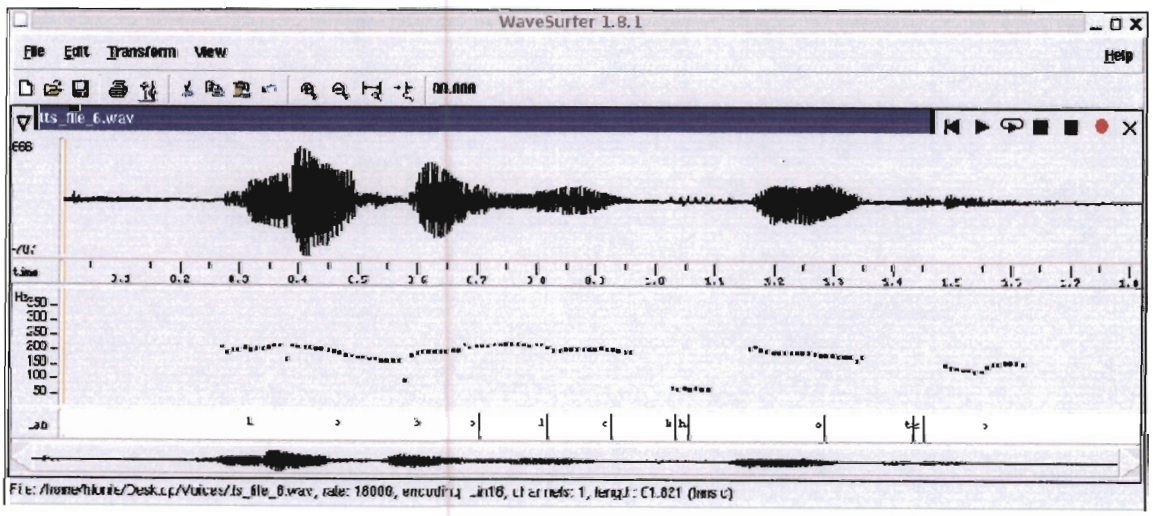


Figure 6-6: A pitch contour for Protest2

6.3.1.1 Contour Analysis

Protest1 voice has wider F0 gaps at Unvoiced 2 and 3. Protest2 has a more continuous F0 contour than Protest1. In comparison with the voices in Section 6.2.1, Protest2's contour shape is very similar to that of Ldom. With the contour shape of Protest2 being almost the same as that of Ldom, we expect a hybrid of Ldom and the model onto which Protest2 was used (Sphinx2) to have minimal glitches. In brief, we expect the Ldom/Sphinx2 hybrid system to be more fluent than the Ldom/Test2 hybrid system.

6.3.2 Duration and F0 Modeling Results

These are the results obtained from duration and F0 modeling of the two open domain systems. It must be noted that the values obtained from duration modeling are not in the absolute domain (i.e. in seconds); they are in the zscore domain.

According to [9, 40], measuring the accuracy of a generated F0 contour and duration modeling is not easy. However, to have some sense of the measure of accuracy, we used what [9, 40] used: the root mean squared error (RMSE) between the generated contour and the original (smoothed) contour and the correlation factor.

Two experiments were carried out, modeling of duration and modeling of F0. F0 modeling consisted of models for *syl_start*, *syl_mid*, and *syl_end* of the vowels. Tables 6-4 and 6-5 below show the results obtained for Protest1 and Protest2.

Table 6-4: Intonation modeling results for Protest1

	Duration Modeling	**F0 Modeling**		
	Duration	***Syl_start***	***Syl_mid***	***Syl_end***
RMSE (Hz)	0.6904	23.4821	21.2811	23.8040
Correlation	0.7211	0.5393	0.5867	0.8927

Table 6-5: Intonation modeling results for Protest2

	Duration Modeling	**F0 Modeling**		
	Duration	***Syl_start***	***Syl_mid***	***Syl_end***
RMSE (Hz)	0.7235	25.6281	21.5318	24.0215
Correlation	0.6907	0.4808	0.5794	0.8975

6.3.2.1 Modeling Evaluation

A low RMSE value and a high correlation value mean a better performance by the model or system in question. It is interesting to see that, from Table 6-4 and Table 6-5 above, Protest2 has

a higher RMSE value (in both duration and F0 models) than Protest1 in general. On the contrary, for correlation, Protest2 has generally smaller values than those of Protest1.

6.3.3 Listening Tests

In this section, subjective listening tests were used to assess the intonation component of speech synthesis systems. In these tests, listeners were played the same set of sentences with and without the original F0 contour. If listeners could not distinguish between the two, then the synthesized F0 contours were deemed to be perceptually equivalent to the originals.

The results below are obtained from the listening tests of the voices Protest1 and Protest2. Analysis of the results is detailed in Section 6.4.1 below.

Table 6-6: Long Sentences for Prosodic Voices

System	Understandability	Naturalness	Pleasantness	Overall
Protest1	4.3	4.5	4.0	3.8
Protest2	4.1	4.4	4.0	3.8

Table 6-7: Short Sentences for Prosodic Voices

System	Understandability	Naturalness	Pleasantness	Overall
Protest1	4.2	4.5	3.7	3.8
Protest2	4.5	4.8	3.9	3.7

Table 6-8: Out-of-context Sentences for Prosodic Voices

System	Understandability	Naturalness	Pleasantness	Overall
Protest1	4.2	4.6	3.8	3.9
Protest2	4.5	4.9	4.1	4.1

6.4 MOS Score Analysis and Evaluation of Overall Results

This section presents the graphical results obtained from the listening tests of all the synthetic voices built.

6.4.1 Graphical Evaluation

Below is the graphical representation (Figures 6-7, 6-8 and 6-9) of the MOS score results for all the six synthetic voices built. The figures are divided according to the tests based on the type of the sentences used as samples; long, short, or out-of-context.

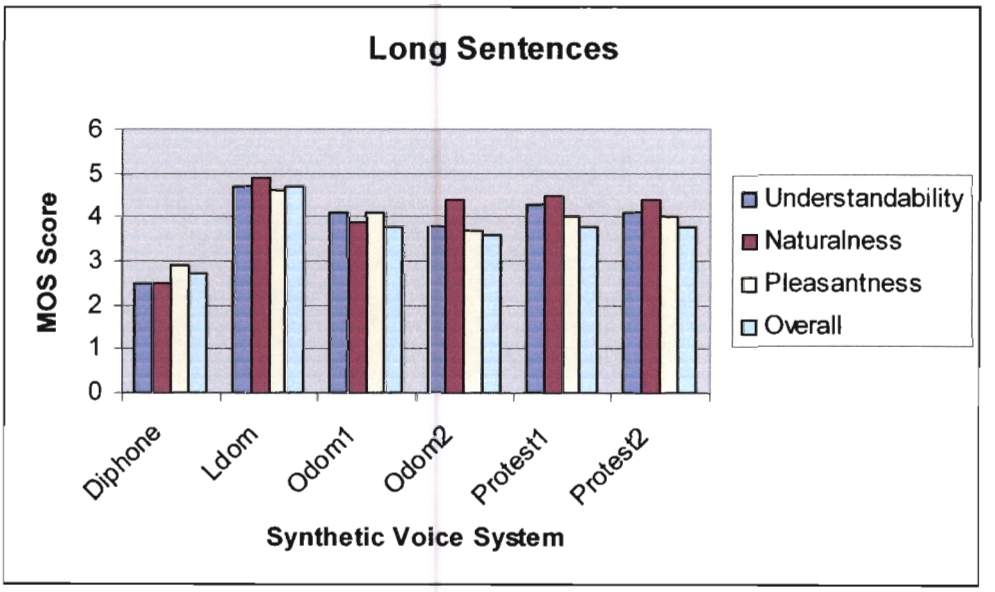


Figure 6-7: **MOS score results from long sentences**

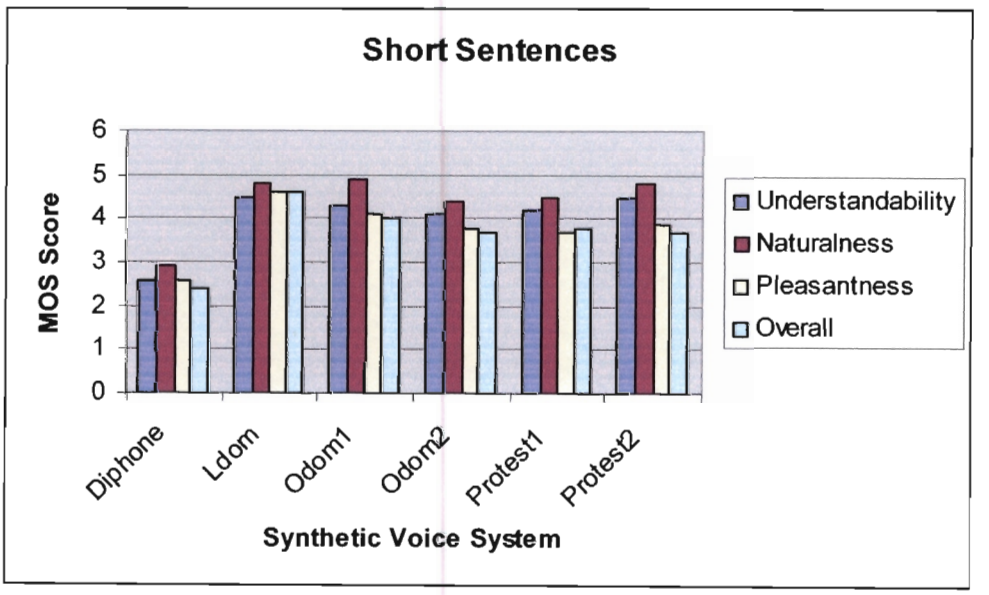


Figure 6-8: **MOS score results from short sentences**

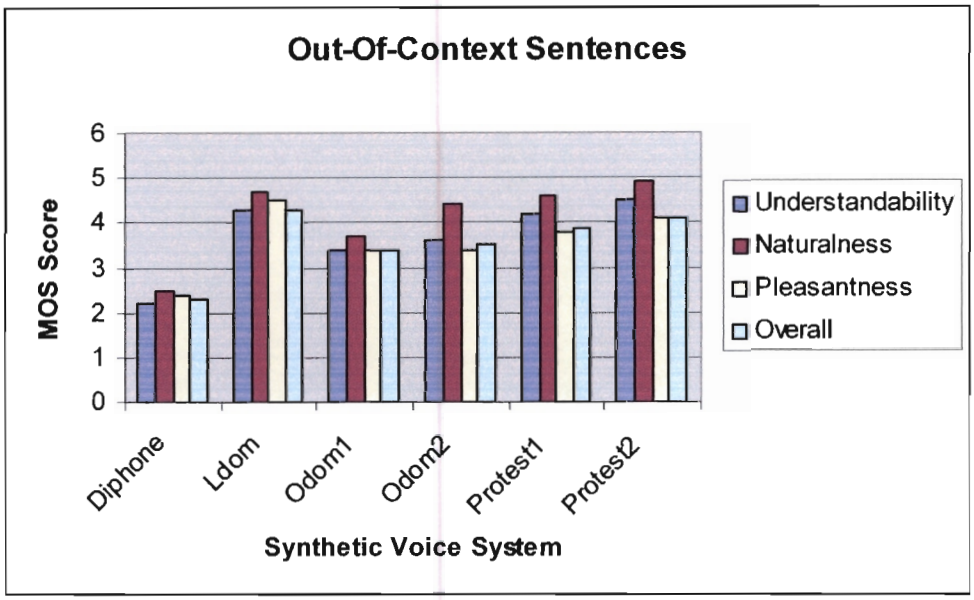


Figure 6-9: **MOS score results from sentences out of context**

6.4.1.1 Analysis and Evaluation of Long Sentences

The criterion with the highest score is naturalness, with the limited domain voice giving the best score of 4.9. The next best condition is understandability with an average of 3.92 for all voice systems. This is closely followed by pleasantness at an average of 3.88. Overall, the average performance of the systems for long sentences is 3.73.

Understandability: The prosodic voices, Protest1 and Protest2, together with Odom1, have the next best understandability score greater than 4, after Ldom. This means that their speech output quality is reasonably clear. It is quite interesting to see that listeners found Odom1 to be more understandable than Odom2.

Naturalness: Odom2, Protest1 and Protest2 systems have naturalness of 4.4, 4.5, 4.4 respectively, making them the best systems which sound close to a human voice, after the limited domain system.

Pleasantness: All the unit selection voices have a pleasantness score well over the acceptable level (3.0), with the diphone system dragging at just below this level (2.9).

Overall, the diphone voice has the worst performance in all categories, with none being acceptable, i.e. all rate below score of 3. Limited domain, on the other hand, has the best performance with all its criteria values close to 5.

6.4.1.2 Analysis and Evaluation of Short Sentences

Understandability: The unit selection voices give out intelligible speech, with the prosodic voices performing slightly better than Odom1 and Odom2. The diphone voice has the worst performance.

Naturalness: For short sentences, Odom1 beats Ldom and Protest2 by a very slight margin (Odom at 4.9 and Protest2 and Ldom both at 4.8) on naturalness. Odom2 and Protest1 are the next best with a diphone voice being the worst system at 2.9.

Pleasantness: Odom1, again, surpasses Odom2, Protest1 and Protest2 on pleasantness. Ldom has the best score and the diphone system performs horribly.

Overall, the speech output of the systems is slightly worse than that of long sentences by a difference of 0.03.

6.4.1.3 Analysis and Evaluation of Out-of-context Sentences

Understandability: Protest2 excels in understandability with the highest score of 4.5. Ldom and Protest1 pursue with scores of 4.3 and 4.2 respectively. Odom1 and Odom2 are acceptably understandable. The diphone system does not give an intelligible speech output.

Naturalness: The prosodic voice, Protest2, has the best naturalness output, (4.9), followed by Ldom. Odom1, Odom2 and Protest1 also show good performance with the diphone voice sounding very unnatural.

Pleasantness: Ldom is the most pleasant voice to listen to, with the other unit selection voices at an acceptable region greater than 3. The diphone system gives a boring speech output.

Overall, Protest2 is the best performer after Ldom, with Protest1 tagging closely behind. Odom1 and Odom2 are not very far off with the diphone system giving a poor performance.

6.4.1.4 Overall Analysis

Naturalness: All the systems give a speech output which sounds close to a human voice, except for a diphone system which sounds robotic.

Understandability: Understandability rates very close to naturalness, though slightly lower. This is an indication that even though the systems sound natural, their intelligibility is not necessarily the best. Some bit of effort needs to be put in order to fully understand what is being synthesized.

Pleasantness: The unit selection systems are quite pleasant to listen to, but not for long periods. This could be due to the fact that the articulation of the words, though understandable, gets boring with time.

Overall: Long and short sentences have an acceptable speech output quality (3.7) though a value higher than this would have been preferred. The performance of out-of-context sentences is inferior in comparison at a score value of 3.6. The reason here could be that because the words are jumbled up, the utterances do not make sense and the listeners find this confusing and as such, do not appreciate such performance.

6.5 Comparison of the Hybrid Systems

6.5.1 Pitch Contour

Figures 6-10 to 6-13 show the F0 contours of the four hybrid systems. The utterance being tested is "morena makhetha ke mofu". The words "morena" and "ke" are in the database, which means they are being synthesized by the limited domain voice. The words "makhetha" and "mofu" are OOV words which are synthesized by the open domain voice.

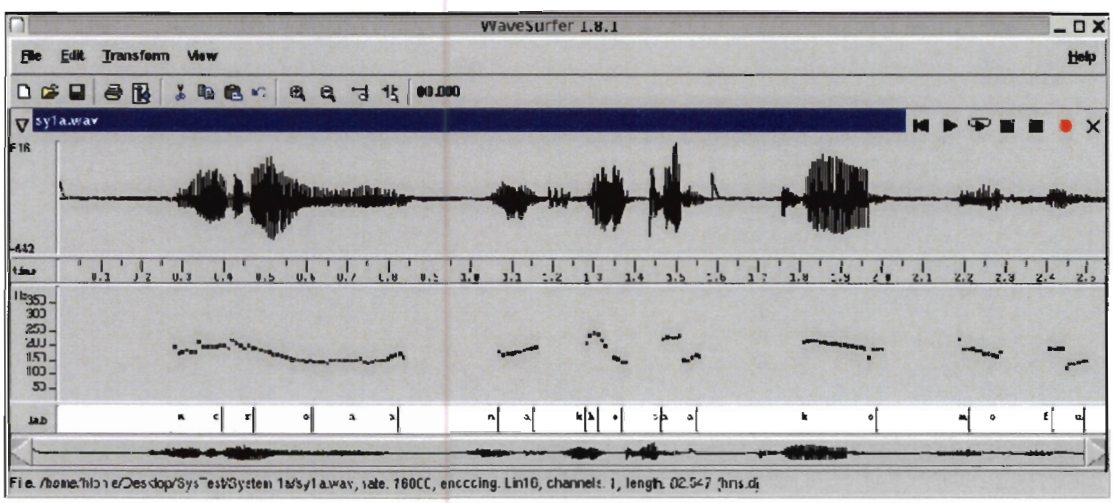


Figure 6-10: A pitch contour for System 1a

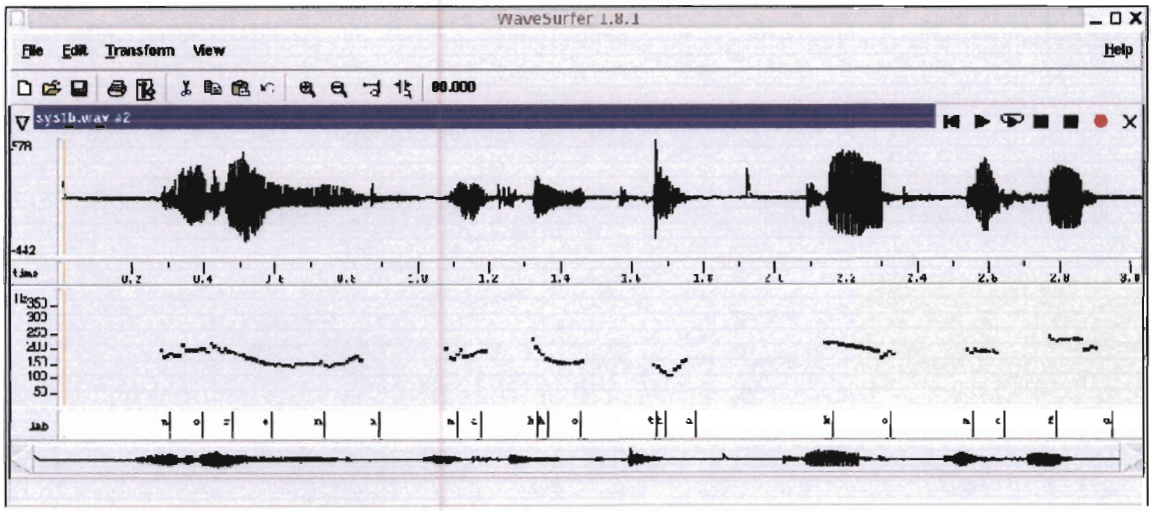


Figure 6-11: A pitch contour for System 1b

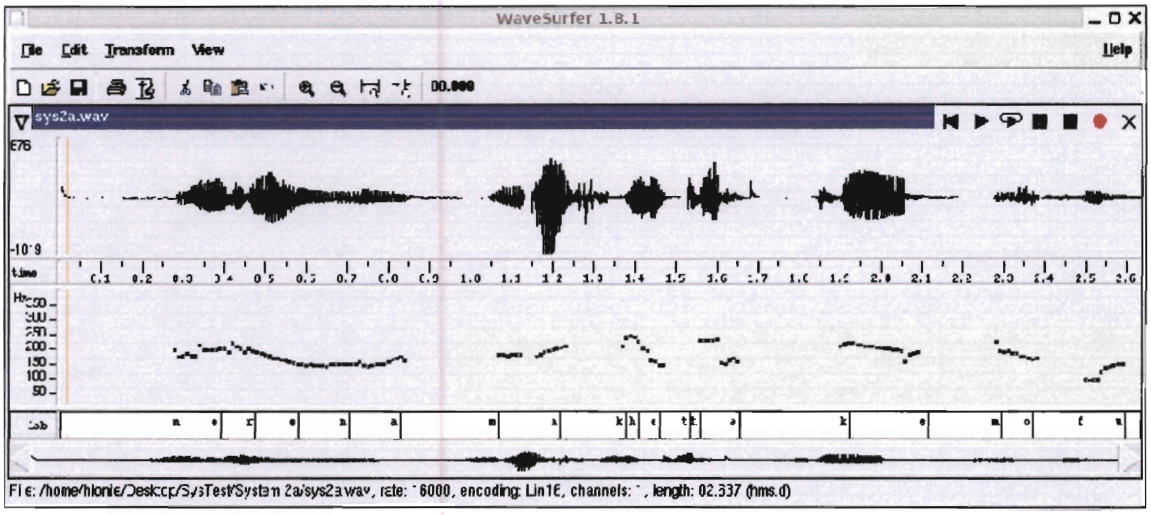


Figure 6-12: A pitch contour for System 2a

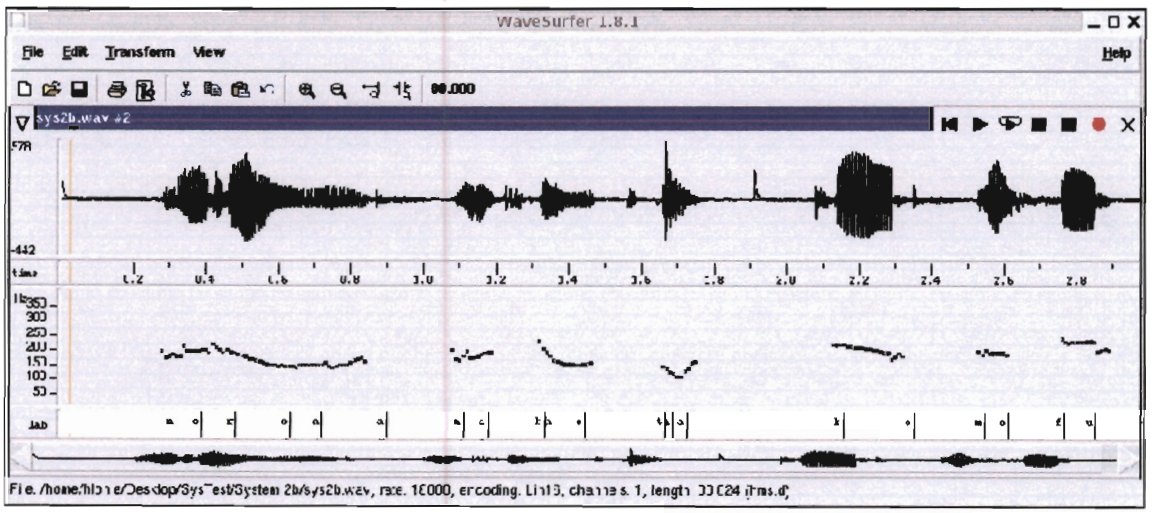


Figure 6-13: A pitch contour for System 2b

6.5.1.1 Analysis of the Pitch Contour

The contour shapes of System 1a and System 2a look very similar, with System 2a getting smoother towards the end. With Systems 1b and 2b, it is difficult to see the difference in the contour shapes.

6.5.2 Listening Tests

The results shown below are taken from the MOS score evaluation of the four hybrid systems by the listeners.

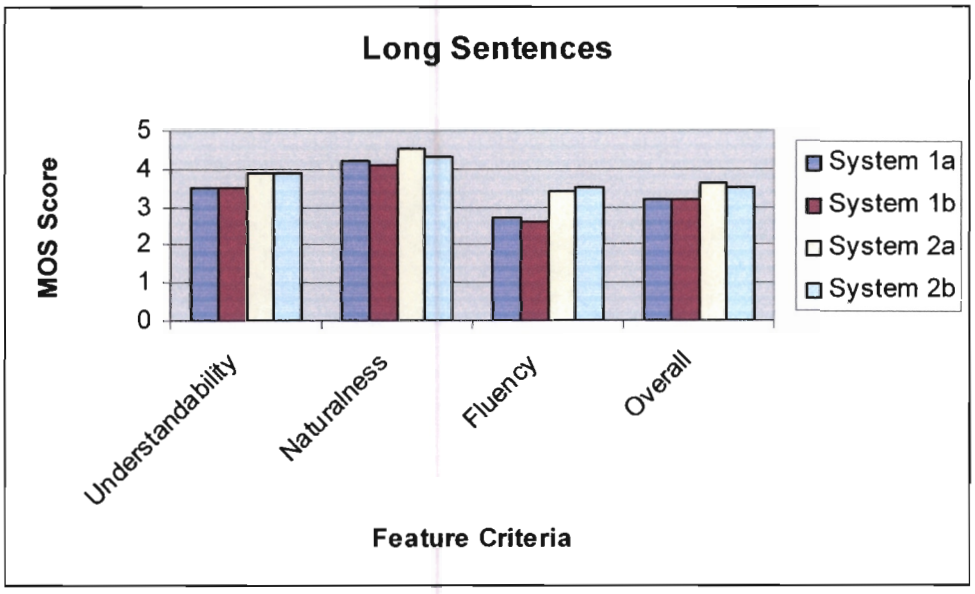


Figure 6-14: Graphical results for long sentences on the hybrid systems

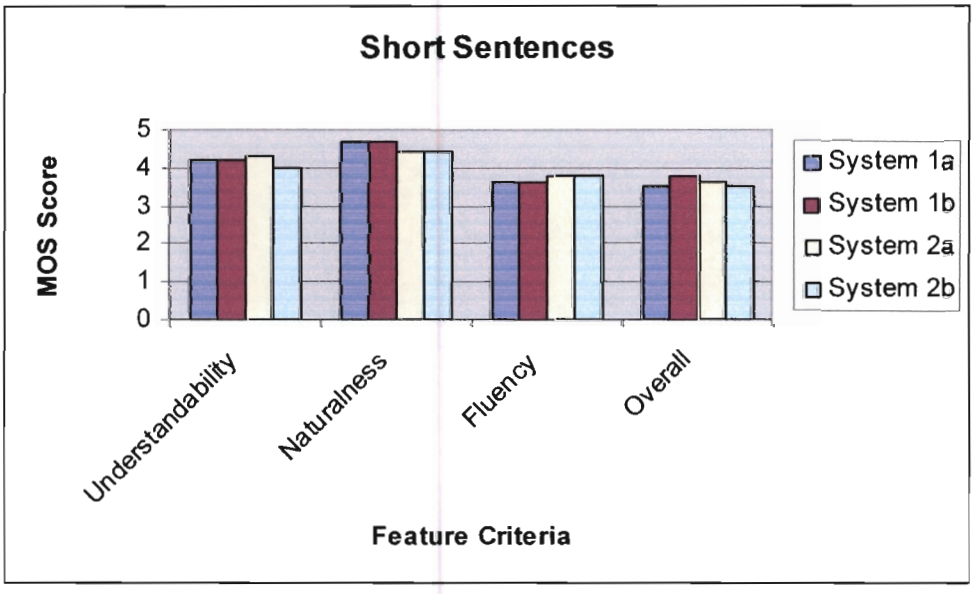


Figure 6-15: Graphical results for short sentences on the hybrid systems

6.5.2.1 Analysis and Evaluation of Long Sentences

Understandability: Systems 2a and 2b require the least effort applied (score = 3.9 for both) in order to understand the text being synthesized. Systems 1a and 1b also perform well, both having equal scores of 3.5.

Naturalness: System 2a sounds more natural than all the systems, with System 2b in close pursuit. Systems 1a and 1b have a reasonable performance as well. Generally, this criterion rates highest for all systems.

Fluency: Systems 2a and 2b are more fluent than their counterparts, Systems 1a and 1b, which have performance that is not acceptable (lower than 3).

Overall, the systems perform quite well (just above 3), with prosodic systems being the better systems.

6.5.2.2 Analysis and Evaluation of Short Sentences

Understandability: In this case, System 2b performs the worst, though all systems have a score greater than 3.

Naturalness: Systems 1a and 1b are far better than the prosodic systems, 2a and 2b. Naturalness also rates the highest in comparison to other features.

Fluency: Fluency here is much better than that for long sentences. This means the concatenation of the voices in each system is less audible than for long sentences. Systems 2a and 2b are more fluent than Systems 1a and 1b with a score of 3.8 compared to 3.45 of the latter systems.

Overall: System 1b has the best performance, followed by System 2a. System 2b and System 1a perform equally at 3.5.

All in all, the systems give a much better output for short sentences than for long ones. Compared to the individual voice systems, the hybrid systems' performance for short sentences matches closely with that of out-of-context sentences.

6.6 Post Analysis Discussion

The aim of this thesis was to improve the naturalness and fluency of a hybrid text-to-speech system for Sesotho language through intonation. This was achieved by applying duration and F0 modeling techniques on the system. The results of the F0 contour shapes of the system on which intonation modeling was applied show that this is possible. The contours shown in Figures 6-10 to 6-13 show the similarity between generated contours and the originals. Subjective listening tests confirm that the naturalness of the systems on which intonation was applied has improved.

There are some aspects which were not covered in the thesis and these are:

- There was no assignment of accent and boundary event labels during the synthesis process, even though according to [9], durations depend on accent.
- Due to an old writing system for Sesotho slowly phasing out, we have only considered the current writing system for our phoneset. For instance, a word such as ***ts'oene*** (monkey) is written as ***tšoene*** in the old writing system.

By analyzing the MOS tests conducted, we have also identified the main problems and limitations of the developed system. The major sources of errors degrading synthesized speech quality are:

- Misalignment of phones in the speech database, though we tried our best.
- Timing errors for phones.

6.7 Summary

This chapter has detailed results obtained from various tests done on the individual synthetic voice systems, and the four hybrid systems. The tests carried out were determination of the pitch contour for all systems, duration and F0 modeling for the two open domain systems, and subjective listening tests for all systems. Evaluation of the listening tests was based on the MOS score system which rated the quality of the speech output of the systems. The criteria rated were: understandability, naturalness, pleasantness, and fluency (for the hybrid systems). Analysis of the systems was made based on these results.

Chapter 7

7. Conclusions and Future Work

In this report, design of a fluent advanced text-to-speech system for Sesotho through intonation has been proposed and implemented. Four different voices based on concatenative synthesis have been built, these being a diphone voice, a limited domain voice, and two open vocabulary voices. Two hybrid systems were built using the unit selection synthesis voices, the limited domain and the open domain systems. The hybrid systems had the qualities of an advanced TTS system in that they produced speech that is understandable, natural, pleasant and flexible. In order to improve the naturalness and fluency of the advanced systems, intonation was added to the systems. This was done by applying duration and F0 modeling on the open domain systems, and forming hybrid systems from these new intonation-modeled systems. Pitch contours of the original and generated models were compared for improvement. Subjective listening tests were carried out to assess the speech output quality of the old systems (with no intonation) and the new systems. Overall, the new hybrid systems, System 2a and System 2b, got an average MOS score of 3.7.

7.1 Conclusions

Conclusions drawn from the experiments are as follows:

- As expected, the systems on which intonation was implemented produced better results which were acceptable to listeners than those which had no intonation. This means that intonation modeling on the unit selection synthesis systems did improve the speech output to sound more natural.
- Overall, the prosodic voices performed as expected; with improvement in naturalness. The prosodic voices performed as expected, with naturalness improved (in particular as the main measure in this thesis) from Odom1 to Protest1, and from Odom2 to Protest2. The speech output of the individual synthetic voice systems for short sentences was slightly worse than that of long sentences by a difference of 0.03. This brings us to the conclusion that the length of the utterance does not really matter much in synthesis, as long as the right intonation has been implemented.
- For hybrid systems, an overall improvement in fluency of Systems 2a and 2b (from 3.42 to 3.6) is an indication that intonation is efficient in enhancing the flow of the speech

output. Also, as expected from the comparison of the F0 contour shapes of Ldom and the voice models for Test2 and Sphinx2, System 2b (Ldom/Sphinx2) is more fluent than System 2a (Ldom/Test2). This is verified by the MOS score obtained from the listening tests.

7.2 Future Work

For future work, the developed advanced hybrid systems for Sesotho can be enhanced by doing the following:

- Intensive studies should be done on duration analysis and modeling of Sesotho language.
- An investigation on stress characteristics in Sesotho and energy differences between phones.
- Before synthesizing text, the hybrid system goes through the input text and checks for words which match those in the dictionary and out of the dictionary. This causes a delay which is not appreciated by listeners. Therefore, in order to reduce runtime complexity of the unit selection process, some methods based on pre-selection of units can be implemented, which can also reduce the size of the speech unit database [17]. This will be appreciated by people who have to use, say, a service where information is read back to them over the phone using this system. This way, people will not get irritated by both the waiting and having to lose money on airtime.
- Work more on the default diphone used during unit selection synthesis.

Finally, an interesting question is this: Does the size of the database affect intonation? This is brought by an idea from [17] who argues that "... for systems that use relatively less amount of speech corpus, using prosody information in unit selection helps to select better units in terms of prosody, hence increasing the overall naturalness of synthetic speech. On the other hand, for larger corpus, we have more units in the corpus and the unit selection is more probable to find a better acoustic and prosodic match. In these systems, using prosody information may cause the unit selection to favor prosody over acoustic appropriateness which is probably more important than prosody for naturalness."

References

[1] "History of Speech Synthesis, 1770 – 1970: Wolfgang von Kempelen's speaking machine and its successors"
Available at: www.acoustics.hut.fi/~slemmett/dippa/chap2.html
(Last accessed on 28 March 2006).

[2] T. Kaneko, "Generating Intonation for a Japanese Speech Synthesizer", Fourth *Year project report*, University of Cambridge, 2001.

[3] S. Lemmetty, "Review of Speech Synthesis Technology", MSc Thesis, Helsinki University of Technology, 1999.

[4] F. Rousseau, "Design of an Advanced TTS System for Afrikaans", MSc Thesis, University of Cape Town, 2006.

[5] L. J. Kock and R. H. Moeketsi, *An Introduction to Sesotho Phonetics*, Marius Lubbe Publishers, 1990.

[6] "Sesotho Language",
Available at: http://fixedreference.org/en/20040424/wikipedia/Sesotho_language
(Last accessed on 12 February 2006).

[7] J. Wells, "IPA transcription systems for English "
Available at: http://www.phon.ucl.ac.uk/home/wells/ipa-english.htm
(Last accessed on 18 February 2006).

[8] S. M. Gama, *An Outline Structure of Southern Sotho*, Shuter & Shooter, Pietermaritzburg, 1975.

[9] A. Black and K. Lenzo, "Building Synthetic Voices", Language Technologies Institute, Carnegie Mellon University, 2003.

[10] A. Black, et. al., "The Festival Speech Synthesis System", System documentation, *Edition 1.4. for Festival Version 1.4.3*, 2002.

[11] P. Taylor, "The Rise/Fall/Connection Model of Intonation", Speech Communication, Vol. 15, Nos. 1&2, pp. 169-186, 1994.

[12] D. Colorado, C. Shih and G. Kochanski, "Prosody and Prosodic Models", *Proceedings of the 7th International Conference on Spoken Language Processing*, ICSLP, 2002.

[13] "Speech Synthesis using Concatenation of Speech Waveforms", Intellectual Property Digital Library, European Patent Office.
Available at: http://www.freepatentsonline.com/6665641.html
(Last accessed: 30 August 2006)

[14] T. Saito, M. Sakamoto, "Applying a Hybrid Intonation Model to a Seamless Speech Synthesizer", *Proceedings of the 7th International Conference on Spoken Language Processing*, ICSLP, 2002.

[15] R. A. Paroz, *Elements of Southern Sotho*, 1946.

[16] A. Louw, "A short guide to Pitch-marking in the Festival Speech Synthesis System and Recommendations for Improvements", Local Languages Speech Technology Initiative, CSIR, Pretoria, 2004.

[17] H. Sak, et. al., "A Corpus-Based Concatenative Speech Synthesis System for Turkish", Turkish Journal of Electrical Engineering, Vol. 14, No.1, 2006.

[18] A. Raux and A. Black, "A Unit Selection Approach to F0 Modeling and its Application to emphasis", ASRU St Thomas, US Virgin, 2003.

[19] J. Schroeter, "The Fundamentals of Text-to-Speech Synthesis", VoiceXML Review Forum, Vol. 1, Issue 3, Article #2, 2001.
Available at: www.voicexmlreview.org/Mar2001/features/tts.html
(Last accessed on 18 October 2005).

[20] A. Edwards, "ITD Notes: Speech Synthesis", Vol. 1, No. 2, 1994.
Available at: http://www.infomotions.com/serials/itd/itd-v1n02-contents.txt
(Last accessed on 18 October 2005).

[21] Edinburgh Speech Tools,
Available at: http://www.cstr.ed.ac.uk/projects/festival/download.html
(Last accessed on 26 January 2006).

[22] Festival Speech Synthesis software, Available at: http://www.cstr.ed.ac.uk/projects/festival/
(Last accessed on 14 April 2006).

[23] A. Black, "Speech Synthesis in Festival: A practical course on making computers talk", *Ed. 1.4.1 for Festival Version 2.0* (last updated May 8th 2000).

[24] "Speech Synthesis", Available at: http://www.fact-index.com/s/sp/speech_synthesis.html
(Last accessed on 10 January 2006).

[25] X. Sun, "The Determination, Analysis, and Synthesis of Fundamental Frequency", *PhD Thesis*, Northwestern University, 2002.

[26] K. Khiba, *Bophelo ke Ntoa*, Morija Sesuto Book Depot, 1986.

[27] Carnegie Mellon's FestVox project,
Available at: http://festvox.org
(Last accessed on 20 March 2006).

[28] R. Clark, "Generating Synthetic Pitch Contours Using Prosodic Structure", *PhD Thesis*, Department of Linguistics, University of Edinburgh, 2003.

[29] "ToBI", Available at: http://www.ling.ohio-state.edu/~tobi/
(Last accessed on 12 April 2006).

[30] M. Tatham, et. al., "Syllable Reconstruction in concatenated waveform Speech synthesis", *Proceedings of the International Congress of Phonetic Sciences*, pp. 2303 – 2306, 1999.

[31] "Speech by the Honourable the Prime Minister Mr. Pakalitha Mosisili, MP at the Launch of the National Vision and Strategy for the Justice Sector National Convention Centre 20 June, 2005",
Available at: http://www.lesotho.gov.ls/articles/2005/Speech_PM_Justice_Vision.htm
(Last accessed on 16 January 2006).

[32] E. Jacottet, *A Practical way of learning Sesuto*, 1972.

[33] MrSci.com – Dynamic Time Warping definition,
Available at: http://www.mrsci.com/Artificial-intelligence/Dynamic_time_warping.php
(Last accessed on 10 February 2006).

[34] A. Black, "Limited Domain Speech Synthesis: Building a Limited domain synthesizer with Festival", Adapted from the Festvox documentation,
Available at:
http://fonpc18.hum.uva.nl/SpeechRecognitionAndSynthesis/WWW/www.cstr.ed.ac.uk
(Last accessed on 10 February 2006).

[35] Sphinx2 and SphinxTrain source, Available at: http://sourceforge.net/projects/cmusphinx/
(Last accessed on 12 January 2006).

[36] Multitel Text-to-Speech Group,
Available at: http://www.multitel.be/TTS/layout.php?page+LiONS_eval_intro
(Last accessed on 18 February 2006).

[37] South African Government Services, Available at: www.services.gov.za
(Last accessed on 21 February 2006).

[38] A. Syrdal, et.al. "Three Methods of Intonation Modelling", *In Third International Workshop on Speech Synthesis*, Australia,1998.

[39] A. Black and K. Lenzo, "Limited Domain Synthesis", *International Conference on Spoken Language Processing*, *ICSLP*, China, 2000.

[40] K. Dusterhoff and A. Black, "Generating F0 contour for Speech Synthesis using the Tilt Intonation Theory", *In Proceedings of ESCA Workshop on Intonation*, 1997.

[41] J. van Santen, et.al "Synthesis of Prosody using Multi-level Unit Sequences", *Speech Communication*, Vol. 36, pp 365-375, 2005.

[42] J. Meron, "Prosodic Unit Selection using an Imitation Speech Database", *4th ISCA Workshop on Speech Synthesis*, pp. 53-57, 2001.

[43] A. Black and P. Taylor, "Automatically Clustering Similar Units for Unit Selection in Speech Synthesis", *In Proceedings of EUROSPEECH'97*, pp. 601 - 604, 1997.

[44] X. Huang, et. al.,"Whistler: A trainable Text-to-Speech system", *In Proceedings of the International Conference on Spoken Language Processing*, Vol. 4, pp.169-172, 1996.

[45] A. Black and K. Lenzo, "Optimal Data Selection for Unit Selection Synthesis", *In 4th ESCA Workshop on Speech Synthesis*, 2001.

[46] X. Sun, "Predicting Underlying Pitch targets for Intonation Modeling", *In Proceedings of the 4th ISCA Tutorial and Research Workshop on Speech Synthesis*, pp.143–148, 2001.

[47] G. Möhler and J. Mayer, "A Discourse Model for Pitch-Range Control", *In 4th ISCA workshop on Speech Synthesis*, 2001.

[48] T. Saito, "Generating F0 Contours by Statistical Manipulation of Natural F0 Shapes", *IEEE Transactions Information and System*, Vol. E89-D, No. 3, March 2006.

[49] T. Saito, "On the Use of F0 features in Automatic Segmentation for Speech Synthesis", ICSLP, Vol. 7, pp. 2839-2842, 1998.

[50] T. Saito and M. Sakamoto, "A Method of creating a New Speaker's Voicefont in a Text-to-Speech system", *ICSLP*, Vol. 2, pp. 771-774, 2000.

[51] Y. Yamashita and T. Ishida, "Stochastic F0 Contour model based on the Clustering of F0 shapes of a Syntactic Unit", *7th European Conference on Speech Communication and Technology*, pp. 533-536, 2001.

[52] N. Govender, et. al., "Developing Intonation corpora for isiXhosa and isiZulu", *In Proceedings of the 16th Annual Symposium of the Pattern Recognition Association of South Africa*, pp. 129-132, 2005.

[53] M. Heldner, et. al., "Automatically extracted F0 Features as Acoustic Correlates of Prosodic Boundaries", *In Proceedings of Fonetik*, 2004.

[54] E. Keller and S. Werner, "Automatic Intonation Extraction and Generation for French", *14th CALICO Annual Symposium*, June 1997.

[55] T. Saito and M. Sakamoto, "Generating F0 contours by statistical manipulation of natural F0 shapes", *In Proceedings of EUROSPEECH*, 2001.

[56] F. Tamburini and C. Caini, "An automatic system for detecting Prosodic Prominence in American English continuous speech", *International Journal of Speech Technology*, Vol. 8, pp 33-44, 2005.

[57] K. Dusterhoff, et. al., "Using decision trees within the Tilt Intonation model to predict F0 contours", *In Proceedings of EUROSPEECH*, pp. 1627–1630, 1999.

[58] C. Tseng, et.al., "Fluent speech prosody: Framework and modeling", *Speech Communication*, Vol. 46, pp 284-309, 2005.

[59] Meraka Institute - Human Language Technologies, Available at: http://www.meraka.org.za/humanLanguage.htm (Last accessed on 18 February 2006).

[60] D. O'Shaughnessy, *Speech Communication*, Addison-Wesley Publishing Company, Reading, Massachusetts, 1987.

[61] H. Fujisaki, "Fujisaki Model: Dynamic characteristics of voice fundamental frequency in speech and singing", In Peter F. MacNeilage (Ed), *The production of speech,* Springer, New York, pp. 39– 55, 1983.

[62] B. Moebius, "Synthesizing German Intonation Contours", In J.P. van Santen, R. Sproat, J. Olive, and J. Hirschberg, editors, *Progress in Speech Synthesis*, pp. 401 - 415, 1996.

[63] G. Möhler and A. Conkie , "Parametric Modeling of Intonation using Vector Quantization", *Proceedings of 3rd ESCA Workshop on Speech Synthesis*, Australia, 1998.

[64] K. Ross and M. Ostendorf, "Prediction of Abstract Prosodic labels for Speech Synthesis", *Computer Speech and Language*, Vol.10, pp155- 185, 1996.

[65] P. Taylor, "The Rise/Fall/Connection model of Intonation", *Speech Communication*, Vol. 15, pp.169-186, 1995.

[66] P. Taylor, "Analysis and Synthesis of Intonation Using the Tilt Model", *Journal of the Acoustical Society of America*, Vol. 107, No. 3, pp. 1697-1714, 2000.

[67] "History: Sesotho – Southern Sotho",
Available at: http://www.cyberserv.co.za/users/~jako/lang/ses.htm
(Last accessed on 21 November 2005).

[68] R. N. J., Veldhuis, and A. Kohlrausch, "Waveform coding and Auditory Masking", *Speech Coding and Synthesis*, Elsevier Science Publishers, Amsterdam, pp. 101-133, 1995.

[69] P. Taylor and A. W. Black., "Synthesizing Conversational Intonation from a Linguistically Rich Input", *In Proc. ESCA Workshop on Speech Synthesis*, pp. 175-178, 1994.

[70] R. O. Duda, et. al., *Pattern Classification*, 2nd Ed., Wiley-Interscience, 2000.

[71] P. G. Underwood, "A Case Study in Development Through Information Literacy", *White paper prepared for UNESCO, the U.S. National Commission on Libraries and Information Science, and the National Forum on Information Literacy,* Prague 2003.

[72] "South Africa grows to 44.8-million"
Available at:
http://www.southafrica.info/ess_info/sa_glance/demographics/census-main.htm
(Last updated 9 July 2003)
(Last accessed on 10 February 2006).

[73] "Free State" – Wikipaedia, the free encyclopaedia,
Available at: http://en.wikipedia.org/wiki/Free_State
(Last accessed on 18 January 2006).

Appendices

Appendix A: An example of the MOS questionnaire for individual voice systems.

Appendix B: An example of the MOS questionnaire for hybrid systems.

Appendix C: An audio DVD of the individual voice systems and the hybrid systems.

Appendix A - An example of the MOS questionnaire for individual voice systems.

Questionnaire A1: Long Sentences

Title: Design of an Advanced and Fluent Sesotho Text-to-Speech System through Intonation
Date:
Citizenship:

Sentence 1: Ke mang ea ka fumanang tlhokomelo ka tlas'a molao oo?

Question 1: How much listening effort is required in understanding the voice?

	Diphone	**Ldom**	**Test**	**Test2**	**Sphinx**	**Sphinx2**
5. No effort required						
4. Little effort required						
3. Fair amount of effort required						
2. Vast amount of effort required						
1. Cannot understand the voice						

Question 2: Does the system sound close to a human voice?

	Diphone	**Ldom**	**Test**	**Test2**	**Sphinx**	**Sphinx2**
5. Sounds like a human voice						
4. Natural enough to listen to						
3. Acceptable but lacking						
2. Unnatural						
1. Completely unnatural						

Question 3: Is the system pleasant to listen to?

	Diphone	**Ldom**	**Test**	**Test2**	**Sphinx**	**Sphinx2**
5. Very pleasant.						
4. Quite pleasant						
3. Acceptable						
2. Unpleasant						
1. Horrible						

Question 4: What is your overall impression of the system?

	Diphone	**Ldom**	**Test**	**Test2**	**Sphinx**	**Sphinx2**
5. Excellent						
4. Good						
3. Acceptable						
2. Poor						
1. Horrible						

Appendix B - An example of the MOS questionnaire for hybrid systems.

Long Sentence 1

Title: Design of an Advanced and Fluent Sesotho Text-to-Speech System through Intonation
Date:
Citizenship:

Question 1: How much listening effort is required in understanding the system?

	System 1a	System 1b	System 2a	System 2b
5. No effort required				
4. Little effort required				
3. Fair amount of effort required				
2. Vast amount of effort required				
1. Cannot understand the voice				

Question 2: Does the system sound close to a human voice?

	System 1a	System 1b	System 2a	System 2b
5. Sounds like a human voice				
4. Natural enough to listen to				
3. Acceptable but lacking				
2. Unnatural				
1. Completely unnatural				

Question 3: How fluent is the system? Take into account concatenation points.

	System 1a	System 1b	System 2a	System 2b
5. Very fluent				
4. Quite fluent				
3. Acceptable				
2. Concatenation points fairly audible				
1. . Concatenation points very audible				

Question 4: What is your overall impression of the system?

	System 1a	System 1b	System 2a	System 2b
5. Excellent				
4. Good				
3. Acceptable				
2. Poor				
1. Horrible				

Appendix C - An audio DVD of the individual voice systems and the hybrid systems.

The audio DVD includes six individual voice systems. The voices are in the folder **Thesis Voice Systems** and they are:

- Diphone voice – ***uct_ses_hlo_diphone***.
- Limited domain voice – ***uct_time_hlo3_ldom***.
- Open domain voice 1 (DTW labeled) – ***uct_ses_test_clunits***.
- Open domain voice 1 with intonation – ***uct_ses_test2_clunits***.
- Open domain voice 2 (Sphinx labeled) – ***uct_ses_sphinx_clunits***.
- Open domain voice 2 with intonation – ***uct_ses_sphinx2_clunits***.

The DVD also contains the four hybrid systems, two of which intonation has been applied. The systems are in the folder **Hybrid Distribution** and they are:

- System 1a whose labeling technique for the open domain voice is based on dynamic time warping. This system has no intonation applied to it.
- System 1b whose labeling technique for the open domain voice is based on Sphinx. This system has no intonation applied to it.
- System 2a which is based on System 1a. This system has intonation applied to it.
- System 2b which is based on System 1b. This system has intonation applied to it.