SPEECH SYNTHESIS

A Thesis report submitted towards the partial fulfillment of requirement for the award of the degree of

Master of Engineering
(Electronics Instrumentation &Control)

Submitted by:

Ritu Sharma

Roll No. 8044214

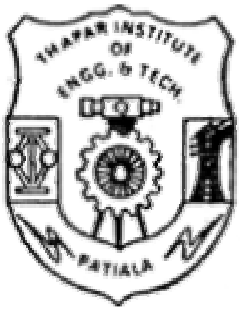
Under Esteemed Guidance:

Mr. Sunil Kumar Singla

Senior Lecturer, EIED TIET Patiala

Mr. Nirbhow Jap Singh

Lecturer, EIED TIET Patiala



Electrical and Instrumentation Engineering Department
THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY,
(Deemed University), PATIALA – 147004, INDIA

June 2006

ABSTRACT

Speech synthesis is the artificial production of human speech. A system used for this purpose is termed a **speech synthesizer**, and can be implemented in software or hardware. Speech synthesis systems are often called **text-to-speech (TTS)** systems in reference to their ability to convert text into speech. However, systems exist that instead render symbolic linguistic representations like phonetic transcriptions into speech.

A **text-to-speech system** is composed of two parts: a **front-end** and a **back-end**. Broadly, the front-end takes input in the form of text and outputs a symbolic linguistic representation. The back-end takes the symbolic linguistic representation as input and outputs the synthesized speech waveform . TTS software can "read" text from a document, Web page or e-Book, generating synthesized speech through a computer's speakers. TTS can also convert text files into audio MP3 files that can then be transferred to a portable MP3 player or CD-ROM. This can save time by allowing the user to listen to reports or background materials while performing other tasks. TTS makes a critical difference to those with disabilities such as poor vision or visual dyslexia. People with speech loss can utilize specialized TTS programs to turn typed words into vocalization. TTS programs provide a valuable edge, particularly for learning new languages.

This thesis aims to study the speech synthesis technology and to develop a cost effective, user friendly **text to speech conversion system** using Laboratory virtual instruments engineering workbench (LabVIEW) graphical programming language.

ACKNOWLEDGEMENT

Words are often to less to revels one's deep regards. An understanding of the work like this is never the outcome of the efforts of a single person. I take this opportunity to express my profound sense of gratitude and respect to all those who helped me through the duration of this thesis.

First, I would like to thank the Supreme Power "The God", one who has always guided me to work on the right path of the life. Without his grace, this would never come to be today's realty.

I am grateful to Head of the Department **Mrs. Manbir Kaur** for providing the encouragement & facilities for the completion of this Thesis.

This work would not have been possible without the encouragement and able guidance of my guides **Mr. Sunil Kumar Singla & Mr.Nirbhow Jap Singh,** their enthusiasm and optimism made this experience both rewarding and enjoyable. Most of the novel ideas and solutions found in this seminar are the result of our numerous stimulating discussions. Their feedback and editorial comments were also invaluable for the same.

At last, I would like to thank to all the members of Electrical and Instrumentation Department whose love and affection made my stay at T.I.E.T campus a memorable.

Place: T.I.E.T. Patiala, India (Ritu Sharma)

DECLARATION

I hereby declare that the Thesis report entitled **"SPEECH SYNTHESIS**" is an authentic record of my own work carried out as requirements for the award of degree of M.E. (Electronics Instrumentation & Control) at *Thapar Institute of Engineering & Technology (Deemed University), Patiala,* under the guidance of "**Mr. Sunil Kumar Singla**, Senior Lecturer, EIED TIET Patiala & **Mr. Nirbhow Jap Singh**" Lecturer, EIED TIET Patiala during January to June 2006.

(Ritu Sharma)
Roll No. 8044214

Date: June 2006

CERTIFICATE

Certified that the above statement made by the student is correct to the best of my knowledge and belief.

(Sunil Kumar Singla)
Senior Lecturer, EIED
TIET,Patiala

(Nirbhow Jap Singh)
Lecturer,EIED
TIET,Patiala

Counter Signed:

(Manbir Kaur)
Head of Department EIED,
T.I.E.T,Patiala

(Dr.T.P.Singh)
Dean of Acadmic Affairs
T.I.E.T,Patiala

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE No.
	Abstract……	i
	Acknowledgement…….....	ii
	Certificate…….	iii
	Table of contents……..	iv
	List of figures……..... ……	vii
Chapter 1	**Introduction**……...…….....	1
	1.1 Introduction to Speech Synthesis…….	1
	1.2. Phonetics and Theory of Speech Production……...	2
	1.2.1 Representation and Analysis of Speech Signals…….....	2
	1.2.2 Speech Production……	5
	1.2.3 Phonetics…….	8
	1.2.3.1 English Articulatory Phonetics …….....	10
	1.3. Problems in Speech Synthesis……..	11
	1.3.1 Text-to-Phonetic Conversion……..	12
	1.3.1.1 Text preprocessing…….	12
	1.3.1.2 Pronunciation……	13
	1.3.1.3Prosody……	14
	1.4 ProblemDefination ……..	15

Chapter 2 Literature Survey ……. 17

2.1 Trainable Text To Speech System……..... 17

2.2 Methods, Techniques, and Algorithms……. 19

2.2.1 Articulatory Synthesis…….. 19

2.2.2 Formant Synthesis…… 20

2.2.3 Concatenative Synthesis…….. 23

2.2.4 Sinusoidal Models…… 26

2.2.5 High-Level Synthesis…… 27

2.2.5.1 Text Preprocessing…… 28

2.2.5.2 Pronunciation…… 29

2.2.5.3 Prosody…… 30

2.3 Other Methods and Techniques..... 34

2.4 Applications of Synthetic Speech..... 35

2.4.1 Applications for the Blind..... 35

2.4.2 Applications for the Deafened and Vocally Handicapped..... 37

2.4.3 Educational Applications..... 38

2.4.4 Applications for Telecommunications and Multimedia..... 38

2.4.5 Other Applications and Future Directions..... 39

2.5 Application Frameworks..... 40

2.5.1 Speech Application Programming Interface..... 41

2.5.2 Internet Speech Markup Languages..... 41

2.5.3 MPEG-4 TTS..... 42

2.5.3.1 Applications of MPEG-4 TTS..... 42

Chapter 3 LabVIEW Graphical Programming Language…… 44

3.1 LabVIEW Program Structure…….. 44

3.2 Dataflow Programming……... 48

3.3 Graphical Programming…….. 49

3.4 Benefits…… 49

Chapter 4 Software Implementation ……. 51

4.1 Text To Speech Interface Module Using Labview…….. 51

4.1.1 ActiveX and LabVIEW…… 51

4.1.2 LabVIEW as an Automation Client…… 52

4.1.2.1 Automation Open ……. 53

4.1.2.2 Invoke Node……. 54

4.1.2.3 Property Node ……. 54

4.1.2.4 Close Reference…… 55

4.1.2.5 Type Cast…….. 55

4.2 wave file player.vi……... 58

4.2.1 SO Start ……. 58

4.2.2 Snd Read Wave File…….. 58

4.2.3 SO Config……. 59

4.2.4 SO Write …….. 60

4.2.5 SO Clear …… 60

4.2.1 Build Screen.vi..... 61

Chapter 5 Results and Conclusions ……..…… 62

5.1Results……. 62

5.2Conclusions…….. 64

5.3Future Scope……..... 65

References and Literature…….. 66

LIST OF FIGURES

FIGURE No.	NAME OF FIGURE	PAGE No.
1.1	The time- and frequency-domain presentation of vowels /a/, /i/, and /u/.	3
1.2	Cepstral analysis.	4
1.3	Hierarchical levels of fundamental frequency	4
1.4	The human vocal organs	5
1.5	Examples of two- and three-tube models for the vocal tract.	7
1.6	The classification of the main vowels in English	10
1.7	Classification of English consonants	11
1.8	Prosodic dependencies	15
2.1	Simple text-to-speech synthesis procedure	18
2.2	Basic structure of cascade formant synthesizer	21
2.3	Basic structure of a parallel formant synthesizer	22
2.4	Sinusoidal analysis / synthesis system	27
2.5	Basic idea of the hybrid synthesis system	34
4.1	Programming flow of ActiveX used in LabVIEW	53
4.2	Flowchart for the text to wave file conversion	56
4.3	Flowchart for wave file player.vi	60
4.4	Front Panel diagram of Build Screen.vi	61
5.1	Front Panel Diagram of TEXT TO SPEECH .VI	62
5.2	Waveform for “GOOD MORNING”	64

CHAPTER 1

Introduction

1.1 INTRODUCTION TO SPEECH SYNTHESIS

Knowledge extraction by just listening to sounds is a distinctive property and has become an important milestone in the evolution of species. Most of the animals are not only equipped with the means to extract information from the rich acoustical content of the environment and act accordingly, but they have the ability to produce sounds to interact with the environment as well. Humans have gone one step further, they have fairly advanced mechanisms that enable interaction within the species by very abstract rules of communication using voice – the language.

Speech is the primary means of communication between people. Speech synthesis, automatic generation of speech waveforms, has been under development for several decades [1].Recent progress in speech synthesis has produced synthesizers with very high intelligibility but the sound quality and naturalness still remain a major problem. However, the quality of present products has reached an adequate level for several applications, such as multimedia and telecommunications. With some audiovisual information or facial animation (talking head) it is possible to increase speech intelligibility considerably [2].

Preparing for research in voice synthesis requires a fairly diverse range of coursework and understanding. Many biological and physiological functions must be understood for use as a background. In order to synthesize human speech, an understanding must be developed of how humans are capable of such a task.

By dissecting speech and its process into measurable and analytical qualities, instruments and techniques can be created to reproduce the aspects of speech.

1.2 PHONETICS AND THEORY OF SPEECH PRODUCTION

Speech processing and language technology contains lots of special concepts and terminology. To understand how different speech synthesis and analysis methods work one must have some knowledge of speech production, articulatory phonetics, and some other related terminology. The basic theory of these topics will be discussed briefly in this topic.

1.2.1 REPRESENTATION AND ANALYSIS OF SPEECH SIGNALS

Continuous speech is a set of complicated audio signals which makes producing them artificially difficult. Speech signals are usually considered as voiced or unvoiced, but in some cases they are something between these two. Voiced sounds consist of fundamental frequency (F0) and its harmonic components produced by vocal cords (vocal folds). The vocal tract modifies this excitation signal causing formant (pole) and sometimes antiformant (zero) frequencies .Each formant frequency has also amplitude and bandwidth and it may be sometimes difficult to define some of these parameters correctly. The fundamental frequency and formant frequencies are probably the most important concepts in speech synthesis and also in speech processing in general.

With purely unvoiced sounds, there is no fundamental frequency in excitation signal and therefore no harmonic structure either and the excitation can be considered as white noise. The airflow is forced through a vocal tract constriction which can occur in several places between glottis and mouth. Some sounds are produced with complete stoppage of airflow followed by a sudden release, producing an impulsive turbulent excitation often followed by a more protracted turbulent excitation [3]. Unvoiced sounds are also usually more silent and less steady than voiced ones. Whispering is the special case of speech. When whispering a voiced sound there is no fundamental frequency in the excitation and the first formant frequencies produced by vocal tract are perceived. Speech signals of the three vowels (/a/ /i/ /u/) are presented in time- and frequency domain in Figure 1.1.

The fundamental frequency is about 100 Hz in all cases and the formant frequencies F1, F2, and F3 with vowel /a/ are approximately 600 Hz, 1000 Hz, and 2500 Hz respectively. With vowel /i/ the first three formants are 200 Hz, 2300 Hz, and 3000 Hz, and with /u/ 300 Hz, 600 Hz, and 2300 Hz. The harmonic structure of the excitation is also easy to perceive from frequency domain presentation.

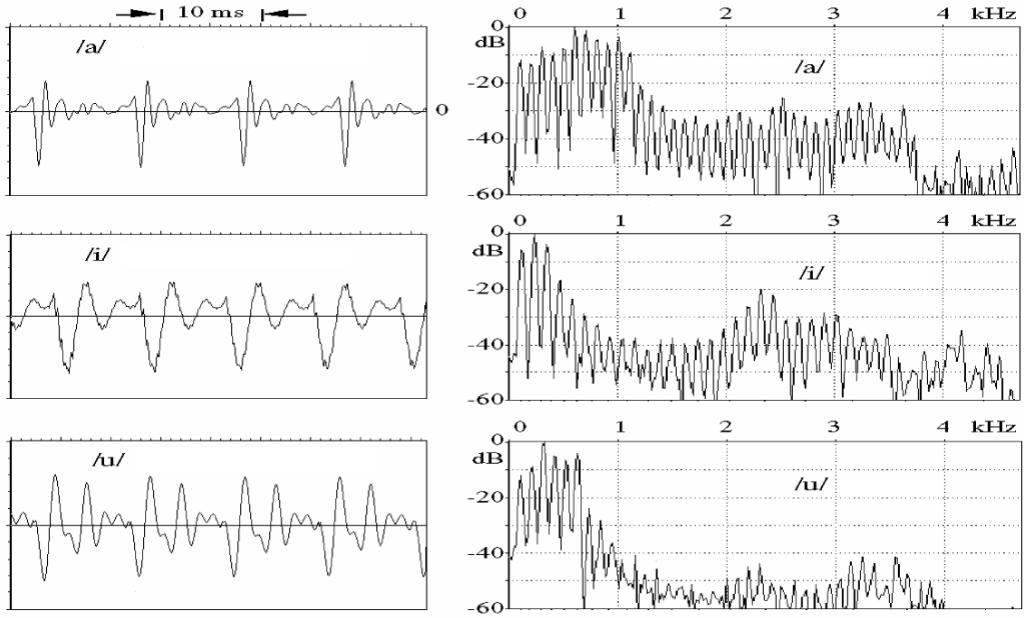


Fig. 1.1. The time- and frequency-domain presentation of vowels /a/, /i/, and /u/.

It can be seen that the first three formants are inside the normal telephone channel (from 300 Hz to 3400 Hz) so the needed bandwidth for intelligible speech is not very wide. For higher quality, up to 10 kHz bandwidth may be used which leads to 20 kHz sampling frequency. Unless, the fundamental frequency is outside the telephone channel, the human hearing system is capable to reconstruct it from its harmonic components.

For determining the fundamental frequency or pitch of speech, for example a method called cepstral analysis may be used [4]. Cepstrum is obtained by first windowing and making Discrete Fourier Transform (DFT) for the signal and then logaritmizing power spectrum and finally transforming it back to the time-domain by Inverse Discrete Fourier Transform (IDFT). The procedure is shown in Figure 1.2

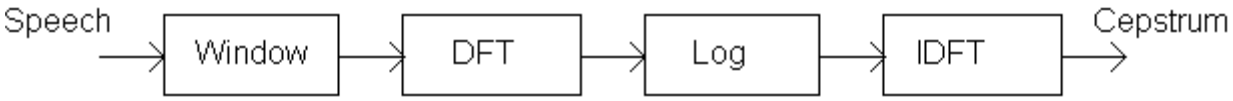


Fig. 1.2. Cepstral analysis.

Cepstral analysis provides a method for separating the vocal tract information from excitation. Thus the reverse transformation can be carried out to provide smoother power spectrum known as homomorphic filtering. Fundamental frequency or intonation contour over the sentence is important for correct prosody and natural sounding speech. The different contours are usually analyzed from natural speech in specific situations and with specific speaker characteristics and then applied to rules to generate the synthetic speech. The fundamental frequency contour can be viewed as the composite set of hierarchical patterns shown in Figure 1.3. The overall contour is generated by the superposition of these patterns [5].

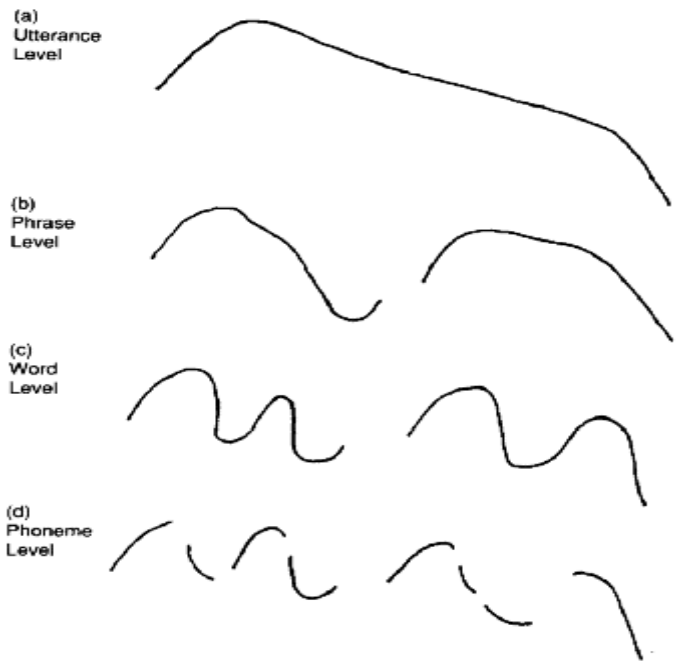


Fig. 1.3. Hierarchical levels of fundamental frequency (Sagisaga 1990).

1.2.2 SPEECH PRODUCTION

Human speech is produced by vocal organs presented in Figure 1.4. The main energy source is the lungs with the diaphragm. When speaking, the 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 V-shaped opening between the vocal cords, called 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 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. 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, such as /s/ or /f/, they may be completely open. An intermediate position may also occur with for example phonemes like /h/.

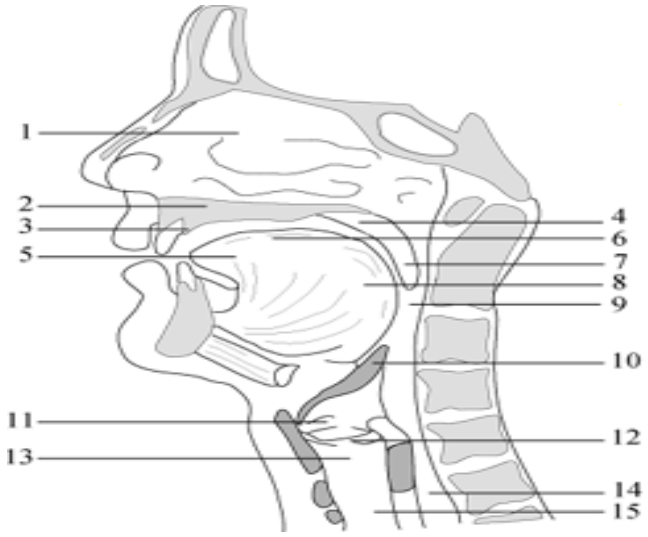


Fig. 1.4 The human vocal organs. (1) Nasal cavity, (2) Hard palate, (3) Alveoral ridge, (4) Soft palate (Velum), (5) Tip of the tongue (Apex), (6) Dorsum, (7) Uvula, (8) Radix, (9) Pharynx, (10) Epiglottis, (11) False vocal cords, (12) Vocal cords, (13) Larynx, (14) Esophagus, and (15) Trachea.

The 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 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 the movements of the palate, the tongue, the lips, the cheeks and the teeth. Especially the tongue is very flexible, 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 12 cm and volume 60 cm^3. The air stream to the nasal cavity is controlled by the soft palate.

From technical point of view, the vocal system may be considered as a single acoustic tube between the glottis and mouth. Glottal excited vocal tract may be then approximated as a straight pipe closed at the vocal cords where the acoustical impedance $Z_g = \infty$ and open at the mouth ($Z_m = 0$).

In this case the volume-velocity transfer function of vocal tract is [6]:

$$V(\omega) = \frac{Z_m}{Z_g} = \frac{U_m}{U_g} = \frac{1}{\cos(\frac{\omega l}{c})}$$

Where l is the length of the tube, ω is radian frequency and c is sound velocity.

Vowels can be approximated with a two-tube model presented on the left in Figure 1.5. For example, with vowel /a/ the narrower tube represents the pharynx opening into wider tube representing the oral cavity. If assumed that both tubes have an equal length of 8.5 cm, formants occur at twice the frequencies noted earlier for a single tube. Due to acoustic coupling, formants do not approach each other by less than 200 Hz so formants F1 and F2 for /a/ are not both at 1000 Hz, but rather 900 Hz and 1100 Hz, respectively [7].

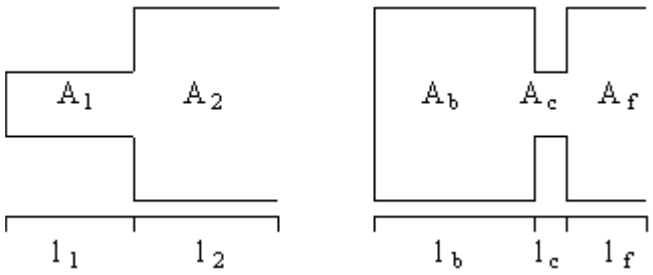


Fig. 1.5. Examples of two- and three-tube models for the vocal tract.

Consonants can be approximated similarly with a three-tube model shown on the right in Figure 1.4., where the narrow middle tube models the vocal tract constriction. The back and middle tubes are half-wavelength resonators and the front tube is a quarter wavelength resonators with resonances:

$$\frac{ci}{2l_b}, \frac{ci}{2l_c}, \frac{c(2i-1)}{4l_f}$$, for i = 1, 2, 3…

where l_b, l_c, and l_f are the length of the back, center, and front tube, respectively. With the typical constriction length of 3 cm the resonances occur at multiples of 5333 Hz and can be ignored in applications that use less than 5 kHz bandwidth.

The excitation signal may be modeled with a two-mass model of the vocal cords which consists of two masses coupled with a spring and connected to the larynx by strings and dampers [8].

Several other methods and systems have been developed to model the human speech production system to produce synthetic speech. These methods are related with articulatory synthesis described in Chapter 2.

1.2.3 PHONETICS

In most languages the written text does not correspond to its pronunciation so that in order to describe correct pronunciation some kind of symbolic 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 [7]. A set of phonemes can be defined as the minimum number of symbols needed to describe every possible word in a language. In English there are about 40 phonemes [9]. Due to complexity and different kind of definitions, the number of phonemes in English and most of the other languages can not be defined exactly.

Phonemes are abstract units and their pronunciation depends on contextual effects, speaker's characteristics, and emotions. During continuous speech, the articulatory movements depend on the preceding and the following phonemes. The articulators are in different position depending on the preceding one and they are preparing to the following phoneme in advance. This causes some variations on how the individual phoneme is pronounced. These variations are called allophones which are the subset of phonemes and the effect is known as coarticulation. For example, a word *lice* contains a light /l/ and *small* contains a dark /l/. These l's are the same phoneme but different allophones and have different vocal tract configurations. Another reason why the phonetic representation is not perfect is that the speech signal is always continuous and phonetic notation is always discrete [10].

Different emotions and speaker characteristics are also impossible to describe with phonemes so the unit called phone is usually defined as an acoustic realization of a phoneme [11].

The 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 [12].

Some efforts to construct language-independent phonemic alphabets were made during last decades. One of the best known is perhaps IPA (International Phonetic Alphabeth) which consists of a huge set of symbols for phonemes, suprasegmentals, tones/word accent contours, and diacritics. For example, there are over twenty symbols for only fricative consonants [13].

Another such kind of phonetic set is SAMPA (Speech Assessment Methods - Phonetic Alphabet) which is designed to map IPA symbols to 7-bit printable ASCII characters. In SAMPA system, the alphabets for each language are designed individually. Originally it covered European Communities languages, but the objective is to make it possible to produce a machine-readable phonetic transcription for every known human language. Alphabet known as Worldbet is another ASCII presentation for IPA symbols which is very similar to SAMPA [14]. Few examples of different phonetic notations are given in Table 1.1.

Table 1.1. Examples of different phonetic notations.

IPA	IPA-ASCII	SAMPA	DECtalk	Example
i	i	i:	iy	beet
I	I	I	ih	bit
ɛ	E	e	ey	bet
æ	&	{	ae	at
ə	@	@	ax	about
ʌ	V	V	ah	but

Several other phonetic representations and alphabets are used in present systems. There is still no single generally accepted phonetic alphabet.

1.2.3.1 English Articulatory Phonetics

The number of phonetic symbols used in English varies by different kind of definitions. Usually there are about ten to fifteen vowels and about twenty to twenty-five consonants.

English vowels may be classified by the manner or place of articulation (front-back) and by the shape of the mouth (open - close). Main vowels in English and their classification are described in Figure 1.6 below. Sometimes also some diphthongs like /ou/ in *tone* or /ei/ in *take* are described separately.

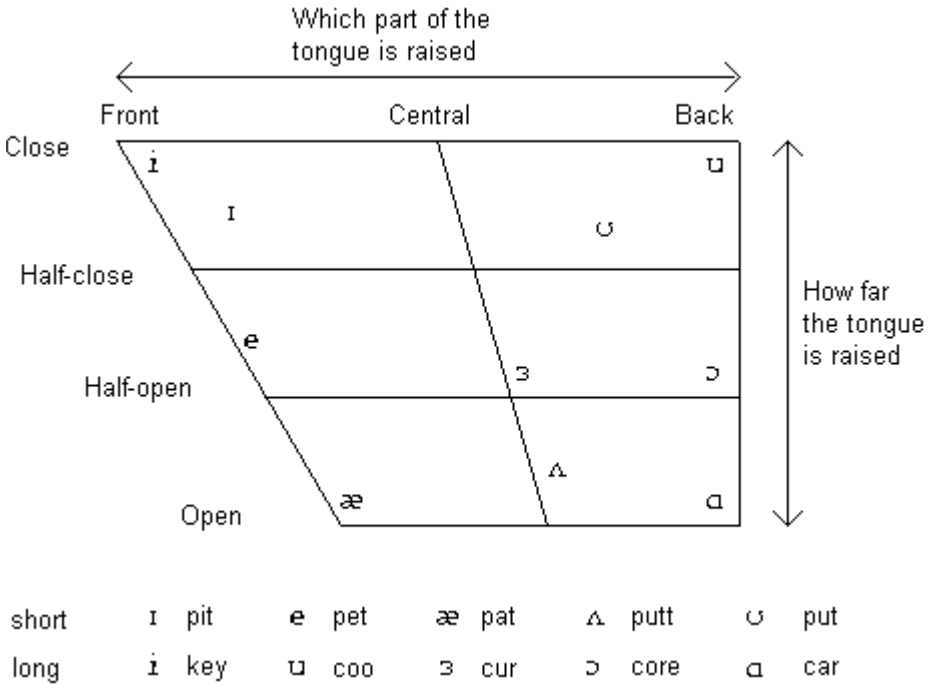


Fig. 1.6. The classification of the main vowels in English (Cowie 1996).

English consonants may be classified by the manner of articulation as plosives, fricatives, nasals, liquids, and semivowels. Plosives are known also as stop consonants. Liquids and semivowels are also defined in some publications as approximants and laterals. Further classification may be made by the place of articulation as labials (lips), dentals (teeth), alveolars (gums), palatals (palate), velars (soft palate), glottal (glottis), and labiodentals (lips and teeth). Classification of English consonants is summarized in Figure 1.7.

place / manner	labial	labio-dental	dental	alveolar	palate-alveoral	palatal	velar	glottal
plosive	p b			t d			k g	
fricative		f v	q ð	s z	ʒ ʃ			h
nasal	m			n			ŋ	
liquid				r l				
semivowel	w					j		

Fig. 1.7. Classification of English consonants (Cowie 1996).

Finally, consonants may be classified as voiced and unvoiced. Voiced consonants are:

/b/, /d/, /g/, /v/, /z/, / ʒ /, / ð /, /l/, /r/, and /j/ others are unvoiced.

1.3 PROBLEMS IN SPEECH SYNTHESIS

The problem area in speech synthesis is very wide. There are several problems in text pre-processing, such as numerals, abbreviations, and acronyms. Correct prosody and pronunciation analysis from written text is also a major problem today. Written text contains no explicit emotions and pronunciation of proper and foreign names is sometimes very anomalous. At the low-level synthesis, the discontinuities and contextual effects in wave concatenation methods are the most problematic. Speech synthesis has been found also more difficult with female and child voices. Female voice has a pitch almost twice as high as with male voice and with children it may be even three times as high.

The higher fundamental frequency makes it more difficult to estimate the formant frequency locations [15] .The evaluation and assessment of synthesized speech is neither a simple task. Speech quality is a multidimensional term and the evaluation method must be chosen carefully to achieve desired results.

1.3.1 TEXT-TO-PHONETIC CONVERSION

The first task faced by any TTS system is the conversion of input text into linguistic representation, usually called text-to-phonetic or grapheme-to-phoneme conversion. The difficulty of conversion is highly language depended and includes many problems. For English and most of the other languages the conversion is much more complicated. A very large set of different rules and their exceptions is needed to produce correct pronunciation and prosody for synthesized speech. Conversion can be divided in three main phases, text preprocessing, creation of linguistic data for correct pronunciation, and the analysis of prosodic features for correct intonation, stress, and duration.

1.3.1.1 Text preprocessing

Text preprocessing is usually a very complex task and includes several language dependent problems. Digits and numerals must be expanded into full words. For example in English, numeral 243 would be expanded as *two hundred and forty-three* and 1750 as *seventeen-fifty* (if year) or *one-thousand seven-hundred and fifty* (if measure). Fractions and dates are also problematic. 5/16 can be expanded as *five-sixteenths* (if fraction) or *May sixteenth* (if date). Expansion ordinal numbers have been found also problematic. The first three ordinals must be expanded differently than the others, 1st as *first*, 2nd as *second*, and 3rd as *third*. Same kind of contextual problems are faced with roman numerals. Chapter III should be expanded as *Chapter three* and Henry III as *Henry the third* and *I* may be either a pronoun or number. Roman numerals may be also confused with some common abbreviations, such as MCM.

Numbers may also have some special forms of expression, such as 22 as *double two* in telephone numbers and 1-0 as *one love* in sports.

Abbreviations may be expanded into full words, pronounced as written or pronounced letter by letter [16]. Some contextual problems for example kg can be either *kilogram* or *kilograms* depending on preceding number, St. can be *saint* or *street* and ft. *Fort*, *foot* or *feet*.

In some cases, the adjacent information may be enough to find out the correct conversion, but to avoid misconversions the best solution in some cases may be the use of letter-to-letter conversion.

Special characters and symbols, such as '$', '%', '&', '/', '-', '+', cause also special kind of problems. In some situations the word order must be changed. For example, *$71.50* must be expanded as *seventy-one dollars and fifty cents* and *$100 million* as *one hundred million dollars*, not as *one hundred dollars million*. The expression '1-2' may be expanded as *one minus two* or *one two*, and character '&' as *end* or *and*. Also special characters and character strings in for example web-sites or e-mail messages must be expanded with special rules. For example, character '@ ' is usually converted as *at* and e-mail messages may contain character strings, such as some header information, which may be omitted. Some languages also include special non ASCII characters, such as accent markers or special symbols.

Written text may also be constructed in several ways, like in several columns and pages as in a normal newspaper article. This may cause insuperable problems especially with optical reading machines.

1.3.1.2 Pronunciation

The second task is to find correct pronunciation for different contexts in the text. Some words, called *homographs*, cause maybe the most difficult problems in TTS systems. Homographs are spelled the same way but they differ in meaning and usually in pronunciation (e.g. fair, lives). The word *lives* is for example pronounced differently in sentences "Three *lives* were lost" and "One *lives* to eat".

Some words e.g. *lead*, has different pronunciations when used as a verb or noun, and between two noun senses (He followed her *lead* / He covered the hull with *lead*). With these kinds of words some semantical information is necessary to achieve correct pronunciation.

The pronunciation of a certain word may also be different due to contextual effects. This is easy to see when comparing phrases *the end* and *the beginning*. The pronunciation of *the* depends on the initial phoneme in the following word. Compound words are also problematic. For example the characters 'th' in mother and hothouse is pronounced differently. Some sounds may also be either voiced or unvoiced in different context. For example, phoneme /s/ in word *dogs* is voiced, but unvoiced in word *cats* [17].

Finding correct pronunciation for proper names, especially when they are borrowed from other languages, is usually one of the most difficult tasks for any TTS system. Some common names, such as Nice and Begin, are ambiguous in capitalized context, including sentence initial position, titles and single text. For example, the sentence *Nice is a nice place* is very problematic because the word *Nice* may be pronounced as /niis/ or /nais/. Some names and places have also special pronunciation, such as Leicester and Arkansas. For correct pronunciation, these kinds of words may be included in a specific exception dictionary. Unfortunately, it is clear that there is no way to build a database of all proper names in the world.

1.3.1.3 Prosody

Finding correct intonation, stress, and duration from written text is probably the most challenging problem for years to come. These features together are called prosodic or suprasegmental features and may be considered as the melody, rhythm, and emphasis of the speech at the perceptual level.

The intonation means how the pitch pattern or fundamental frequency changes during speech. The prosody of continuous speech depends on many separate aspects, such as the meaning of the sentence and the speaker characteristics and emotions. The prosodic dependencies are shown in Figure 1.8. Unfortunately, written text usually contains very little information of these features and some of them change dynamically during speech. However, with some specific control characters this information may be given to a speech synthesizer.

Timing at sentence level or grouping of words into phrases correctly is difficult because prosodic phrasing is not always marked in text by punctuation, and phrasal accentuation is almost never marked .If there is no breath pauses in speech or if they are in wrong places, the speech may sound very unnatural or even the meaning of the sentence may be misunderstood.

For example, the input string "John says Peter is a liar" can be spoken as two different ways giving two different meanings as "John says: Peter is a liar" or "John, says Peter, is a liar". In the first sentence Peter is a liar, and in the second one the liar is John.

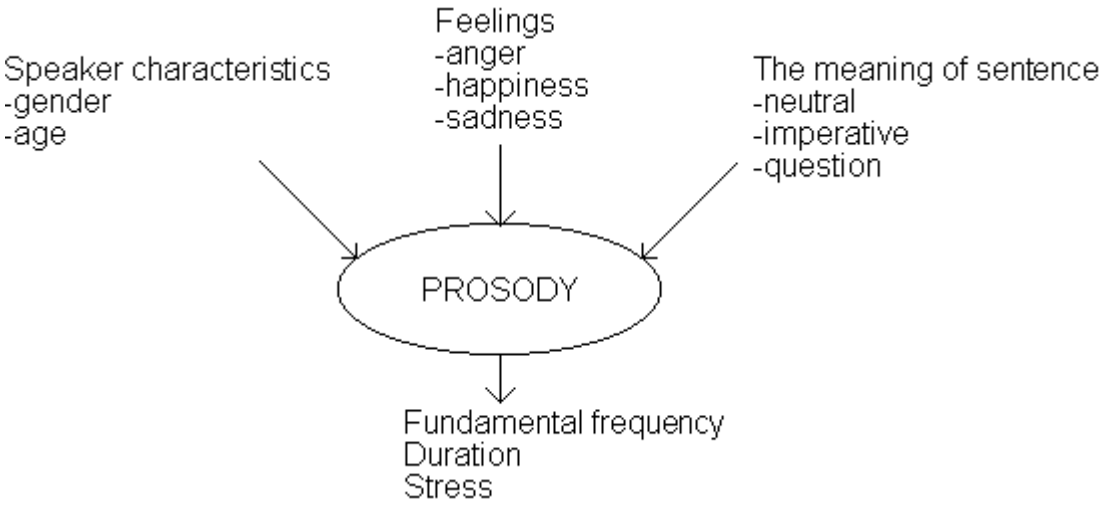


Fig. 1.8. Prosodic dependencies.

Problem Definition

Speech synthesis is the artificial production of human speech through the use computer. A very large set of different rules and their exceptions is needed to produce correct pronunciation and prosody for synthesized speech. The main objective of this report is to:

i) Study the speech synthesis technology,

ii) Develop text to speech module using LabVIEW software

The developed system must be user friendly, cost effective and gives the result in the real time. Moreover, the program must have the required flexibility to be modified easily if the need arises.

CHAPTER 2

Literature Survey

2.1 TRAINABLE TEXT-TO-SPEECH (TTS) SYSTEM

A trainable Text-to-Speech (TTS) system is one which automatically learns the model parameters from a corpus. Both prosody parameters and concatenative speech units are derived through the use of probabilistic learning methods that have been successfully used for speech recognition.

Traditionally, Text-to-Speech (TTS) systems convert input text into voice by using a set of manually derived rules for prosody generation and/or voice synthesis .While these systems can achieve a high level of intelligibility, they typically sound unnatural. The process of deriving these rules is not only labor intensive but also difficult to generalize to a new language, a new voice, or a new speech style.

TTS can "read" text from a document, Web page or e-Book, generating synthesized speech .TTS programs can be useful for a variety of applications. For example, proofreading with TTS allows the author to catch awkward phrases, missing words or pacing problems.

TTS can also convert text files into audio MP3 files that can then be transferred to a portable MP3 player or CD-ROM. This can save time by allowing the user to listen to reports or background materials in bed, *en route* to a meeting, or while performing other tasks.

Even top screenwriting software includes TTS functionality so that a writer can assign different voices to characters in his or her script. The writer can then *listen* to the dialog to weed out stilted sentences. In the area of education, TTS programs provide a valuable edge, particularly for learning new languages. Speech engines are available in a variety of languages, including English, Spanish, German, French, and dozens more [18].

TTS makes a critical difference to those with disabilities such as poor vision or visual dyslexia. People with speech loss can utilize specialized TTS programs to turn typed words into vocalization. Third party TTS programs can allow for great flexibility and, in many cases, much-improved voice quality. Most of these affordable programs come with a variety of voices to choose from, both male and female. Some offer voices with accents. For specialized fields of vocabulary such as medical terminology, a medical TTS program is required. TTS vocalization has come a long way and will continue to improve. The art of designing software that can provide context-dependent pronunciation and inflection is a highly complicated, code-intensive task.

The text-to-speech (TTS) synthesis procedure consists of two main phases. The first one is text analysis, where the input text is transcribed into a phonetic or some other linguistic representation, and the second one is the generation of speech waveforms, where the acoustic output is produced from this phonetic and prosodic information. These two phases are usually called as high- and low-level synthesis.

A simplified version of the procedure is presented in Figure 2.1.The 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 from high-level one.

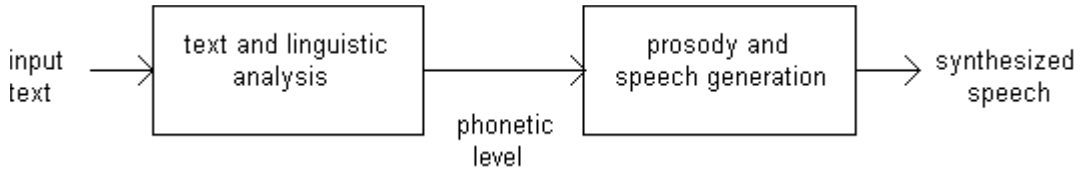


Fig. 2.1 Simple text-to-speech synthesis procedure.

The simplest way to produce synthetic speech is to play long prerecorded samples of natural speech, such as single words or sentences. This concatenation method provides high quality and naturalness, but has a limited vocabulary and usually only one voice. The method is very suitable for some announcing and information systems.

However, it is quite clear that we can not create a database of all words and common names in the world. It is maybe even inappropriate to call this speech synthesis because it contains only recordings. Thus , for unrestricted speech synthesis (text-to-speech) we have to use shorter pieces of speech signal, such as syllables, phonemes, diaphones or even shorter segments.

2.2 METHODS, TECHNIQUES, AND ALGORITHMS

Synthesized speech can be produced by several different methods. All of these have some benefits and deficiencies that are discussed in this and previous chapters. The methods are usually classified into three groups:

- Articulatory synthesis, which attempts to model the human speech production system directly.
- Formant synthesis, which models the pole frequencies of speech signal or transfer function of vocal tract based on source-filter-model.
- Concatenative synthesis, which uses different length prerecorded samples derived from natural speech.

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

2.2.1 ARTICULATORY SYNTHESIS

Articulatory synthesis tries to model the human vocal organs as perfectly as possible, so it is potentially the most satisfying method to produce high-quality synthetic speech.

On the other hand, it is also one of the most difficult methods to implement and the computational load is also considerably higher than with other common methods [19].

Thus, it has received less attention than other synthesis methods and has not yet achieved the same level of success.

Articulatory synthesis typically involves models of the human articulators and vocal cords. The articulators are usually modeled with a set of area functions between glottis and mouth. The first articulatory model was based on a table of vocal tract area functions from larynx to lips for each phonetic segment.

For rule-based synthesis the articulatory control parameters may be for example lip aperture, lip protrusion, tongue tip height, tongue tip position, tongue height, tongue position and velic aperture. Phonatory or excitation parameters may be glottal aperture, cord tension, and lung pressure

When speaking, the vocal tract muscles cause articulators to move and change shape of the vocal tract which causes different sounds. The data for articulatory model is usually derived from X-ray analysis of natural speech. However, this data is usually only 2-D when the real vocal tract is naturally 3-D, so the rule-based articulatory synthesis is very difficult to optimize due to the unavailability of sufficient data of the motions of the articulators during speech. Other deficiency with articulatory synthesis is that X-ray data do not characterize the masses or degrees of freedom of the articulators [15]. Also, the movements of tongue are so complicated that it is almost impossible to model them precisely.

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 [7]. 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, it might be a potential synthesis method in the future.

2.2.2 FORMANT SYNTHESIS

Probably the most widely used synthesis method during last decades has been formant synthesis which is based on the source-filter-model of speech.

There are two basic structures in general, parallel and cascade, but for better performance some kind of combination of these is usually used. Formant synthesis also provides infinite number of sounds which makes it more flexible than for example concatenation methods.

At least three formants are generally required to produce intelligible speech and up to five formants to produce high quality speech.

Rule-based formant synthesis is based on a set of rules used to determine the parameters necessary to synthesize a desired utterance using a formant synthesizer [17]. The input parameters may be for example the following, where the open quotient means the ratio of the open-glottis time to the total period duration [20].

- Voicing fundamental frequency (F0)
- Voiced excitation open quotient (OQ)
- Degree of voicing in excitation (VO)
- Formant frequencies and amplitudes (F1...F3 and A1...A3)
- Frequency of an additional low-frequency resonator (FN)
- Intensity of low- and high-frequency region (ALF, AHF)

A cascade formant synthesizer (Figure 2.2) consists of band-pass resonators connected in series and the output of each formant resonator is applied to the input of the following one. The cascade structure needs only formant frequencies as control information. The main advantage of the cascade structure is that the relative formant amplitudes for vowels do not need individual controls [17].

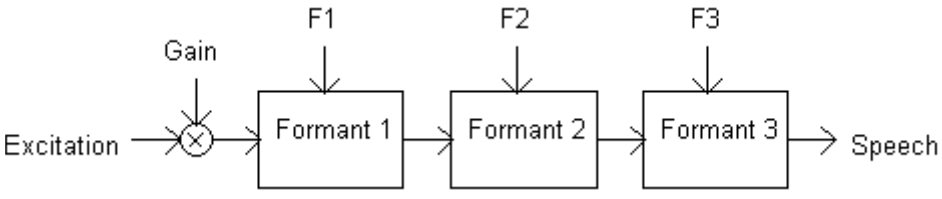


Fig. 2.2. Basic structure of cascade formant synthesizer.

The cascade structure has been found better for non-nasal voiced sounds and because it needs less control information than parallel structure, it is then simpler to implement. However, with cascade model the generation of fricatives and plosive bursts is a problem.

A parallel formant synthesizer (Figure 2.3) consists of resonators connected in parallel. Sometimes extra resonators for nasals are used. The excitation signal is applied to all formants simultaneously and their outputs are summed.

Adjacent outputs of formant resonators must be summed in opposite phase to avoid unwanted zeros or anti resonances in the frequency response.

The parallel structure enables controlling of bandwidth and gains for each formant individually and thus needs also more control information.

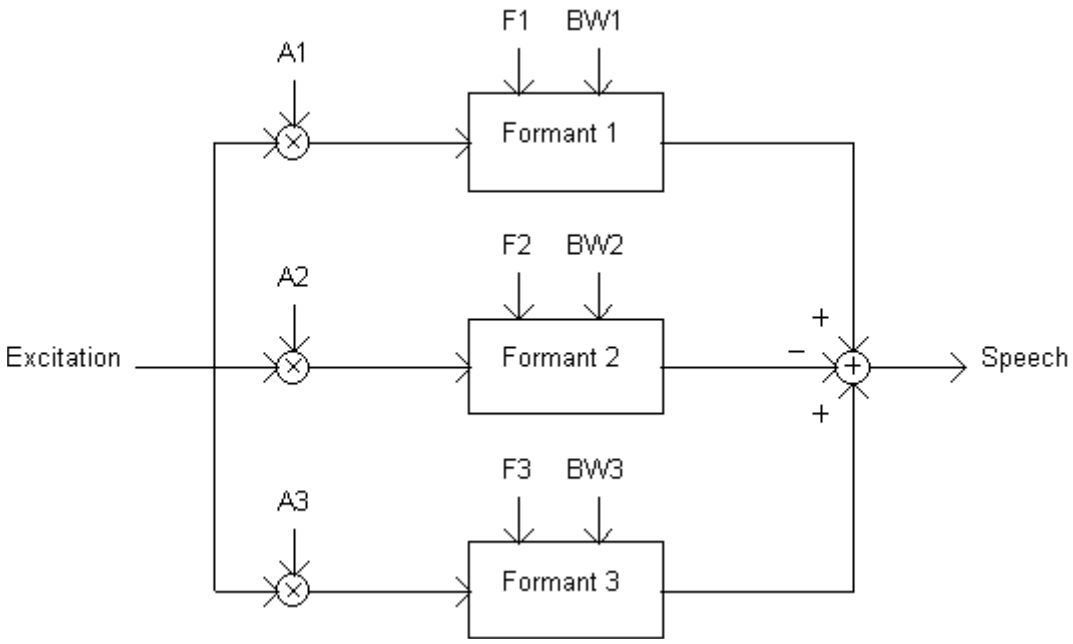


Fig. 2.3. Basic structure of a parallel formant synthesizer.

The parallel structure has been found to be better for nasals, fricatives, and stop-consonants, but some vowels can not be modeled with parallel formant synthesizer as well as with the cascade one.

There has been widespread controversy over the quality and suitably characteristics of these two structures.

It is easy to see that good results with only one basic method is difficult to achieve so some efforts have been made to improve and combine these basic models. The correct and carefully selected excitation is important especially when good controlling of speech characteristics is wanted.

The formant filters represent only the resonances of the vocal tract, so additional provision is needed for the effects of the shape of the glottal waveform and the radiation characteristics of the mouth. Usually the glottal waveform is approximated simply with -12dB/octave filter and radiation characteristics with simple +6dB/octave filter.

2.2.3 CONCATENATIVE SYNTHESIS

Connecting prerecorded natural utterances is probably the easiest way to produce intelligible and natural sounding synthetic speech. However, concatenative synthesizers are usually limited to one speaker and one voice and usually require more memory capacity than other methods.

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

Word is perhaps the most natural unit for written text and some messaging systems with very limited vocabulary. Concatenation of words is relative easy to perform and co articulation effects within a word are captured in the stored units. However, there is a great difference with words spoken in isolation and in continuous sentence which makes the continuous speech to sound very unnatural [21].

Because there are hundreds of thousands of different words and proper names in each language, word is not a suitable unit for any kind of unrestricted TTS system.

The number of different syllables in each language is considerably smaller than the number of words, but the size of unit database is usually still too large for TTS systems. For example, there are about 10,000 syllables in English.

Unlike with words, the coarticulation effect is not included in stored units, so using syllables as a basic unit is not very reasonable. There is also no way to control prosodic contours over the sentence. At the moment, no word or syllable based full TTS system exists. The current synthesis systems are mostly based on using phonemes, diaphones, demisyllables or some kind of combinations of these.

Demisyllables represents the initial and final parts of syllables. One advantage of demisyllables is that only about 1,000 of them is needed to construct the 10,000 syllables of English .Using demisyllables, instead of for example phonemes and diphones, requires considerably less concatenation points. Demisyllables also take account of most transitions and then also a large number of coarticulation effects and also covers a large number of allophonic variations due to separation of initial and final consonant clusters. However, the memory requirements are still quite high, but tolerable. Compared to phonemes and diphones, the exact number of demisyllables in a language can not be defined. With purely demisyllable based system, all possible words can not be synthesized properly. This problem is faced at least with some proper names.

Phonemes are probably the most commonly used units in speech synthesis because they are the normal linguistic presentation of speech. The inventory of basic units is usually between 40 and 50, which is clearly the smallest compared to other units .Using phonemes gives maximum flexibility with the rule-based systems. However, some phones that do not have a steady-state target position, such as plosives, are difficult to synthesize. The articulation must also be formulated as rules. Phonemes are sometimes used as an input for speech synthesizer to drive for example diaphone-based synthesizer.

Diaphones (or dyads) are defined to extend the central point of the steady state part of the phone to the central point of the following one, so they contain the transitions between adjacent phones. That means that the concatenation point will be in the most steady state region of the signal, which reduces the distortion from concatenation points.

Another advantage with diaphones is that the coarticulation effect needs no more to be formulated as rules. In principle, the number of diaphones is the square of the number of phonemes (plus allophones), but not all combinations of phonemes are needed. However, the number of data is still tolerable and with other advantages, diaphone is a very suitable unit for sample-based text-to-speech synthesis.

Longer segmental units, such as triphones or tetraphones, are quite rarely used. Triphones are like diaphones, but contains one phoneme between steady-state points (half phoneme - phoneme - half phoneme). In other words, a triphone is a phoneme with a specific left and right context. For English, more than 10,000 units are required [22].

Building the unit inventory consists of three main phases [23]. First, the natural speech must be recorded so that all used units (phonemes) within all possible contexts (allophones) are included. After this, the units must be labeled or segmented from spoken speech data, and finally, the most appropriate units must be chosen. Gathering the samples from natural speech is usually very time-consuming. However, some is this work may be done automatically by choosing the input text for analysis phase properly. The implementation of rules to select correct samples for concatenation must also be done very carefully.

There are several problems in concatenative synthesis compared to other methods.

- Distortion from discontinuities in concatenation points, which can be reduced using diaphones or some special methods for smoothing signal.
- 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. In theory, all possible allophones should be included in the material, but trade-offs between the quality and the number of samples must be made.

Some of the problems may be solved with methods described below and the use of concatenative method is increasing due to better computer capabilities.

2.2.4 SINUSOIDAL MODELS

Sinusoidal models are based n a well known assumption that the speech signal can be represented as a sum of sine waves with time-varying amplitudes and frequencies [24]. In the basic model, the speech signal *s(n)* is modeled as the sum of a small number *L* of sinusoids

$$s(n) = \sum_{l=1}^{L} A_l \cos(\omega_l n + \phi_l)$$

where $A_l(n)$ and $\phi_l(n)$ represent the amplitude and phase of each sinusoidal component associated with the frequency track ω_l. To find these parameters $A_l(n)$ and $\phi_l(n)$, the DFT of windowed signal frames is calculated, and the peaks of the spectral magnitude are selected from each frame (see Figure 2.4). The basic model is also known as the McAulay/Quatieri Model. The basic model has also some modifications such as ABS/OLA (Analysis by Synthesis / Overlap Add) and Hybrid / Sinusoidal Noise models.

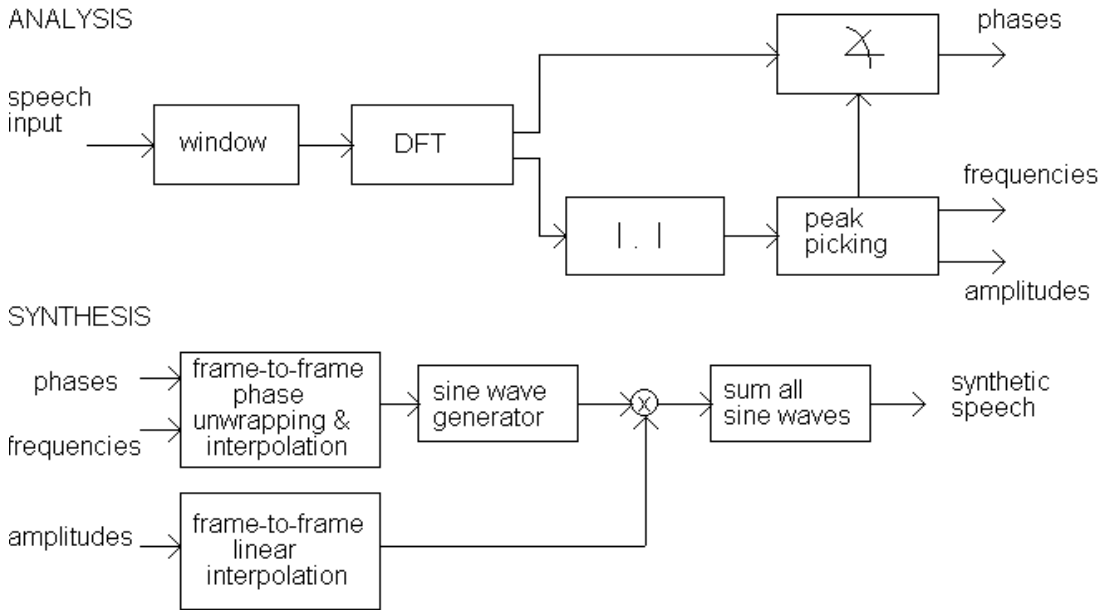


Fig. 2.4. Sinusoidal analysis / synthesis system (Macon 1996).

While the sinusoidal models are perhaps very suitable for representing periodic signals, such as vowels and voiced consonants, the representation of unvoiced speech becomes problematic.

Sinusoidal models are also used successfully in singing voice synthesis [25]. The synthesis of singing differs from speech synthesis in many ways. In singing, the intelligibility of the phonemic message is often secondary to the intonation and musical qualities. Vowels are usually sustained longer in singing than in normal speech, and naturally, easy and independent controlling of pitch and loudness is also required.

2.2.5 HIGH-LEVEL SYNTHESIS

With high-level synthesis the input text or information is transcribed in such format that low-level voice synthesizer is capable to produce the acoustic output. A proper implementation of this is the fundamental challenge in all present systems and will probably be for years to come.

The procedure consists of three main phases:

- Text preprocessing where numerals, special characters, abbreviations, and acronyms are expanded into full words.
- Pronunciation analysis where the pronunciation of certain words, including homographs and proper names, are determined.
- Prosodic analysis where the prosodic features of speech are determined.

After high-level synthesizer, the information is delivered to drive some low-level system. The type of used data depends on the driven system.

For example, for formant synthesizer, at least fundamental frequency, formant frequencies, duration, and amplitude of each sound segment are needed.

2.2.5.1 Text Preprocessing

The first task of all TTS systems is to convert input data to proper form for a synthesizer. In this stage, all non-alphabetical characters, numbers, abbreviations, and acronyms must be converted into a full spelled-out format. Text preprocessing is usually made with simple one-to-one lookup tables, but in some cases additional information of neighboring words or characters is needed. This may lead to a large database and complicated set of rules and may cause some problems with real-time systems. Input text may also contain some control characters which must be delivered through the text parser without modifications. The conversion must neither affect abbreviations which are a part of another. For example, if the character *M* is in some context converted as *mega*, the abbreviation *MTV* should not be converted as *megaTV*. However, character strings or abbreviations which are not in a lookup table and consist only of consonants can be always converted letter-by-letter because those kinds of words do no exist in any language.

Numbers are perhaps the most difficult to convert correctly into spelled-out format. Numbers are used in several relations, such as digits, dates, roman numerals, measures, and mathematical expressions.

Numbers between 1100 and 1999 are usually converted as years like 1910 as *nineteen-ten*. Expressions in form 12/12/99 or 11/11/1999 may be converted as dates, if the numbers are within acceptable values. However, the expression 2/5 is more difficult because it may be either *two divided by five* or *the second of may*.

In some cases, the correct conversion is possible to conclude from compounding information (measures etc.) or from the length of the number (dates, phone numbers etc.). However, there will be always some ambiguous situations.

In some cases with measures, usually currencies, the order of some character and value is changed. For example $3.02 is converted as *three dollars and two cents*. In these situations, the numerical expressions which are already in spelled-out format must be recognized to avoid the misconversion like *$100 million* to *one hundred dollars million*.

Some abbreviations and acronyms are ambiguous in different context like described in Chapter 3. For common abbreviation like st., the first thing to do is to check if it is followed by a capitalized word (potential name), when it will be expanded as *saint*. Otherwise, if it is preceded a capitalized word, an alphanumeric (5th), or a number, it will be expanded as *street* [3].

Sometimes different special modes, especially with numbers, are used to make this stage more accurate, for example, math mode for mathematical expressions and date mode for dates and so on. Another situation where the specific rules are needed is for example the E-mail messages where the header information needs special attention.

2.2.5.2 Pronunciation

Analysis for correct pronunciation from written text has also been one of the most challenging tasks in speech synthesis field. Especially, with some telephony applications where almost all words are common names or street addresses. One method is to store as much names as possible into a specific pronunciation table.

Due to the amount of existing names, this is quite unreasonable. So rule-based system with an exception dictionary for words that fail with those letter-to-phoneme rules may be a much more reasonable approach [12]. This approach is also suitable for normal pronunciation analysis. With morphemic analysis, a certain word can be divided in several independent parts which are considered as the minimal meaningful subpart of words as prefix, root, and affix. About 12 000 morphemes are needed for covering 95 percent of English [17]. However, the morphemic analysis may fail with word pairs, such as heal/health or sign/signal.

Another perhaps relatively good approach to the pronunciation problem is a method called *pronunciation by analogy* where a novel word is recognized as parts of the known words and the part pronunciations are built up to produce the pronunciation of a new word, for example pronunciation of word *grip* may be constructed from *grin* and *rip* [26].

2.2.5.3 Prosody

Prosodic or suprasegmental features consist of pitch, duration, and stress over the time. With good controlling of these genders, age, emotions, and other features in speech can be well modeled. However, almost everything seems to have effect on prosodic features of natural speech which makes accurate modeling very difficult. 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 (intonation) in natural speech is a combination of many factors. 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 when the sentence is in a question form, the pitch pattern will raise to the end of sentence. In the end of sentence there may also be a continuation rise which indicates that there is more speech to come. A raise or fall in fundamental frequency can also indicate a stressed syllable .Finally, the pitch contour is also affected by gender, physical and emotional state, and attitude of the speaker.

The duration or time characteristics can also be investigated at several levels from phoneme (segmental) durations to sentence level timing, speaking rate, and rhythm. The segmental duration is determined by a set of rules to determine correct timing. Usually some inherent duration for phoneme is modified by rules between maximum and minimum durations. For example, consonants in non-word-initial position are shortened, emphasized words are significantly lengthened, or a stressed vowel or sonorant preceded by a voiceless plosive is lengthened [17].In general, the phoneme duration differs due to neighboring phonemes. At sentence level, the speech rate, rhythm, and correct placing of pauses for correct phrase boundaries are important. For example, a missing phrase boundary just makes speech sound rushed which is not as bad as an extra boundary which can be confusing.

The intensity pattern is perceived as a loudness of speech over the time. At syllable level vowels are usually more intense than consonants and at a phrase level syllables at the end of an utterance can become weaker in intensity.

The intensity pattern in speech is highly related with fundamental frequency. The intensity of a voiced sound goes up in proportion to fundamental frequency [15].

The speaker's feelings and emotional state affect speech in many ways and the proper implementation of these features in synthesized speech may increase the quality considerably. With text-to-speech systems this is rather difficult because written text usually contains no information of these features. However, this kind of information may be provided to a synthesizer with some specific control characters or character strings. The users of speech synthesizers may also need to express their feelings in "real-time". For example, deafened people can not express their feelings when communicating with speech synthesizer through a telephone line.

This section shortly introduces how some basic emotional states affect voice characteristics.

The voice parameters affected by emotions are usually categorized in three main types [27]:

- *Voice quality* which contains largely constant voice characteristics over the spoken utterance, such as loudness and breathiness. For example, angry voice is breathy, loud, and has a tense articulation with abrupt changes while 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 the general rhythm, speech rate, the lengthening and shortening of the stressed syllables, the length of content words, and the duration and placing of pauses.

The number of possible emotions is very large, but there are five discrete emotional states which are commonly referred as the primary or basic emotions and the others are altered or mixed forms of these. These are anger, happiness, sadness, fear, and disgust. The secondary emotional states are for example whispering, shouting, grief, and tiredness.

Anger in speech causes increased intensity with dynamic changes [28]. The voice is very breathy and has tense articulation with abrupt changes. The average pitch pattern is higher and there is a strong downward inflection at the end of the sentence. The pitch range and its variations are also wider than in normal speech and the average speech rate is also a little bit faster.

Happiness or *joy* causes slightly increased intensity and articulation for content words. The voice is breathy and light without tension. Happiness also leads to increase in pitch and pitch range. The peak values of pitch and the speech rate are the highest of basic emotions [29].

Fear or *anxiety* makes the intensity of speech lower with no dynamic changes. Articulation is precise and the voice is irregular and energy at lower frequencies is reduced. The average pitch and pitch range are slightly higher than in neutral speech. The speech rate is slightly faster than in normal speech and contains pauses between words forming almost one third of the total speaking time [28].

Sadness or *sorrowness* in speech decreases the speech intensity and its dynamic changes. The average pitch is at the same level as in neutral speech, but there are almost no dynamic changes. The articulation precision and the speech rate are also decreased. High ratio of pauses to phonation time also occurs [4]. Grief is an extreme form of sadness where the average pitch is lowered and the pitch range is very narrow. Speech rate is very slow and pauses form almost a half of the total speaking time [30].

Disgust or *contempt* in speech also decreases the speech intensity and its dynamic range. The average pitch level and the speech rate are also lower compared to normal speech and the number of pauses is high. Articulation precision and phonation time are increased and the stressed syllables in stressed content words are lengthened [27].

Whispering and *shouting* are also common versions of expression. Whispering is produced by speaking with high breathiness without fundamental frequency, but the emotions can still be conveyed [30]. Shouted speech causes an increased pitch range, intensity and greater variability in it. Tiredness causes a loss of elasticity of articulatory muscles leading to lower voice and narrow pitch range.

2.3 OTHER METHODS AND TECHNIQUES

Several other methods and experiments to improve the quality of synthetic speech have been made. Variations and combinations of previously described methods have been studied widely, but there is still no single method to be considered distinctly the best.

Synthesized speech can also be manipulated afterwards with normal speech processing algorithms. For example, adding some echo may produce more pleasant speech.

However, this approach may easily increase the computational load of the system. Some experiments to show the use of a combination of the basic synthesis methods have been made, because different methods show different success in generating individual phonemes. Time domain synthesis can produce high-quality and natural sounding speech segments, but in some segment combinations the synthesized speech is discontinuous at the segment boundaries and if a wide-range variation of fundamental frequency is required, overall complexity will increase. On the other hand, formant synthesis yields more homogeneous speech allowing a good control of fundamental frequency, but the voice-timbre sounds more synthetic. This approach leads to the hybrid system which combines the time- and frequency-domain methods. The basic idea of a hybrid system is shown in Figure 2.5 [31].

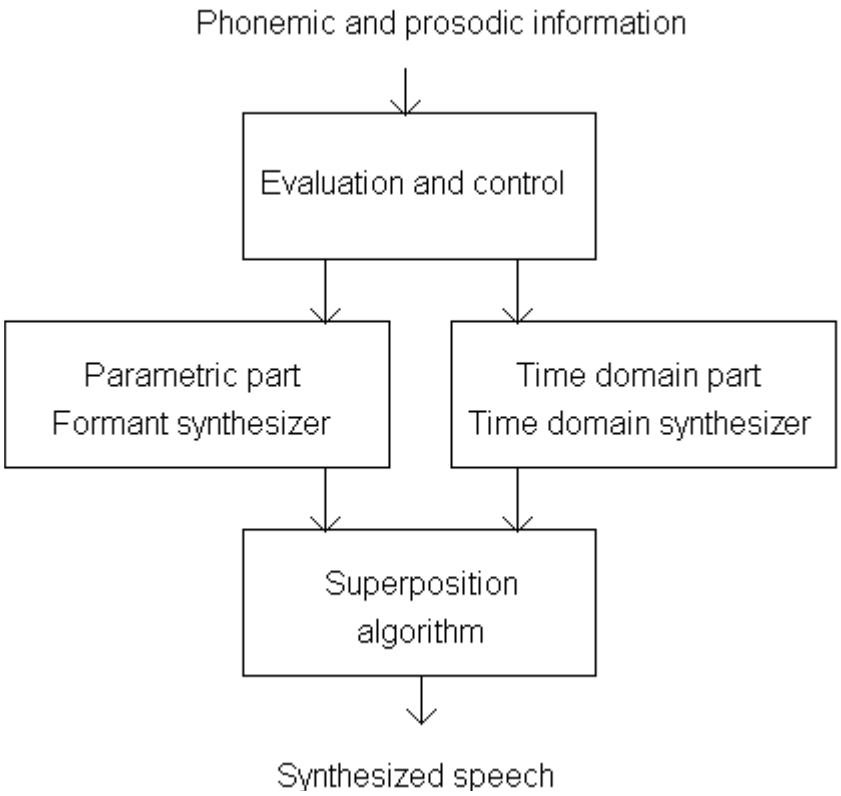


Fig. 2.5. Basic idea of the hybrid synthesis system

Another common method in speech synthesis, and especially in speech recognition and analysis of prosodic parameters from speech, is for example the use of hidden Markov models (HMMs).

The method is based on a statistical approach to simulate real life stochastic processes [5].A hidden Markov model is a collection of states connected by transitions. Each transition carries two sets of probabilities: a transition probability, which provides the probability for taking the transition, and an output probability density function, which defines the conditional probability of emitting each output symbol from a finite alphabet, given that that the transition is taken [32].

2.4 APPLICATIONS OF SYNTHETIC SPEECH

Synthetic speech may be used in several applications. Communication aids have developed from low quality talking calculators to modern 3D applications, such as talking heads. The implementation method depends mostly on used application. In some cases, such as announcement or warning systems, unrestricted vocabulary is not necessary and the best result is usually achieved with some simple messaging system. With suitable implementation some funds may also be saved. On the other hand, some applications, such as reading machines for the blind or electronic-mail readers, require unlimited vocabulary and a TTS system is needed.

The application field of synthetic speech is expanding fast whilst the quality of TTS systems is also increasing steadily. Speech synthesis systems are also becoming more affordable for common customers, which makes these systems more suitable for everyday use. For example, better availability of TTS systems may increase employing possibilities for people with communication difficulties.

2.4.1 APPLICATIONS FOR THE BLIND

Probably the most important and useful application field in speech synthesis is the reading and communication aids for the blind. Before synthesized speech, specific audio books were used where the content of the book was read into audio tape.

It is clear that making such spoken copy of any large book takes several months and is very expensive.

It is also easier to get information from computer with speech instead of using special bliss symbol keyboard, which is an interface for reading the Braille characters.

The first commercial TTS application was probably the Kurzweil reading machine for the blind introduced by Raymond Kurzweil in the late 1970's. It consisted of an optical scanner and text recognition software and was capable to produce quite intelligible speech from written multi font text [15]. The prices of the first reading machines were far too high for average user and these machines were used mostly in libraries or related places.

Today, the quality of reading machines has reached acceptable level and prices have become affordable for single individual, so a speech synthesizer will be very helpful and common device among visually impaired people in the future. Current systems are mostly software based, so with scanner and OCR (optical character recognition) system, it is easy to construct a reading machine for any computer environment with tolerable expenses. Regardless of how fast the development of reading and communication aids is, there are always some improvements to do.

The most crucial factor with reading machines is speech intelligibility which should be maintained with speaking rates ranging from less than half to at least three times normal rate [33]. Naturalness is also an important feature and makes the synthetic speech more acceptable. Although the naturalness is one of the most important features, it may sometimes be desirable that the listener is able to identify that speech is coming from machine, so the synthetic speech should sound natural but somehow "neutral".

When the output from a speech synthesizer is listened for the first time, it may sound intelligible and pleasant. However, during longer listening period, single clicks or other weak points in the system may arise very annoying. This is called an annoying effect and it is difficult to perceive with any short-term evaluation method, so for these kinds of cases, the feedback from long-term users is sometimes very essential.

Speech synthesis is currently used to read www-pages or other forms of media with normal personal computer.

Information services may also be implemented through a normal telephone interface with keypad-control similar to text-tv. With modern computers it is also possible to add new features into reading aids.

It is possible to implement software to read standard check forms or find the information how the newspaper article is constructed. However, sometimes it may be impossible to find correct construction of the newspaper article if it is for example divided in several pages or has an anomalous structure.

A blind person can not also see the length of an input text when starting to listen it with a speech synthesizer, so an important feature is to give in advance some information of the text to be read. For example, the synthesizer may check the document and calculate the estimated duration of reading and speak it to the listener. Also the information of bold or underlined text may be given by for example with slight change of intonation or loudness.

2.4.2 APPLICATIONS FOR THE DEAFENED AND VOCALLY HANDICAPPED

People who are born-deaf can not learn to speak properly and people with hearing difficulties have usually speaking difficulties. Synthesized speech gives the deafened and vocally handicapped an opportunity to communicate with people who do not understand the sign language. With a talking head it is possible to improve the quality of the communication situation even more because the visual information is the most important with the deaf and dumb. A speech synthesis system may also be used with communication over the telephone line.

Adjustable voice characteristics are very important in order to achieve individual sounding voice. Users of talking aids may also be very frustrated by an inability to convey emotions, such as happiness, sadness, urgency, or friendliness by voice. Some tools, such as HAMLET (Helpful Automatic Machine for Language and Emotional Talk) have been developed to help users to express their feelings.

The HAMLET system is designed to operate on a PC with high quality speech synthesizer, such as DECtalk.With keyboard it is usually much slower to communicate than with normal speech.

One way to speed up this is to use the predictive input system that always displays the most frequent word for any typed word fragment, and the user can then hit a special key to accept the prediction. Even individual pre-composed phrases, such as greetings or salutes, may be used.

2.4.3 EDUCATIONAL APPLICATIONS

Synthesized speech can be used also in many educational situations. A computer with speech synthesizer can teach 24 hours a day and 365 days a year. It can be programmed for special tasks like spelling and pronunciation teaching for different languages. It can also be used with interactive educational applications.

Especially with people who are impaired to read (dyslexics), speech synthesis may be very helpful because especially some children may feel themselves very embarrassing when they have to be helped by a teacher [15]. It is also almost impossible to learn write and read without spoken help. With proper computer software, unsupervised training for these problems is easy and inexpensive to arrange.

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

2.4.4 APPLICATIONS FOR TELECOMMUNICATIONS AND MULTIMEDIA

The newest applications in speech synthesis are in the area of multimedia. Synthesized speech has been used for decades in all kind of telephone enquiry systems, but the quality has been far from good for common customers. Today, the quality has reached the level that normal customers are adopting it for everyday use.

Electronic mail has become very usual in last few years. However, it is sometimes impossible to read those E-mail messages when being for example abroad. There may be no proper computer available or some security problems exist.

With synthetic speech e-mail messages may be listened to via normal telephone line. Synthesized speech may also be used to speak out short text messages (sms) in mobile phones.

For totally interactive multimedia applications an automatic speech recognition system is also needed. The automatic recognition of fluent speech is still far away, but the quality of current systems is at least so good that it can be used to give some control commands, such as yes/no, on/off, or ok/cancel.

2.4.5 OTHER APPLICATIONS AND FUTURE DIRECTIONS

In principle, speech synthesis may be used in all kind 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. Using speech instead of warning lights or buzzers gives an opportunity to reach the warning signal for example from a different room. Speech synthesizer may also be used to receive some desktop messages from a computer, such as printer activity or received e-mail.

In the future, if speech recognition techniques reach adequate level, synthesized speech may also be used in language interpreters or several other communication systems, such as videophones, videoconferencing, or talking mobile phones. If it is possible to recognize speech, transcribe it into ASCII string, and then re synthesize it back to speech, a large amount of transmission capacity may be saved. With talking mobile phones it is possible to increase the usability considerably for example with visually impaired users or in situations where it is difficult or even dangerous to try to reach the visual information. It is obvious that it is less dangerous to listen than to read the output from mobile phone for example when driving a car.

During last few decades the communication aids have been developed from talking calculators to modern three-dimensional audiovisual applications [34]. The application field for speech synthesis is becoming wider all the time which brings also more funds into research and development areas.

2.5 APPLICATION FRAMEWORKS

Several methods and interfaces for making the implementation of synthesized speech in desired applications easier have been developed during this decade. It is quite clear that it is impossible to create a standard for speech synthesis methods because most systems act as stand alone device which means they are incompatible with each other and do not share common parts. However, it is possible to standardize the interface of data flow between the application and the synthesizer.

Usually, the interface contains a set of control characters or variables for controlling the synthesizer output and features. The output is usually controlled by normal play, stop, pause, and resume type commands and the controllable features are usually pitch baseline and range, speech rate, volume, and in some cases even different voices, ages, and genders are available. In most frameworks it is also possible to control other external applications, such as a talking head or video.

In this chapter, three approaches to standardize the communication between a speech synthesizer and applications are introduced. Most of the present synthesis systems support so called Speech Application Programming Interface (SAPI) which makes easier the implementation of speech in any kind of application. For Internet purposes several kind of speech synthesis markup languages have been developed to make it possible to listen to synthesized speech without having to transfer the actual speech signal through network. Finally, one of the most interesting approaches is probably the TTS subpart of MPEG-4 multimedia standard which will be introduced in the near future.

2.5.1 SPEECH APPLICATION PROGRAMMING INTERFACE

SAPI is an interface between applications and speech technology engines, both text-to-speech and speech recognition [35]. The interface allows multiple applications to share the available speech resources on a computer without having to program the speech engine itself.

Speech synthesis and recognition applications usually require plenty of computational resources and with SAPI approach lots of these resources may be saved.

The user of an application can also choose the synthesizer used as long as it supports SAPI. Currently SAPIs are available for several environments, such as MS-SAPI for Microsoft Windows operating systems and Sun Microsystems Java SAPI (JSAPI) for JAVA based applications. In this chapter, only the speech synthesis part is discussed.

SAPI text-to-speech part consists of three interfaces. The *voice text* interface which provides methods to start, pause, resume, fast forward, rewind, and stop the TTS engine during speech. The *attribute interface* allows access to control the basic behavior of the TTS engine, such as the audio device to be used, the playback speed (in words per minute), and turning the speech on and off.

With some TTS systems the attribute interface may also be used to select the speaking mode from predefined list of voices, such as female, male, child, or alien. Finally, the *dialog interface* can be used to set and retrieve information regarding the TTS engine to for example identify the TTS engine and alter the pronunciation lexicon.

2.5.2 INTERNET SPEECH MARKUP LANGUAGES

Most synthesizers accept only plain text as input. However, it is difficult to analyze the text and find correct pronunciation and prosody from written text. In some cases there is also need to include the speaker features or emotional information in the output speech. With some additional information in input data it is possible to control these features of speech easily.

For example, with some information about if the input sentence is in a question, imperative, or neutral form, the controlling of prosody may become significantly easier. Some commercial systems allow the user to place same kind of annotations in the text to produce more natural sounding speech.

The first attempt to develop a TTS markup language was called SSML (Speech Synthesis Markup Language), developed at the Centre for Speech Technology Research (CSTR) in the University of Edinburgh, England, in 1995.

It included control tags for phase boundaries, language, and made possible to define a pronunciation of a specific word and include emphasis tags in the sentence.

Markup languages provide some advantages compared to for example SAPI which provides tags only for speaker directives, not for any text description. In theory, anything specifiable in the text which can give an instruction or description to a TTS system could be in a synthesis markup language. Unfortunately, several systems are under development and the making of an international standard is a considerable problem.

2.5.3 MPEG-4 TTS

The MPEG-4 Text-to-Speech (M-TTS) is a subpart of the standard which is currently under development in ISO MPEG-4 committee (ISO 1997). It specifies the interface between the bit stream and listener. Naturally, due to various existing speech synthesis techniques the exact synthesis method is not under standardization.

The main objective is to make it possible to include narration in any multimedia content without having to record natural speech. Also controlling of facial animation (FA) and moving picture (MP) is supported. MPEG-4 TTS system is still under development.

2.5.3.1 Applications of Mpeg-4 TTS

M-TTS presents two application scenarios for the M-TTS Decoder, MPEG-4 Story Teller on Demand (STOD) and MPEG-4 Audio Text-to-Speech with Moving Picture.

These scenarios are only informative and they are not under standardization process. Naturally, MPEG-4 TTS may be used in several other audio-visual related applications, such as dubbing-tools for animated pictures or Internet voice.

Story Teller on Demand is an application where user can select huge databases or story libraries stored on hard disk, CD-ROM or other media. The system reads the story via M-TTS decoder with the MPEG-4 facial animation or with appropriately selected images. The user can stop and resume speaking at any moment he wants with for example mouse or keyboard. The gender, age, and the speech rate of the story teller are also easily adjustable.

With the STOD system, the narration with several features can be easily composed without recording the natural speech and so the required disk space is considerably reduced.

Audio Text-to-Speech with Moving picture is an application where the synchronized playback of the M-TTS decoder and encoded moving picture is the main objective.

The decoder can provide several granularities of synchronization for different situations. Aligning only the composition time of each sentence, coarse granularity of synchronization and trick mode functionality can be easily achieved. For finer synchronization granularity the lip shape information may be utilized. The finest granularity can be achieved by using the prosody and video-related information. With this synchronization capability, the M-TTS decoder may be used for moving picture dubbing by utilizing the lip shape pattern information.

In the future M-TTS or other similar approaches may be used in several multimedia and telecommunication applications. However, it may take some time before we have full synthetic newsreaders and narrators. Some of the present synthesizers are using a same kind of controlling approach in their system, but there is still no effort for widespread standard [36].

CHAPTER 3

Labview Graphical Programming Language

Graphical programming is a natural extension of the way scientists and engineers design equipment, experiments, and also communicate each other. Most of the time when an idea is developed, a general diagram is drawn or sketched and discussed. It is a very productive tool for rapidly designing and obtaining results.

With this in mind, *LabVIEW* (an acronym for ***Lab***oratory ***V***irtual ***I***nstrument ***E***ngineering ***W***orkbench) was developed in late 1980's by National Instruments, for use in the desktop/workstation computer environment. *LabVIEW* soon became very popular and widely accepted in scientific and engineering circles. Graphical programming software packages are also available from many other companies. LabVIEW is a versatile, powerful graphical programming language, which greatly simplified and facilitated the production of efficient, graphical user interface (GUI) style programs.

In *LabVIEW* one can create a virtual instrument (VI) by selecting various icons from libraries and connecting them together on the computer screen. Each icon (and theirinterconnections) is translated into code by an internal code generator, such that theprogram can then be run on the computer. *LabVIEW* uses a graphical programming language called *G*, however you do not need to know this language to use *LabVIEW*.

3.1 LABVIEW PROGRAM STRUCTURE

A *LabVIEW* program is similar to a text-based program with functions and subroutines; however, in appearance it functions like a *virtual instrument* (VI). A real instrument may accept an input, process on it and then output a result. Similarly, a *LabVIEW* VI behaves in the same manner.

A *LabVIEW* VI has 3 main parts*:*

a) *Front Panel* window,

b) *Block Diagram* window and

c) *Controls*, *Functions* and *Tools Palette* windows, which contain icons associated with extensive libraries of software functions, subroutines, etc.

The *Controls Palette* is associated with the *Front Panel* window, the *Functions Palette* is associated with the *Block Diagram* window, and the *Tools Palette* is associated with both *Front Panel* and *Block Diagram* windows.

The appearance of the *Front Panel* window is similar to that of the front panel of an actual instrument. It may contain icons (libraries of which are contained in the *Controls Palette*) associated with graphical elements such as knobs, switches, pushbuttons, x-y plots/charts, etc. Data is input through the keyboard or mouse to interact with those virtual components (for example a switch could be turned on). The result is shown through various output components of the VI (for example on a graph). The F*ront Panel* window is actually the main user interface of the VI.

The *Block Diagram* window is the VI's equivalent of source code. It itself is an executable program. Each component of a *LabVIEW Block Diagram* window is represented by an icon (libraries of which are contained in the *Functions Palette*). Here, an icon used in the *Block Diagram* window may be a function, a subroutine (i.e. a lower level VI), or a control structure, etc. By connecting together the icons in the *Block Diagram* window with wires, one can control the flow/sequencing of the program.

Thus, by choosing specific icons associated with various *Controls* and *Functions* in conjunction with the use of various *Tools*, each segment of the *LabVIEW* VI can be built up in succession, very easily and rapidly, once some familiarity and experience with using *LabVIEW* has been gained by the program developer.

In addition, a *LabVIEW* VI is also inherently modular in nature. A series of sub-tasks can be acted upon to run the desired program. Each VI may be designated as a sub-VI of a larger program and so on to accomplish multitudes of tasks.

In a text-based computer program, we routinely use subroutines (written e.g. in FORTRAN, BASIC), procedures (written in Pascal) or functions (written in C and/or C++).

Their purpose is to make the overall program clear and modular. Similarly, we can add modularity to our VI in *LabVIEW*.

Many VI's can be formed into a group to create another VI. These VI's can also be used later on in other programs, thereby reducing overall program development time.

LabVIEW is a graphical program development application developed by National Instruments in 1986 to integrate engineering tasks like;

a. Interfacing computers with the instruments,
b. Collecting, storing, analyzing, transmitting measured data,
c. Developing program in a graphical environment,
d. Providing an effective user interface.

LabVIEW delivers real solutions to the practical problems faster than any other graphical environment.

More than software package for controlling an instrument, LabVIEW can be used to integrate GPIB and VXI for data acquisition, automation, motion control, vision almost everything to build a system.

The programs written in LabVIEW are called "Virtual Instruments" or VI's due to the instrumentation-related origin. The programs created are independent of the type of machine that they are created for so programs can be transferred between different operating systems.

Additionally LabVIEW has a large set of built-in mathematical functions and graphical data visualization and data input objects typically found in data acquisition and analysis applications. You can write most of your "code" with only the mouse. If structured "properly", this "code" can pass as your flow chart.

Some features of LabVIEW:

- Graphical programming called as "G" language,
- Data-flow-controlled execution, as compared to sequential execution of text-line based languages,
- Real time visual debugging features
- Built in drivers and function libraries for the serial, parallel and network computer ports.

- Simple file input-output operations,
- "Plug-and-play" interface devices for most types of external equipment.
- Direct program portability (binary files) between different platforms: PCs, Macintosh, Sun, HP-UX, and operating systems.
- A wealth of visual debugging tools,
- Add-on software packages for specific extension of the program features, for instance image processing,
- Built-in interactive graphic control and display
- Database (SQL) interfacing, libraries for industrial PLCs
- Ready to use analysis functions including;
- Communication (TCP, UDP, DDE, OLE, HiQ)
- Signal generation (sine wave, triangle wave, square wave, sawtooth, uniform, gaussian white and periodic white noise, etc.)
- Digital signal processing (FFT, power spectrum, hilbert transform, convolution, derivative x(t), integral x(t), etc.)
- Measurement (power spectrum, time domain windowing, transfer function, harmonic analyzer, pulse parameters, peak detection, etc.)
- Filtering (butterworth, IIR, chebyshev, bessel filter, median filter etc.)
- Windows (hanning, hamming, triangle, flat top, force window, exponential window, etc.)
- Curve fitting (linear, exp., poly. nonlinear Lev-Mar.fit, interpolation, etc.)
- Probability and statistics functions (mean, standard deviation, RMS, histogram, distributions (chi square, F, t, inverse distributions, erfc(x), erf(x), contingency table, etc), ANOVA (1D, 2D, 3D), etc.
- Linear algebra (many functions including some advanced linear algebra functions)
- Array operations (numerical methods, roots, etc.)
- Code interface function to use DLL's written any other language. This feature gives the opportunity to use the codes written in conventional languages (C/C++, Visual Basic, etc) to be used in a LabVIEW program.

- Add-on software packages for specific extension of the program features, for instance image processing,

3.2 DATAFLOW PROGRAMMING

The programming language used in LabVIEW, called "G", is a dataflow language. Execution is determined by the structure of a graphical block diagram (the LV-source code) on which the programmer connects different function-nodes by drawing wires. These wires propagate variables and any node can execute as soon as all its input data become available. Since this might be the case for multiple nodes simultaneously, "G" is inherently capable of parallel execution. Multi-processing and multi-threading hardware is automatically exploited by the built-in scheduler, which multiplexes multiple OS threads over the nodes ready for execution.

Programmers with a background in conventional programming often show a certain reluctance to adopt the LabVIEW dataflow scheme, claiming that LabVIEW is prone to race conditions. In reality, this stems from a misunderstanding of the data-flow paradigm. The afore-mentioned data-flow (which can be "forced", typically by linking inputs and outputs of nodes) completely defines the execution sequence, and that can be fully controlled by the programmer. Thus the execution sequence of the LabVIEW graphical syntax is as well-defined as with any textually coded language such as C, Visual BASIC, or Python etc. Furthermore, LabVIEW does not require type definition of the variables; the wire type is defined by the data-supplying node. LabVIEW supports polymorphism in that wires automatically adjust to various types of data.

Screenshot of a simple LabVIEW program that generates, synthesizes, analyzes and displays waveforms, showing the block diagram and front panel. Each symbol on the block diagram represents a LabVIEW subroutine (subVI) which can be another LabVIEW program or a LV library function.

3.3 GRAPHICAL PROGRAMMING

LabVIEW ties the creation of user interfaces (called front panels) into the development cycle. LabVIEW programs/subroutines are called virtual instruments (VIs). Each VI has three components: a block diagram, a front panel and a connector pane. The latter may represent the VI as a subVI in block diagrams of calling VIs. Controls and indicators on the front panel allow an operator to input data into or extract data from a running virtual instrument. However, the front panel can also serve as a programmatic interface. Thus a virtual instrument can either be run as a program, with the front panel serving as a user interface, or, when dropped as a node onto the block diagram, the front panel defines the inputs and outputs for the given node through the connector pane. This implies each VI can be easily tested before being embedded as a subroutine into a larger program.

The graphical approach also allows non-programmers to build programs by simply dragging and dropping virtual representations of the lab equipment with which they are already familiar. The LabVIEW programming environment, with the included examples and the documentation, makes it easy to create small applications. This is a benefit on one side but there is also a certain danger of underestimating the expertise needed for good quality "G" programming. For complex algorithms or large-scale code it is important that the programmer understands the special LabVIEW syntax and the topology of its memory management well. The most advanced LabVIEW development systems offer the possibility of building stand alone applications. Furthermore, it is possible to create distributed applications which communicate by a server/client scheme, which the inherently parallel nature of *G*-code makes easy.

3.4 BENEFITS

One benefit of LabVIEW over other development environments is the extensive support for accessing instrumentation hardware. Drivers and abstraction layers for many different types of instruments and buses are included or available. These present themselves as graphical nodes. The abstraction layers offer standard software interfaces to communicate with hardware devices. The provided driver interfaces save program development time.

The sales pitch of National Instruments is, therefore, that even people with limited coding experience can write programs and deploy test solutions in a reduced time frame when compared to more conventional or competing systems.

In terms of performance, LabVIEW includes a compiler that produces native code for the CPU platform. The graphical code is translated into executable machine code by interpreting the syntax and by compilation. The LabVIEW syntax is strictly enforced during the editing process and compiled into the executable machine code when requested to run or upon saving. In the latter case, the executable and the source code are merged into a single file. The executable runs with the help of the LabVIEW run-time engine, which contains some precompiled code to perform common tasks that are defined by the G language. The run-time engine reduces compile time and also provides a consistent interface to various operating systems, graphic systems, hardware components, etc. The run-time environment makes the code portable across platforms. Generally, LV code can be slower than equivalent compiled C code, although the differences often lie more with program optimization than inherent execution speed.

Many libraries with a large number of functions for data acquisition, signal generation, mathematics, statistics, signal conditioning, analysis, etc., along with numerous graphical interface elements are provided in several LabVIEW package options.

The LabVIEW Professional Development System allows creating stand-alone executables and the resultant executable can be distributed an unlimited number of times. The run-time engine and its libraries can be provided freely along with the executable.

A benefit of the LabVIEW environment is the platform independent nature of the *G*-code, which is (with the exception of a few platform specific functions) portable between the different LabVIEW systems for different operating systems (Windows, MacOSX and Linux).

CHAPTER 4

Software Implementation

4.1 TEXT TO SPEECH MODULE USING LABVIEW

In this chapter development of text to speech module will be discussed. Any system requires two types of components:

a.) Hardware

b) Software

In Text to Speech system the hardware requirements are very less. It requires only a good quality speaker for the production of sound signal. The software part is developed using the LabVIEW Software.

Here, in text to speech module a text box is created so that user can write text which is to be converted into speech in .wav file format and creates a wave file named output .wav , which can be listen by using wave file player.

We have used the ACTIVE X sub pallet in Communication pallet and its functions to exchange data between applications. ActiveX technology provides a standard model for interapplication communication that different programming languages can implement on different platforms

4.1.1 ActiveX and LabVIEW

ActiveX is the general name for a set of Microsoft Technologies that allows users to re-use code and link individual programs together to suit their computing needs. Based on COM (Component Object Model) technologies, ActiveX is an extension of a previous technology called OLE (Object Linking and Embedding).

The principle is that components need not be regenerated by each program, but rather, reused to fully give the user the power to combine applications together.

ActiveX Automation

ActiveX refers to the process of controlling one program from another via ActiveX. Much like networking, one program acts as the client and the other as the server. LabVIEW supports automation both as the client and the server. Both programs (client and server) exist independent of each other, but they are able to share information between each other. This sharing of information is achieved through communication of the automation client with the ActiveX objects that the automation server exposes.

The objects have properties and methods that can be accessed by the automation client. Properties are simply attributes of an object whose values can be set or retrieved from other programs. Similarly, methods are functions that are performed on objects, and they can be invoked from other programs. An example of an ActiveX property would be the program name, height or width, and likewise, an example method could be the save or print method.

Here we have used LabVIEW as an Automation Client.

4.1.2 LabVIEW as an Automation Client

LabVIEW provides functions in its Application Programmable Interface (API) that allow it to act as an automation client with any automation server. The figure 4.1 shows the programming flow used in LabVIEW, and gives the associated functions with each block.

In general, information about a program's ActiveX automation server can be obtained from the program's documentation or by browsing the program's type library.

Often, LabVIEW is used as an automation client for Microsoft programs, and their object models are available from Microsoft . Here we have used *Microsoft Speech Object Library (Version 5.1)* to build speech-enabled applications, which retrieve the voice and audio output information available for computer. This library allows to select the voice and audio device one would like to use, enter the text to be read, and adjust the rate and volume of the selected voice

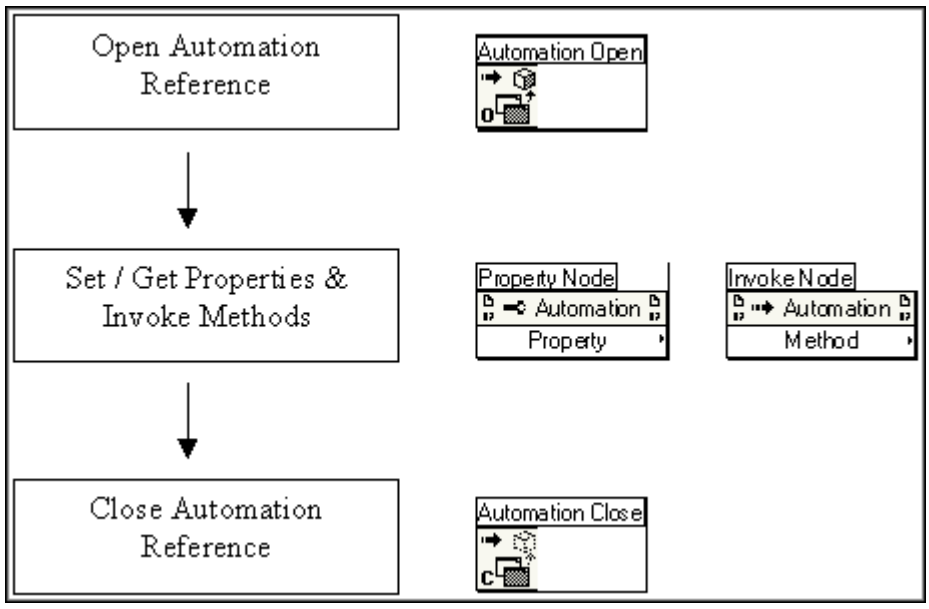
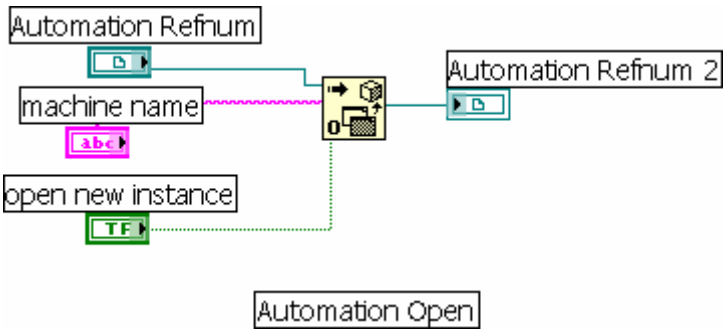


Fig 4.1 Programming flow of ActiveX used in LabVIEW

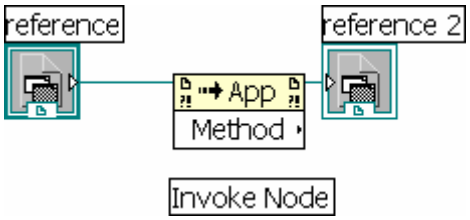
4.1.2.1 Automation Open

Returns an automation refnum, which points to a specific ActiveX object. In Text to Speech VI, it gives refnum for Microsoft speech object library.



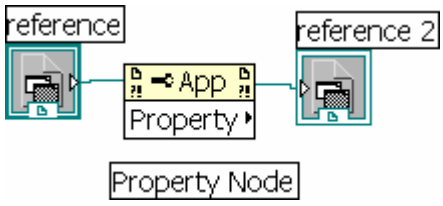
4.1.2.2 Invoke Node

Invokes a method or action on a reference. Most methods have associated parameters. If the node is configured for VI Server Application class or Virtual Instrument class and **reference** is unwired, **reference** defaults to the current Application or VI. In Text to Speech VI we have invoked method for creating and closing wave file and text to speech conversion.



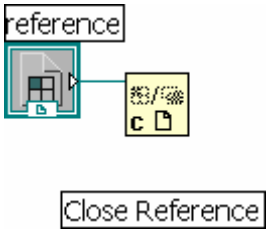
4.1.2.3 Property Node

Gets (reads) and/or sets (writes) properties of a reference. The Property Node automatically adapts to the class of the object that you **reference**. LabVIEW includes Property Nodes preconfigured to access VISA properties and properties. In Text to Speech VI property node utilizes the property of creating audio output stream.



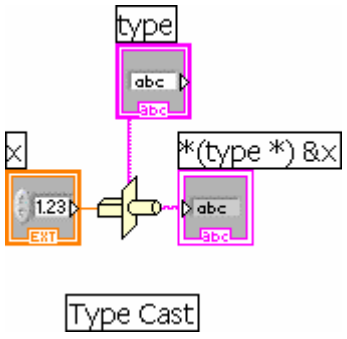
4.1.2.4 Close Reference

Closes a refnum associated with an open VI, VI object, an open instance of LabVIEW, or an ActiveX or .NET object. It closes the Microsoft speech object library.



4.1.2.5 Type Cast

Casts **x** to the data type, **type**, by flattening it and unflattening it using the new data type. The connector pane displays the default data types for this polymorphic function. In Text to Speech VI it provides the refnum to other nodes.



Flowchart for the text to wave file conversion is given in Figure 4.2. This VI creates a wave format file named output .wav which consist the typed text converted into speech.

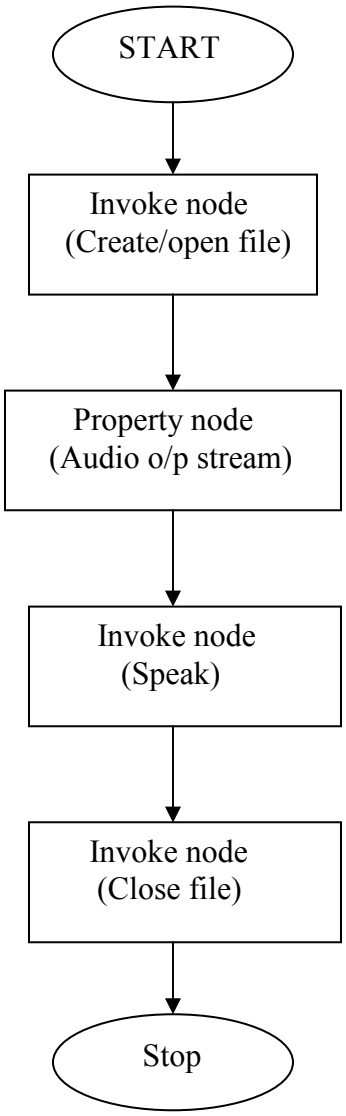


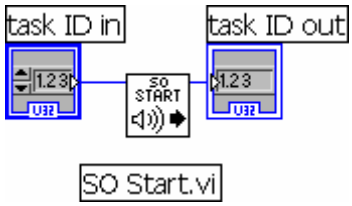
Fig.4.2 Flowchart for the text to wave file conversion

4.2 WAVE FILE PLAYER .VI

This VI illustrates how to playback a wave file using sound board, information such as file name, sound quality rate & bits/sample are also displayed. We are using this VI to listen the speech generated by text to speech conversion. Various Functions used in wave file player.vi are described here.

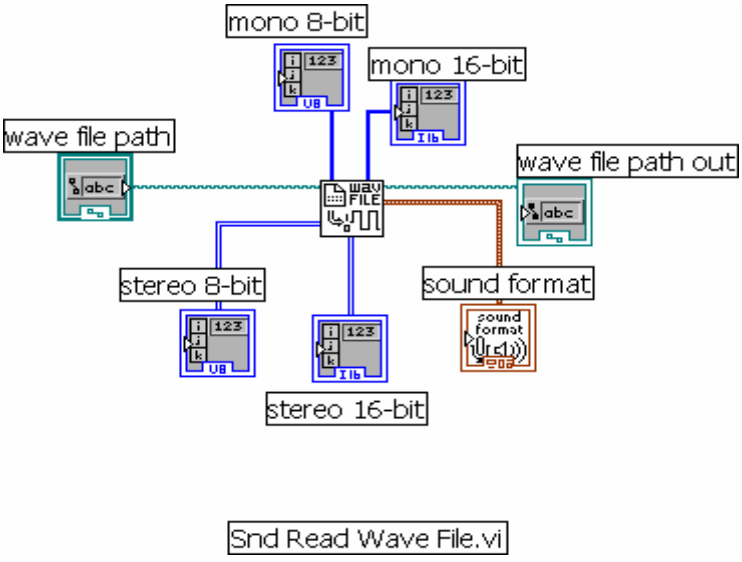
4.2.1 SO Start

Starts a sound output operation associated with the **task ID in**. If the device is running already, calling this VI has no effect. If the device is in pause mode, this VI continues the output operation.



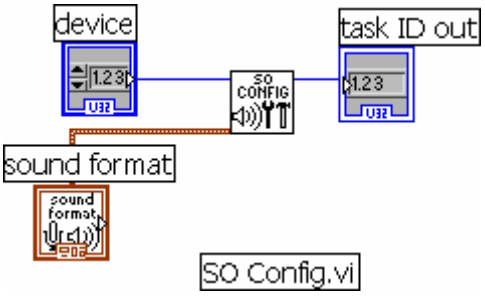
4.2.2 Snd Read Wave File

Retrieves a PC wave file (.wav) specified in **wave file path**. The information returned includes both waveform data and sound format data, which is necessary for configuring a sound device to play the waveform. This VI only retrieves uncompressed wave files.



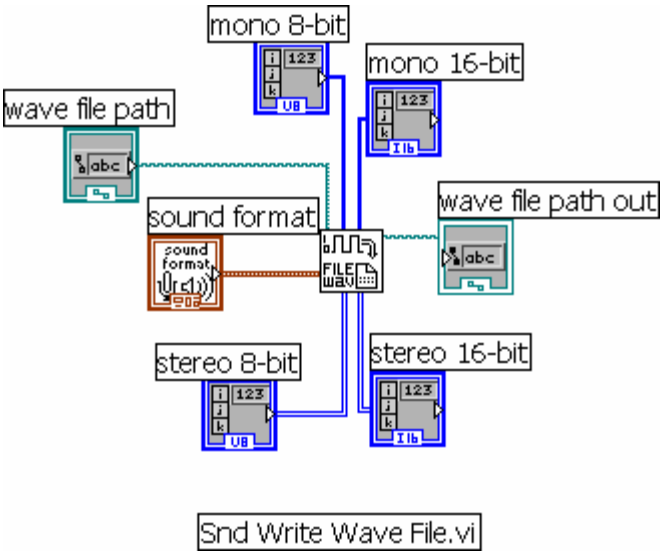
4.2.3 SO Config

Configures a sound output device and creates a sound output task ID. After you use this VI to configure the sound device, the device is in pause mode. You can use the SO Write and SO Start VIs to play the application data.



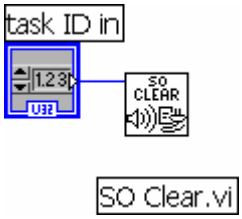
4.2.4 SO Write

Writes data to the sound output device associated with the **task ID in**. If the device is running already, the data move to the buffer immediately. If the device is in pause mode, the data do not start playing until SO Start runs.



4.2.5 SO Clear

Closes the output sound device associated with the **task ID in** and releases any resource the device uses to the computer system.



4.2.6 Disabled Property

Property of Control.

Indicates if control can be operated: 0-Enabled, 1-Disabled, 2-Disabled and Grayed Out.

The following table lists the characteristics of this property.

Settable when the VI is running	Yes
Need to authenticate before use	No
Requires the block diagram to be loaded	No
Available on local LabVIEW only	No
Requires the front panel to be loaded	No
Must wait until user interface is idle	No
Available with control VIs	No
Available with global VIs	No
Available with strict type definitions	Yes
Available with polymorphic VIs	No
Available in Run-Time Engine	Yes (Read/Write)
Permissions	Read/Write

Flowchart for wave file player.vi is shown in Figure 4.3

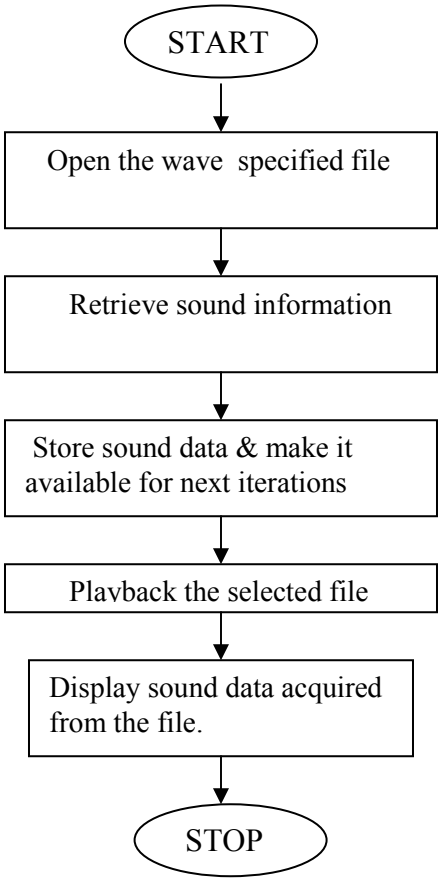


Fig 4.3 Flowchart for wave file player.vi

4.3 BUILD SCREEN.VI

This Sub VI gives information about sound format of wave file, which includes information like Sound Quality , Sampling Rate , Channel Information , File Name , Bits/Sample. Fig 4.4 shows the Front Panel diagram of Build screen.vi

Fig.4.4 Front Panel diagram of Build Screen.vi

CHAPTER 5

Results and Conclusions

5.1 RESULTS

The front panel of text to speech.VI is shown in fig 5.1.

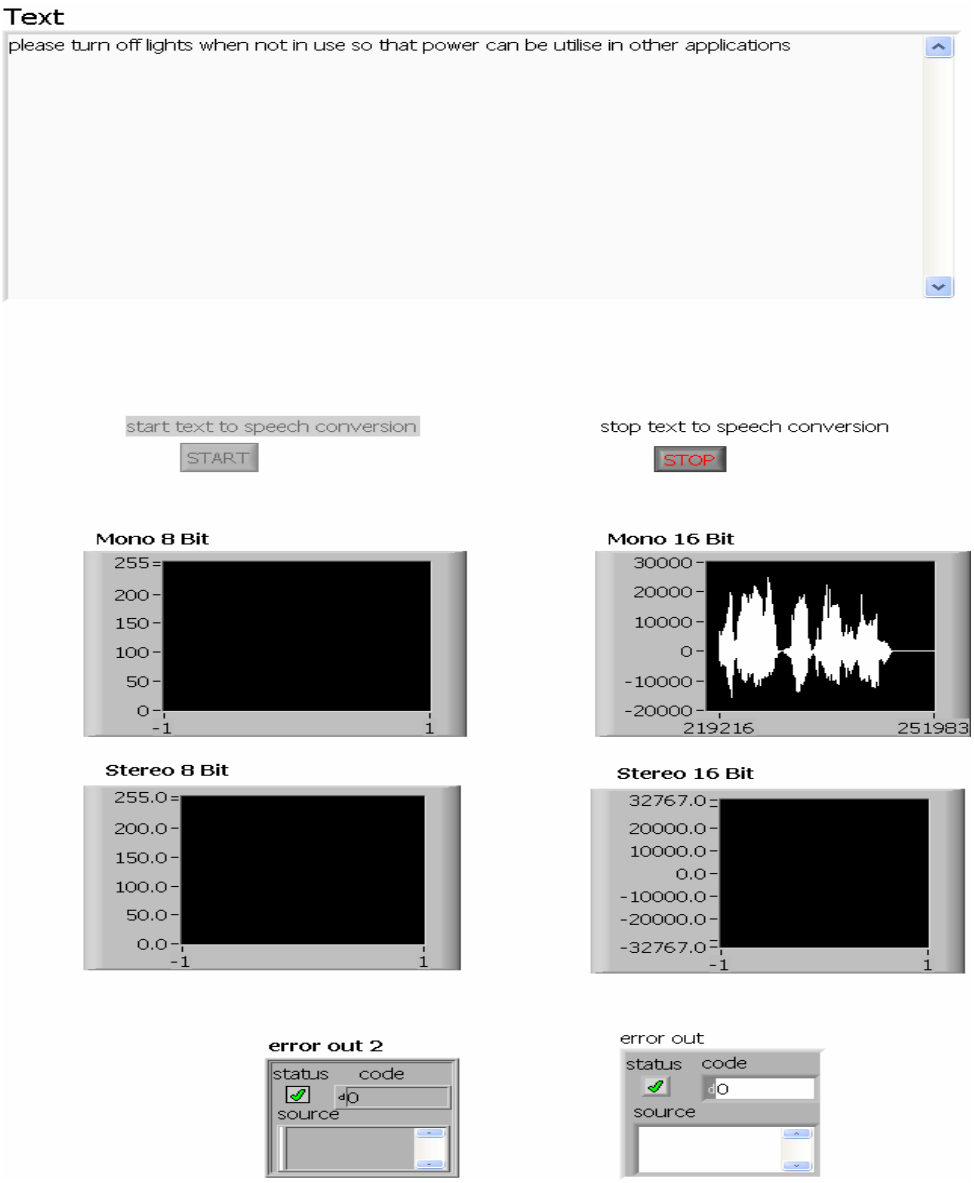


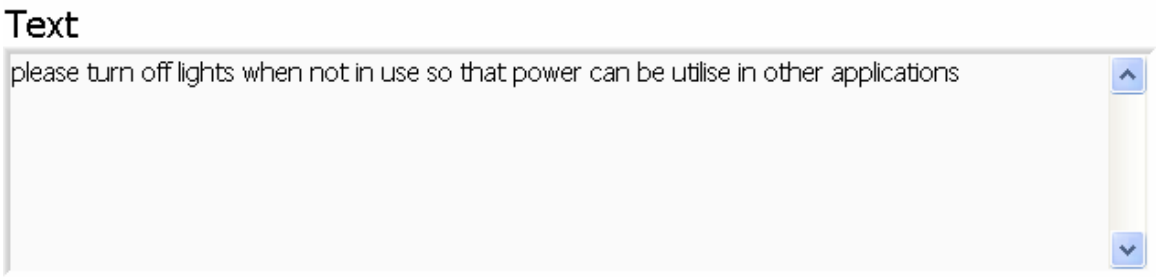
Fig 5.1 Front Panel Diagram of TEXT TO SPEECH .VI

Various steps for the execution of the program are listed below:

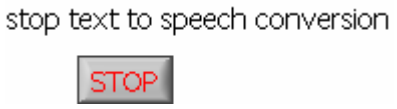
STEP1: Run the VI.

STEP2: Press the START button

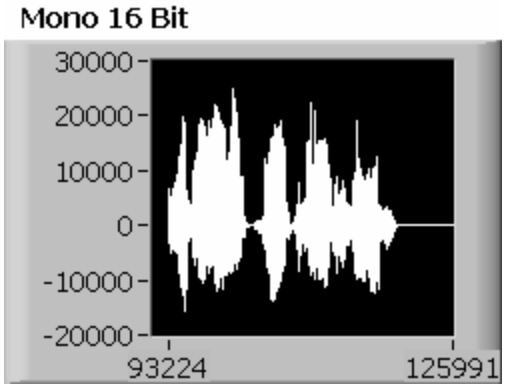
STEP3: Write some text into the given text box.



STEP4: Press STOP button to stop the VI.



STEP5: A wave file output.wav is created containing text converted into speech which can listen using wave file player.



The waveform will vary according to the different text typed in the text box and can be listened on the speaker. The wave form for **"GOOD MORNING"** is shown in figure 5.2.

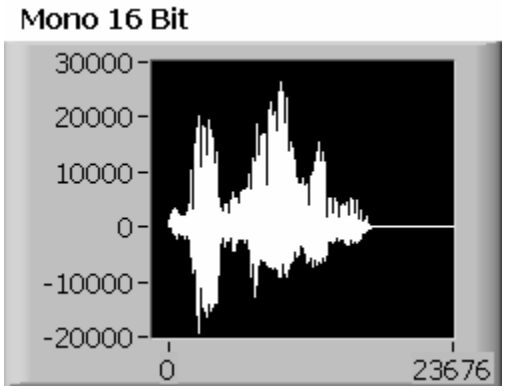


Fig.5.2 waveform for "GOOD MORNING"

5.2 CONCLUSIONS

The LabVIEW tool allows us to implement the Text to Speech conversion .LabVIEW has its strong inbuilt Speech library to implement the Text to Speech conversion. Various methods & techniques are available for speech synthesis which is discussed in this report. Here, in text to speech conversion VI ,a text box is created so that user can write text which is to be converted into speech in .wav file format and creates a wave file named output .wav , which can be listen by using wave file player.

The ACTIVE X sub pallet in Communication pallet is used to exchange data between applications. ActiveX technology provides a standard model for interapplication communication that different programming languages can implement on different platforms. *Microsoft Speech Object Library (Version 5.1)* has been used to build speech-enabled applications, which retrieve the voice and audio output information available for computer. This library allows to select the voice and audio device one would like to use, enter the text to be read, and adjust the rate and volume of the selected voice.

The application developed is user friendly, cost effective and gives the result in the real time. Moreover, the program has the required flexibility to be modified easily if the need arises.

5.3 FUTURE SCOPE

A speech synthesize system i.e. a text to speech conversion system is developed using the LabVIEW. Still some more work can be done in this field as mentioned below:

1) By adding some reverberation it may be possible to increase the pleasantness of synthetic speech afterwards

2) Different sound wave format files such as .mp3 etc. or other format required by user, can produced using different techniques

3) Different methods for correct pronunciation can be used to improve better and correct pronunciation.

4) Provision for controlling the bits/samples, and hence the speech speed, pitch control etc can be added as new feature.

5) A new module can be added for the voice activated remote control applications.

References and Literature

1. Santen J., Sproat R., Olive J., Hirschberg J. (editors) (1997). *Progress in Speech Synthesis*, Springer-Verlag New York Inc.
2. Beskow K., Elenius K., McGlashan S. (1997). The OLGA Project: An Animated Talking Agent in a Dialogue System. *Proceedings of Eurospeech 97*. http://www.speech.kth.se/multimodal/papers/
3. Kleijn K., Paliwal K. (Editors) (1998). *Speech Coding and Synthesis*. Elsevier Science B.V., The Netherlands
4. Cowie R., Douglas-Cowie E. (1996). Automatic Statistical Analysis of the Signal and Prosodic Signs of Emotion in Speech. *Proceedings of ICSLP 96* (3).
5. Sagisaga Y. (1990). Speech Synthesis from Text.
6. Flanagan J., Rabiner L. (Editors) (1973). *Speech Synthesis*. Dowden, Hutchinson & Ross, Inc., Pennsylvania.
7. O'Saughnessy D. (1987). *Speech Communication - Human and Machine*, Addison-Wesley
8. Fant G. (1970). *Acoustic Theory of Speech Production*. Mouton, The Hague.
9. Breen A., Bowers E., Welsh W. (1996). An Investigation into the Generation of Mouth Shapes for a Talking Head. *Proceedings of ICSLP 96* (4).
10. Veldhuis R., Bogaert I., Lous N. (1995). Two-Mass Models for Speech Synthesis. *Proceedings of Eurospeech 95* (3): 1853-1856.
11. Donovan R. (1996). *Trainable Speech Synthesis*. PhD. Thesis. Cambridge University Engineering Department, England. http://svr-ftp.eng.cam.ac.uk/pub/reports/donovan_thesis.ps.Z.
12. Belhoula K. (1993). Rule-Based Grapheme-to-Phoneme Conversion of Names. *Proceedings of Eurospeech 93* (2): 881-884.
13. IPA (1998). International Phonetic Association Homepage. http://www.arts.gla.ac.uk/IPA/ipa.html.
14. Altosaar T., Karjalainen M., Vainio M. (1996). A Multilingual Phonetic Representation and Analysis for Different Speech Databases. *Proceedings of ICSLP 96* (3).

15. Klatt D. (1987) Review of Text-to-Speech Conversion for English. *Journal of the Acoustical Society of America, JASA* vol. 82 (3), pp.737-793.

16. Macon M. (1996). *Speech Synthesis Based on Sinusoidal Modeling*. Doctorial Thesis, Georgia Institute of Technology

17. Allen J., Hunnicutt S., Klatt D. (1987). *From Text to Speech: The MITalk System*. Cambridge University Press, Inc

18. http://www.wiseGEEK.com/What is TTS.htm

19. Kröger B. (1992). Minimal Rules for Articulatory Speech Synthesis. *Proceedings of EUSIPCO92* (1): 331-334.

20. Holmes W., Holmes J., Judd M. (1990). Extension of the Bandwith of the JSRU Parallel-Formant Synthesizer for High Quality Synthesis of Male and Female Speech. *Proceedings of ICASSP 90* (1): 313-316.

21. Dutoit T., Pagel V., Pierret N., Bataille F., Vrecken O. (1996). The MBROLA Project: Towards a Set of High Quality Speech Synthesizers Free of Use for Non Commercial Purposes. *Proceedings of ICSLP 96* (3).

22. Huang X., Acero A., Adcock J., Hon H., Goldsmith J., Liu J., Plumpe M. (1996). Whistler: A Trainable Text-to-Speech System. *Proceedings of ICSLP96* (4).

23. Hon H., Acero A., Huang X., Liu J., Plumpe M. (1998). Automatic Generation of Synthesis Units for Trainable Text-to-Speech Systems. *Proceedings of ICASSP 98 (CD-ROM).*

24. McAulay R., Quatieri T. (1986). Speech Analysis-Synthesis Based on Sinusoidal Representation. *Proceedings of ASSP-34* (4): 744-754.

25. Macon M., Clements C. (1996). Speech Concatenation and Synthesis Using an Overlap-Add Sinusoidal Model. *Proceedings of ICASSP 96*: 361-364.

26. Gaved M. (1993). Pronunciation and Text Normalisation in Applied Text-to-Speech Systems. *Proceedings of Eurospeech 93* (2): 897-900.

27. Abadjieva E., Murray I., Arnott J. (1993). Applying Analysis of Human Emotion Speech to Enhance Synthetic Speech. *Proceedings of Eurospeech 93* (2): 909-912.

28. Scherer K. (1996). Adding the Affective Dimension: A New Look in Speech Analysis and Synthesis. *Proceedings of ICSLP 96* (3).

29. Ohala J. (1996). Ethological Theory and the Voice Expression of Emotion in the Voice. *Proceedings of ICSLP 96* (3).

30. Murray I., Arnott L. (1993). Toward the Simulation of Emotions in Synthetic Speech: A Review of the Literature on Human Vocal Emotion. *Journal of the Acoustical Society of America, JASA* vol. 93 (2): 1097-1108.

31. Fries G. (1993). Phoneme-Depended Speech Synthesis in the Time and Frequency Domains. *Proceedings of Eurospeech 93* (2): 921-924.

32. Lee K. (1989). Hidden Markov Models: Past, Present, and Future. *Proceedings of Eurospeech 89* (1): 148-155.

33. Portele T., Krämer J. (1996). Adapting a TTS System to a Reading Machine for the Blind. *Proceedings of ICSLP 96* (1).

34. Klatt D. (1980). Software for a Cascade/Parallel Formant Synthesizer. *Journal of the Acoustical Society of America, JASA*, Vol. 67: 971-995.

35. Amundsen M. (1996). *MAPI, SAPI, and TAPI Developers Guide*. Sams Publishing. http://book.ygm.itu.edu.tr/Book/mapi_sapi/index.htm

36. MPEG Homepage (1998). http://drogo.cselt.stet.it/mpeg/