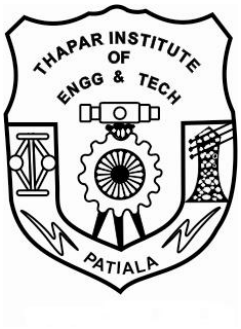
TEXT-TO-SPEECH SYNTHESIS FOR PUNJABI LANGUAGE

A Thesis
submitted in partial fulfillment of the
requirements for the award of degree of

Master of Engineering
in
Software Engineering



under the supervision of
Dr. R. K. Sharma
Professor and Head
School of Mathematics and Computer Applications
Thapar Institute of Engineering and Technology, Patiala

Submitted By
Pardeep Gera
(8043115)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY
(DEEMED UNIVERSITY)
PATIALA – 147004(India)

May 2006

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Text-to-Speech Synthesis for Punjabi Language"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Software Engineering submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology (Deemed University), Patiala, is an authentic record of my own work carried out under the supervision of Dr. R. K. Sharma.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

Pardeep Gera

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Dr. R. K. Sharma
Professor and Head
School of Mathematics and Computer Applications
Thapar Institute of Engineering and Technology, Patiala

Countersigned by

Dr. (Mrs.) Seema Bawa
Professor and Head
Computer Science & Engineering Department
Thapar Institute of Engg. and Technology
Patiala

Dr. T. P. Singh
Dean (Academic Affairs)
Thapar Institute of Engg. and Technology
Patiala

Acknowledgement

A journey is easier when traveled together. Interdependence is certainly more valuable than independence. This thesis is the result of work carried out during the final semester of my course whereby I have been accompanied and supported by many people. It is a pleasant aspect that I now have the opportunity to express my gratitude for all of them.

No amount of words can adequately express the debt, I owe to Dr. R.K. Sharma, Professor and Head, School of Mathematics and Computer Applications, for his kind support, motivation and inspiration that triggered me for the thesis work. Every interaction with him was an inspiration. At every step he was there to help me choose the right path. I feel proud and honoured of having got this opportunity of working under his able guidance.

I wish to express my gratitude to Dr. (Mrs.) Seema Bawa, Professor and Head, Computer Science and Engineering Department, for encouraging us right from the beginning of the course. I am also thankful to all the faculty and staff members of the Computer Sc. & Engg. Department for providing me all the facilities required for the completion of this work.

No thesis could be written without being influenced by the thoughts of others. I would like to thank my friends who were always there at the hour of the need and provided with all the help and support, which I needed.

Pardeep Gera
(8043115)

Abstract

In recent years, the use of computers in speech synthesis and speech recognition has become an important area of study among speech and computer scientists. The primary motivations are to provide users with a friendly vocal interface with the computer and to allow people with certain handicaps (such as blindness) to use the computer. The tremendous recent developments in speech and computer technology have produced unrestricted-vocabulary speech synthesis on PCs in English and some other European languages. In India, very few institutions are working in the field of speech synthesis. Indian Institute of Technology, Kharagpur, has developed an Indian language speech synthesizer named “Shruti” for Hindi and Bengali languages. Another popular speech synthesizer available for Indian languages is “Dhvani”, which is for Hindi and Kannada languages.

In speech synthesis, the accuracy of information extraction is crucial in producing high quality synthesized speech. Speech synthesis involves the algorithmic conversion of input text data to speech waveforms. Speech Synthesizers are characterized by the method used for storage, encoding and synthesis of the speech. The synthesis method is determined by the vocabulary size as all possible utterances of the language need to be modeled. It is a well-established fact that text-to- speech (TTS) synthesizers designed for use in a restricted domain always perform better than their general-purpose counterparts. The design of such general purpose synthesizers are complicated by the fact that the sound output needs to be close to natural speech.

The presented work attempts to achieve this type of speech synthesis for Punjabi language. The synthesizer uses an existing TTS named Dhvani that has been developed for Hindi and Kannada Language, as a model. Dhvani acts as a prototype of a phonetics-to-speech engine, which can serve as a back-end for speech synthesizers in many Indian languages. One just has to associate it with the corresponding language specific text-to-phonetics module. The synthesizer has been implemented and tested successfully on Fedora Core 3 platform. The results are evaluated to determine the performance of the synthesizer developed in terms of intelligibility and naturalness.

Table of Contents

List of Figures

List of Tables

Chapter

1

INTRODUCTION

Speech is the primary means of communication between people. Speech synthesis has been under development for several decades from now. Recent progress in speech synthesis has produced synthesizers with very high intelligibility. In this chapter, a brief introduction to speech processing has been given, which includes speech synthesis and speech recognition. The chapter starts with a brief history of speech processing. Further sections give an outline of the field of speech synthesis and text-to-speech (TTS) conversion. A black-box view of TTS synthesizer is also presented and applications of synthesized speech are discussed. The chapter concludes with a summary and overview of the remaining thesis.

1.1 SPEECH PROCESSING

In the year 1960, the world first time witnessed the idea of a talking computer. It was demonstrated in a movie theater through a space odyssey with two astronauts and a computer. This computer was named HAL. HAL could not only speak but was also friendly and understanding.

Before HAL, an actor speaking as a computer deliberately created a stylized, mechanical, "robotic" voice. That mechanical sound was the viewer's perception that a computer was speaking. However, HAL presented the possibility that future computers would speak and function like human beings.

For most of the people, who were present in the demonstration of HAL, a computer was something out of science fiction. The way we interact with computers today - by typing on a keyboard to input information and receiving responses on a video screen - was just being designed at that time. With the invention of new technologies such as speech synthesis and speech recognition, we are now moving into an era of more effective human-computer interaction.

1.1.1 SPEECH TECHNOLOGY

Speech technology consists of the following two key components:

- Speech synthesis – Speech synthesis can be described in the simple words as computers speaking to people. This mainly requires computers to understand the language speaking rules. TTS synthesizers belong to this category.
- Speech recognition – Speech recognition can be considered as people speaking to computers. This requires computers to understand the speech. SPEECH-TO-TEXT systems belong to this category.

In the present work, the speech synthesis component of speech technology has been considered with reference to Punjabi language.

1.1.2 SPEECH SYNTHESIS

Speech synthesis can also be considered as the generation of an acoustic speech signal by a computer. The application areas of speech synthesis may include:

- TTS synthesizers like e-mail or news readers, etc.
- Dialogue systems, for example, enquiry for train schedule information or information about flight reservation.
- Automatic translation (speech-to-speech) systems.
- Concept/content-to-speech (CTS), for example, weather forecasting.

In this thesis work, a TTS synthesizer for Punjabi language has been developed.

1.1.3 TTS SYNTHESIZER

A Text-To-Speech (TTS) synthesizer is a computer-based system that should be able to read any text aloud, whether it was directly introduced in the computer by an operator or scanned and submitted to an Optical Character Recognition (OCR) system [Dutoit, 1996].

As such, the process of TTS conversion allows the transformation of a string of phonetic and prosodic symbols into a synthetic speech signal. The quality of the result produced by a TTS synthesizer is a function of the quality of the string, as well as of the quality of the generation process.

1.1.4 COMPONENTS OF A TTS SYNTHESIZER

As depicted in Fig. 1.1, a TTS synthesizer is composed of two parts:

- A front-end that takes input in the form of text and outputs a symbolic linguistic representation.

- A back-end that takes the symbolic linguistic representation as input and outputs the synthesized speech in waveform.

These two phases are also called as high-level synthesis phase and low-level synthesis phase, respectively.

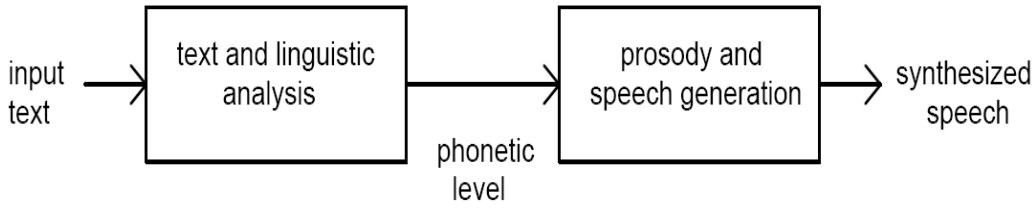


Fig. 1.1: Black-box view of a TTS synthesizer

1.1.5 TTS SYNTHESIZER VS. TALKING MACHINE/VOICE RESPONSE SYSTEMS

A TTS synthesizer differs from any other type of "talking machine" in the sense that it should be able to automatically produce "new sentences", thus, it should also be able to read unrestricted text [Allen, 1976]. It is also important to note that a TTS synthesizer differs from any of the so-called voice response systems (such as announcements in a train etc.) in the sense that the vocabulary in a TTS synthesizer is not limited. The set consisting of utterance types is also not limited in TTS synthesizer.

1.1.6 QUALITY OF A TTS SYNTHESIZER

Usually, two quality criteria are proposed for deciding the quality of a TTS synthesizer.

- **Intelligibility** – it refers to how easily the output can be understood. It can be measured by taking into account several kinds of units (phonemes, syllables, words, phrases, etc.).
- **Naturalness** or pleasantness – it refers to how much the output sounds like the speech of a real person.

Naturalness may be related to the concept of realism in the field of image synthesis: the goal is not to restitute the reality but to suggest it. Thus, listening to a synthetic voice must allow the listener to attribute this voice to some pseudo-speaker and to perceive some kind of expressivities as well as some indices characterizing the speaking style and the particular situation of elocution. For this purpose, the corresponding extra-linguistic information must be supplied to the synthesizer.

Most of the existing TTS synthesizers produce an acceptable level of intelligibility, but the naturalness dimension, the ability to control expressivities, speech style and pseudo-speaker identity are still now the areas of concern and need improvements. However, users' demands vary to a large extent according to the field of application: general public applications such as telephonic information retrieval need maximal realism and naturalness, whereas some applications involving professionals (process or vehicle control) or highly motivated persons (visually impaired, applications in hostile environments) demand intelligibility with the highest priority.

1.2 Applications Of Synthesized Speech

Synthetic speech can be used in a number of applications. Communication aids have developed from low quality talking calculators to modern 3D applications, such as talking heads. The implementation method varies from one application to the other. 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. On the other hand, some applications, such as reading machines for the blind or electronic-mail readers, require unlimited vocabulary and a TTS synthesizer is needed.

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

1.2.1 Applications For The Blind – Motivation For The Current Work

Probably the most important and useful application field in speech synthesis is the reading and communication aids for the blind persons. Before the usage of 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.

These days, the synthesizers are mostly software based. As such with scanner and OCR system, it is easy to construct a reading machine for any computer environment with tolerable expenses. Speech synthesis is currently used to read www-pages or other forms of media with normal personal computer.

1.2.2 Applications For The Deafened And Vocally Handicapped

Voice handicaps originate in mental or motor/sensation disorders. Machines can be an invaluable support in the latter case. With the help of an especially designed keyboard and a fast sentence-assembling program, synthetic speech can be produced in a few seconds to remedy these impediments.

1.2.3 Educational Applications

Synthesized speech can also be used 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. TTS synthesis coupled with a Computer Aided Learning system can provide a helpful tool to learn a new language. Synthesized speech can also be used with interactive educational applications.

Especially, with people who are impaired to read, speech synthesis may be very helpful because especially some children may feel themselves very embarrassing when they have to be helped by a teacher [Klatt, 1987]. 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.

1.2.4 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. Texts might range from simple messages, such as local cultural events to huge databases, which can hardly be read and stored as digitized speech. Synthetic speech is key factor in voice mail systems. This is a well known fact that electronic mails have become very usual in last few years. However, it is not possible to read the mails without proper connectivity. This type of situation may exist at various places, e.g., a person traveling in the plane. 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 synthesizers is at least so good that it can be used to give some control commands, such as yes/no, on/off, or ok/cancel.

1.2.5 Fundamental And Applied Research

TTS synthesizers possess a very peculiar feature, which makes them wonderful laboratory tools for linguists. These are completely under control, so that repeated experiences provide identical results (as is hardly the case with human beings). Consequently, they allow investigating the efficiency of intonative and rhythmic models. A particular type of TTS synthesizer, that is based on a description of the vocal tract through its resonant frequencies (its formants) and denoted as formant synthesizer, has also been extensively used by phoneticians to study speech in terms of acoustical rules.

1.2.6 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 UNICODE/ASCII string, and then resynthesize 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 audio-visual applications. The application field for speech synthesis is becoming wider, which also brings more funds into research and development areas.

1.3 OVERVIEW OF THE THESIS

Summary of Chapter 1: Introduction - The aim of this chapter is to introduce the field of speech synthesis. The chapter began with the basic definitions of important terms, in section 1.1. To understand the relevance of TTS, the applications of TTS synthesizers are presented in section 1.2. The motivation for this thesis happens to be the application of TTS synthesizers for increasing human-computer interaction for people using Punjabi as their native language.

The overview of remaining chapters of the thesis is as follows:

Chapter 2: TTS Architecture-Detailed Overview - This chapter takes a look into the architecture of the TTS synthesizer. The sections in this chapter discuss in detail the techniques involved in the designing and development of a TTS synthesizer. For a better understanding of TTS field, a brief history of the important developments in the TTS field is given in section 2.2. Finally, section 2.3 and section 2.4 list some of the generic frameworks, and tools available for TTS development.

Chapter 3*:* TTS Synthesizers for Indian Languages - This chapter is a survey of the work done by different institutions in India in the field of TTS for Indian languages. The techniques used for Indian Language TTS development are presented here. The work done by following Institutions is reviewed as the part of literature survey: C-DAC Bangalore, IIT Mumbai, HP Labs, IIT Kharagpur, Simputer Trust, IIT Madras, IIIT Hyderabad, HCU, IISc Bangalore, Utkal University, TIFR Mumbai, C-DAC Noida and College of Engineering, Guindy.

Chapter 4: Development and Implementation of Punjabi TTS Synthesizer - This chapter presents the implementation details of the Punjabi TTS synthesizer, using Dhvani TTS synthesizer as a model. Section 4.1 gives a brief outline of an Arabic TTS synthesizer. Section 4.2 describes in detail the architecture of Dhvani. The subsequent sections describe the development of Punjabi TTS synthesizer along with various modules, the phonetic description of the input and the structure of the sound database.

Chapter 5: TTS Evaluation - This chapter presents the results of TTS evaluation tests performed on the Punjabi TTS. Some standard types of tests for assessing the quality of the speech are carried out. These tests basically include tests for the intelligibility and naturalness of the speech output of the TTS synthesizer. The outcomes of these results are analyzed.

Chapter 6: Conclusion and Future Work – This chapter has been divided into three sections. Section 6.1 describes various limitations that persist in the Punjabi TTS synthesizer developed. Section 6.2 gives a brief conclusion of the thesis and section 6.3 briefly describe the future directions of the work presented in this thesis.

Chapter 2

TTS ARCHITECTURE: DETAILED OVERVIEW

2.1 ARCHITECTURE OF A TTS SYNTHESIZER

How does a computer read? This apparently difficult question can be answered by looking into the architecture of a typical TTS synthesizer. This section discusses in more detail the two components, namely, Natural Language Processing (NLP) component and Digital Signal Processing (DSP) component (Fig. 2.1) of a typical TTS synthesizer [Dutoit, 1996].

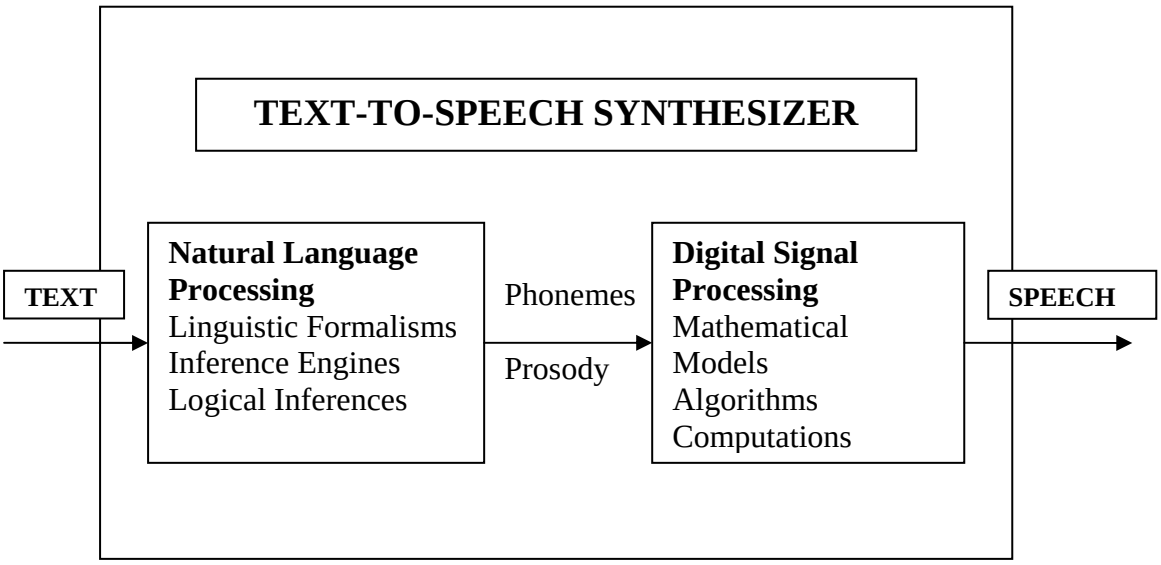


Fig. 2.1: Basic components of a TTS synthesizer

2.1.1 NLP COMPONENT

A general NLP module can be seen as consisting of three major subcomponents (Fig. 2.2) [Dutoit, 1996].

Component 1: TEXT ANALYSIS

The text analyzer consists of the following four modules:

1. A pre-processing module, which organizes the input sentences into manageable lists of words. It identifies numbers, abbreviations, acronyms and idiomatics and transforms them into full text where so ever needed.
2. A morphological analysis module, the task of which is to propose all possible part of speech categories for each word taken individually, on the basis of their spelling.
3. The contextual analysis module considers words in their context, which allows it to reduce the list of their possible part of speech categories to a very restricted number of highly probable hypotheses, given the corresponding possible parts of speech of neighboring words.
4. Finally, a syntactic-prosodic parser, which examines the remaining search space and finds the text structure (i.e., its organization into clause and phrase-like constituents) which more closely relates to its expected prosodic realization.

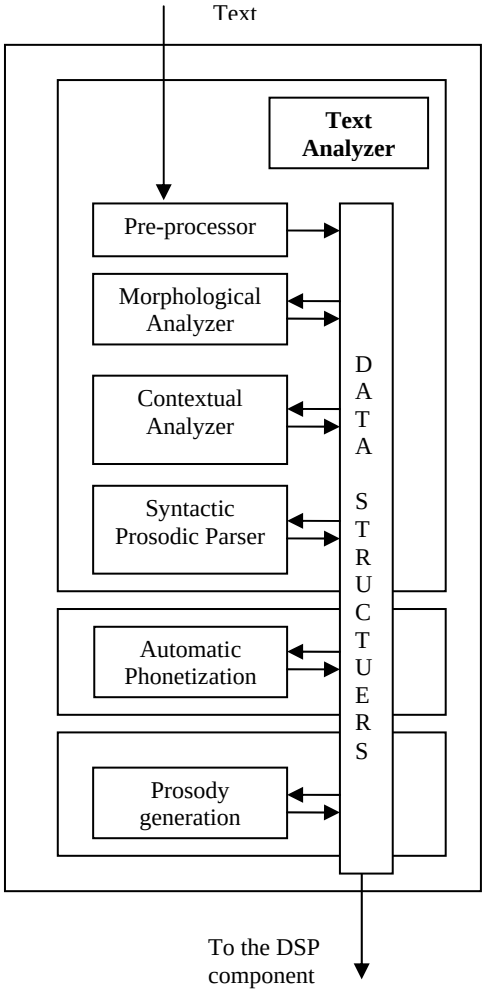


Fig. 2.2: NLP Component

Component 2: AUTOMATIC PHONETIZATION

The Automatic Phonetization module (also known as Letter-To-Sound (LTS) module) is responsible for the automatic determination of the phonetic transcription of the incoming text. Pronunciation dictionaries may not always help in this module due to the following facts:

- The pronunciation dictionaries refer to word roots only. These do not explicitly account for morphological variations (i.e. plural, feminine, conjugations, especially for highly inflected languages).
- Words embedded into sentences are not pronounced as if they were isolated. It can also be noted that not all words can be found in a phonetic dictionary. The pronunciation of new words and of many proper names has to be deduced from the one of already known words.

The two popular ways of implementing an Automatic Phonetization module are:

1. Dictionary-based solutions that consist of storing a maximum of phonological knowledge into a lexicon.
2. Rule-based transcription systems that transfer most of the phonological competence of dictionaries into a set of letter-to-sound (or grapheme-to-phoneme) rules. This time, only those words that are pronounced in such a particular way that they constitute a rule on their own are stored in an exceptions dictionary.

Component 3: PROSODY GENERATION

Prosodic features consist of pitch, duration, and stress over the time. With a good control over these features, gender, age, emotions, and other features can be incorporated in the speech and very natural sounding speech can be modeled. However, almost everything seems to have an effect on prosodic features of natural speech and it makes accurate modeling very difficult (Fig. 2.3).

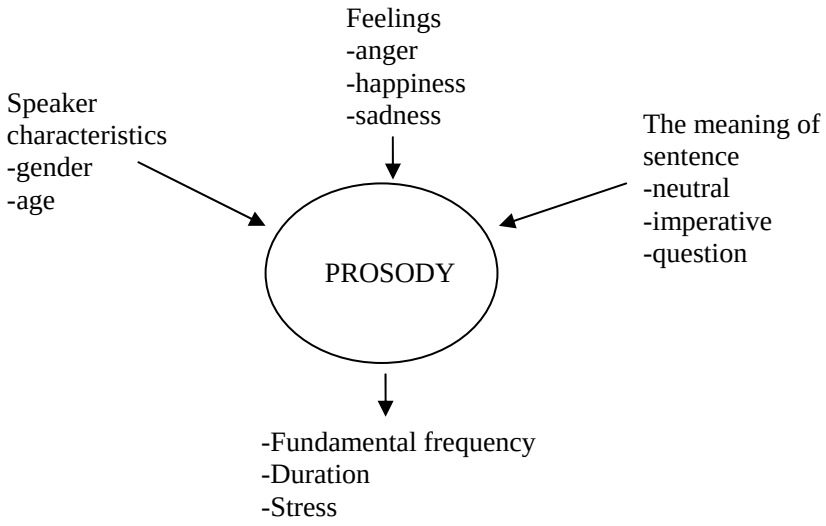


Fig. 2.3: Prosodic dependencies

Prosodic features can be divided into several levels such as syllable, word, and 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 three features of the prosody are described below in brief.

Pitch: 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 towards 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 continuous rise which indicates that there is more speech to come. A raise or fall in fundamental frequency can also indicate a stressed syllable [Klatt, 1987]. Finally, the pitch contour is also affected by gender, physical and emotional state, and attitude of the speaker.

Duration: 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. 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.

Intensity: 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 [Klatt, 1987].

2.1.2 DSP COMPONENT

The DSP component is also called as the synthesizer component. Different TTS synthesizers can be classified according to the type of synthesis technique that is used to synthesize the speech. The methods are usually classified into three groups:

Articulatory synthesis – this attempts to model the human speech production system directly and is thus, potentially the most satisfying method to produce high quality synthetic speech. Articulatory synthesizer generates speech by mathematically modeling the movement of articulators, e.g., lips, tongue, and jaws.

Formant synthesis - this models the pole frequencies of speech signal or transfer function of vocal tract based on source-filter-model. Thus speech is generated by an acoustic-phonetic production model, based on formant of the vocal tract. The acoustic-phonetic parameters such as energy, pitch and resonance (formant) frequencies associated with speech and heuristic rules are used to derive the model.

Concatenative synthesis -this uses different length prerecorded samples derived from natural speech. Here, speech is generated by combining splices of pre-recorded natural speech.

The articulatory and formant synthesis are also classified as rule-based synthesis methods whereas the concatenative technique falls under database driven synthesis method. The formant and concatenative methods are the most commonly used in present synthesizers. The formant synthesis was dominant for long time, but these days, 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.

Some other synthesis methods are:

- **Hybrid synthesis** marries aspects of formant and concatenative synthesis to minimize the acoustic glitches when speech segments are concatenated.
- **HMM-based synthesis** is a synthesis method based on Hidden Markov Models (HMMs). In this type of synthesizer, speech frequency spectrum

(vocal tract), Fundamental frequency (vocal source), and duration (prosody) are modeled simultaneously by HMMs. Speech waveforms are generated from HMMs themselves based on Maximum likelihood criterion.

- **Sinusoidal Model based Synthesis** - Sinusoidal models are based on a well known assumption that the speech signal can be represented as a sum of sine waves with time varying amplitude and frequencies.
- **Linear predictive methods** - Linear predictive methods are originally designed for speech coding systems, but may also be used in speech synthesis. In fact, the first speech synthesizers were developed from speech coders. Like formant synthesis, the basic LPC is based on the source-filter-model of speech described. The digital filter coefficients are estimated automatically from a frame of natural speech.

The presented work uses concatenative synthesis with syllable-like units for the development of Punjabi TTS synthesizer [Rao, 2005], [Kishore, 2002]. So, in the next section, details only about concatenative synthesis are given.

2.1.3 CONCATENATIVE SYNTHESIS

In the last decade, there has been a significant trend for development of speech synthesizers using Concatenative Synthesis techniques. There are a number of different methodologies for Concatenative Synthesis such as TDPSOLA, PSOLA, MBROLA and Epoch Synchronous Non Over Lapping Add (ESNOLA) [Dutoit, 1996].

There are three main subtypes of concatenative synthesis:

1. **Unit selection synthesis:** This type of synthesis uses large speech databases (more than one hour of recorded speech). During database creation, each recorded utterance is segmented into some or all of the following: individual phones, syllables, morphemes, words, phrases, and sentences. Typically, the division into segments is done using a specially modified speech recognizer set to a "forced alignment" mode with some hand correction afterward, using visual representations such as the waveform and spectrogram. An index of the units in the speech database is then created based on the segmentation and acoustic parameters like the fundamental frequency (pitch), duration, position in the syllable, and neighboring phones [Kishore, 2002].

At runtime, the desired target utterance is created by determining the best chain of candidate units from the database (unit selection). This process is typically achieved using a specially-weighted decision tree.

Unit selection gives the greatest naturalness due to the fact that it does not apply a large amount of digital signal processing to the recorded speech, which often makes recorded speech sound less natural, although some synthesizers may use a small amount of signal processing at the point of concatenation to smooth the waveform.

2. **Diphone synthesis:** It uses a minimal speech database containing all the Diphones (sound-to-sound transitions) occurring in a given language. The number of diphones depends on the phonotactics of the language: Spanish has about 800 diphones and German has about 2500 diphones. In diphone synthesis, only one example of each diphone is contained in the speech database.

   At runtime, the target prosody of a sentence is superimposed on these minimal units by means of digital signal processing techniques such as Linear predictive coding, PSOLA or MBROLA.

   The quality of the resulting speech is generally not as good as that from unit selection but more natural-sounding than the output of formant synthesizers. Diphone synthesis suffers from the sonic glitches of concatenative synthesis and the robotic-sounding nature of formant synthesis, and has few of the advantages of either approach other than small size. As such, its use in commercial applications is declining, although it continues to be used in research because there are a number of freely available implementations.

3. **Domain-specific synthesis:** It concatenates pre-recorded words and phrases to create complete utterances. It is used in applications where the variety of texts the synthesizer will output is limited to a particular domain, like trains schedule announcements or weather reports. This technology is very simple to implement, and has been in commercial use for a long time: this is the technology used by gadgets like talking clocks and calculators.

The naturalness of these synthesizers can potentially be very high because the variety of sentence types is limited and closely matches the prosody and intonation of the original recordings. However, since these synthesizers are limited by the words and phrases in their database, these are not general-purpose and can only synthesize the combinations of words and phrases they have been pre-programmed with. Fig. 2.4

gives the sub-modules of a general concatenative synthesizer, as proposed by [Dutoit, 1996].

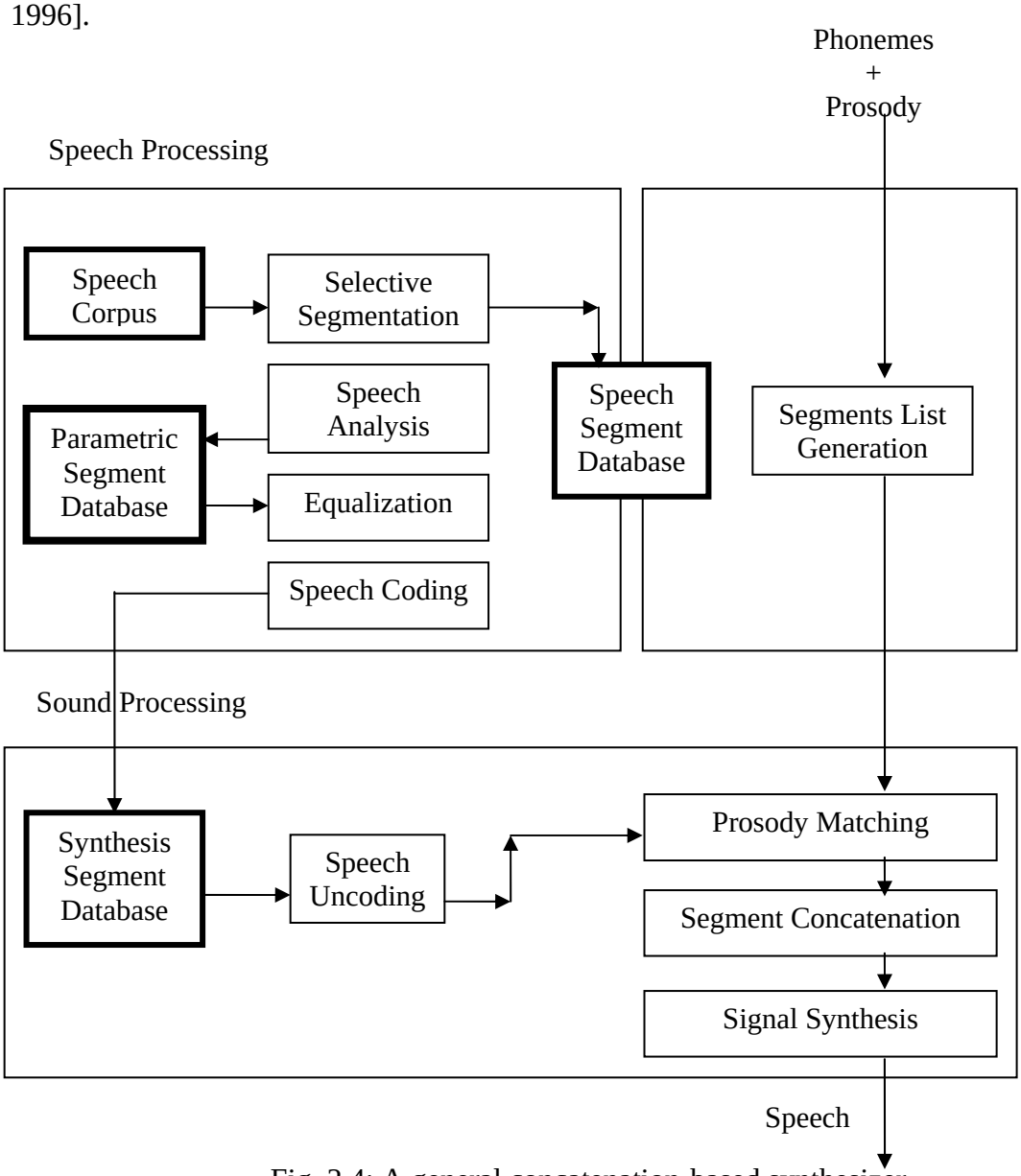


Fig. 2.4: A general concatenation-based synthesizer

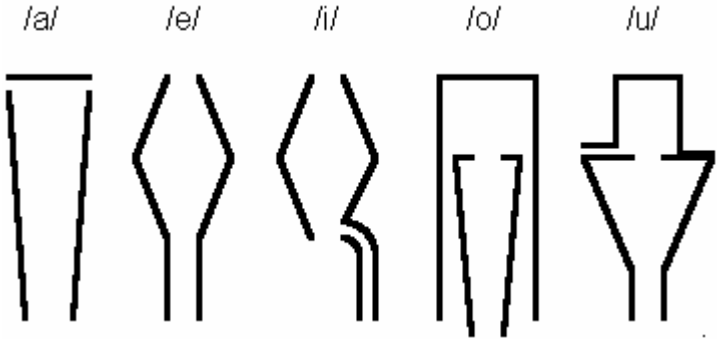
2.2 Brief History Of Speech Synthesis

Human fascination with talking machines is not new. For centuries, people have tried to empower machines with the ability to speak; prior to the machine age humans even hoped to create speech for inanimate objects. The early men attempted to show that their idols could speak, usually by hiding a person behind the figure or channeling voices through air tubes.

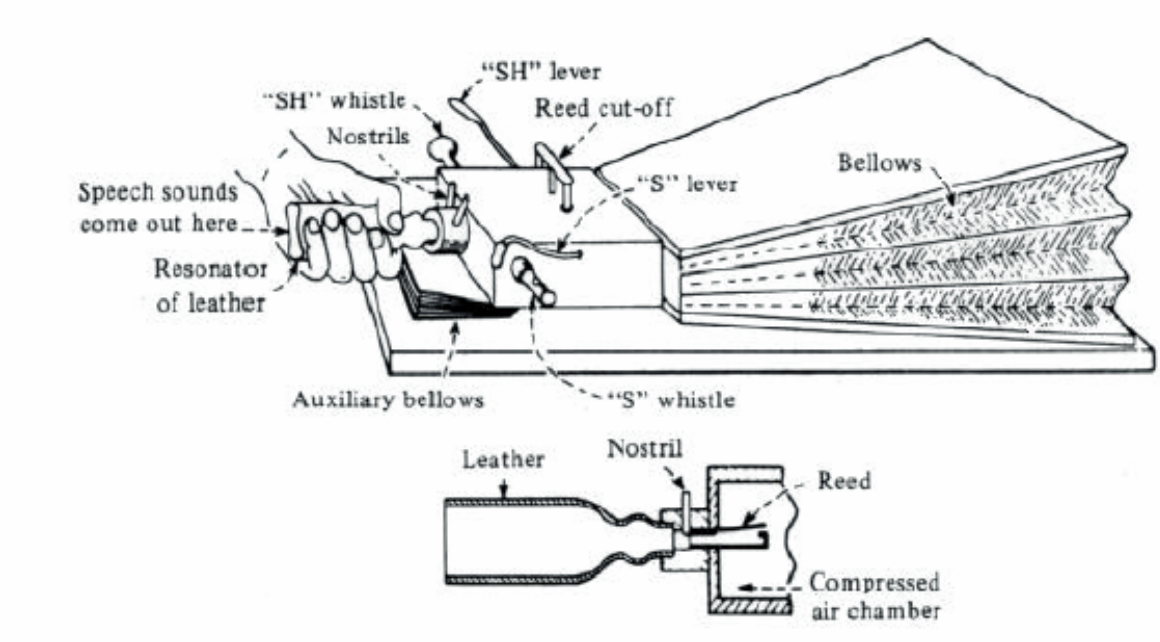
Producing artificial speech with mechanical devices, (i.e., talking machines) has been the goal of scientists for centuries. The first attempts to produce human speech by machine were made in the 18th century. The first talking machines were mechanical

speech devices that imitated the human vocal tract. A brief history of these machines is given below:

1) In 1773 Christian Kratzenstein in St. Petersburgh explained the physiological differences between five long vowels (/a/, /e/, /i/, /o/, and /u/) by an apparatus that could produce them artificially. A set of acoustic resonator tubes connected to organ pipes imitated the human vocal tract.



2) Wolfgang von Kempelen (the first experimental phonetician), wrote *Mechanismus der menschlichen Sprache nebst Beschreibung einer sprechenden Maschine* that appeared in 1791 in Vienna. The 1791 book had a detailed description of a talking machine (Fig. 2.5). Kempelen's studies led to the theory that the vocal tract, a cavity between the vocal cords and the lips, is the main site of acoustic articulation. The essential parts of the machine were a pressure chamber for the lungs, a vibrating reed to act as vocal cords, and a leather tube for the vocal tract action. The machine was hand operable and could produce not only single sounds but also whole words (and even short phrases).



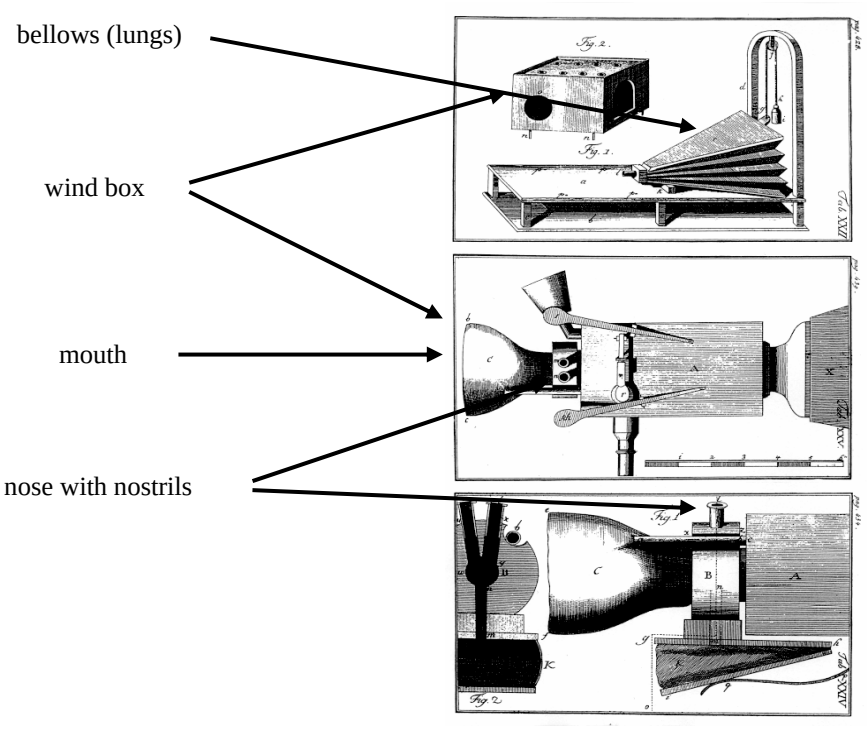


Fig. 2.5: Charles Wheatstone's version of von Kempelen's speaking machine

3) **Electrical synthesizers**: The first device to be considered as a speech synthesizer was an electric synthesizer called VODER (Voice Operating Demonstrator) which was presented by Homer Dudley at the World Fair in New York in 1939. It had a voicing/noise source with a foot pedal for fundamental frequency control. Signals, in this synthesizer, routed through ten bandpass filters. The device was played like a musical instrument (Fig. 2.6).

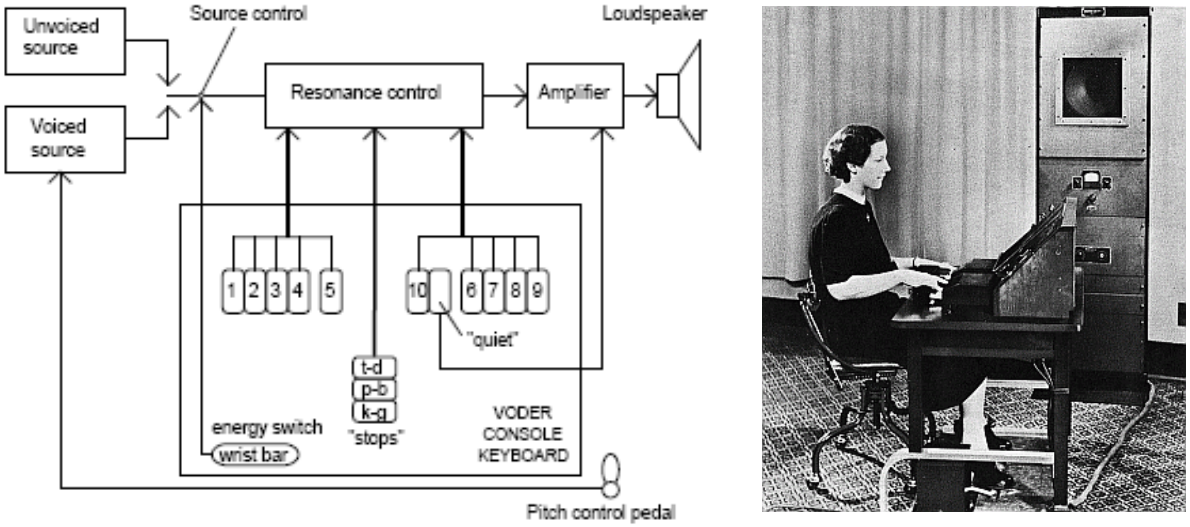


Fig. 2.6: The VODER speech synthesizer

4) **Formant synthesizers**: The first formant synthesizer, PAT (Parametric Artificial Talker), was introduced by Walter Lawrence in 1953 [Klatt, 1987]. PAT consisted of three electronic formant resonators connected in parallel. The input signal was either a buzz or noise. A moving glass slide was used to convert painted patterns into six time functions to control the three formant frequencies, voicing amplitude, fundamental frequency, and noise amplitude.

5) At about the same time when PAT was introduced, Gunnar Fant introduced the first cascade formant synthesizer, OVE (Orator Verbis Electris) which consisted of formant resonators connected in cascade.

6) **Articulatory synthesizer**: First articulatory synthesizer was introduced in 1958 by George Rosen at the Massachusetts Institute of Technology [Klatt, 1987]. The DAVO (Dynamic Analog of the VOcal tract) was controlled by tape recording of control signals created by hand.

7) The first full TTS synthesizer for English was developed in the Electro technical Laboratory, Japan in 1968 by Noriko Umeda and his companions [Klatt, 1987]. It was based on an articulatory model and included a syntactic analysis module with sophisticated heuristics. The speech was quite intelligible but monotonous and far away from the quality of present synthesizers.

8) The first reading aid with optical scanner was introduced by Kurzweil in 1976. The Kurzweil Reading Machines for the Blind were capable to read quite well the multifont written text. However, the synthesizer was far too expensive for average customers, but was used in libraries and service centers for visually impaired people [Klatt, 1987].

9) In 1979 Allen, Hunnicutt, and Klatt demonstrated the MITalk laboratory TTS synthesizer developed at M.I.T. [Allen, 1979].

More details on the history of Speech Synthesis, in a chronological order are given in [Klatt, 1987]. Some milestones of speech synthesis development are shown in Fig. 2.7.

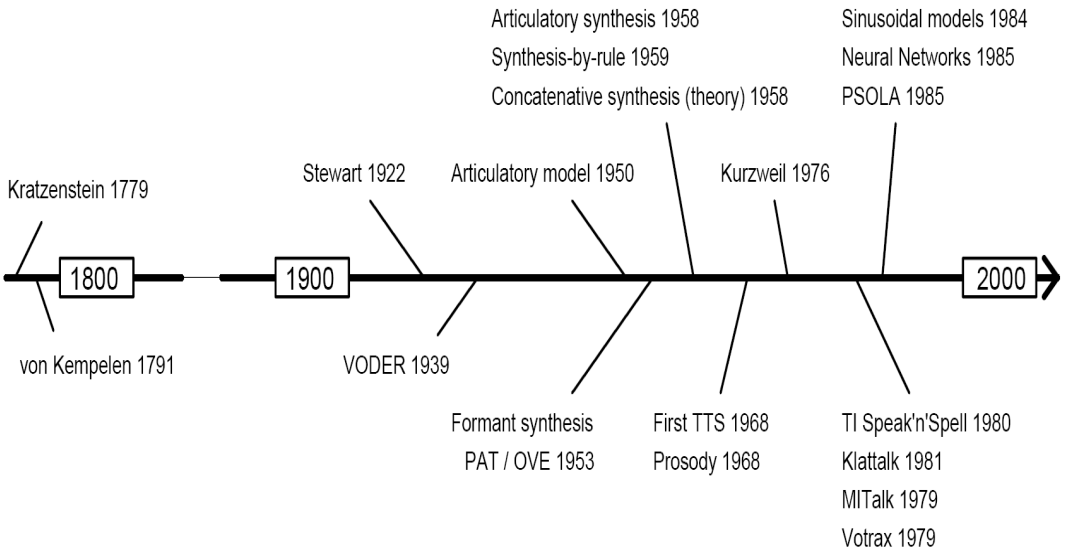


Fig. 2.7: Some milestones in speech synthesis

2.3 SOME GENERIC TTS FRAMEWORKS

This section gives some of the generic frameworks available in public domain for the development of a TTS synthesizer. Some of these act as back-end engines and others are full-featured commercial TTS frameworks.

2.3.1 MBROLA SYNTHESIZER

MBROLA is a high-quality, diphone-based speech synthesizer that is available in public domain. It is provided by the TCTS Lab of the Faculte Polytechnique de Mons (Belgium) which aims to obtain a set of speech synthesizers for as many languages as possible. The MBROLA speech synthesizer is free of charge for non-commercial, non-military applications. Anyone can send in his or her own speech recordings and an MBROLA database for synthesis is prepared. There are presently diphone databases existing for several languages: American English, Brazilian Portuguese, Breton, British English, Dutch, French, German, Greek, Romanian, Spanish and Swedish.

TCTS also provides speech database labeling software: MBROLIGN, a fast MBROLA-based TTS aligner. MBROLIGN can also be used to produce input files for the MBROLA v2.05 speech synthesizer. More information and demos of the different voices and languages and also comparisons between MBROLA and other synthesis methods can be found on the MBROLA project home page [MBROLA, 1998].

2.3.2 FESTIVAL

The Festival TTS synthesizer was developed in CSTR at the University of Edinburgh by Alan Black and Paul Taylor and in co-operation with CHATR, Japan [Black et al., 2001]. It is a freely available complete diphone concatenation and unit selection TTS synthesizer. Festival is the most complete freeware synthesis system and it includes a comprehensive manual. Festival offers a general framework for building speech synthesis systems as well as including examples of various modules. As a whole, it offers full TTS synthesizer through a number of APIs. Festival is multi-lingual (currently English, Spanish and Welsh). The English version is most advanced and the developments for this version are very fast.

The synthesizer is written in C++ and uses the Edinburgh Speech Tools for low-level architecture and has a Scheme (SIOD)-based command interpreter for control. The latest details and a full software distribution of the Festival Speech Synthesis System are available through the Festival home page [Black et al., 2001].

2.3.3 FLITE

Flite (Festival-lite) is a smaller, faster alternative version of Festival designed for embedded systems and high volume servers. More information is available at:
http://www.speech.cs.cmu.edu/flite/

2.4 TOOLS AVAILABLE FOR DEVELOPMENT OF A TTS SYNTHESIZER

The tools available for developing a TTS synthesizer include speech API's provided by different vendors, and different markup languages. There exist many different APIs for speech output but there is a trend towards using the Microsoft API for synthesizers running on Windows. Another API that is not so frequently used is the Sun-Java Speech API. These two are described below.

2.4.1 SUN - JAVA SPEECH API

The Java Speech API is being developed to allow Java applications and applets to incorporate speech technology. The API defines a cross-platform API to support command and control recognizers, dictation systems and speech synthesizers. Java Speech Grammar Format provides a cross-platform control of speech recognizers. Java Speech Markup Language provides a cross-platform control of speech synthesizers.

Text is provided to a speech synthesizer as a Java String object. The Java Platform uses the Unicode character set for all strings. Unicode provides excellent multi-lingual

support and also includes the full International Phonetic Alphabet (IPA), which can be used to accurately define the pronunciations of words and phrases. More information can be found on the Java homepage:
http://java.sun.com/products/java-media/speech/

2.4.2 SAPI MICROSOFT'S SPEECH API

The leading vendors are beginning to support Microsoft's Speech API, or SAPI, which is based on the COM specification and is being adopted as the industry standard. The motive of SAPI is to eventually allow interoperability between the speech engines. The Microsoft Speech API provides applications with the ability to incorporate speech recognition (command & control dictation) or TTS, using either C/C++ or Visual Basic. SAPI follows the OLE Component Object Model (COM) architecture. It is supported by many major speech technology vendors. The major interfaces are:

- Voice Commands: high-level speech recognition API for command and control.
- Voice Text: simple high-level TTS API. The Voice Text object is available in two forms: a standard COM interface IVoiceText and companion interfaces, and also an ActiveX COM object, VtxtAuto.dll
- Multimedia Audio Objects: audio I/O for microphones, headphones, speakers, telephone lines, files etc.

With the Microsoft Speech SDK, and in particular, the TTS VtxtAuto ActiveX COM object, any developer can create a TTS-enabled application using a few simple commands, such as register and speak. More information can be found on the website:
http://msdn.microsoft.com/library/sdkdoc/sapicom/html/intro2sapi.html

2.4.3 MARKUP LANGUAGES

The input to a TTS synthesizer is often a string of words but sometimes it also contains information in the form of markers to indicate emphasis, stress placement, speech rate, voice etc. System providers normally have their own markup codes but there is some co-operation between providers to develop standards for markups.
A number of markup languages have been established for rendition of text as speech in an XML compliant format.

SSML-SSML is the most recent markup language, proposed by the W3C.

JSML-The Java Synthesis Markup Language (JSML), an SGML-based mark-up language, is being specified for formatting text input to speech synthesizers. JSML

allows applications to control important characteristics of the speech produced by a synthesizer. Pronunciations can be specified for words, phrases, acronyms and abbreviations to ensure comprehension. Explicit control of pauses, boundaries and emphasis can be provided to improve naturalness. Explicit control of pitch, speaking rate and loudness at the word and phrase level can be used to improve naturalness and comprehension.

Sable-Sable is consortium aimed at providing a single standard for speech synthesis markup. The consortium's principle aim is to merge the two existing proposed standard, namely SSML developed by Bell Labs and Edinburgh, and JSML, developed by Sun.

2.5 SUMMARY

In this chapter, we have presented the detailed architecture of a TTS along with its major components. For a better understanding of TTS field, a brief history of the important developments in the TTS field is also incorporated in this chapter. In section 2.3, we have listed some of the generic frameworks available for TTS synthesizer development.

In the next chapter, we shall study the work done in the field of TTS for Indian languages by various organizations and educational institutions in India.

Chapter

3

TTS SYNTHESIZERS FOR INDIAN LANGUAGES

3.1 LINGUISTIC STUDIES IN INDIA

India is a country with more than 15 official languages and each language is spoken in different forms across different places. According to experts, Speech recognition and Speech synthesis technology can be very useful in the Indian context as it provides an easy interface for interacting with computers. Using such a convenient means of rendering information to or from the machine would mean that the end-user need not be computer literate and still can use the power of the IT industry.

There are several language problems in India which IT can solve. In this regard Government of India has funded several projects taken up by various institutions all over the country.

From the point of view of TTS development, the most helpful aspect of Indian scripts is that they are basically phonetic in nature, and there is one-to-one correspondence between the written and spoken forms of most of the Indian languages. This makes the task of automatic phonetization simpler [Krishna, 2002].

This chapter is a survey of the work done in the field of TTS for Indian languages. The goal of this survey is to study the techniques used for Indian language TTS development and identify suitable techniques for development of Punjabi TTS synthesizer.

Even among Indian Languages, focus is given more to the work done for Hindi TTS synthesizer, keeping in mind that Hindi text and speech resembles with that of Punjabi text and speech. The different works have been studied from the point of view of following aspects:

- Is the code available?
- Can software be run for demo?
- Is it a framework?

- Input file format (Unicode/text)?
- What is the NLP component?
- What is the DSP component (concatenative/rule based)?
- Synthesis unit (syllable/diphone etc)?
- OS support-Windows or Linux?
- Language support?

Several institutions in India are working in the field of TTS for Indian languages. The work done in the following institutions is discussed in detail.

- C-DAC Bangalore - Matrubhasha API [Raman, 2004]
- IIT Mumbai - Vani Framework
- HP Labs - Hindi TTS
- IIT Kharagpur - SHRUTI TTS
- Simputer Trust - Dhvani TTS [Dhvani, 2001], [Hariharan, 2002]

Some other institutions where the TTS development is going on, include – IIT Madras, IIIT Hyderabad, HCU, IISc Bangalore, Utkal University, TIFR Mumbai, C-DAC Noida and College of Engineering, Guindy.

3.2 C-DAC BANGALORE – MATRUBHASHA API

C-DAC Bangalore (formerly National Centre for Software Technology (NCST)) is an autonomous society, involved in Research & Development, under the administrative purview of Department of Information Technology, Ministry of Communications & Information Technology, and Government of India.

Matrubhasha is a project carried out at C-DAC Bangalore, as a part of the digital divide bridging activities [Raman, 2004]. It was undertaken with the intention of making end user applications speak and listen to the masses in any Indian language that they are comfortable to communicate in.

Matrubhasha is a Unicode and MBROLA based software solution for TTS synthesis and, CMU Sphinx based Speech Recognizer for Indian languages. It is visualized with the objective of building a framework, which can be used by any software developer to incorporate speech capabilities (in Indian languages) into his/her software thus increasing its usability across different sections of society.

The Matrubhasha project is an activity of the ICT Research and Training Centre (India) of Development Gateway Foundation. The Government of India is a member of the Development Gateway Foundation, a World Bank initiative. An Information

and Communication Technologies (ICT) - Research & Training (R&T) Centre of the Foundation has been set up in Bangalore, India with Centre for Development of Advanced Computing (C-DAC) as the Project Implementing Agency and the Indian Institute of Technology (IIT), Bombay as the first collaborating institution.

Matrubhasha provides:

- One single engine and different language bases. By just using different language rule files, the same engine speaks different languages, and thus speaks multiple languages from the same document.
- Tools with a linguist friendly user interface which help the linguists to create language bases for any language and any dialect and create rules in the language base so as to bring out correct pronunciation by imposing those rules during conversion of written text to spoken phonemic form. These tools include Anuvaachak, Uchharak and Bhavna.
- An application programming interface (API) that speech enables any application. This API can also be extended to create plug-ins to general purpose utilities like office products and internet browsers.

3.3 IIT MUMBAI– VANI FRAMEWORK

Vani is a TTS synthesizer proposed by IIT Mumbai. It is primarily developed for Hindi, but with minor modification could directly be modified for any language which is phonetic in nature. The approach is similar to concatenation synthesis, with phonemes as the basic unit. However, the phonemes are not selected from a database, but are generated from a more basic unit, which they call fract-phoneme in the case of vowels. The basic observation is that vowels look like a continuous repetition of these very small segment phonemes called fract-phonemes. Since fract-phonemes are very small in size, they are a good choice for acting as a basic unit.

The aim of Vani was to allow complete specification of speech, i.e., one can get the software to speak exactly what they want to. This means typically that the software can also sing if so desired. Also emotions can be expressed. In order to give complete freedom of expression to the user Vani represents the speech by a new encoding scheme called vTrans which is an extension of iTrans encoding scheme. Vani generates speech from a given vTrans file.

For Schwa deletion algorithms for Hindi, Vani uses a rule-based algorithm [Basu, 2002]. Vani is built using java to enable platform independence and uses Java Sound API (JSAPI).

3.4 HP LABS – HINDI TTS

HP Labs, India have developed a Hindi TTS synthesizer based on Festival framework. This effort is a part of the Local Language Speech Technology Initiative (LLSTI), which facilitates collaboration between motivated groups around the world, by enabling sharing of tools, expertise, support and training for TTS development in local languages. It aims to develop a TTS framework around Festival that will allow for rapid development of TTS synthesizers in any language.

Since Festival does not provide complete language processing support specific to various languages, it needs to be augmented to facilitate the development of TTS synthesizers in certain new languages. Because of this, a generic G2P converter has been developed at HP Labs India as part of the LLSTI initiative. The system consists of a rule processing engine which is language independent. Language specific information is fed into the system in the form of lexicon, rules and mapping.

3.5 IIT KHARAGPUR- SHRUTI TTS

An Indian language TTS synthesizer (named SHRUTI) that accepts text inputs in two Indian languages namely Hindi and Bengali and produces near natural audio output has been developed at IIT Kharagpur [Mukhopadhyay, 2006]. The synthesizer runs on a Compaq iPaq PDA built around the Intel Strong Arm-1110 processor running Microsoft Pocket PC, a customized version of Microsoft's operating system WinCE for mobiles and other handheld devices. The synthesizer has also been ported to a Casio Cassioepeia built around a MIPS 124 processor running Pocket PC. Two versions of the synthesizer have been built, one which resides on the system memory and another which runs from a storage card.

3.6 SIMPUTER TRUST – DHVANI TTS

The Simputer Trust is a registered (1999) non-profit charitable trust with the broad goal of harnessing the potential of Information Technology for the benefit of the weaker sections of society. It has brought together people from two entities: Computer Science and Automation Department of the Indian Institute of Science and Encore Software (formerly Ncore technologies).

Simputer is a low-cost multilingual, mass access handheld device that uses Indian Language User Interfaces to provide Information technology based services to the multilingual population of India [Hariharan, 2002]. It is a low cost portable alternative to PCs, by which the benefits of IT can reach the common man. The Amida Simputer was launched in Bangalore on 26th March 2004 by Picopeta Simputers Pvt. Ltd. It is a retail product.

DHVANI is the TTS effort of the Simputer Trust. The aim of this effort is to ensure that literacy and knowledge of English are not essential for using the Simputer. Using images in conjunction with voice output in local languages makes the Simputer accessible to a larger fraction of the Indian population.

Dhvani has a C/Linux implementation. Currently, it has a phonetic-to-speech engine which is capable of generating intelligible speech from a suitable phonetic description in many Indian languages. In addition, it is capable of converting UTF-8 text in Hindi and Kannada to the phonetic description, and then speaking it out using the phonetic-to-speech engine.

3.7 OTHER INSTITUTIONS

IIT Madras, IIIT Hyderabad, HCU, IISc Bangalore, Utkal University, TIFR Mumbai, C-DAC Noida and College of Engineering, Guindy: These institutes are also working on TTS synthesizers for various Indian Languages. Following is a brief description of the techniques used by these institutions for developing their TTS synthesizers.

3.7.1 IIT MADRAS

Systems Development Laboratory, IIT Madras has come up with a speech enabled multilingual editor (http://acharya.iitm.ac.in) which supports text processing and speech generation for several Indian languages including Telugu. The speech is produced using the MBROLA speech engine. Since the databases required for Indian languages are not yet available for use with MBROLA, the IITM team had experimented with other available data bases where the phonemes are close to the phonemes of Indian languages. In this regard, Swedish database was used for generating the speech for Indian languages as the phonemes of Swedish are well suited to produce speech output in Indian Languages.

The Telecommunication and Computer Networking (TeNeT) Group, a coalition of 14 faculty members from the Electrical Engineering and Computer Science & Engineering Departments of IIT-Madras, has taken initiative for developing local

language speech interface system. In this regard, they have worked with Festival to perform speech synthesis. Their system has following features:

- Common phoneset for Hindi and Telugu, also usable for other Indian language.
- Diphone unit selection for synthesis.
- Data-driven prosody modeling using Classification and Regression Tree (CART).
- Concatenative synthesis technique is used to produce natural sounding speech.

Also, a separate group in Speech and Vision Lab, Department of Computer Science and Engineering, IIT Madras, is working with an objective to address different issues in acquisition and incorporation of duration and intonation knowledge in the context of TTS synthesis for Indian languages.

3.7.2 IIIT HYDERABAD

Language Technologies Research Centre (LTRC), IIIT Hyderabad has as its goal the development of technologies dealing with language. It includes technologies pertaining to translation and other NLP areas, speech processing, optical character recognition, etc. Their research and development focus is to develop speech interfaces for Indian Languages. Currently, they are working on TTS synthesizers for Indian languages. They already have some restricted domain TTS synthesizers developed for Telugu and Hindi working in a number of domains.

LTRC has used data-driven (Corpus - based / Example - based) approach using festvox for the development of TTS synthesizers. They have used Unicode as the input to the front end for the text processing. They experimented with different choices of units: syllable, diphone, phone and half phone, for unit selection speech synthesis for Hindi. The observation they have come up with is that the syllable unit performs better than diphone, phone and half phone, and seems to be a better representation for languages such as Hindi. Also, the half phone synthesizer performs better than diphone and phone synthesizers. They have also proposed a data-driven synthesis method for Indian languages using syllables as basic units for concatenation. Some applications developed by LTRC based on TTS technology are:

- SAMACHAR VANI – a framework for automated spoken news service. It is TTS based newsreader software.

- TALKING TOURIST AID – an application which allows a non native Hindi/Telugu person to express his queries/concern related to city, travel, accommodation etc., in the language of the native speaker. This application is ported to Simputer.
- SCREEN READER -for visually handicapped. A preliminary version of it is developed using Hindi voices.

They are also working on a Hindi speech synthesizer that fits in 1.45 MB flat. Such a small speech synthesizer can be put on small systems like PDAs and embedded devices including mobiles. The only other speech synthesizer available at comparable sizes for Indian languages is the Dhvani speech synthesizer which is more than 2 MB in size. LTRC has also come up with an implementation of an API for the TTS synthesizer in Windows for robust application development in Windows environments.

3.7.3 HYDERABAD CENTRAL UNIVERSITY (HCU) - VAANI

Language Engineering Research Centre (LERC) at Resource Centre for Indian Language Technology Solutions (RCILTS), HCU has developed several products related to speech technologies which include an experimental TTS synthesizer for Telugu -VAANI. Vaani uses MBROLA as a back end. They have used the diphone segmentation approach. The Telugu voice used by them is developed by Jahnavi Ayachitam and Kalpana Reddy KVK of ISS College of Info. Tech. and Engineering for Women, Hyderabad, AP, INDIA. Presently, this is the only Telugu diphone database available for MBROLA synthesizer.

3.7.4 IISC BANGALORE -THIRUKKURAL & VAACHAKA

IISc Bangalore has developed a complete Tamil and Kannada TTS synthesizer named Thirukkural and Vaachaka respectively. Similar techniques are applied in the development of both the TTS synthesizers. Syllables of different lengths have been selected as units. Automatic segmentation algorithm has been devised for segmenting syllables into consonant and vowel. Thirukkural is designed in VC++ and runs on windows 95/98/NT.

3.7.5 UTKAL UNIVERSITY, ORISSA

Speech Tech Group (STG), RC-ILTS-ORIYA Utkal University, Orissa has developed an ORIYA TTS synthesizer. This TTS synthesizer is designed by the Character Based Concatenation technique. In this method, the words are parsed into characters and a

character is segregated into pure consonant and vowel. For example, ‘ka’ is obtained from the word ‘kataka’ and further this is separated as pure consonant ‘k’ and ‘a’. Likewise, all the pure consonants and vowels are stored in the ‘.wav’ format. As the vowels are dominant in the utterance, they are stored for different durations as they occur in the word.

Using the techniques of Artificial Intelligence the synthesis of sound is obtained. Further the transition between two characters is stored by taking the help of Paninian philology to give a natural shape to the output. These rules help to slot in different levels of pitch for incorporating prosody in the output. The classifications on the silence region and the unvoiced regions have also been studied to place them in proper places. Intonation modeling is also incorporated in the synthesis system.

3.7.6 TATA INSTITUTE OF FUNDAMENTAL RESEARCH (TIFR), MUMBAI

Work has been done in TIFR, Mumbai for TTS synthesis in Indian English. The important components of the language processor used are the parser to categorize words, an Indian English phonetic dictionary, morphological analyzer, letter-to-sound rules, phonological rules, prosody rules and Indian name detector. The relevant rules are formulated with the aid of a large CMU pronunciation dictionary and a language tool GENEX, developed in TIFR, which can generate a sub-dictionary following a set of specified constraints.

3.7.7 C-DAC, NOIDA

The TTS project at C-DAC, Noida, is based on the concatenative approach and aims at developing a TTS synthesizer for Indian languages primarily, Hindi. The input in Unicode is processed by the Text processing unit and the speech-processing unit resulting in synthesized speech. The input text is also normalized and converted to their orthographic form, before breaking into the basic units required for synthesis.

3.7.8 COLLEGE OF ENGINEERING, GUINDY, CHENNAI

A preliminary version of TTS synthesizer using concatenative synthesis and diphones is developed for Tamil. They are still working on improving the quality of the speech.

3.8 SUMMARY

In this chapter, we have presented an exhaustive survey of the work done in the field of TTS for Indian languages. The goal of this survey is to study the techniques used for Indian Language TTS development. A lot of work is done and a lot of work is in progress in the field of TTS for Indian languages. Most of the TTS synthesizers for

Indian languages were able to generate intelligent speech; however getting speech close to natural speech is still a challenge. The Hindi and Kannada TTS synthesizer, DHVANI developed by Simputer Trust, India was found to be producing a very high quality speech which was quite close to natural speech.

It has been found during the literature survey that there is not any TTS synthesizer available for the Punjabi language. After the study of various TTS works by Indian institutions, attempts were made to develop a TTS synthesizer for the Punjabi language. The ground work for this is presented in chapter 4.

In chapter 4, we shall discuss the implementation details of the Punjabi TTS synthesizer using an existing TTS synthesizer named DHVANI (available for Hindi and Kannada languages) developed by Simputer Trust. The Hindi language module of DHVANI has been used as a model for developing Punjabi TTS synthesizer as Hindi text and speech resembles with that of Punjabi text and speech.

Chapter 4

DEVELOPMENT AND IMPLEMENTATION OF PUNJABI TTS SYNTHESIZER

This chapter provides the implementation details of Punjabi TTS synthesizer. For the development of Punjabi TTS synthesizer, two existing TTS synthesizers have been studied in detail. These synthesizers are Dhvani (for Hindi and Kannada language) and an Arabic TTS synthesizer. Section 4.1 gives a brief outline of the Arabic TTS synthesizer and section 4.2 gives the architecture of Dhvani. The next section gives the implementation details of the Punjabi TTS synthesizer using Dhvani.

4.1 ARABIC TTS

The Arabic TTS synthesizer studied by us has been developed at Kuwait Scientific center, Kuwait University. It uses International Phonetic Alphabet (IPA) notation to convert raw text entered by the user into phonetic notation. After conversion into phonemes, it uses a homogeneous set of prerecorded sub-syllabic synthesis units to generate tokens. It contains 1372 sub-syllabic synthesis units that consist of open syllables, vowel-consonant clusters, vowel-consonant-vowel clusters, and long duration consonants [Banat, 1990], [El-Imam, 1987], [El-Imam, 1989]. The synthesizer component concatenates these synthesis units to produce the final speech. For developing Punjabi TTS synthesizer, we have used similar synthesis units consisting of CV, VC, CVC clusters along with vowels (V) and consonants (0C). The Punjabi TTS synthesizer presented in section 4.3 also uses the IPA notation similar to Arabic TTS synthesizer for performing the grapheme-to-phoneme conversion. Section 4.3 gives more details of the synthesis units and other modules of Punjabi TTS.

4.2 SIMPUTER TRUST-DHVANI TTS

The Simputer Trust has developed an Indian language TTS synthesizer, Dhvani for Hindi and Kannada Language. Dhvani has a C/Linux implementation. Currently, it has a phonetic-to-speech engine which is capable of generating intelligible speech

from a suitable phonetic description in many Indian languages. In addition, it is capable of converting UTF-8 text in Hindi and Kannada to this phonetic description, and then speaking it out using the phonetic-to-speech engine. The architecture of the Dhvani TTS synthesizer is shown in Fig. 4.1 [Dhvani, 2001], [Hariharan, 2002].

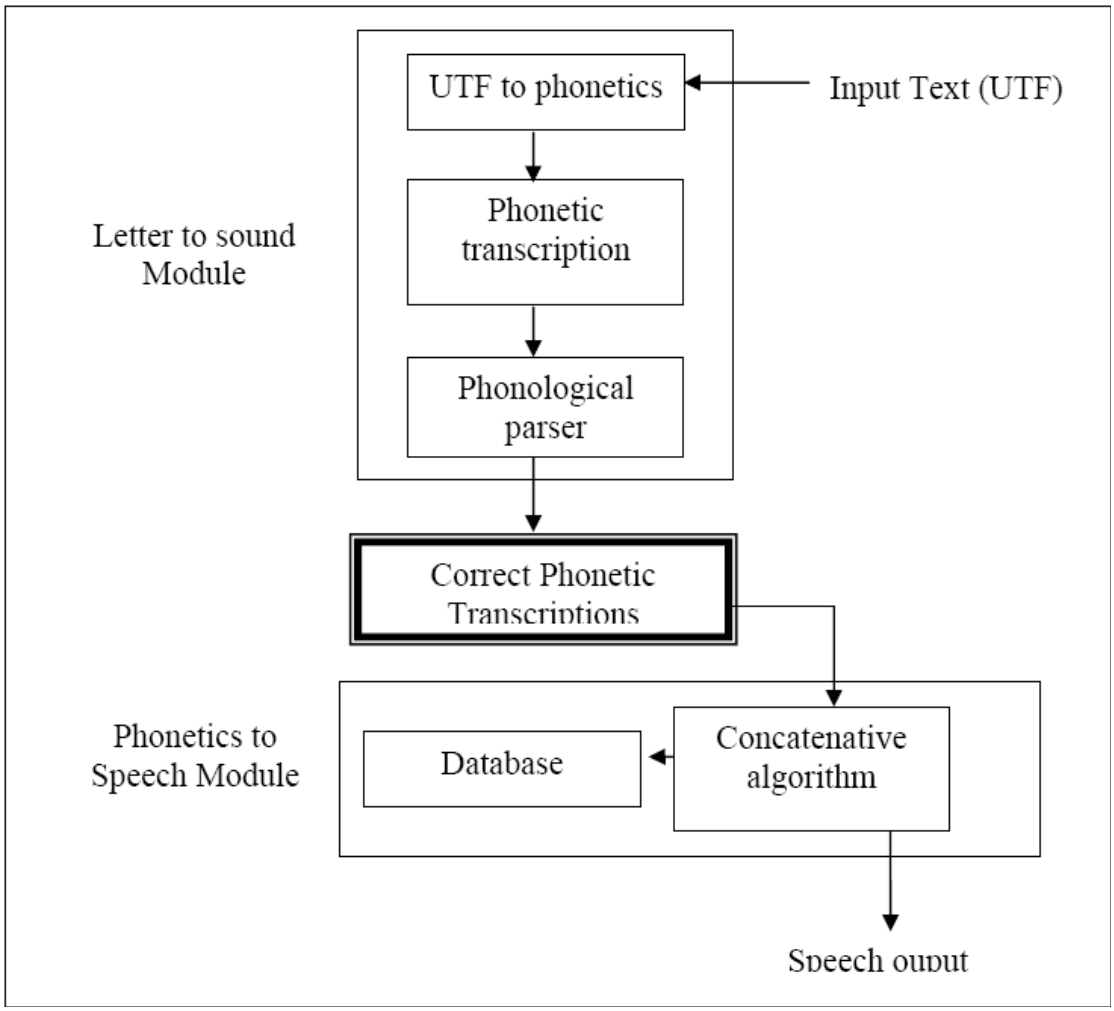


Fig. 4.1: The architecture of Dhvani TTS synthesizer

The Phonetics-to-Speech Engine works by diphone-concatenation. It uses a database of about 800 basic sounds, which are pitch-marked. The engine reads the phonetic description, identifies the appropriate diphones, concatenates them at pitch-marks, and plays out the resulting signal. To reduce the database size, an open-source implementation of the GSM 06.10 RPELTP compression standard by The Communications and Operating Systems Research Group (KBS) at the Technische Universitaet Berlin, was used. This reduces the database size to about 1MB (even though it appears to be 2MB due to fragmentation) which is about a factor of 10 compression. All basic sounds are recorded at 16000Hz as 16 bit samples.

The Text-to-Phonetics routine for Hindi/Kannada reads in a text in UTF-8 format and converts this text into the phonetic description. Currently, only Hindi and Kannada Text-to-phonetics are provided.

The Dhvani TTS synthesizer was tested and found to be fairly intelligent. However, it lacked naturalness. This is because the speech engine doesn't do prosody on the output. It simply concatenates basic sound units at pitch periods and plays them out. Moreover, a syllable based phonetic transcription is used which is quite cumbersome.

4.3 PUNJABI TTS SYNTHESIZER AS AN EXTENSION OF DHVANI TTS ENGINE

For developing the TTS synthesizer for Punjabi language, we have used Dhvani as a model. Dhvani is a prototype of a phonetic-to-speech engine which can serve as a back-end for speech synthesizers in many other Indian languages. We have to associate it with the corresponding language-specific text-to-phonetics module.

Dhvani accepts text in UTF-8 format only as input to it. As such, we need to convert our raw text to UTF-8 format. For this purpose, we have used an existing utility called SC Unipad (Scipad) but unfortunately the problem which arises is that of integration of Dhvani TTS with this utility as Dhvani is Linux-based and the above mentioned utility is Windows-based. For speaking the Punjabi UTF-8 file, we can use the Hindi prosodic rules with some modifications with the assumption that the Hindi text and speech resembles with that of Punjabi text and speech. The synthesizer consists of three components, two of which act as server units and the third one being a client sending input to them. The servers are named primarily as "synthserv" (for synthesizing speech units) and "hindiphonserv" (for generating the phonemic units from the raw text sent by the client as a file encoded in UTF-8 format). The "hindiphonserv" server after converting the raw text into phonemic notation sends it to "synthserv" server for speech generation. Briefly, the TTS engine works as follows.

4.3.1 HINDIPHONSERV UNIT

This is the server used for generating the phonetic notation from the UTF-8 input sent by the client program. It replaces the input UTF text to the corresponding phonetic symbols in the database by careful mapping the UTF symbols for Punjabi onto the phonetic symbols in the database. Simultaneously, each symbol is tagged as a consonant(C) or Vowel (V). It denotes Consonant as one and Vowel as zero. Then, it parses the phonetic strings thus obtained to produce speakable tokens. The main

challenge lies in a peculiarity of the Punjabi language - the occasional presence of special characters like tippi, adhak etc. After identifying the tokens and generating the phonetic notation it acts as a client for the “synthserv” unit, and gives input to it.

The major work that has been carried out in the present thesis has been to modify this “hindiphonserv” program to migrate the Hindi language rules to Punjabi language rules and adding many new rules specific to only Punjabi language such as rules for tippi, etc. Some of the rules which were specifically designed for Hindi have also been excluded.

4.3.2 SYNTHSERV UNIT

This is the server unit used to synthesize speech units from the phonetic notation and speak out appropriate sounds from the database. This is the unit where prosody rules can be applied to enhance the quality of speech and to add emotions etc. to the synthesized speech.

4.3.3 CLIENT UNIT

The client unit is responsible for creating socket connections with the server units and sending the UTF-8 Punjabi text to them as input. Fig. 4.2 shows the flow of information in the Punjabi speech synthesizer.

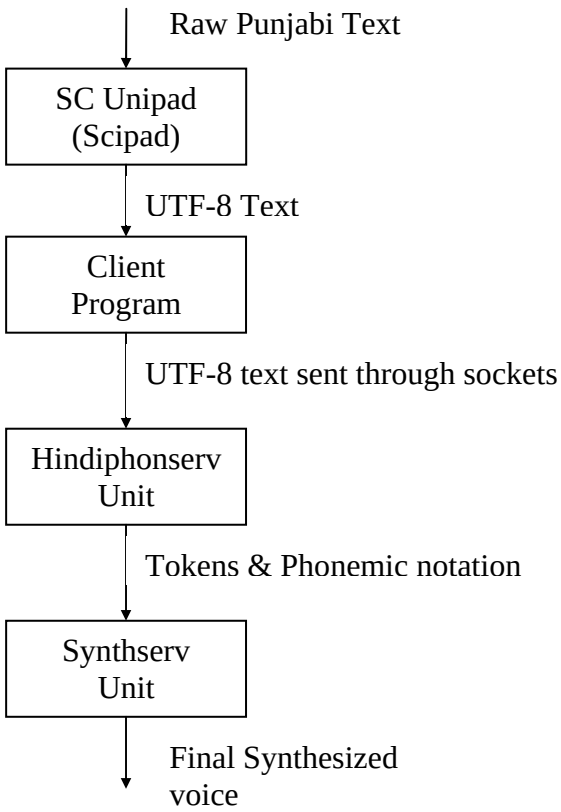


Fig. 4.2: Information flow in Punjabi TTS synthesizer

The main data structures used in the synthesizer are:

struct code {int type; unsigned short alpha; unsigned char beta;};

struct wordtemplate{int type; char * letter;};

char * word[100]

A word is assumed to be not more than 100 letters. ‘char* word’ stores letters of one word.

‘type’ is for distinguishing between ASCII and Unicode letters.

Structure ‘code’ is used to pass the characters between different modules.

The main modules used in “hindiphonserv” program and their flow is as follows:

1) The function readfile() uses getnext() function repeatedly to identify tokens in Punjabi text. A token is delimited by either space/comma/tab/carriage-return etc. and must be in Punjabi. Once a token is identified, it is processed and synthesized using “synthserv” program. At each delimiter, a G1500 voice is also synthesized which stands for a gap or delay of 1500 milliseconds.
2) The function getnext() gets the next “character” from the UTF-8 encoded file. Here, the modification is made to take input as Punjabi UTF file instead of Hindi file.
3) After reading the tokens, the synthesizer reads the first character of each token, if it is between 0-9, then the synthesizer sends the token to replacenum() function and then to parsenum() function to generate phonemic units for numeric input. Otherwise, the synthesizer calls replace() function and the parseword() function and these functions replace the input characters by the corresponding phonetic strings depending upon the International Phonetic Alphabet (IPA) and synthesize them accordingly.
4) The replacenum() function takes as input a Unicode array of numbers, assigns them the corresponding phonetic string and returns the size of the array i.e. number of digits in the corresponding number.
5) The parsenum() function uses the results of replacenum() to decide what should be the pronunciation of that number depending upon number of digits in the number.

4.3.4 PHONETIC DESCRIPTION OF THE INPUT

The phonetic description of the input is syllable-based. Following kinds of sounds are allowed (C stands for consonant, V for Vowel). The text to be spoken out must be expressed in terms of these sound units.

V: a plain vowel

CV: a consonant followed by a vowel

VC: a vowel followed by a consonant

CVC: a consonant followed by a vowel followed by a consonant.

0C: a consonant alone

G [0-9]: a silence gap of the specified length (typical gaps between words would be between G1500 and G3000 depending upon the speed required; maximum value that is allowed is G15000; larger gaps can be given by repeating G15000 as many times as required).

4.3.5 THE STRUCTURE OF THE SOUND DATABASE

The database has the following structure. All sound files stored in the database are gsm compressed .gsm files recorded at 16 KHz as 16bit signed linear samples. The total size of the database is currently around 1MB, though we can possibly work to get it down to about half the size by storing only parts of vowels and extending them on the fly. We are using gsm compression which gives about a factor of 10 compressions. The sound database has the following architecture:

CV files are named x.y.gsm where x is the consonant number and y is the vowel number.

VC files are named x.y.gsm where x is the vowel number and y is the consonant number.

V files are named x.gsm where x is the vowel number.

0C files are named x.gsm where x is the consonant number.

CV files are in the cv/ directory within database/

VC files are in the vc/ directory within database/

V files are in the v/ directory within database/

0C files are in the c/ directory within database/

All vowels and consonants are assigned particular numbers to identify them.

Vowels allowed are:

1) a as is pun
2) aa as in the Punjabi word saal (meaning year)
3) i as in pin
4) ii as in keen
5) u as in pull
6) uu as in pool

7) e as in met

8) ee as in meet

9) ae as in mate

10) ai as in height

11) o as in the gold

12) oo as in court

13) au as in call

14) ow as in cow

The phonetic description uses the numbers 1-14 instead of the pnemonics given above.

Consonants allowed are:

ka kha ga gha

nga ca cha ja jha nya

ta tha da dha na

tta ttha dda ddha nna

pa fa ba bha ma ya ra la lla va sha sa ha

khha ghha za rra fa

These consonants are sequentially given the numbers starting from 1 to 38. The phonetic description however uses the pnemonics above. Within the program and in the database nomenclature, the numbers are used.

All files other than the 0C files have been pitch marked and the marks appear in the corresponding .marks files, one mark per byte as an unsigned char.

In addition to the sound files, there are three files in database/, namely cvoffsets, vcoffsets and voffsets, which store various attributes of the sound files.

4.3.6 PROGRAMS

To play a demo Punjabi file in UTF-8 format, we need to invoke the server versions of the back end speech engine and the Punjabi UTF to phonetics converter.

To compile these programs, one should use the following commands:

```
>gcc -I. -g synthserv.c -lm ../gsm/lib/*.o -o synthserv
>gcc -I. -g hindiphonserv.c -lm ../gsm/lib/*.o -o hindiphonserv
>gcc -I. -g client.c -lm ../gsm/lib/*.o -o client
```

Now, one has to start the server programs. After that, one should invoke the client to send the demo file as input using the following commands.

```
>./synthserv
```

>./hindiphonserv

>./client /dhvani/demos/utffile, for Punjabi UTF-8 demo files (examples of demo files are stored in ../dhvani/demos)

4.4 SUMMARY

A study of the Arabic speech synthesis system using sub-syllabic synthesis units for generating Arabic speech, study of Dhvani TTS synthesizer for Hindi and Kannada (also using sub-syllabic synthesis units for speech generation), enabled us to decide appropriate synthesis method for Punjabi TTS synthesizer as explained in this chapter. All the stages of the TTS conversion process have been investigated, and rules governing the production of phonemes from input text, conversion of the phonemes to a phonetic string, generation of the synthesis units from the phonetic string, and concatenation of the synthesis units, are developed as algorithms suited to the computer handling of the speech synthesis for Punjabi language. When tested with Punjabi UTF-8 sample file using the modified NLP component of Hindi, the Punjabi speech synthesizer gave speech which was quite intelligible.

Chapter

5

EVALUATION OF PUNJABI TTS SYNTHESIZER

5.1 INTRODUCTION

Evaluation of TTS synthesizers is a very difficult problem. After years of discussion and research in the academic community, there is still no convincing solution. As a result, assessment contains the ideas that are subjective to a great extent.

The main contributing factors that may affect the overall quality of a TTS synthesizer are given below [ScanSoft, 2004].

- **Intelligibility –** Intelligibility answers the questions such as, how much of the spoken output can a user understand? How quickly does the user become fatigued as a listener, etc?
- **Naturalness –** Naturalness means how close to human speech does the output of the TTS synthesizer sound?
- **Front-end processing –** The ability of the synthesizer to deal intelligently with commonly used challenges in text such as abbreviations, numerical sequences and so on, from the part of front-end processing.

This chapter presents the results of some of the evaluation techniques applied to test the quality of the Punjabi TTS synthesizer developed in the present work. Different listeners respond differently to the elements of the speech; some are most influenced by intelligibility; some by how human the signal sounds; some by how well the text analysis component performs text analysis. Intelligibility is a must for any synthesis system output. For a visually impaired user of a reading machine, intelligibility of speech plays more important role than naturalness. However, the importance of naturalness cannot be underestimated. A robotic speech in the long run leads to annoying affect. In this chapter, the evaluation test for intelligibility and the naturalness testing are presented. Same subject number in different tables used in the

following sections may not refer to same person. However, different subjects are distinguished by giving different numbers within a single table.

5.2 TESTS FOR INTELLIGIBILITY

Intelligibility tests are concerned with the ability to identify what was spoken or synthesized. They are less concerned with the naturalness of the signal, although naturalness is indirectly related to, and influences, intelligibility. Comprehension tasks, phonetic tasks and transcription tasks have been performed to check the intelligibility of the synthesizer.

5.2.1 COMPREHENSION TASKS

Often listeners of TTS output follow what is spoken but do not comprehend it. The following test is used to check whether the listener has comprehended what was heard. The better the intelligibility of the TTS output, the better would be the comprehension. A passage of TTS output was played to the subjects and they were asked five questions from the passage read. Based on how many questions the subjects could answer correctly, we get an idea of the intelligibility of the TTS synthesizer. Table 5.1 shows the results of the above test applied to the Punjabi TTS synthesizer developed.

Table 5.1: Results of Comprehension Task

Subject	**Number of Correct Answers (Out of 5 Questions)**
1	3
2	5
3	4
4	5
5	4
Average	**4.2**

5.2.2 PHONETIC TASKS

Phonetic tasks deal with identifying the precise sounds within a specific word that are synthesized. They tend not to deal with intelligibility of whole words or phrases or sentences. Following tests have been conducted on the TTS synthesizer for Punjabi language to decide the phonetic quality of the synthesizer.

5.2.2.1 DIAGNOSTIC RHYME TEST (DRT)

DRT is a test of the intelligibility of word-initial consonants. Pairs of words with a different first consonant are played to subjects and asked to identify which word they heard. Depending upon the error rate, we can have an idea of the intelligibility of the TTS synthesizer under test.

The subjects were asked to identify five test pairs read by the synthesizer. The following five test pairs were used for testing Punjabi TTS synthesizer.

kmlj	qmlj
dkan	mkan
varl	barl
mwa	kwa
#da	jada

Table 5.2 shows the results of DRT applied for these five test pairs.

Table 5.2: Results of DRT

Subject	**Number of Correctly Identified Pairs (Out of 5 word pairs)**
1	3
2	4
3	3.5
4	4
5	3
Average	**3.5**

5.2.2.2 MODIFIED RHYME TEST (MRT)

MRT is like DRT, but includes tests for word-final intelligibility. The subjects were asked to identify five test pairs read by the synthesizer. The following five test pairs were used for testing Punjabi TTS synthesizer.

badr	badl
kapl	kaFl
pqa	pta
mr	mP
klqa	klPa

Table 5.3 shows the results of MRT applied for these five test pairs.

Table 5.3: Results of MRT

Subject	**Number of Correctly Identified Pairs (Out of 5 word pairs)**
1	1
2	1
3	2.5
4	2
5	2.5
Average	**1.8**

5.2.3 TRANSCRIPTION TASKS

Semantically Unpredictable Sentence Test (SUS) – In this test, subjects are asked to transcribe sentences that have no inherent meaning or context, and therefore minimize the possibility of deriving phonetic information from any source but the speech signal itself, e.g., the following Punjabi sentence.

Drvaja kl bhq hsNa chda h \

Here, subjects were given a sentence of six words and were asked to identify the words within the sentence. Table 5.4 shows the number of words identified from the sentence by the subjects.

Table 5.4: Results of SUS Test

Subject	**Number of Correctly Identified Words (Out of 6 words)**
1	4
2	5
3	6
4	6
5	5
Average	**5.2**

5.3 TESTS FOR NATURALNESS

5.3.1 MEAN OPINION SCORE (MOS)

MOS is a well-known subjective scoring method. Listeners were asked to rate the naturalness of the synthesis on a scale of one to five, where one is very poor and five is completely natural. All results were summed, and a mean score between one and five was derived, which was meant to represent the overall naturalness rating for the synthesizer. The results revealed that the naturalness of the synthesized speech is quite satisfactory. These results are depicted in Table 5.5.

Table 5.5: Results of MOS

Subject	**Rating by the Subject (Plain Text as Input)**	**Rating by the Subject (Numeric Input)**
1	2	4
2	3	5
3	3	4
4	3.5	4
5	4	5
Average	**3.1**	**4.4**

5.4 SUMMARY

In this chapter, the evaluation test for intelligibility and the naturalness testing of the Punjabi TTS synthesizer are presented. Testing the TTS helped in understanding how the variation in input text can affect the intelligibility and naturalness of the synthesized speech. The tests reveal that the synthesizer performs quite satisfactory in all the cases except in the case of MRT where an average of 1.8 depicts poor performance. There are occasional abruptnesses for some words in the speech output of Punjabi TTS synthesizer. This does cause annoying affect and makes the speech difficult to understand at times.

Chapter

6

CONCLUSION AND FUTURE WORK

This chapter describes various limitations that persist in the Punjabi TTS synthesizer developed. Section 6.2 gives a brief conclusion of the thesis and finally future directions to the current system are presented in the next section.

6.1 LIMITATIONS OF PUNJABI TTS SYNTHESIZER

Although the Punjabi speech synthesizer developed works well for all types of inputs, still some problems persist in it. These are as follows:

- The speech output appears rather slow. The reason for this is that basic sounds have been recorded individually instead of being recorded as parts of words.
- The input module is not integrated with the rest of the synthesis system.
- Dynamic speed and pitch modification are not yet incorporated.
- The synthesizer produces occasional abruptness in case of some words.
- Consonant-Consonant junctions are currently not very satisfactory; either there are large gaps, or sounds get gobbled up.
- Multiple voices need to be added. A major challenge here is to automate the process of going from recording to setting up the database. The main obstacle here is that of automatic segmentation of the recording.
- The speech engine does not make any attempt to do prosody effect on the output. It simply concatenates basic synthesis units at pitch periods and plays them out.
- The phonetic description used is cumbersome.

6.2 CONCLUSION

The current work has been aimed at developing a TTS synthesizer for Punjabi language. As a first step towards this end, a study of the different architectures for different TTS synthesizers for Indian languages was done. This study helped in deciding that Dhvani TTS synthesizer for Hindi language can be a suitable choice for

extending it to support Punjabi language. The entire development has been carried out on Fedora Core 3. The implementation details in this context are explained in chapter 4. The Punjabi TTS synthesizer was tested on sample data and evaluated for the quality of the speech generated. The test results showed that the synthesized speech from Punjabi TTS synthesizer is quite satisfactory.

6.3 FUTURE ENHANCEMENTS

- The integration of the input module (SC Unipad utility available only for windows at the moment) with the rest of the synthesis system is a major task to be done in future. The system then can be extended to take directly the Punjabi text as input.
- The speech synthesizer needs to be migrated to other platforms. At present the synthesizer is only available for Linux platform.
- The speech synthesizer can be integrated with Punjabi OCR output to read handwritten text also.
- The synthesizer is currently not making any prosody attempt to synthesized speech. This is also a major area where work is to be done so that voice quality can be improved.
- Though the current TTS synthesizer is using the concatenative-based synthesis, articulatory-based techniques can also be used to develop TTS synthesizer for Punjabi language because of its phonetic nature.
- The facility to stop, pause and continue reading can be provided, i.e., the user should be able to pause the synthesizer at any time and then continue reading using just a mouse-click.
- Accent variation and multiple voices (both male and female versions) can be provided to users to choose depending upon their interest.

REFERENCES

1. [Allen, 1976] Allen J (1976) Synthesis of speech from unrestricted text. IEEE Journal, Vol.64, Issue 4, pp 432-42

2. [Allen, 1987] Allen J, Hunnicutt S, Klatt D (1987) From text-to-speech: the MITalk system**.** Cambridge University Press, Inc.

3. [Banat, 1990] Banat Karima (Kuwait University), El-Imam Yousif A (IBM Kuwait Scientific Center) (1990) Text-to-speech conversion on a personal computer. IEEE Micro, Vol. 10, Issue 4, pp 62-74

4. [Black et al., 2001] User Manual for the Festival Speech Synthesis System, version 1.4.3 http://fife.speech.cs.cmu.edu/festival/cstr/festival/1.4.3/

5. [Black et al., 2001] Black A, Taylor P, Caley R (2001) The Festival speech synthesis system: system documentation. University of Edinburgh http://www.cstr.ed.ac.uk/projects/festival/

6. [Basu, 2002] Basu A, Choudhury M (2002) A rule based schwa deletion algorithm for Hindi. In: Proceedings of international conference on knowledge-based computer systems, IKON 2002, December, pp 343-53

7. [Dhvani, 2001] Dhvani-TTS system for Indian Languages http://dhvani.sourceforge.net

8. [Dutoit, 1996] Dutoit T (1996) An Introduction to Text-to-Speech Synthesis, First edition, Kluwer Academic Publishers

9. [Dutoit, 1996] Dutoit T (1996) High-quality text-to-speech synthesis: an overview. Journal of Electrical & Electronics Engineering, Australia: Special Issue on Speech Recognition and Synthesis, vol. 17, pp 25-37

10. [El-Imam, 1987] El-Imam Y A (1987) A personal computer-based speech analysis and synthesis system. IEEE Micro, Vol. 7, No. 3, June, pp 4-21

11. [El-Imam, 1987] El-Imam Y A (1987) Speech synthesis by concatenating sub-syllabic sound units. In: Proceedings of ICASSP, Vol. 4, IEEE Press, pp 2416-17

12. [El-Imam, 1989] El-Imam Y A (1989) An unrestricted vocabulary Arabic speech synthesis system. In: IEEE transactions on acoustics, speech, and signal processing, Vol. 37, No. 12, December, pp 1829-45

13. [Hariharan, 2002] Hariharan Ramesh, Bali Kalika, Manohar Swami, Vinay V, Vivek K (2002) Language technology solutions in Simputer: an overview. In:

Proceedings of the Language Engineering Conference (LEC'02), IEEE Computer Society, pp 189-96

14. [Kishore, 2002] Kishore S P, Kumar Rohit, Sangal Rajeev (2002) A data-driven synthesis approach for Indian languages using syllable as basic unit. In: Proceedings of International Conference on Natural Language Processing, ICON-2002, Mumbai, December 18-21, pp 311-16

15. [Klatt, 1987] Klatt D (1987) Review of text-to-speech conversion for English. Journal of the Acoustical Society of America, vol. 82, pp 737-93

16. [Krishna, 2002] Krishna N Sridhar, Murthy Hema A, Gonsalves Timothy A (2002) Text-to-speech in Indian languages. In: Proceedings of International Conference on Natural Language Processing, ICON-2002, Mumbai, December 18-21, pp 317-26.

17. [MBROLA, 1998] MBROLA project homepage (1998) http://tcts.fpms.ac.be/synthesis/mbrola.html

18. [Mukhopadhyay, 2006] Mukhopadhyay Arijit, Chakraborty Soumen, Choudhury Monojit, Lahiri Anirban, Dey Soumyajit, Basu Anupam (2006) Shruti-an embedded text-to-speech system for Indian languages. In: IEE proceedings-software, vol.153, issue: 2, pp 75-79

19. [Rao, 2005] Rao M Nageshwara, Thomas S, Nagarajan T, Murthy Hema A (2005) Text-to-Speech Synthesis using syllable-like units. In: Proceedings of NCC-2005, pp 277-80

20. [Raman, 2004] Raman R K V S, Sarma Sireesha, Sridevi S, Thomas Rekha (2004) Matrubhasha - An integrated speech framework for Indian languages. http://www.ncb.ernet.in/matrubhasha/

21. [ScanSoft, 2004] Assessing Text-to-Speech System Quality. SpeechWorks solutions from ScanSoft, www.ScanSoft.com

PAPER ACCEPTED/COMMUNICATED

Pardeep Gera, Dr. R.K. Sharma, “Speech synthesis for Punjabi language using syllable-like units”, National Conference on Recent Trends in Information Systems (ReTIS-06), Jadavpur University, Calcutta to be held on 14-15 July,2006 [**COMMUNICATED**].

Appendix

A

GLOSSARY OF TERMS

Articulation- It is the approach or contact of two speech organs, such as the tip of the tongue and the upper teeth.

Consonant- A consonant is any articulation that is acoustically produced by forming a constriction by one or more articulators along the vocal tract to the flow of air from the lungs. A long-duration consonant is a syllable closing sound, while a short-duration consonant occurs between two vowels.

Diphone- A diphone is a sound that starts from the middle of one phoneme and ends in the middle of a neighboring phoneme.

Formant- A formant is the resonating frequency of the air in the vocal tract.

Intonation- Intonation is the pattern of fundamental frequency changes that occur during a phrase or a sentence.

Linguistics- The scientific study of language, which may be undertaken from many aspects, for ex: - structure of words (morphology), meanings (semantics) etc.

Prosody- The study of speech-rhythms.

Phoneme- The smallest unit of sound in a language. It is an abstract unit that represents sounds and writing in a systematic, unambiguous way.

Phonetic Transcription (Phonetic Notation) - It is the visual system of symbolization of the sounds occurring in spoken human language. The most common type of phonetic transcription uses an International Phonetic Alphabet.

With phonetic transcriptions, dictionaries tell you about the pronunciation of words. Phonetic transcription is necessary, because the spelling of a word doesn't tell you how to pronounce it.

Phonetics- Phonetics is the representation of the sounds of a language.

Phonology-It is the description of the systems and patterns of sounds that occur in a language.

Syllable- A syllable is a word or a part of a word that is uttered by a single effort of the voice. A syllable is structured from phonemes that are consonants and vowels. A vowel is usually the syllable nucleus while a consonant usually represents the syllable margin.

Synthesis unit- A synthesis unit is an acoustical entity that represents a complete or partial sound. These units join to form syllables, words, or phrases.

Vocal Tract- The vocal tract is the air passage above the larynx. It consists of the oral and nasal tracts.

Vowel- A vowel consists of a single vocal sound that is produced by a continuous passage of air from the lungs on an open vocal tract. This definition applies to both short and long vowels.