DESIGN AND DEVELOPMENT OF DATABASE AND SEARCH ALGORITHM FOR CONCATENATIVE SYNTHESIS OF HINDI SPEECH

A DISSERTATION

Submitted in partial fulfilment of the
requirements for the award of the degree
of
MASTER OF TECHNOLOGY
in
INFORMATION TECHNOLOGY

By

V.S. SRIDHAR

ER & DCI
NOIDA

IIT Roorkee-ER&DCI, Noida
C-56/1, "Anusandhan Bhawan"
Sector 62, Noida-201 307
FEBRUARY, 2003

Enrolment No. 019052

CANDIDATE'S DECLARATION

This is to certify that the work, which is being presented in this dissertation, entitled "**DESIGN AND DEVELOPMENT OF DATABASE AND SEARCH ALGORITHM FOR CONCATENATIVE SYNTHESIS OF HINDI SPEECH**", in partial fulfillment of the requirements for the award of the degree of **Master of Technology** in **Information Technology** submitted in **IIT, Roorkee – ER&DCI Campus, Noida,** is an authentic record of my own work carried out from August 2002 to February 2003, under the supervision of Dr. S.S. Aggarwal, Emeritus Scientist, CSIO New Delhi.

I have not submitted the matter embodied in this dissertation for the award of any other degree.

Date: 24/02/03

Place: Noida

Sridhar

(V.S.Sridhar)

CERTIFICATE

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

Date: 24/02/03

Place: Noida

(Dr S.S. Aggarwal)
Emeritus Scientist,
CSIO, New Delhi.

ACKNOWLEDGEMENT

I hereby take the privilege to express my deepest sense of gratitude to **Dr. Prem Vrat**, Director, Indian Institute of Technology, Roorkee, and **Mr. R.K. Verma,** Executive Director, ER&DCI, Noida for providing me with this valuable opportunity to carry out this work. I am also very grateful to **Dr. A.K. Awasthi,** our Programme Coordinator and Dr. R.P. **Agrawal,** our course coordinator for providing the best of the facilities for the completion of this work and constant encouragement towards the goal.

I have no words to thank my guide, **Dr S.S. Aggarwal,** Emeritus Scientist , CSIO New Delhi for his guidance and invaluable suggestions during the entire course of this work.

I am highly indebted to **Mrs. Sunita Arora**, Senior Project Engineer, Natural Language Processing Laboratory, ER&DCI, Noida, for without their constant support this work could not have been completed. My sincere thanks are due to **Dr. P.R. Gupta** for the continuous inspiration and support she provided me with throughout the course of this dissertation. I am also grateful to **Mr. Munish Kumar** for the cooperation extended by him in the successful completion of this work.

It is impossible to mention the names of all those persons who have been involved, directly or indirectly, with this work and I extend my gratitude to all of them. However, I feel, I owe special thanks to all my friends who have helped me formulate my ideas and have been a constant support. I find myself short of words to thank my father, mother and sister who have always been by my side throughout my life.

Sridhar
(V.S.Sridhar)
Enrolment No. 019052

CONTENTS

ABSTRACT

Synthetic or artificial speech has been developed steadily during the last decades. Especially, the intelligibility has reached an adequate level for most applications, especially for communication-impaired people. The intelligibility of synthetic speech may also be increased considerably with visual information. The objective of this work is to map the current situation of speech synthesis technology. Speech synthesis may be categorized as restricted (messaging) and unrestricted (text-to-speech) synthesis. The first one is suitable for announcing and information systems while the latter is needed for example in applications for the visually impaired. The text-to-speech procedure consists of two main phases, usually called high- and low-level synthesis. In high-level synthesis the input text is converted into such form that the low-level synthesizer can produce the output speech. The three basic methods for low-level synthesis are the formant, concatenative, and articulatory synthesis. The formant synthesis is based on the modeling of the resonances in the vocal tract and is perhaps the most commonly used during last decades. However, the concatenative synthesis, which is based on playing prerecorded samples from natural speech, is becoming more popular. In theory, the most accurate method is articulatory synthesis, which models the human speech production system directly, but it is also the most difficult approach. Here an attempt is made to develop a Hindi text to speech synthesizer with minimal errors by using concatenation approach.

Chapter 1

INTRODUCTION

1.1 Overview

A Text-To-Speech (TTS) synthesizer is a computer-based system that should be able to read any text aloud. There is a fundamental difference between the system we are about to discuss here and any other talking machine (as a cassette-player for example) in the sense that we are interested in the automatic production of new sentences. Systems that simply concatenate isolated words or parts of sentences, denoted as voice response systems, are only applicable when a limited vocabulary is required (typically a few one hundreds of words), and when the sentences to be pronounced respect a very restricted structure, as is the case for the announcement of arrivals in train stations for instance. In the context of TTS synthesis, it is impossible to record and store all the words of the language. It is thus more suitable to define Text-To-Speech as the automatic production of speech through grapheme to phoneme transcription of the sentences to utter.

1.2 Objective of the Dissertation

The main objective of this report is to study the situation of today's speech synthesis technology and to focus on potential methods for the future of this project. The objective of the whole project is to develop high quality audio speech synthesis with a well-synchronized talking head. Other aspects, such as naturalness, personality, platform independence, and quality assessment are also under investigation. Most synthesizers today are so called stand-alones and they do not work platform independently and usually do not share common parts, thus we can not just put together the best parts of present systems to make a state-of-the-art synthesizer. Hence, with good modularity characteristics a synthesis system can be achieved, which is easier to develop and improve.

1.3 Scope of the work

Text to speech synthesizers possess a very peculiar feature, which makes them wonderful laboratory tools for linguists: they 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 into native and rhythmic models. A particular type of Text To Speech systems, which are based on a description of the vocal tract through its resonant frequencies (its formants) and denoted as formant synthesizers, has also been extensively used by phoneticians to study speech in terms of acoustical rules. This project can be extended to provide Telecommunications services, Language education, Talking books and toys Multimedia, man-machine communication etc.

1.4 Organization of the thesis

The report starts with a brief description of different speech synthesis methods and speech synthesizers. The second chapter includes a short theory section of human speech production, articulatory phonetics, and some other related concepts. The speech synthesis procedure involves lots of different kinds of problems described in Chapter 3. The design procedure is described in the chapter 4 . The implementation details are described in the chapter 5. The result screens are shown in chapter 6 and, finally, the last chapters contain conclusion, and future discussion.

Chapter 2

LITERATURE SURVEY

Speech synthesis, the automatic generation of speech waveforms, has developed rapidly in recent years. Before special-purpose DSP chips were introduced , synthetic speech was generated primarily on large computers, sometimes interfaced with an analog vocal tract model. Now "speech synthesizer" devices range from inexpensive software programs for home computers to reading machines for the blind but still they all represent tradeoffs among the conflicting demands of maximizing speech quality, while minimizing memory space, algorithmic complexity, and computation time. The quality of the final synthetic speech depends on all the stages of the synthetic speech development process; neat speech editing and segmentation, accurate analysis and encoding, and complete strategy rules present better sounds.

Compared to speech recognition, there has been much less speaker recognition research because fewer applications exist than for speech recognition and less is understood about which aspects of a speech signal identify a speaker than about segmental-phonetics. For speech recognition, much is known about the speech production process linking a text and its phonemes to the spectra and prosodies of a corresponding speech signal. Each phoneme has specific articulatory targets, and the corresponding acoustic events have been well studied but remain far from fully understood. For speaker recognition, the acoustic aspects of what characterizes the differences between voices are obscure and difficult to separate from signal aspects that reflect segment recognition.

There are three sources of variation among speakers: differences in vocal cords and vocal tract shape, differences in speaking style (including variations in both target positions for phonemes and dynamic aspects of coarticulation such as speaking rate), and differences in what speakers choose to say. Automatic speaker recognizers exploit only the first two variation sources but not fully, examining low-level acoustic features of speech since a speaker's tendency to use certain words and syntactic structures (the third source) is difficult to quantify or control in an experiment. But the synthesized speech can never have emotions in it, it can never sound like a human being's voice.

Since eyes cannot directly see speech, it is difficult to intuitively understand the difficulty of speech recognition. However, the difficulty of recognizing speech is comparable to that of reading text in running hand and deciding who wrote it. In running hand style each character varies slightly depending on the neighboring characters. In speech this phenomenon is called coarticulation. With written characters, the variation among different writers and due to additive noise, such as spots on the paper, are very large. There is similar variation in conversational speech; everyday conversational speech includes botched utterances and repetition more frequently than writing.

If we reconstruct a voice by using sampled sounds from a speaker, it should result in a recognizable reconstructed voice. Such technology would make it possible to edit voices, i.e. add or delete sections without the speaker being present there. In other words, it would enable users to retain the voice aspect of personality for an indefinite period of time.

To communicate information to a listener, a speaker produces a speech signa*l* in the form of pressure waves that travel from the speaker's head to the listener's ears. The signal is nonstationary, or time-varying, changing characteristics as the muscles of the vocal tract contract and relax. Speech can be divided into sound segments, which share some common acoustic properties with one another for a short interval of time. These are the smallest units of speech that serve to distinguish one utterance from another in a language, commonly known as phonemes; for example, the \t\ of ten and the \p\ of pen are distinctive in English language[6].

2.1 Classification of sounds

Sounds are typically divided into two broad classes: (a) vowels, which allow unrestricted airflow in the vocal tract, and (b) consonants, which restrict airflow at some point and are weaker than vowels.

(a) Vowels

Vowels (including diphthongs) are voiced; usually have the largest amplitudes among phonemes, and range in duration from – milli seconds in normal speech. These are primarily described in terms of tongue position and lip rounding.

(b) Consonants

Place of articulation is most often associated with consonants, rather than vowels, because consonants use a relatively narrow constriction. Along the vocal tract, approximately eight regions or points are traditionally associated with consonant constrictions, as follows and the various organs of the mouth are shown in figure 2.1:

1. **Labials**: if both lips constrict, the sound is bilabial; if the lower lip contacts the upper teeth, it is labiodental.[11]

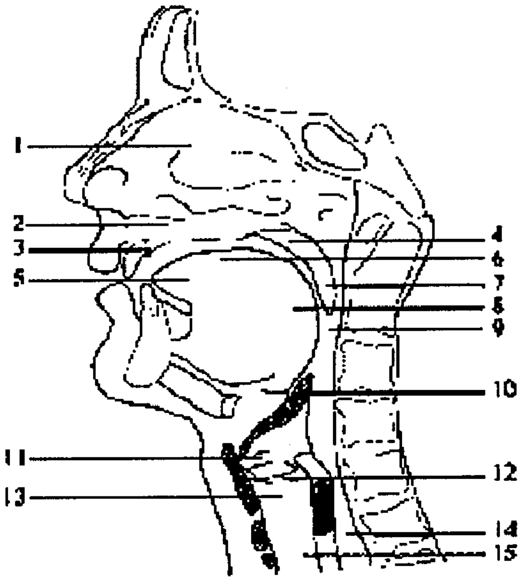


Figure 2.1. The human vocal organs. (1) Nasal cavity, (2) Hard palate, (3) Alveolar ridge, (4) Soft palate (Velum), (5) Tip of the tongue (Apex), (6) Dorsum, (7) Uvula, (8) Radix, (9) Pharynx, (10) Epiglottis, (11) False vocal cords, (12) Vocal cords, (13) Larynx, (14) Esophagus, and (15) Trachea.

2. **Dental:** the tongue tip or blade touches the edge or back of the upper incisor teeth (if the tip protrudes between upper and lower teeth, as in */q/*, the sound is interdentally).

3. **Alveolar:** the tongue tip or blade approaches or touches the alveolar ridge.

4. **Palatals:** the tongue blade or dorsum constricts with the hard palate; if the tongue tip curls up, the sound is retroflex.

5. **Velar:** the dorsum approaches the soft palate. Some linguists use the term compact for velars because their spectra concentrate energy in one frequency region.

6. **Uvular***:* the dorsum approaches the uvula.

7. **Pharyneal:** the pharynx constricts.

8. **Glottal:** the vocal folds either close or constrict.

2.2 Simple text to speech procedure

Speech is the primary means of communication between people. Speech synthesis, automatic generation of speech waveforms, has been under development for several decades. A Text-to-Speech (TTS) synthesizer is a computer-based program in which the system processes through the text and read it aloud. In the context of TTS systems, it is impossible to store all the words of the language unlike the Voice Response Systems that need a few words. Thus TTS system is the production of speech, through a grapheme-to-phoneme transcription of the sentences. TTS systems are better than the audio recording systems in the sense that they audibly communicate information with the users when the recordings are too large to store and the order of the time in speaking is important. By the definitions above, a basic idea is given about the Text-to-Speech Systems. Simple text to speech synthesiser modules are explained with the help of figure 2.2.

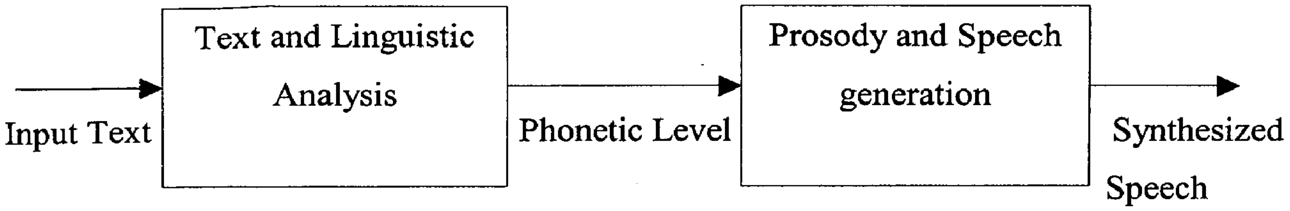


Fig. 2.2. Simple text-to-speech synthesis procedure.

1. **Transcription:** This module transcribes the orthographic input text into a sequence of phoneme or allophone codes, which specify the sounds to be produced.
2. **Prosody:** This module computes phoneme/allophone durations, determines word and sentence-level stress and assigns a fundamental frequency contour to the utterance.
3. **Synthesis:** This module synthesizes the desired utterance from the specifications provided by the transcription and the prosody module.

A Text To Speech (TTS) system needs detailed information in order to synthesize a desired utterance correctly. When the input comes from a natural language generation system, it may already be annotated with much of the necessary information. But most TTS systems are standalone; they have to extract all information systematically from the written text with the help of linguistic knowledge. This linguistic knowledge is usually stored in the form of rules, sometimes in the form of statistical models. Since designing linguistic rule sets is difficult and time consuming, statistical models are to be preferred for minority language TTS. Although it is also time consuming to key in the training data for these models, it should take less time and effort than thorough linguistic analyses.

Utterances are not just simple realizations of sequences of phonemes. Variations in fundamental frequency, intensity, and duration as well as pauses can express a variety of information. These parameters are commonly subsumed under the common heading prosody. They mirror sentence structure, pragmatics (questions have a different fundamental frequency contour than declarative sentences), emphasis, and so on. To generate an adequate prosody, we therefore need this high-level linguistic information. We also need morphological information, e.g. about the part of speech of each word, to support syntactic and semantic analyses. For minority languages, these modules are mostly far too difficult to write and design, but a few simple heuristics can nevertheless be implemented to cover the most common sentence patterns[7].

Next, we need to know the sequence of phonemes to be synthesized. Most words will already be stored in a lexicon; unknown words have to be transcribed automatically. Since lexical for minority languages are smaller and less sophisticated than those for well researched languages like English, more words will be unknown. Therefore, the automatic transcription module will have to be especially reliable.

There are two fundamental types of TTS Synthesis; Rule-Based Synthesis and Concatenative Synthesis.

1. Rule-Based Synthesis

Rule-based synthesis systems generate utterances from a set of parameters that are used to control the synthesizers proper, which produce the speech sounds. In a simple model, there are two major components of a rule-based synthesizer: a generator for the excitation signal and a filter that simulates the effect of the vocal tract. Usually, the parameters of the filter are derived from acoustic specifications. For each allophone, two sets of parameters are specified: the acoustic target that should be reached and the transitions to and from neighboring allophones. Alternatively, the transitions can be specified via rules only.

2. Concatenative Synthesis

Concatenative speech synthesis systems generate speech by concatenating and manipulating prerecorded units of speech. Three design decisions are particularly important: choice of units, storage of units, and concatenation method. As the second decision is relatively independent of the other two. Concatenating single phones, the first approach to concatenative synthesis, yields rather poor quality. Quality improves dramatically when diphones are used instead. Diphone units consist mainly of the transition between two phones. The unit boundaries are in the steady states of these phones in order to allow smooth concatenation. For example, if we want to synthesise the word test /test/, we have to concatenate the diphones #-t, t-e, e-s, s-t, and t-#. Diphone synthesis is based on two assumptions:

1. The transition between the two phones is sufficient to model all necessary coarticulatory effects.
2. The spectra of the steady states of the phones are consistent enough across different instances to avoid grave spectral mismatches at the concatenation points.

Another category of approaches in concatenation synthesis, exemplified by, derives the units from a phonetically balanced speech corpus, where each phone has been labeled with name, pitch, and other relevant information. For a given utterance, the synthesis algorithm now searches for a sequence of speech from the corpus that minimizes concatenation costs. For minority languages, a completely corpus-based approach would be ideal for several reasons:

1. **Recording:** meaningful text is easier to read than nonsense words.
2. **Unit selection and concatenation:** It would not be necessary to write a separate unit selection algorithm, if the standard algorithm is flexible enough.

On the other hand, phonetically balanced texts that provide all necessary units are very hard to design. By the definitions above, a basic idea is given about the Text-to-Speech Systems.

2.3 Potential Applications of TTS Systems

TTS systems have lots of uses in the today's intelligent systems like expert systems that explain their result and reasoning, intelligent assistants that collaborate with users and etc. These applications require that a system be capable of generating coherent multisentential responses, and interpreting and responding the user's subsequent expressions in the context of the ongoing interaction. These high quality TTS systems have numerous applications like the examples below:

- **Telecommunication services:** In these systems textual information can be accessed over the telephone. Mostly they are used when the requirement of interactivity is little and texts range from simple messages. Queries can be given through the user's voice (needs speech recognition) or through the telephone keyboard.
- **Language Education:** They provide a helpful tool to learn a new language known as computer aided learning system.
- **Aid to handicapped people:** the help of an especially designed keyboard and a fast sentence-assembling program can produce synthetic speech produced in a few seconds to remedy the voice handicaps. Also blind people can benefit from TTS systems, which gave them access to written information.
- Talking books and toys
- Vocal monitoring
- Multimedia, man-machine communication

In addition to these intelligent systems, there exist some uses than can be used in interactive situations such as:

- The user's hands and/or eyes are busy;
- Screen real estate is at a premium;
- Time is critical; or
- System and user are communicating via a primarily audio channel such as the telephone.

Chapter3

ANALYSIS OF THE PROBLEM

Text to speech (TTS) technology, the automatic conversion of stored text to synthetic speech, has progressed due in part to extensive theoretical and empirical contributions from the behavioral sciences. Human factors specialists also play an important role in determining appropriate applications for text to speech technology and in further tailoring the technology for a targeted application. The various stages of text to speech conversion are:

1. Text normalization.
2. Syntactic parsing.
3. Word pronunciation.
4. Determination of prosody.
5. Speech Synthesis.

3.1 Text Normalization

There are many abbreviations, acronyms, nonalphabetic characters, punctuations and digit strings in the unrestricted text. Periods may indicate an abbreviation or the end of a sentence, and must be disambiguated. Many abbreviations are expanded differently depending on the context. For example Dr. Can be expanded as doctor. Similarly common punctuation characters and numbers may be expanded differently, depending upon the context. There are two general ways existing TTS systems have dealt with the problems of text normalization. In some systems the user can set logical switches to determine how various types of nonalphabetic strings are handled[11].

3.2 Syntactic parsing

Syntactic analysis is important for word pronunciation and as one of the several components in determination of prosody. A partial syntactic analysis is also important for text normalization. A fairly large class of common words can be either nouns or verbs in which the first syllable is stressed for the noun, and the second syllable is stressed for a verb. In order to pronounce these words correctly, the syntactic structure of the input sentence must be determined. Both durational and intonational aspects of prosody are also affected by syntactic structure. This affects both segment durations and intonation contours. A syntactic analysis is necessary for appropriate prosody. Natural prosodic phrasing depends on locating syntactic location boundaries, particularly in sentences with long clauses. Inappropriate prosodic phrasing can result in impairs sentence comprehension. In txt normalization, the correct interpretation of non-word input may depend upon syntactic content of a standard or specialized type such as syntax of an addresses or telephone numbers. Syntactic analysis for text to speech conversion involves both the generation of possible syntactic parsing for a sentence and the determination of the most probable parsing of the sentence from among the syntactically correct alternatives of a syntactically ambiguous sentence. A simple strategy adopted here in this regard is to use function words to locate phrase boundaries, although this method is insufficient to locate all boundaries[11].

3.3 Word pronunciation

The correct pronunciation of words, which involves appropriate phoneme selection and lexical stress assignment, is critical to the intelligibility and acceptability of a text to speech system. Pronunciation habits have evolved considerably over time while spelling has remained more stable. Consequently letter to sound rules cannot yield accurate phonemic transcriptions for a sizable proportion of the words encountered in the text.

Letter to sound rules predict word pronunciation strictly from orthography. A set of conversion rules is devised assuming that there are general correspondences between alphabetic characters or pair of characters in the spelled word and the phonemes in its pronunciation. These correspondences may vary depending upon the neighboring letters in the word, which can be handled by context sensitive rules.

Stress assignment rules are an important constituent of TTS systems. Incorrect lexical stress assignment is disruptive to the listener because the stress pattern is salient feature in the recognition of a word. Stress assignment rules are generally modeled after linguistic theories, which predict lexical stress on the basis of general rules that vary considerably in their complexity[10].

3.4 Determination of prosody

Prosody refers to the way in which something is said, rather than what words are spoken. Intonation (pitch contours), timing and intensity are aspects of prosody that affects the sensation of pitch, length and loudness. Appropriate prosody is very important for a sentence to sound intelligible and natural. Prosody conveys both linguistic information and extra linguistic information about the speaker's attitude, intentions and the physical and emotional states.

In text to speech systems, the goal of intonation component is to generate an appropriate intonation contour for each spoken phrase. Listeners perceive an intonation contour as a pitch pattern that rises and falls at different points in a phrase.

Speech timing is an important aspect of prosody and affects both intelligibility and naturalness. Text to speech systems typically use duration rules to determine target duration for each phonetic segment in a utterance.

Intensity is the least important of the components of the prosody and is not modeled in most text to speech systems. Intensity typically becomes weaker at the end of an utterance but again this appears to be a secondary effect of changing voice source characteristics[8].

3.5 Speech Synthesis

Synthesis involves the transformation of a linguistic message from a visual modality to an auditory modality. This phonetic to acoustic conversion is achieved in several ways .The methods are usually classified into three groups:

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

Articulatory Synthesis

Articulatory synthesis tries to model the human vocal organs as perfectly as possible, so it is potentially the most satisfying method to produce high-quality synthetic speech. On the other hand, it is also one of the most difficult methods to implement and the computational load is also considerably higher than with other common methods. Thus, it has received less attention than other synthesis methods and has not yet achieved the same level of success. Articulatory synthesis typically involves models of the human articulators and vocal cords. The articulators are usually modeled with a set of area functions between glottis and mouth. The first articulatory model was based on a table of vocal tract area functions from larynx to lips for each phonetic segment. For rule-based synthesis the articulatory control parameters may be for example lip aperture, lip protrusion, tongue tip height, tongue tip position, tongue height, tongue position and velic aperture. Phonatory or excitation parameters may be glottal aperture, cord tension, and lung pressure.

When speaking, the vocal tract muscles cause articulators to move and change shape of the vocal tract, which causes different sounds. The data for articulatory model is usually derived from X-ray analysis of natural speech.

Advantages of articulatory synthesis are that the vocal tract models allow accurate modeling of transients due to abrupt area changes, whereas formant synthesis models only spectral behavior. The articulatory synthesis is quite rarely used in present systems, since the analysis methods are developing fast and the computational resources are increasing rapidly, it might be a potential synthesis method in the future [4].

Formant Synthesis

Probably the most widely used synthesis method during last decades has been formant synthesis, which is based on the source-filter-model of speech. There are two basic structures in general, parallel and cascade, but for better performance some kind of combination of these is usually used. Formant synthesis also provides infinite number of sounds, which makes it more flexible than for example concatenation methods. At least three formants are generally required to produce intelligible speech and up to five formants to produce high quality speech. Each formant is usually modeled with a two-pole resonator, which enables both the formant frequency (pole-pair frequency) and its bandwidth to be specified. Rule-based formant synthesis is based on a set of rules used to determine the parameters necessary to synthesize a desired utterance using a formant synthesizer. The input parameters may be for example the following, where the open quotient means the ratio of the open-glottis time to the total period duration:

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

A cascade formant synthesizer consists of band-pass resonators connected in series and the output of each formant resonator is applied to the input of the following one. The cascade structure needs only formant frequencies as control information. The main advantage of the cascade structure is that the relative formant amplitudes for vowels do not need individual controls. The cascade structure has been found better for non-nasal voiced sounds and because it needs less control information than parallel structure, it is then simpler to implement. However, with cascade model the generation of fricatives and plosive bursts is a problem [10].

Concatenative Synthesis

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

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

Word is perhaps the most natural unit for written text and some messaging systems with very limited vocabulary. Concatenation of words is relative easy to perform and coarticulation effects within a word are captured in the stored units. However, there is a great difference with words spoken in isolation and in continuos sentence, which makes the continuous speech to sound very unnatural. Because there are hundreds of thousands of different words and proper names in each language, word is not a suitable unit for any kind of unrestricted TTS system.The number of different syllables in each language is considerably smaller than the number of words, but the size of unit database is usually still too large for TTS systems.

For example, there are about 10,000 syllables in Hindi. Unlike with words, the coarticulation effect is not included in stored units, so using syllables as a basic unit is not very reasonable. There is also no way to control prosodic contours over the sentence. At the moment, no word or syllable based full TTS system exists. The current synthesis systems are mostly based on using phonemes, diphones, demisyllables or some kind of combinations of these. Demisyllables represents the initial and final parts of syllables. One advantage of demisyllables is that only about 1,000 of them is needed to construct the 10,000 syllables of Hindi. Using demisyllables, instead of for example phonemes and diphones, requires considerably less concatenation points. Demisyllables also take account of most transitions and then also a large number of coarticulation effects and also covers a large number of allophonic variations due to separation of initial and final consonant clusters. However, the memory requirements are still quite high, but tolerable. Compared to phonemes and diphones, the exact number of demisyllables in a language cannot be defined. With purely demisyllable-based system, all possible words cannot be synthesized properly. Phonemes are probably the most commonly used units in speech synthesis because they are the normal linguistic presentation of speech. The inventory of basic units is usually between 40 and 50, which is clearly the smallest compared to other units. Using phonemes gives maximum flexibility with the rule-based systems.

However, some phones that do not have a steady-state target position, such as plosives, are difficult to synthesize. The articulation must also be formulated as rules. Phonemes are sometimes used as an input for speech synthesizer to drive for example diphonebased synthesizer[10].

Chapter 4

DESIGN OF THE PROPOSED SOLUTION

Text-to-speech (TSS) conversion has to be performed in two steps:

(a) Text to phoneme conversion

(b) Phoneme to speech conversion.

In the second step, we face the core problem of speech synthesis, viz. those acoustic manifestations of phonemes are context dependent and transitions between phonemes carry vital perceptual cues. Synthesis of continuous speech therefore cannot be done just by cut-and-paste of individual words or phonemes. Two parallel approaches to overcome this problem are as below.

(a) Concatenation method uses cut-and-paste, but larger splices of natural speech, e.g. syllables, diphones are selected as units, so that immediate contextual effects are captured. Also, the splices are to be concatenated at steady-state of speech.

(b) In the Generative method, speech is generated from some kind of speech production model and the control parameters of the model are varied usually by context-dependent rules. Formant synthesizer is a very widely used generative synthesizer, which uses a speech production model based on formant (i.e. resonance) frequencies.

4.1 Text analyzer

Input to this stage is Romanized equivalent if Hindi text and the output is a set of phoneme symbols and stress markers, indicating pronunciation and accent in either Hindi. The stress markers currently include full stop, semi-colon, comma, interrogation, exclamation and end-of-word symbols. Hindi and Indian English text analyzers are two completely different units. But the underlying methodology of text processing is by and large the same for both and is being described hereafter.

The core problem is to obtain pronunciation of language words, a task that is non-trivial. We are currently accomplishing the task in the following steps.

(a) The word is looked up in a phonetic dictionary. The dictionary access is done by indexing and searching, which makes the look-up quite fast even on platforms with memory constraints. If the look-up succeeds, the pronunciation is obtained from the dictionary.

(b) If no match is found, a morphological analysis is done to separate various components of the word i.e. prefixes, suffixes and roots. The prefixes and suffixes, however, have quite different connotations for Hindi and other languages. It is possible to have ambiguities in morphological analysis. In such cases, a few strong root alternatives are generated as per the letter context and each of them is searched for in the dictionary, in an order determined by some heuristic rules. If a match is found, the pronunciation of the root is obtained from the dictionary. The pronunciation of the prefixes and suffixes are then merged with it, following suitable contextual modifications.

(c) If there is no match even now, we select the best root alternative and fall back to a set of Letter-to-Phoneme rules. These rules are quite different for Hindi and other languages and are quite elaborate for the latter, as its script is not phonetic. Basic (default) pronunciation of letter(s) is obtained from a symbol table. In Hindi, each character is assigned a pronunciation. Prior to table look-up, context-dependent rules are applied on the text character string. These are fewer and simpler for Hindi.

(d) The pronunciation of various morphs of the word (obtained in whichever way) are then merged together to form the pronunciation of the whole word.

(e) Ultimately, 'Phonological Rules' are applied to the entire phoneme string thus obtained.

These take care of contextual modification of pronunciation, e.g. of Sandi's (compound Words) in Hindi. The sets of phonological rules are completely different for different languages. Also, no clear-cut convention to represent 'typical' Indian phonemes (e.g. retroflexed/dental, aspirated/non-aspirated stop consonants, nasal vowels) in Roman script is followed. To do the best in the given situation, we have adopted a method which generates a 'score', depending on the position of the word in the sentence (first or not), whether the word starts with a capital, whether it is there in the (domain specific) 'Name dictionary' and which prefixes/suffixes (if any) were detected. The method, although not foolproof, works reasonably well. Currently, we are making a statistical analysis of letter clusters present in Indian names and English words. For example, 'bh' or 'sr' clusters are rare in English, whereas 'ous', 'tion' etc. are uncommon in Indian names. This, coupled with grammatical validity check for noun, will improve this decision [8].

Phoneme to acoustic–phonetic parameter conversion

This module generates a set of acoustic–phonetic parameters, corresponding to the input phoneme string, for every small time interval (currently, 5 ms) of speech. The parameters are of two types: fixed, which decides the voice type, and variable, which corresponds to the changes of phonetic contexts during speech.

The variable parameters used are:

(a) Amplitude of voicing for vocal sounds,

(b) Three 'formant' (resonance) frequencies,

(c) The corresponding bandwidths,

(d) Amplitude of 'frication', for noise-like sounds,

(e) Five amplitudes for various bands of energy in the noise spectrum,

(f) Amplitude of 'aspiration', for h-type of sounds,

(g) Spectral tilt and 'open quotient' of voicing, to shape the spectrum, and

(h) pitch.

Acoustic manifestations of various phonemes (in terms of the above parameters) are not fixed and are greatly influenced by the context. Separate sets of rules, usually corresponding to the immediate diphone contexts, were thereby formulated to generate each parameter track.. To avoid combinatorial explosion, phonemes are grouped into different classes, according to the places of articulation (e.g. vowel, stop consonant, nasal). Rules are selected as per the two adjacent phoneme classes (e.g. silence-to-vowel, nasal-to-stop), but they operate on the phoneme data of individual elements.

The rules are encapsulated in a very compact and flexible organization of a three dimensional interpolation table. Each entry corresponds to the current phoneme class, the next phoneme class and the parameter type. Each entry specifies an interpolation type (i.e. the manner in which the parameter should be varied) in the given situation and the parameter track is computed according to the interpolation type selected. For example, the interpolation may be linear (where the change is gradual) or abrupt, like a sudden jump and then a slow glide. New interpolation types are introduced and the old ones modified as per the results for acoustic–phonetic studies. Formant frequency is singularly the most important parameter type for speech perception and synthesis.

4.2 Database preparation

A series of preliminary stages have to be fulfilled before the synthesizer can produce its first utterance. At first, segments are chosen so as to minimize future concatenation problems. A combination of diphones (i.e. units that begin in the middle of the stable state of a phone and end in the middle of the following one), half-syllables, and triphones (which differ from diphones in that they include a complete central phone) are often chosen as speech units, since they involve most of the transitions and co-articulations while requiring an affordable amount of memory[1]. When a complete list of segments has emerged, a corresponding list of words is carefully completed, in such a way that each segment appears at least once (twice is better, for security). Unfavorable positions like inside stressed syllables or in strongly reduced (i.e. over-co-articulated) contexts, are excluded. A corpus is then digitally recorded and stored, and the elected segments are spotted, either manually with the help of signal visualization tools, or with the help of segmentation algorithms, the decisions of which are checked and corrected interactively. A segment database finally centralizes the results, in the form of the segment names, waveforms, durations, and internal sub-splitting. In the case of diphones, for example, the position of the border between phones should be stored, so as to be able to modify the duration of one half-phone without affecting the length of the other one. The database has the following structure. All sound files stored in the database are wave files recorded at 16KHz as 16bit signed linear samples. All the vowels, consonants half consonants as mentioned below are recorded with the help of speech station software through which we can record, segment audio files in wave format. Before giving examples of the above, we need to enumerate the consonants and vowels we allow [1].

Vowels

Vowels allowed are:

1	a	as is pun
2	aa	as in the hindi 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 mate
9	ae	as in mat
10	ai	as in height
11	o	as in the tamil word ponni (meaning gold)
12	oo	as in court
13	au	as in call
14	ow	as in cow

Consonants

k	kh	g
gh	ch	chh
j	jh	t
th	d	dh
n	tt	tth
dd	ddh	nna
p	f	b
bh	m	y
rl	ll	v
sh	s	h
zh	z	an

These consonants are numbered . the phonetic description however uses the pnemonics above. Within the program and in the database nomenclature, the numbers are used.

Examples

khana (food in hindi)	kh - a - n - a
maun (silence in hindi)	m - au - n
kahaan (where in hindi)	k – a - h – aa - n

A note on Half Characters

The various half characters that appear in the Hindi literature are as follows :

Ky kr kl
kll kv ksh
khy khr khl
khv gy gr
gl gv gn
ghy ghr ghv
ghn chy chr
chv jy jv
ty tr tv
thy thr dy
dr dv dhy
dhr dhv ny
nr nv tty
ttr ttv ddy
ddr ddv py
pr pl pll
fr fl by
br bl bhy
bhr bhl my
mr vy vr vl

4.3 Search Algorithm

The hash function

The purpose of the hash function is to assign at each word a number, computed from the letters in the word. This number must be smaller than the number of entries in the hash table. If the number of entries is smaller than the total number of words, some words will receive the same index, that's what is called collision. The number of collisions must be as small as possible given the number of entries so the hash function should try to spread the indexes into the full range as much as possible.

The one used is of the form:

```
h0 = word_let[0];

for(i=1 ; i<nb_of_letters ; i++)

 h0 = (h0*Hash_Mult + word_let[i]) % NbEntry;

  h = h0 % NbEntry;
```

Where NbEntry is the number of entries in the table and Hash_Mult an arbitrary number which is chosen in order to reduce the number of collisions.

For our program, the numbers finally chosen were:

Hash_Mult = 100.

NbEntry = number of words in the dictionary

The word retrieval

When the program starts, it loads the light hash table with the hash file . In the array hash_table, the program can access to the file position corresponding to a given hash index.

For each word in the sentence, the hash index is computed, the dictionary opened at the corresponding location. The first word on this line is then read. If it corresponds to the word we were looking for, the phonetic form is retrieved. If not, we look at the next one, and so on, until the end of the line is reached. This way, we roughly do two or three accesses per word given the size of the table we chose.

Once all the phonetic forms of the words are retrieved, the following task can occur. If the word is not found in the dictionary or in the table, the program stops.

4.4 Concatenative synthesizer

As opposed to rule-based ones, concatenative synthesizers possess a very limited knowledge of the data they handle : most of it is embedded in the segments to be chained up. Segments are then often given a parametric form, in the form of a temporal sequence of vectors of parameters collected at the output of a speech analyzer and stored in a parametric segment database. The advantage of using a speech model originates in the fact that:

- Well chosen speech models allow data size reduction, an advantage which is hardly negligible in the context of concatenation-based synthesis given the amount of data to be stored. Consequently, a parametric speech coder often follows the analyzer.
- A number of models explicitly separate the contributions of respectively the source and the vocal tract, an operation, which remains helpful for the pre-synthesis operations: prosody matching and segments concatenation.

Indeed, the actual task of the synthesizer is to produce, in real-time, an adequate sequence of concatenated segments, extracted from its parametric segment database and the prosody of which has been adjusted from their stored value, i.e. the intonation and the duration they appeared with in the original speech corpus, to the one imposed by the language processing module. Consequently, the respective parts played by the prosody matching and segments concatenation modules are considerably alleviated when input segments are presented in a form that allows easy modification of their pitch, duration, and spectral envelope, as is hardly the case with crude waveform samples.

Since segments to be chained up have generally been extracted from different words, which are in different phonetic contexts, they often present amplitude and timbre mismatches. Even in the case of stationary vocalic sounds, for instance, a rough sequencing of parameters typically leads to audible discontinuities. These can be coped with during the constitution of the synthesis segments database, thanks to an equalization in which related endings of segments are imposed similar amplitude spectra, the difference being distributed on their neighborhood. In practice, however, this operation, is restricted to amplitude parameters: the equalization stage smoothly modifies the energy levels at the beginning and at the end of segments, in such a way as to eliminate amplitude mismatches (by setting the energy of all the phones of a given phoneme to their average value). In contrast, timbre conflicts are better tackled at run-time, by smoothing individual couples of segments when necessary rather than equalizing them once for all, so that some of the phonetic variability naturally introduced by co-articulation is still maintained. In practice, amplitude equalization can be performed either before or after speech analysis (i.e. on crude samples or on speech parameters).

The overall design procedure can be explained with the help of block diagram as shown below in figure 4.1

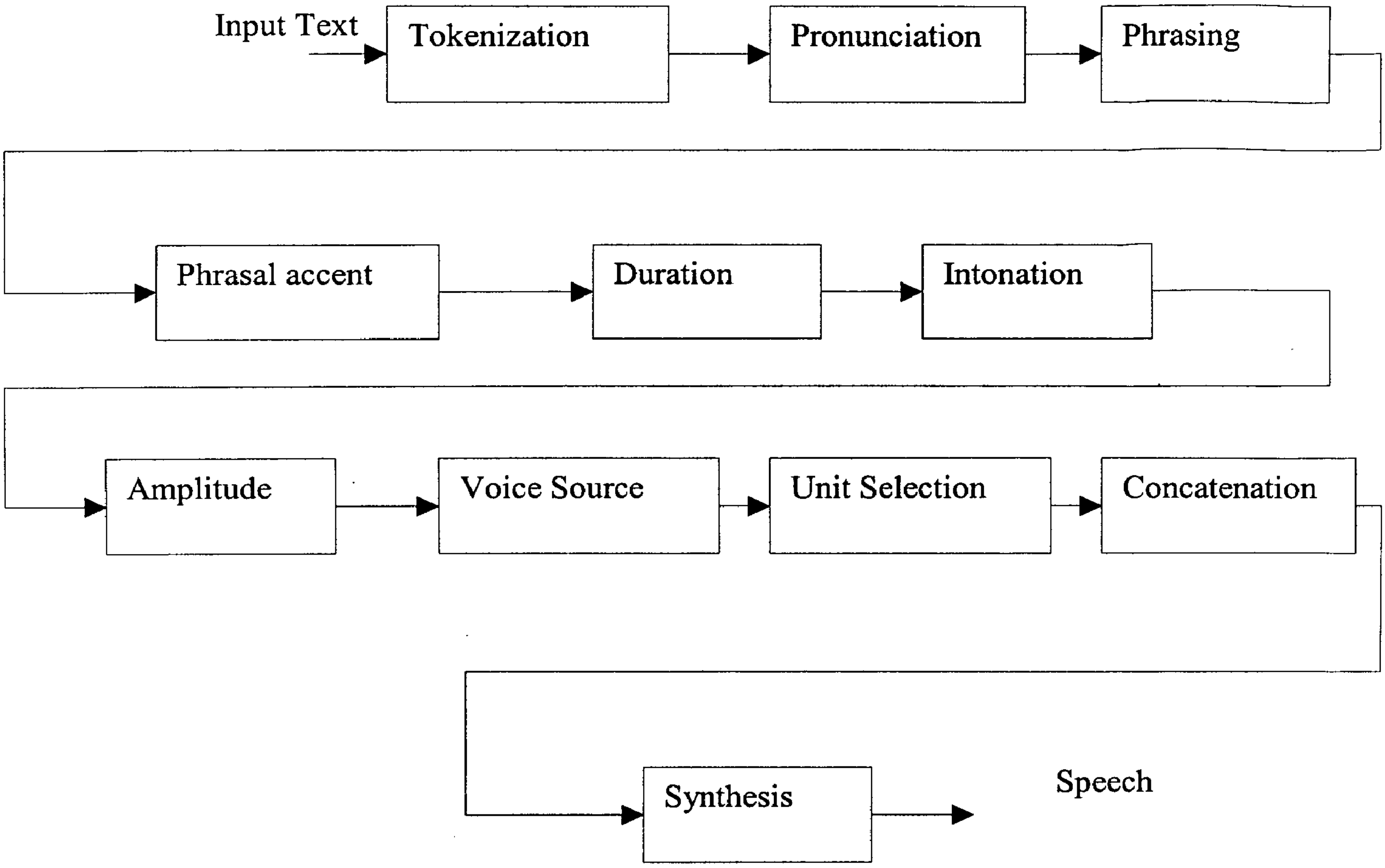


Figure 4.1 modules of text to speech system.

As noted above, one of the first stages of analysis of the text input is the tokenization of the input into words. For many languages, including Hindi, this problem is fairly easy in that one can to a first approximation assume that word boundaries coincide with white space or punctuation in the input text. A minimal requirement for word segmentation would appear to be an on-line dictionary that enumerates the word forms of the language.

Once the input is tokenized into words, the next obvious thing that must be done is to compute a pronunciation (or a set of possible pronunciations) for the words, given the orthographic representation of those words. The simplest approach is to have a set of letter-to-sound rules that simply map sequences of graphemes into sequences of phonemes, along with possible diacritic information, such as stress placement. This approach is naturally best suited where there is a relatively simple relation between orthography and phonology. Of course, the same problems of coverage as were noted in the Chinese segmentation problem also apply in the case of pronouncing dictionaries: many text words occur that are not to be found in the dictionary, the most important of these being morphological derivatives from known words, or previously unseen personal names.

For morphological derivatives, standard techniques for morphological analysis can be applied to achieve a morphological decomposition for a word. The pronunciation of the whole can then in general be computed from the pronunciation of the morphological parts, applying appropriate phonological rules of the language. Morphological analysis is of some use in the prediction of name pronunciation too, since some names are derived from others via fairly productive morphological processes however, this is not always the case, and one must also rely on other methods. One such method involves computing the pronunciation of a new name by analogy with the pronunciation of a similar name for a more general application of analogical reasoning to word pronunciation).

In many languages various words in a sentence are associated with accents, which are often manifested as upward or downward movements of fundamental frequency. Usually, not every word in the sentence bears an accent, however, and the decision of which words should be accented and which ones should not is one of the problems that must be addressed by a TTS system.

A good first step in assigning accents is to make the accentual determination on the basis of broad lexical categories or parts of speech of words. Content words---nouns, verbs, adjectives and perhaps adverbs, tend in general to be accented; function words, including auxiliary verbs and prepositions tend to be deaccented; short function words tend to be criticized. Naturally this presumes some method for assigning parts of speech, and in particular for disambiguating words which can be content words (in this case, a verb or a noun). But accenting has a wider function than merely communicating lexical category distinctions between words. Accenting is not only sensitive to syntactic structure and semantics, but also to properties of the discourse.

In reading a long sentence, speakers will normally break the sentence up into several phrases, each of which can be said to stand alone as an intonational unit. If punctuation were used liberally so that there are relatively few words between the commas, semicolons or periods, then a reasonable guess at an appropriate phrasing would be simply to break the sentence at the punctuation marks. The real problem comes when long stretches occur without punctuation; in such cases, human readers would normally break the string of words into phrases, and the problem then arises of where to place these breaks.

The simplest approach is to have a list of words, typically function words, that are likely indicators of good places to break [6]. One has to use some caution however, since while a particular function word may coincide with a plausible phrase break in some cases, in other cases it might coincide with a particularly place to break.

IMPLEMENTATION DETAILS

The recording of speech:

The Hindi speech was recorded in my own voice using a microphone and an 8-bit sound blaster card on a 486 machine. The data was sampled at 16 kHz with 8-bit quantization. Then using Speech station version 2.0 plotted the sound wave graph.

As silence being word separator, by changing the sequence of words and deleting a word of the sentence .The speech station software possess the capabilities of cutting and pasting waveforms. Hence a long sentence was recorded and syllables were extracted from the recorded sample.

example :

कुक = क. + उक

कक = क + अक

The word was splitted into smaller units and the sounds of all the following were extracted from various sentences:

a aa i
ii u uu
e ee ae
ai o oo
au ow
k kh g
gh ch chh

j jh t

th d dh

n tt tth

dd ddh nna

p f b

bh m y

rl ll v

sh s h

zh z an

ky kr kl

kll kv ksh

khy khr khl

khv gy gr

gl gv gn

ghy ghr ghv

ghn chy chr

chv jy jv

ty tr tv

thy thr dy

dr dv dhy

dhr dhv ny

nr nv tty

ttr ttv ddy

ddr ddv py

pr pl pll

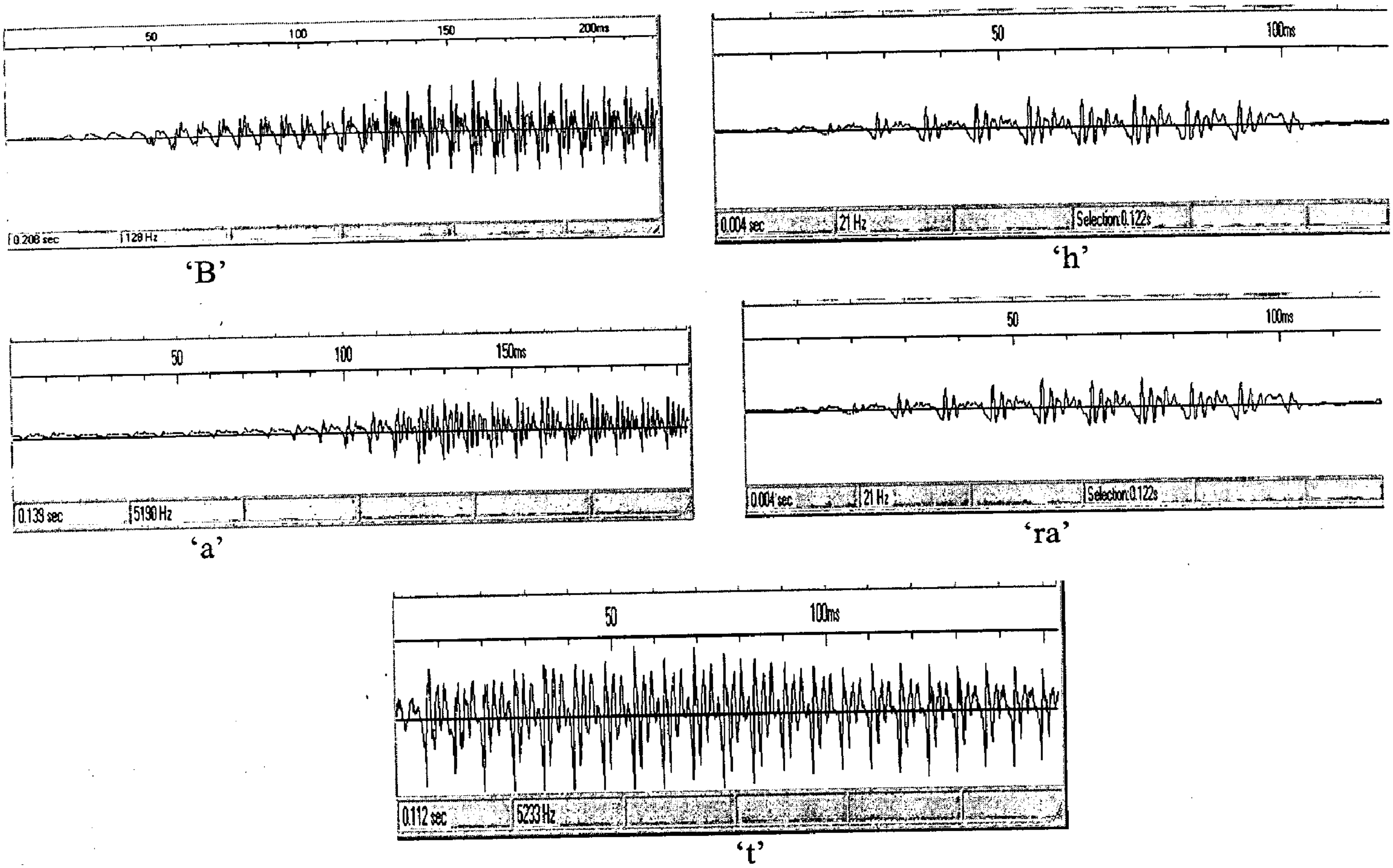
fr fl by

br bl bhy

bhr bhl my

mr vy vr vl

The wave files of the above units and many wave files that were recorded were y were numbered. So a digital code was assigned to each wave file.



'B'

'h'

'a'

'ra'

't'

figure 5.1 The segments that were extracted from word Bharat.

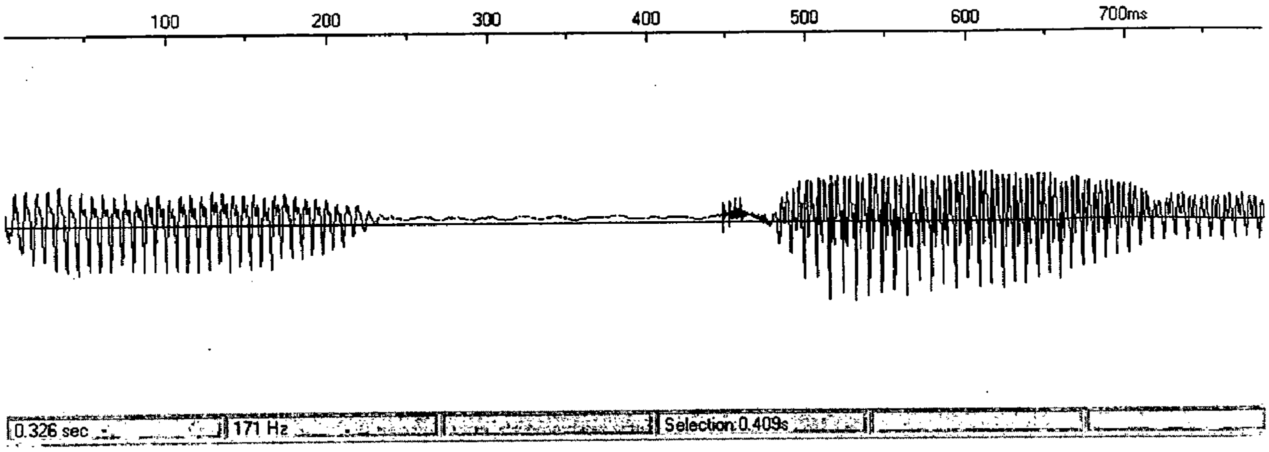


figure 5.2 Waveform for the word Bharat

the segmentation waveforms are shown in figure 5.1 for the word Bharat which is shown in figure 5.2.

Selection of Units:

The input sentence is passed through a parser function, which splits the given sentence into smaller units that are matching to our database.

So the output.of this phase is we get the list of files that are to be played for a particular sentence. Then all the names of the wave files to be played are taken in a data structure so as it represent a play list.

The voices of all the units that occur in the Hindi text are recorded and kept in the database. I have taken the length of the smaller unit as four. So the algorithm is as follows:

Step 1: Start.

Step 2: Get the input sentence.

Step 3: send first 4 characters of the input sentence.

Step 4: If a match is found, send it to play list remove the first four characters of the input string.

Else

Send the first three characters, If a match is found, send it to play list remove the first three characters of the input string.

```
Else
Send the first two characters, If a match is found, send it to play list remove the first two characters of the input string.
Else
Send the first character, If a match is found, send it to play list remove the first character of the input string
Step 5: if (end of sentence) exit
          Else go to step 3.
Step 6: Stop
```

Concatenation:

Here in this phase the length of each wave file is determined which will be in the order of milliseconds or a few seconds. So durations of all the wave files are kept in another file and while playing the wave files the duration of each wave files that are present in the play list is extracted and the wave file is played for that amount of time. Just concatenating different phonemes together, did produce recognizable words but did not produce recognizable speech that is everybody could understand what is said but could not recognize the speaker. Since each word must originally be pronounced with timing and intonation appropriate for that particular sentence in which it's being used, silence was added wherever required. To avoid jumpy, discontinuous speech phonemes were smoothed at their boundaries. Keeping this in mind that the pronunciation of a phoneme in a word or phrase is heavily dependent on its phonetic context (e.g., on neighboring phonemes, intonation and speaking rate), each phoneme was amplified, smoothed and adjusted accordingly.

Actually a speech signal conveys information about the identity of the speaker, the state of his health, mood and the environment etc. in which he is speaking. Every speaker speaks on the basis of individuality information in speech waves.

This means that there is something else apart from the phonemes in actual speech, which is missing, in the reconstructed one. In order to discover that, I took a speech sample, simply deleted the phonemes from that, and then studied the left over sound wave graph. Now this left over speech sample was having different sounds like breathy voices, tongue clicks, murmur, etc., which I saved separately. And then combined these sounds with the reconstructed sentences and got excellent results.

Thus while combining these special sounds with phonemes to reconstruct speech following things should be kept in mind:

Normally at what places the speaker gives pauses (e.g., in middle of some words like 'be cause', in the end of some words like 'so',etc.).

create some noise in the beginning of the speech (e.g., uuu.. or mhmh..,etc.) .

--Normally at what phonemes the speaker stresses (e.g., t, sh, d, etc.) .

At last a smoothing is done to improve the quality of speech by varying the durations between the wave files played during the testing process.

Chapter 6

RESULTS AND DISCUSSIONS

The input to the system is romanized equivalent of Hindi text (as shown in appendix) and the output is an audio stream. The figure 6.1 shows the graphical user interface of this Hindi text to speech system. The input is given at the text box which is provided here and on clicking play event the list of files to be played for the successful pronunciation of the word get displayed and the and the audio files get played. The stop, clear and exit buttons are also provided here.

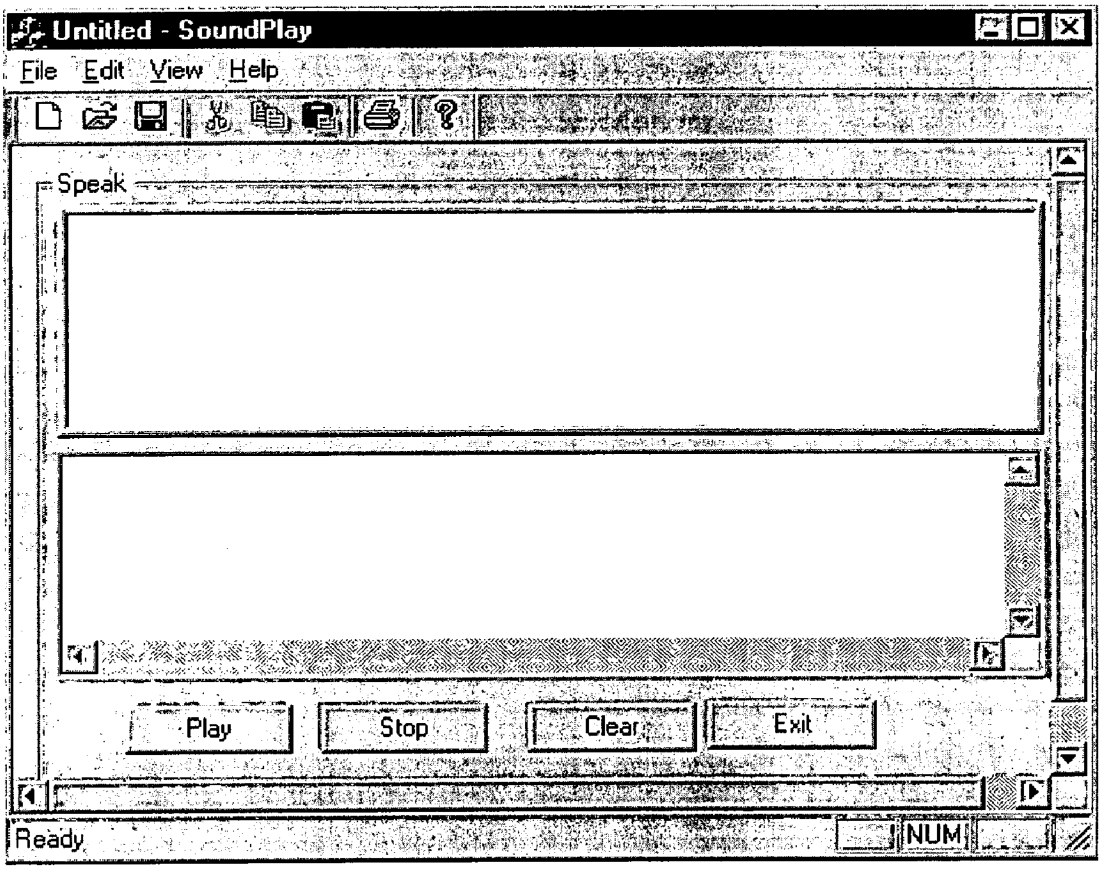
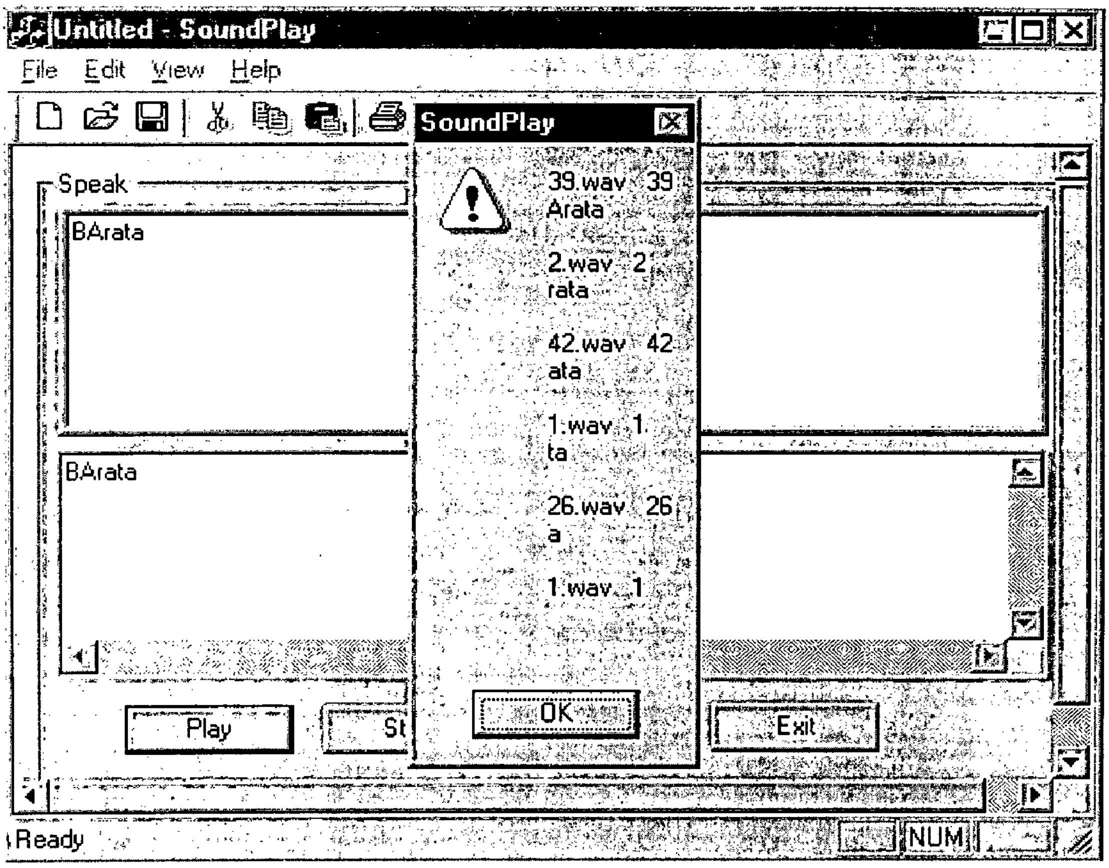


Figure 6.1 GUI for the text to speech system.

After giving the input and clicking the play button the list of wave files to be concatenated are displayed on the space provided. Then on clicking ok the audio corresponding to the input sentence is played. The corresponding figures are given below in figure 6.2 and figure 6.3.

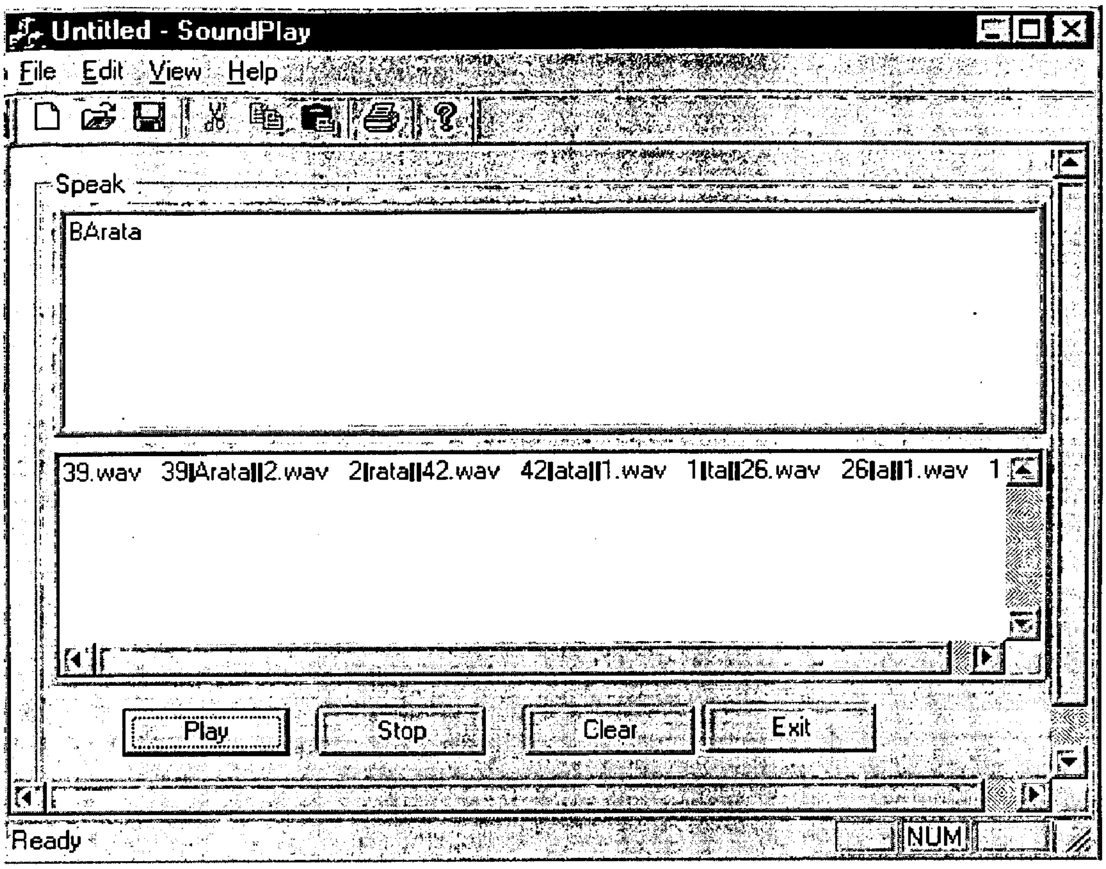


BArata

BArata

```
39.wav 39
Arata

2.wav 2
rata

42.wav 42
ata

1.wav 1
ta

26.wav 26
a

1.wav 1
```

Figure 6.2 GUI after giving the input text.



```
BArata
```

```
39.wav   39|Arata||2.wav   2|rata||42.wav   42|ata||1.wav   1|ta||26.wav   26|a||1.wav   1
```

figure 6.3 GUI while playing the input sentence.

Chapter 7

CONCLUSION

Speech synthesis has been developed steadily over the last decades and it has been incorporated into several new applications. For most applications, the intelligibility and comprehensibility of synthetic speech have reached the acceptable level. However, in prosodic, text preprocessing, and pronunciation fields there is still much work and improvements to be done to achieve more natural sounding speech. Natural speech has so many dynamic changes that perfect naturalness may be impossible to achieve. The most commonly used techniques in present systems are based on formant and concatenative synthesis. The latter one is becoming more and more popular since the methods to minimize the problems with the discontinuity effects in concatenation points are becoming more effective. The concatenative method provides more natural and individual sounding speech, but the quality with some consonants may vary considerably and the controlling of pitch and duration may be in some cases difficult, especially with longer units. Most information of the speech signal is focused at the frequency range less than 10 kHz. However, using higher sample rate than necessary, the speech may sound slightly more pleasant. For example, in warning and alarm systems synthesized speech may be used to give more accurate information of the current situation. Using speech instead of warning lights or buzzers gives an opportunity to reach the warning signal for example from a different room. Speech synthesizer may also be used to receive some desktop messages from a computer, such as printer activity or received e-mail. In the future, if speech recognition techniques reach adequate level, synthesized speech may also be used in language interpreters or several other communication systems, such as videophones, videoconferencing, or talking mobile phones. If it is possible to recognize speech, transcribe it into ASCII string, and then 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.

References:

[1] Agrawal S S, Stevens K 1992: Towards synthesis of Hindi consonants using KLSYN88. Proc. Int. Conf. on Spoken Language Processing 92, Alberta, Canada, pp 177–180

[2] Allen J, Hunnicutt M S, Klatt D H 1987 From text to speech: The MIT talk system (Cambridge, MA: University Press)

[3] Bhaskararao P, Peri V N, Udpikar V 1994 A text-to-speech system for application by visually handicapped and illiterate. Proc. Int. Conf. on Spoken Language Processing 94, Tokyo, Japan, pp 1239–1241

[4] Dan T K, Datta A K, Mukherjee B 1995 Speech synthesis using signal concatenation. J. Acoust. Soc.India, 18: 141–145

[5] Klatt D H 1980 Software for a cascade/parallel formant synthesizer. J. Acoust. Soc. Am. 67: 971–995

[6] Klatt D H 1987 Review of text-to-speech conversion for English. J. Acoust. Soc. Am. 82: 737–793

[7] Rao P V S, Bhiksha Raj, Sen A, Mallavadhani G R 1996 A computer tutor with voice I/O in Hindi.

[8] Knowledge based computer systems research and applications (eds) K S R Anjaneyulu, MSasikumar,R N Ramani (New Delhi: Narosa) pp 491–502

[9] Samudravijaya K, Ahuja R, Bondale N, Jose T, Krishnan S, Poddar P, Rao P V S, Raveendran R 1998.

[10] T. Dutoit, H. Leich. MBR-PSOLA: Text-To-Speech synthesis based on an MBE re-synthesis of the segments database. Speech Communication 13. pp. 435-440. 1993.

[11] Sproat. R. (1996) Multilingual text analysis for text-to-speech synthesis. In W.Wahltster (Ed.), Budapest, Hungary, pp.75-81.

Hindi Romanization scheme :

Hindi character	Equivalent English character
अ	a
आ	A
इ	i
ई	I
उ	u
ऊ	U
ृ	q
ए	e
ऐ	E

ओ	o
औ	O
।	M
:	H
◡	z
क	k
ख	K
ग	g
घ	G
ङ	f
.	Z
च	c

छ	C
ज	j
झ	J
ट	t
ठ	T
ड	d
ढ	D
ण	N
त	w
थ	W
द	x

ध	X
न	n
प	p
फ	P
ब	b
भ	B
म	m
य	y
र	r
ल	l
व	v

श	S
त्र	F
स	s
ह	h
क्ष	kR
ज्ञ	jF
श्र	Sr
त्त	wwa
प्र	pra
क्र	kra