INTRODUCTION TO PROJECT

A Text-To-Speech (TTS) synthesizer is a computer-based system that should be able to read any text aloud. Let us try to be clear. 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. This definition still needs some refinements. 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 (and luckily useless) 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 a grapheme-to-phoneme transcription of the sentences to utter.

Text-To-Speech (TTS) systems aim to transform arbitrary1 textual input into spoken output. At first glance, this may seem a relatively simple task of determining the phonetic sounds of the input and outputting a corresponding sequence of audible signals. However, it is in fact quite a difficult task to produce intelligible and natural results in the general case. This is due to linguistic and vocalization subtleties at various levels that human speakers take for granted when interpreting written text.

The potential application of such functionality is varied, including its use for language education, as an aid to handicapped persons, for implementing talking books and toys, for vocal monitoring and other man-machine communication facilities. The area is currently being explored in order to address its application for the HINDI language as is being done in the project here.

INTRODUCTION TO SYNTHESIS

A Text-to-Speech system (or TTS system) is the process which converts text into voice, spoken out loud. A TTS system does basically the same thing as when you read a page of text. This may seem easy but it took you several years to learn and you still make some mistakes from time-to-time.

Text-to-speech (TTS) is the ability of the operating system to play back printed text as spoken words. An internal driver, called a TTS engine, recognizes the text and using a synthesized voice chosen from several pre-generated voices, speaks the written text.

The primary goal of a TTS system is to deliver the message contained in the text in the best possible way in order to insure that the listener grasps the general sense. The meaning of the text according to its context must also be conveyed. This means a TTS system must be capable of analyzing the structure of the text, abbreviations or the acronyms used, extract some indication about the way it should be read, based on punctuation and text structure. It must also recognize exceptions to the generic language rules. The system will then apply text to phoneme conversion based on the language dependant rules.

The final stage of a TTS system is the part that generates audio signals. Several options are possible to generate speech. There are 3 main technologies for synthesizing speech, which produce different speech quality:

- Format synthesis Technology
- Concatinative Synthesis Technology
- Phrase Splicing Technology

Formant Synthesis Technology:

Formant is a speech synthesis technology which produces speech using linguistic rules and knowledge of how the human vocal tract works. The vocal tract has certain major resonant frequencies, which change as the configuration of the vocal tract changes. These changes in frequency are known as "formants". With Formant technology, the resulting sound is all synthesized causing some to criticize the technology as being mechanical sounding. However, the Formant technology is very intelligible, and, because it is totally generated by the computer, is very flexible. An unlimited number of voices can be created by modifying voice characteristics such as gender, pitch, speed, and roughness.

Concatinative Synthesis Technology:

A Concatenative TTS system (also known as 3rd Generation) produces speech from recordings of small snippets of actual human speech. These units (possibly phonemes, syllables, words, or phrases) are then combined (or concatenated) according to linguistic rules formulated from analyzed text. When these recorded speech units are entire phrases or sentences, the output can be very natural, human-sounding speech. If the recorded speech units are phonemes or words, the output may sound a bit choppy. Because Concatenative TTS requires prerecorded units of speech, it requires a significant amount of memory and disk storage. In addition, applications that use this type of TTS are usually limited to one voice and to a very specific domain. A Concatenative TTS system can only produce audio output based on the words and phrases in its dictionaries. If it cannot find a way to concatenate existing units to form a word, it falls back on the Formant system to generate them.

Phrase Splicing Technology

Rather than gluing together small snippets of sounds to comprise a word (like Concatenative TTS), phrase splicing glues together whole words and phrases to create a remarkably natural sounding voice. When the phrase splicing system cannot find words and phrases in its dictionaries, it falls back on the Concatenative system to generate the words. Phrase splicing is best suited for customers with very limited amount of text to synthesize. It also requires significantly more memory than Concatenative TTS.

Here we have made use of concatenative synthesis technology and hence it is discussed in detail:

TTS using Concatenative Technique

Figure 1 introduces the functional diagram of a very general TTS synthesizer. As for human reading, it comprises a Natural Language Processing module (NLP), capable of producing a phonetic transcription of the text read, together with the desired intonation and rhythm (often termed as prosody), and a Digital Signal Processing module (DSP), which transforms the symbolic information it receives into speech. But the formalisms and algorithms applied often manage, thanks to a judicious use of mathematical and linguistic knowledge of developers, to short-circuit certain processing steps. This is occasionally achieved at the expense of some restrictions on the text to pronounce, or results in some reduction of the "emotional dynamics" of the synthetic voice (at least in comparison with human performances), but it generally allows to solve the problem in real time with limited memory requirements.

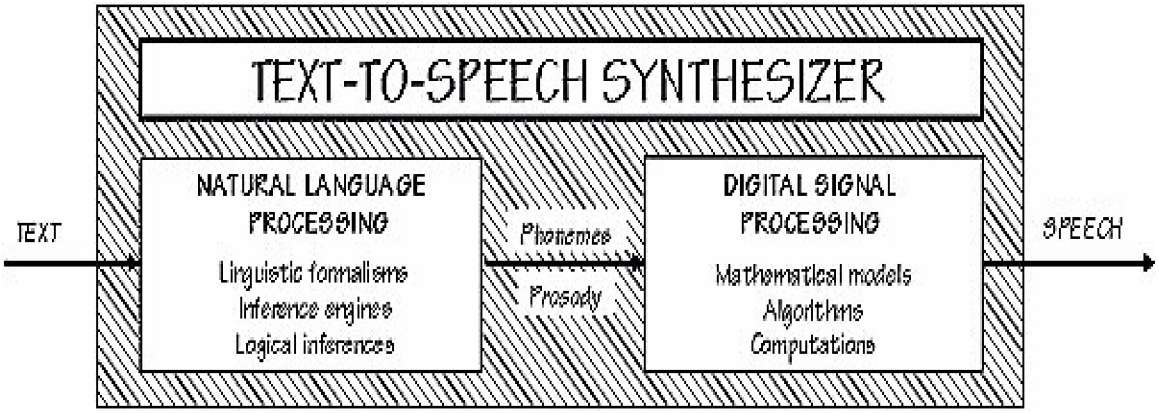


Fig: 1

THE NLP COMPONENT

Figure 2 introduces the skeleton of a general NLP module for TTS purposes. One immediately notices that, in addition with the expected letter-to-sound and prosody generation blocks, it comprises a morpho-syntactic analyser, underlying the need for some syntactic processing in a high quality Text-To-Speech system. Indeed, being able to reduce a given sentence into something like the sequence of its parts-of-speech, and to further describe it in the form of a syntax tree, which unveils its internal structure, is required for at least two reasons:

1. Accurate phonetic transcription can only be achieved provided the part of speech category of some words is available, as well as if the dependency relationship between successive words is known.

2. Natural prosody heavily relies on syntax. It also obviously has a lot to do with semantics and pragmatics, but since very few data is currently available on the generative aspects of this dependence, TTS systems merely concentrate on syntax. Yet few of them are actually provided with full disambiguation and structuration capabilities.

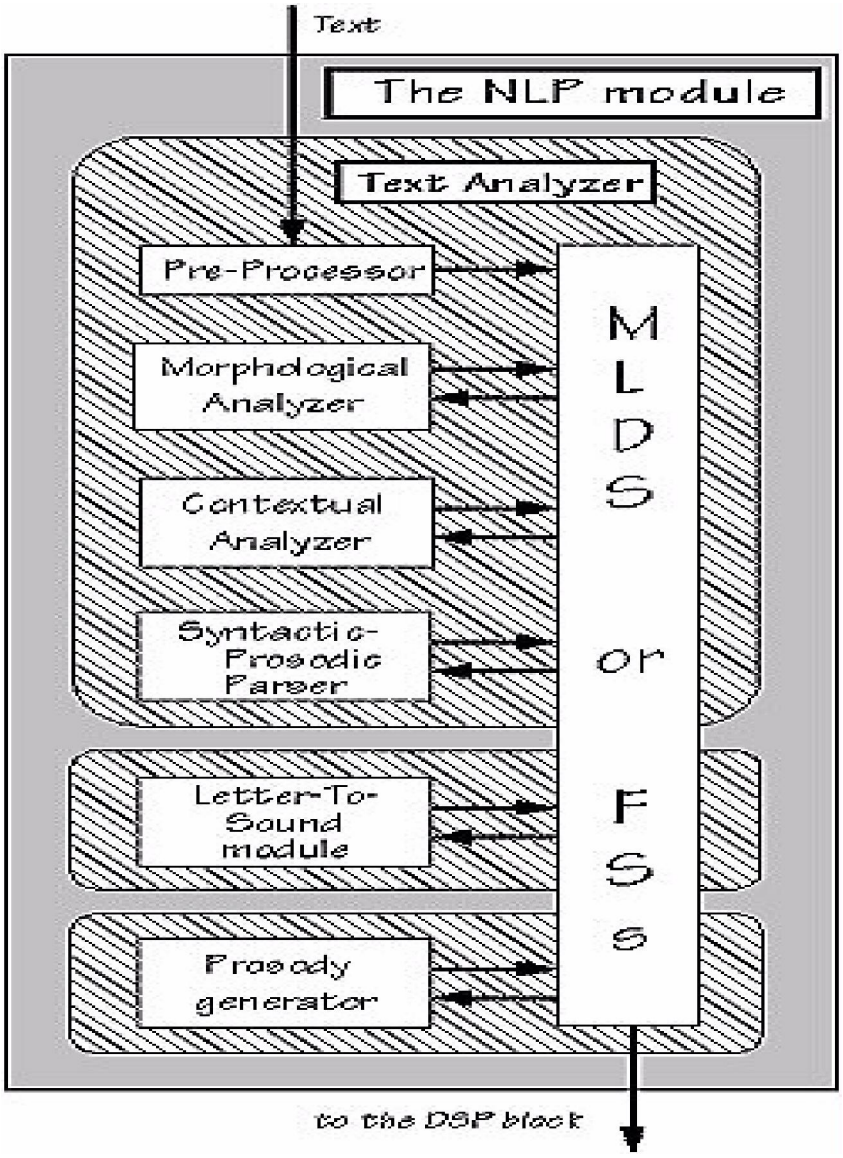

Text
The NLP module
Text Analyzer
Pre-Processor
Morphological Analyzer
Contextual Analyzer
Syntactic-Prosodic Parser
Letter-To-Sound module
Prosody generator
MLDS
or
FSs
to the DSP block

Fig: 2

TEXT ANALYSIS

The analysis of the written text for speech conversion is done as described in he steps described below:

- 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 when needed. An important problem is encountered as soon as the character level: that of punctuation ambiguity (including the critical case of sentence end detection). It can be solved, to some extent, with elementary regular grammars.
- 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. Inflected, derived, and compound words are decomposed into their elementery graphemic units (their morphs) by simple regular grammars exploiting lexicons of stems and affixes.
- 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 neighbouring words. This can be achieved either with n-grams, which describe local syntactic dependences in the form of probabilistic finite state automation, to a lesser extent with multi-layer perceptrons (i.e., neural networks) trained to uncover contextual rewrite rules or with local, non-stochastic grammars provided by expert linguists or automatically inferred from a training data set with classification and regression tree (CART) techniques.
- 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.

LETTER TO SOUND (LTS) MODULE

The Letter-To-Sound (LTS) module is responsible for the automatic determination of the phonetic transcription of the incoming text. It thus seems, at first sight, that its task is as simple as performing the equivalent of a dictionary look-up ! From a deeper examination, however, one quickly realizes that most words appear in genuine speech with several phonetic transcriptions, many of which are not even mentioned in pronunciation dictionaries namely:

1. Pronunciation dictionaries refer to word roots only. They do not explicitly account for morphological variations (i.e. plural, feminine, conjugations, especially for highly inflected languages, such as French), which therefore have to be dealt with by a specific component of phonology, called morphophonology
2. Some words actually correspond to several entries in the dictionary, or more generally to several morphological analyses, generally with different pronunciations. This is typically the case of heterophonic homographs, i.e. words that are pronounced differently even though they have the same spelling. Their correct pronunciation generally depends on their part-of-speech and most frequently contrasts verbs and non-verbs.
3. Pronunciation dictionaries merely provide something that is closer to a phonemic transcription than from a phonetic one (i.e. they refer to phonemes rather than to phones).
4. Words embedded into sentences are not pronounced as if they were isolated.
5. Finally, 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.

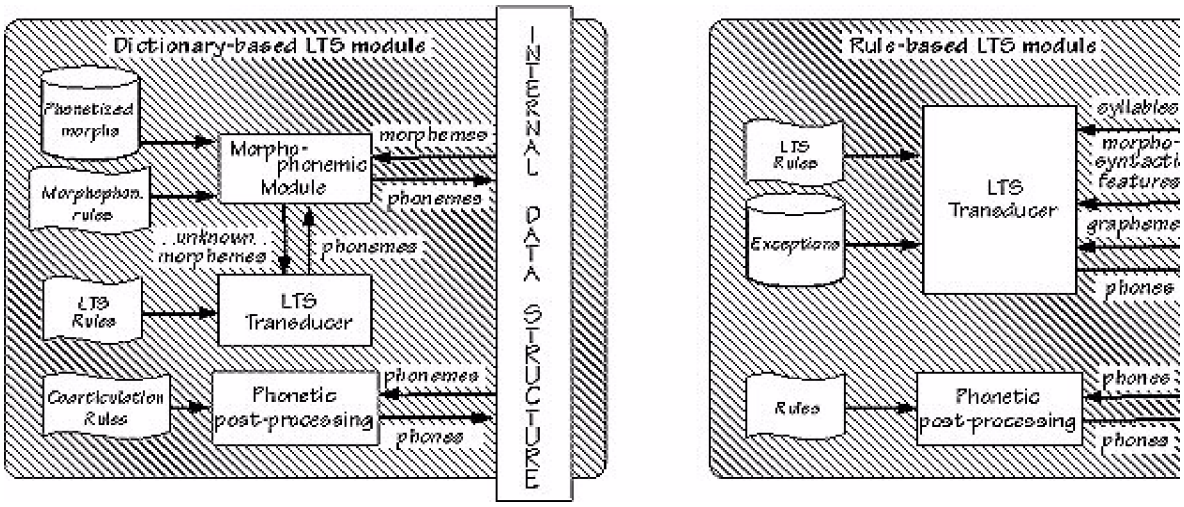
Clearly, points 1 and 2 heavily rely on a preliminary morphosyntactic (and possibly semantic) analysis of the sentences to read. To a lesser extent, it also happens to be the case for point 3 as well, since reduction processes are not only a matter of context-sensitive phonation, but they also rely on morphological

Structure and on word grouping, that is on morphosyntax. Point 4 puts a strong Demand on sentence analysis, whether syntactic or metrical, and point 5 can be partially solved by addressing morphology and/or by finding graphemic analogies between words.

It is then possible to organize the task of the LTS module in many ways (Fig. 3), often roughly classified into dictionary-based and rule-based strategies, although many intermediate solutions exist.

Dictionary-based solutions consist of storing a maximum of phonological knowledge into a lexicon. In order to keep its size reasonably small, entries are generally restricted to morphemes, and the pronunciation of surface forms is accounted for by inflectional, derivational, and compounding morphophonemic rules which describe how the phonetic transcriptions of their morphemic constituents are modified when they are combined into words. After a first phonemic transcription of each word has been obtained, some phonetic post-processing is generally applied, so as to account for coarticulatory smoothing phenomena.

A rather different strategy is adopted in **rule-based** transcription systems, which 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. Notice that, since many exceptions are found in the most frequent words, a reasonably small exceptions dictionary can account for a large fraction of the words in a running text.



PROSODY GENERATION

The term prosody refers to certain properties of the speech signal which are related to audible changes in pitch, loudness, syllable length.Prosodic features have specific functions in speech communication. The most apparent effect of prosody is that of focus. For instance, there are certain pitch events which make a syllable stand out within the utterance, and indirectly the word or syntactic group it belongs to will be highlighted as an important or new component in the meaning of that utterance. The presence of a focus marking may have various effects, such as contrast, depending on the place where it occurs, or the semantic context of the utterance.

Prosodic features create a segmentation of the speech chain into groups of syllables, or, put the other way round, they give rise to the grouping of syllables and words into larger chunks. Moreover, there are prosodic features which indicate relationships between such groups, indicating that two or more groups of syllables are linked in some way. This grouping effect is hierarchical, although not necessarily identical to the syntactic structuring of the utterance. The prosody generator, then, is responsible for identifying the intonational phrases

Corresponding to the text and assigning the appropriate prosody contour.

THE DSP COMPONENT

Intuitively, the operations involved in the DSP module are the computer analogue of dynamically controlling the articulatory muscles and the vibratory frequency of the vocal folds so that the output signal matches the input requirements. In order to do it properly, the DSP module should obviously, in some way, take articulatory constraints into account, since it has been known for a long time that phonetic transitions are more important than stable states for the understanding of speech. This, in turn can be achieved in two ways viz:

- Explicitly, in the form of a series of rules which formally describe the influence of phonemes on one another;
- Implicitly, by storing examples of phonetic transitions and co-articulations into a speech segment database, and using them just as they are, as ultimate acoustic units (i.e. in place of phonemes).

Two main classes of TTS systems have emerged from this alternative, which quickly turned into synthesis philosophies given the divergences they present in their means and objectives: synthesis-by-rule and synthesis-by-concatenation.

Rule based synthesis

Rule-based synthesizers are mostly in favor with phoneticians and phonologists, as they constitute a cognitive, generative approach of the phonation mechanism. For historical and practical reasons (mainly the need for a physical interpretability of the model), rule synthesizers always appear in the form of formant synthesizers. These describe speech as the dynamic evolution of up to 60 parameters [Stevens 90], mostly related to formant and anti-formant frequencies and bandwidths together with glottal waveforms. Clearly, the large number of

(coupled) parameters complicates the analysis stage and tends to produce analysis errors. What is more, formant frequencies and bandwidths are inherently difficult to estimate from speech data. The need for intensive trials and errors in order to cope with analysis errors, makes them time-consuming systems to develop (several years are commonplace). Yet, the synthesis quality achieved up to now reveals typical buzzyness problems, which originate from the rules themselves: introducing a high degree of naturalness is theoretically possible, but the rules to do so are still to be discovered. Rule-based synthesizers remain, however, a potentially powerful approach to speech synthesis. They allow, for instance, studying speaker-dependent voice features so that switching from one synthetic voice into another can be achieved with the help of specialized rules in the rule database. Following the same idea, synthesis-by-rule seems to be a natural way of handling the articulatory aspects of changes in speaking styles (as opposed to their prosodic counterpart, which can be accounted for by concatenation-based synthesizers as well). No wonder then that it has been widely integrated into TTS systems.

Concatenation based synthesis

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. This process utilizes a speech database consisting of prerecorded elementary speech elements (typically diphones, two sequential phoneme units) as the basis for synthesis. In practice, the output is generated by concatenating a sequence of elementary speech elements corresponding to the

phoneme sequence identified by the NLP block. DSP algorithms are then applied in order to smooth the resulting waveform and apply the intended prosody. By the contrast to the rule-based approach, concatenative synthesis is strictly speaker dependent, as the speech database is generated from recordings of appropriately chosen text by a professional speaker.

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. 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 automatically thanks to 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-splittings. 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.

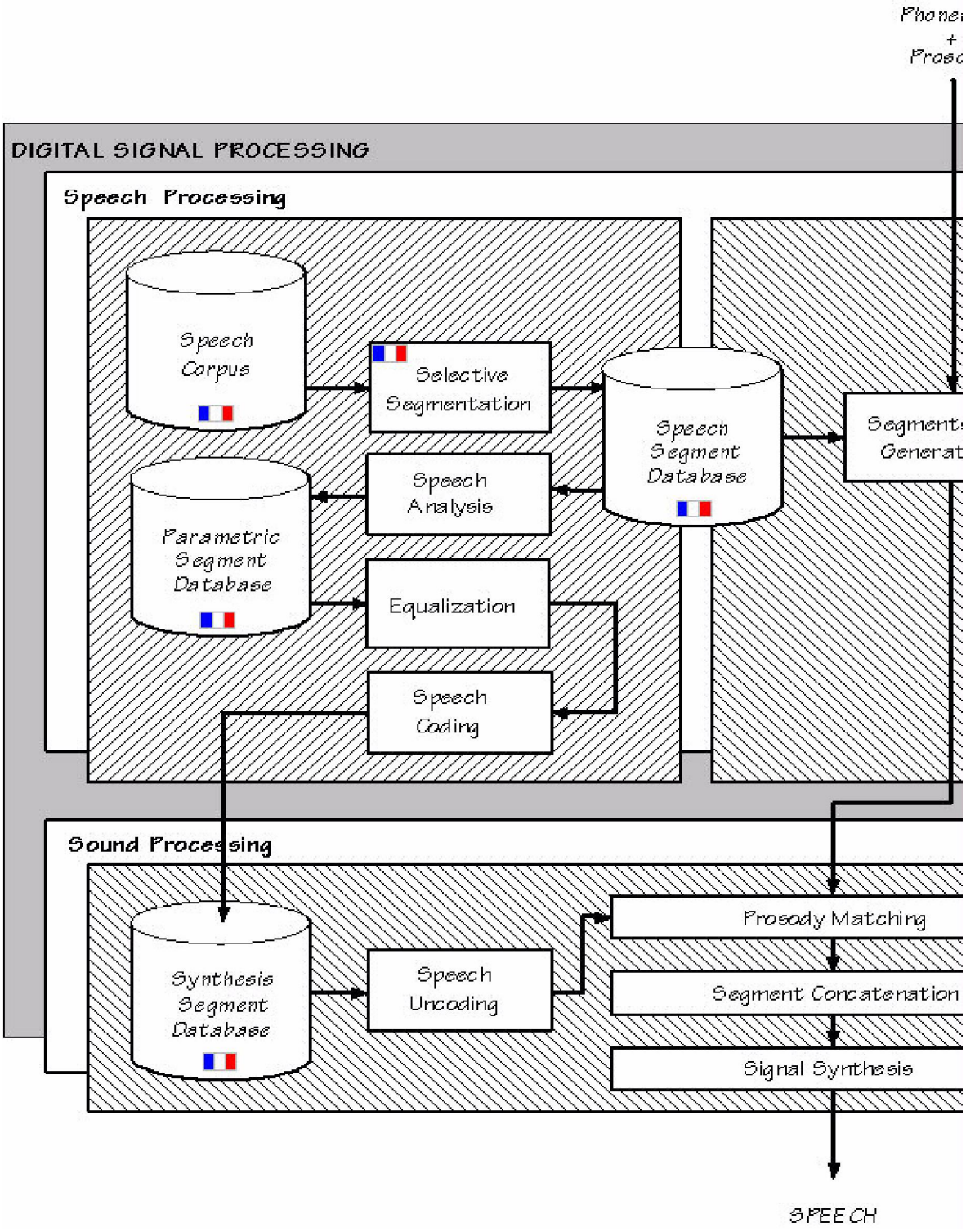

Phone
+
Prosc
DIGITAL SIGNAL PROCESSING
Speech Processing
Speech Corpus
Selective Segmentation
Speech Segment Database
Segments Generat
Speech Analysis
Parametric Segment Database
Equalization
Speech Coding
Sound Processing
Synthesis Segment Database
Speech Uncoding
Prosody Matching
Segment Concatenation
Signal Synthesis
SPEECH

fig:4

Figure 4. shows a general concatenation-based synthesizer. The upper left hatched

block corresponds to the development of the synthesizer (i.e. it is processed once for all). Other blocks correspond to run-time operations. 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 helps 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, the analyzer is often followed by a parametric speech coder.
- 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, that is 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, 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). Once the parametric segment database has been completed, synthesis itself can begin.

SPEECH SYNTHESIS

A sequence of segments is first deduced from the phonemic input of the synthesizer, in a block termed as segment list generation in Fig. 4, which interfaces the NLP and DSP modules. Once prosodic events have been correctly assigned to individual segments, the prosody matching module queries the synthesis segment database for the actual parameters, adequately uncoded, of the elementary sounds to be used, and adapts them one by one to the required prosody. The segment concatenation block is then in charge of dynamically matching segments to one another, by smoothing discontinuities. Here again, an adequate modelization of speech is highly profitable, provided simple interpolation schemes performed on its parameters approximately correspond to smooth acoustical transitions between sounds. The resulting stream of parameters is finally presented at the input of a synthesis block, the exact counterpart of the analysis one. Its task is to produce speech.

The efficiency of concatenative synthesizers to produce high quality speech is mainly subordinated to:

The type of segment chosen

Segments should exhibit some basic properties. They should allow accounting for as many co-articulatory effects as possible.Given the restricted smoothing capabilities of the concatenation block, they should be easily connectable. Their number and length should be kept as small as possible. On the other hand, longer units decrease the density of concatenation points, therefore providing better speech quality.

Similarly, an obvious way of accounting for articulatory phenomena is to provide many variants for each phoneme. This is clearly in contradiction with the limited memory constraint. Some trade-off is necessary. Diphones are often chosen. They are generally not too numerous for any language including hindi.

The model of speech signal, to which the analysis and synthesis algorithms refer

The models used in the context of concatenative synthesis can be roughly classified into two groups, depending on their relationship with the actual phonation process. Production models provide mathematical substitutes for the part respectively played by vocal folds, nasal and vocal tracts, and by the lips radiation. Their most representative members are Linear Prediction Coding (LPC) synthesizers, and the formant synthesizers. On the contrary, phenomenological models intentionally discard any reference to the human production mechanism. Among these pure digital signal processing tools, spectral and time-domain approaches are increasingly encountered in TTS systems. Two leading such models exist: the hybrid Harmonic/Stochastic (H/S) model and the Time-Domain Pitch-Synchronous-Overlap-Add (TD-PSOLA) one. The latter is a time-domain algorithm: it virtually uses no speech explicit speech model. It exhibits very interesting practical features: a very high speech quality (the best currently available) combined with a very low computational cost (7

operations per sample on the average). The hybrid Harmonic/stochastic model is intrinsically more powerful than the TD-PSOLA one, but it is also about ten times more computationally intensive. PSOLA synthesizers are now widely used in the speech synthesis community. The recently developed MBROLA algorithm even provides a time-domain algorithm which exhibits the very efficient smoothing capabilities of the H/S model (for the spectral envelope mismatches that cannot be avoided at concatenation points) as well as its very high data compression ratios (up to 10 with almost no additional computational cost) while keeping the computational complexity of PSOLA.

MATLAB 6.1

MATLAB is a high-performance language for technical computing. It integrates
computation, visualization, and programming in an easy-to-use environment where
problems and solutions are expressed in familiar mathematical notation. Typical uses include:

Math and computation
Algorithm development
Modeling, simulation, and prototyping
Data analysis, exploration, and visualization
Scientific and engineering graphics
Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems,
especially those with matrix and vector formulations, in a fraction of the time it would
take to write a program in a scalar non interactive language such as C or Fortran.

The name **MATLAB** stands for matrix laboratory. **MATLAB** was originally

written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, **MATLAB** uses software developed by the LAPACK and ARPACK projects, which together represent the state-of-the-art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, **MATLAB** is the tool of choice for high-productivity research, development, and analysis.

MATLAB features a family of application-specific solutions called toolboxes. Very important to most users of **MATLAB**, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of **MATLAB** functions (M-files) that extend the **MATLAB** environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

CONCLUSION

Let us bow to the facts : there is still a long way to the brilliant talking computer. A number of advances in the area of NLP or DSP, however, have recently boosted up the quality and naturalness of available voices, and this is likely to continue.

The particular module give to us is working fine. The given wave files are concatenated effectively. The Speech Synthesis Process is completely analyzed and concatenation of wave files is put into effect. Testing has been done and everything has gone according to our requirements. Further, the concatenated files are to be filtered and smoothened using TD-PSOLA and other filtering options. Also the approach whether to be Syllable based or Diphone based is to be decided for preparing the data base. Ours is just a step in the direction of '**M**ultilingual **T**ext to **S**peech **A**nalysis and **S**ynthesis' and a lot still needs to be done for developing an efficient system.

BIBLIOGRAPHY

SITES REFERRED

http://www.mathworks.com

http://www.google.co.in

BOOKS REFERRED

MATLAB Programming for engineers	**C**hapman j. Stephen
Getting Started With MATLAB	Pratap Rudra
Speech Synthesis	Streetmann

PAPERS REFERRED

A Short Introduction to Text to Speech Synthesis	Dutoit T.
Durational Characteristics of Hindi Consonant Clusters	Agrawal S. S.
A Wavelet-Domain PSOLA Approach	M. Holzapfel

TABLE OF CONTENTS

11. Overview of the Current Project

12. Conclusion

13. Bibliography

MAJOR PROJECT REPORT ON

MULTILINGUAL TEXT TO SPEECH ANALYSIS & SYNTHESIS

SUBMITTED IN PARTIAL FULFILLMENT OF REQUIREMENT
FOR THE AWARD OF THE DEGREE

MASTER OF ENGINEERING
IN
ELECTRONICS & COMMUNICATION
OF
DELHI COLLEGE OF ENGINEERING
DELHI

UNDER THE SUPERVISION OF

DR. ASOK BHATTACHARYA
H.O.D (E& C) DCE

MR. O.P. VERMA
ASTT. PROFESSOR
(E&C) DCE

DEVELOPED BY: -

JITENDER KADIAN

(17/E&C/2001)

DEPARTMENT OF ELECTRONICS & COMMUNICATION
DELHI COLLEGE OF ENGINEERING

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without a mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success.

I am grateful to **Dr. ASOK BHATTACHARYA** (HOD, ECE Deptt.) for providing us an opportunity to undertake this project. I am highly indebted to **Mr. O.P. VERMA** (Astt.Professor ECE Deptt.), my project guide for taking keen interest in my work and for his constant monitoring and invaluable guidance and support throughout the course of my project. I profusely thank him for having patience to clear our doubts and channelise my efforts his cheerful disposition made my work all the more enjoyable.

I would be failing in my duty if I do not thank my family members, friends and above all the lord almighty on whom I have relied on, for help and encouragement during my project work.

JITENDER KADIAN

INTRODUCTION

TOOLS USED

BIBLIOGRAPHY

GLOSSARY

Overview

Cool Edit Pro is a comprehensive multi track digital audio editor for Windows 95 and Windows NT. To use it, one needs only a PC compatible computer running one of these operating systems, one or more Windows-compatible sound cards, and some imagination.

Designed as a complete audio environment, Cool Edit Pro puts all of the functionality needed for taking an audio project from conception to completion in one package, eliminating the need for separate applications, or "plug-ins". From recording to mix down, you'll find everything to do the job in Cool Edit Pro, such as session notes for planning and sketching out your session and the flexibility of unlimited multiple sound card support, for complete freedom in routing your audio. Once one have recorded (…up to 64 tracks!), one can apply Cool Edit Pro's unmatched DSP and editing power. Tweak EQ, and other effects in real-time to get things just right. Arrange your tracks with drag and drop, down-to-the-sample or frame precision; snap-to guides, cross fades, and images will help you get it done. When you're ready to mix, there's separate level, pan, mute, solo, and routing for each track.

While Cool Edit Pro does provide an all in one audio solution, it also works with the rest of your audio tools as part of your studio. Support for Microsoft's DirectX/ActiveMovie means you can use DSP modules from leading manufacturers like Waves and Tools from within Cool Edit Pro. If you're needs go beyond audio to include working with MIDI or video, one find Cool Edit Pro's MIDI/SMPTE synchronization provides seamless integration with these mediums.

Though Cool Edit Pro is loaded with features, its layout is such that you're not encumbered by excess windows; you won't need a 20-inch monitor to get the job done. Get at just the features you want. You'll find this under-the-hood Architecture throughout Cool Edit Pro. **Above all, Cool Edit Pro is fun!** Once one played with it for a while, you'll see how it got its name!

Cool Features

Cool Edit Pro's combination of digital recording and editing features includes:

· Multi**-track editing and mixing**—up to 64 tracks!

· 30 **distinct effects modules**, including Delay and Echo effects, Noise/Hiss/Click-and-Pop Reduction, Reverberation, Pitch and Tempo Adjustment, Graphic and Parametric Equalizers, and more

· Support **for more than 16 distinct file formats** (not including sub formats)

· Noise**, Tone, and DTMF Tone generation**

· 32**-bit sample resolution support and full 32-bit internal processing**

· "**Phrase" recognition (auto cue generation)** (will change somewhat)

· Cue **and Play List** support for multiple files and for segments within files

· Scientific **filters, text export capability, and file-statistics** gathering Functions for data analysis

· Scripting **and Batch-Processing** capability

· Waveform **and Spectral View options**

· Multi**-track SMPTE/MTC chase locking synchronization**

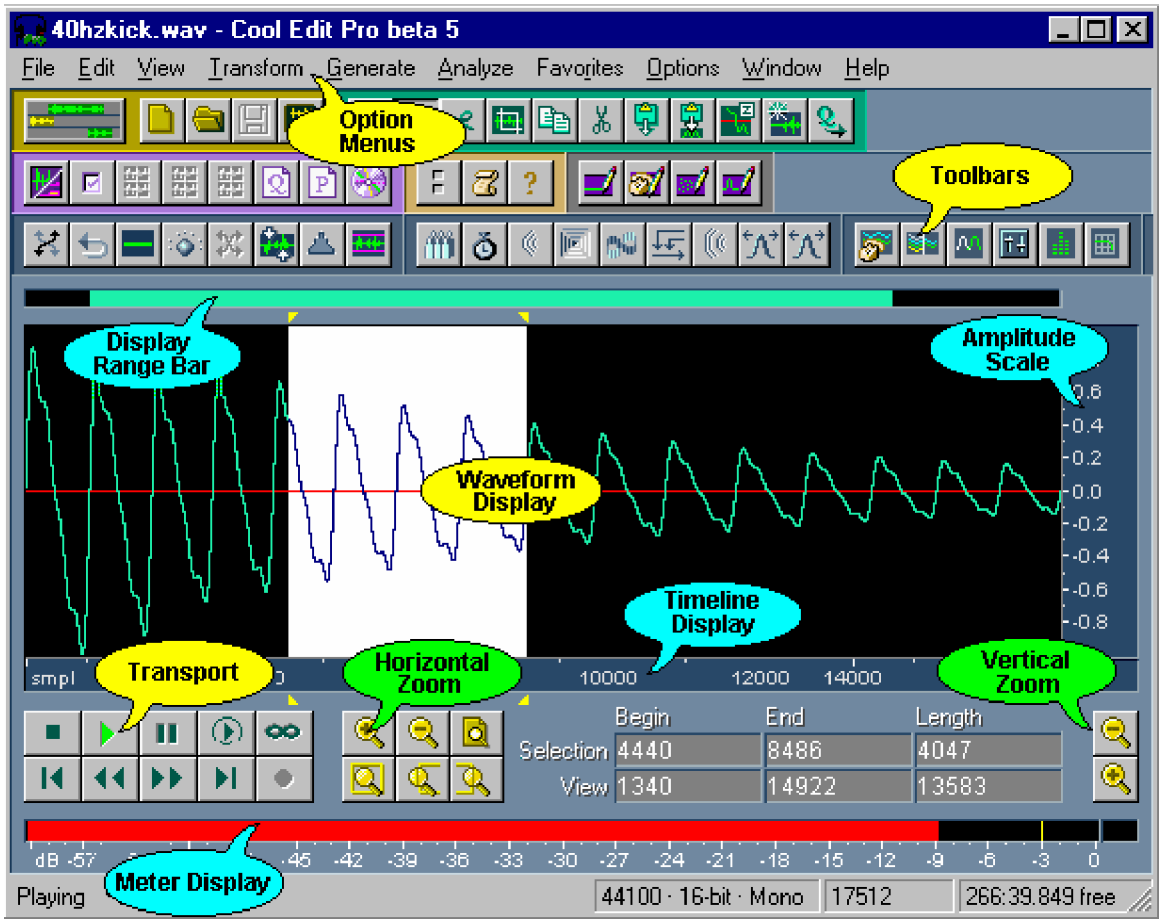
With Cool Edit Pro and any Windows sound card, you have the power of a Complete digital recording studio under your fingertips!

Notes on how to navigate your way around the manual

· The icon gives hints on "noteworthy" information.

· The icon alerts you to hints that are just too cool to pass up!

· The icon alerts you to a short simple exercise that can be used to Quickly learn about a specific Cool Edit Pro option or function.


40hzkick.wav - Cool Edit Pro beta 5
File Edit View Transform Generate Analyze Favorites Options Window Help
Option Menus
Toolbars
Display Range Bar
Amplitude Scale
Waveform Display
Timeline Display
Transport
Horizontal Zoom
Vertical Zoom
Begin End Length
Selection 4440 8486 4047
View 1340 14922 13583
Meter Display
Playing
44100 · 16-bit · Mono
17512
266:39.849 free

Glossary

CONCATENATION: - Addition of two or more equal or unequal sized sound waves in time domain is termed as concatenation.

COOL EDIT: - Cool Edit Pro is a comprehensive multi track digital audio editor for Windows 95 and Windows NT. To use it, one needs only a PC compatible computer running one of these operating systems, one or more Windows-compatible sound cards, and some imagination.

INTONATION: - The patterns of variation of the pitch of the voice (i.e the way in which the pitch varies) constitute the intonation of language.

NLP: - In addition with the expected letter to sound and prosody generation blocks, it comprises an analyzer the need for syntactic processing in a high quality TTS system.

PROSODY: - The term prosody refers to certain properties of the speech signal which are related to audible changes in pitch, loudness and any syllable length.

PSOLA: - Pitch synchronous overlap and add.

SYLLABLE: - While analyzing speech, we will have to consider units higher than individual speech sounds. The unit that is next in hierarchy to the speech sound is the syllable. A word is made up of one or more syllable.

SPLICING:- Splicing glues together whole words and phrases to create a remarkably natural sounding voice.

TTS: - Text to speech as the automatic production of speech , through a grapheme to phoneme transcription of the sentences to utter.