FACTA UNIVERSITATIS (NIŠ)
SER.: ELEC. ENERG. vol. 16, Dec. 2003, 355-364

Speech Signal Processing in ASR&TTS Algorithms

Vlado Delić, Darko Pekar, Radovan Obradović, and Milan Sečujski

Abstract: Speech signal processing and modeling in systems for continuous speech recognition and Text-to-Speech synthesis in Serbian language are described in this paper. Both systems are fully developed by the authors and do not use any third party software. Accuracy of the speech recognizer and intelligibility of the TTS system are in the range of the best solutions in the world, and all conditions are met for commercial use of these solutions.

Keywords: Signal processing, speech processing, speech recognition, text-to-speech synthesis.

1 Introduction

Automatic Speech Recognition (ASR) has a goal to train computers to understand human speech. On the other side, Text-to-Speech (TTS) synthesis has to teach computers to read any text. These tasks are considered very complex due to great variability of speech signal. But great possibilities for ASR&TTS applications is a challenge for many researchers all over the world.

Speech signal processing on PC was first performed on specialized hardware (sound blaster or Computer Telephony Integration (CTI) card) and later for CPU. Audio cards perform analog-to-digital and digital-to-analog conversion, speech signal coding, automatic gain control, adaptive channel equalization, echo canceling, voice activity detection etc. ASR and TTS

Manuscript received June 22, 2003. An earlier version of this paper was presented at the 10th Telecommunications Forum TELFOR 2002, November 26-28, 2002, Belgrade, Serbia.

The authors are with Faculty of Technical Sciences, University of Novi Sad (e-mail: `tlk_delic@uns.ns.ac.yu`).

algorithms are CPU based. They include some specific speech signal processing that requires much calculation. That is why CPUs could not run ASR and TTS in real time until several years ago. Now, Pentium IV PC is able to run several ASR and TTS algorithms simultaneously in acceptable time. Faster R& D and progress in both ASR and TTS are enabled by faster CPUs and larger memory, as well as by quality sound cards and CD writers.

Experiences in speech signal modeling and processing in our ASR and TTS algorithms are presented in this paper.

2 Signal Processing in AlfaNumaASR

2.1 Speech database for ASR

Large speech databases are necessary for R& D in ASR. They usually contain words, phrases and sentences, uttered by several hundreds or, sometimes, several thousands of spe- akers. Their utterances are recorded either via telephone line using a CTI card or in a speech studio or, sometimes in office environment . After recording of a speech database, every record should be listened to, and correct phonetic transcription should be created. To make this job easier and more comfortable, we have created a special software tool which is a part of our C++ Signal Processing Library - slib [1].

Slib is based on the idea of separating a DSP system into blocks with predefined functions, inputs and outputs. There are FIFO buffers between the blocks, whose size is chosen optimally according to features of CPU and available cache memory. This library can be used for common DSP operations like FIR, IIR, windowing, FFT, as well as for more complex signal processing such as ASR and TTS. For example, front end processing for ASR just requires block processing of speech frames. Blocks for extraction of more commonly used speech signal features have been already made and are available. Using slib, a programmer has only to choose and connect desired blocks and speech features are extracted and available for ASR. Slib is available in open source form at `www.alfanum.com`.

2.2 Front end processing

The aim of front end speech signal processing is to extract features, i.e. to code speech signal frames in a form suited for ASR. Feature extraction in our ASR is based on slib library and its script version slib_script. Feature extractor is actually a text file with described blocks, used for feature calculation. Many speech signal features are available: cepstral coefficients (several forms), energy, zero crossing rate (ZCR), pitch, voiced/unvoiced (V/UV) feature, as well as derivatives of these static features.

Our software saves extracted features in HTK parameter file format [5] and enables comparative analysis with this well known ASR software. AlfaNumASR in several tests outperformed HTK in the same conditions [4]. Our software supports other possibilities such as energy normalization, cepstral mean substraction etc.

AlfaNumASR recognizes phonemes in context. Phonemes and subphonemes are modeled with continuous and semi-continuous HMMs. A sequential model is used, so it is possible to transfer only to the next HMM state. This is the case with states within one phoneme. Transfers between different phonemes and words can be arbitrarily complex.

Standard Gaussian mixtures with diagonal covariance matrix are used. Two approaches for state probabilities calculation are used: calculation over a weighted sum of Gaussian mixtures, and winner take all method [2]. Second approach has proved equally good as the first one, while providing significant time savings.

After state probability calculation in certain time instants, optimal sequence is found by Viterbi algorithm. One well known method for modeling state durations is via probability for staying in current state and probability for transition to the next state, Fig. 1.

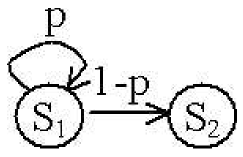

Fig. 1. Classical state duration modeling.

It is also known that this model gives an exponential distribution of state durations that doesn't match real, natural distribution which would optimally model phoneme durations.

Besides this classical approach, AlfaNumASR supports another model which models state durations optimally. Let us suppose that state S1 had a duration distribution in the training database as shown in Fig. 2.

It could then be divided into several states, as shown in Fig. 3.

Unmarked transition probabilities (those within the same macro state) are 1. It is obvious that this model acquires the very distribution we had in training database. However, the number of states is drastically increased

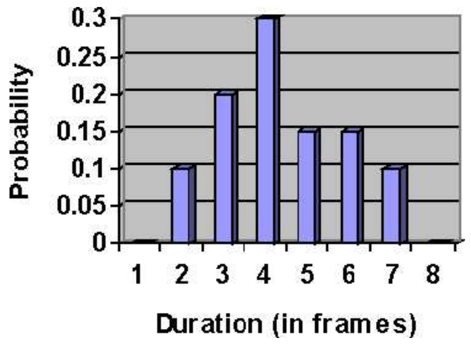

Fig. 2. Duration distribution of state S_1.

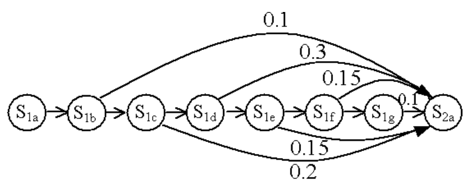

Fig. 3. Expanded model for duration modeling.

(for some states up to several dozen times). These are the same states, from the acoustic point of view, but nevertheless significantly burden Viterbi algorithm and lag recognition. By using this model, certain accuracy improvement is gained, but accuracy vs. CPU time ratio is not satisfactory. Hence it should be used when CPU power is not a problem.

Although better than standard, this method still does not treat another problem of state duration modeling. Joint state probability distribution is not modeled here, and they are treated as statistically independent. That is not correct. In the end the overall probability of certain state is calculated as

$$P = P_t(T = N|S)^K \prod_{n=1}^{N} P_a(S_1|t = n) \tag{1}$$

where P_t is the probability that the state S_1 lasted N frames, and P_a is the acoustical probability of state S_1 in the instant n. If K equaled 1, this would be a classical case, in accordance with probability theory. However, if this coefficient were given some values bigger than 1, it would increase the importance of duration probability compared to the acoustical, and the experiments showed that this can lead to recognition accuracy improvement. This coefficient can be set in the configuration file of AlfaNumASR system.

As once mentioned, part of the program for training requires presence of labeled audio sequences, i.e. a label file with correctly set boundaries between speech units. It is obvious that this would require huge amount of work if all done by hand. Therefore we provided an option for automatic labeling, if some initial acoustic models are provided, as well as a label file with correct phonetic transcription. This is done by building a trellis which would force the recognizer to run through all phonemes in the label, and then let Viterbi algorithm to find the optimal path. Obtained segmentation is recorded into the file.

Of course, some initial labeling would have to be done by hand. By using a relatively small number of labels done by hand, an initial model can be built. It will be relatively poor, but can serve to automatize the process of labeling in the next iteration. Then we can perform automatic labeling on the whole base, and then train the system with those labels, discarding the outliers during the training. In the next iteration we do the same, but with the new model etc. During automatic labeling several details should be observed

1. Models should have small number of mixtures (not more than two).
2. It is wise to decrease frame_duration parameter in order to obtain more precise segmentation.
3. Speakers should be divided into several groups, at least by gender.
4. For duration modeling it is better to use second approach, using real duration probability.

2.3 Word spotting and wildcards usage

In this terminology the term wildcard represents a model which could describe any phoneme, sufficiently well, but its acoustic probability should be less than that of the correct model. If we had such model at our disposal we could make the grammar shown in Fig. 4.

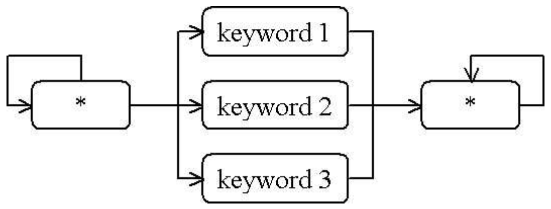

Fig. 4. Transition diagram of the word-spotter.

If any of the keywords is pronunced in test sequence, optimal path through such grammar would include that keyword and it would be recognized ("spotted"). If no keyword is spoken, staying in wildcard (*) state would have the highest probability.

3 Signal Processing in AlfaNumTTS

3.1 Speech database for TTS

The speech database contains approximately two hours of continuous speech, pronounced by a single female speaker. Having in mind possible applications of such a system, and the fact that concatenation of longer speech segments yields more intelligible speech, it was decided that the database should include phrases such as first and last names, addresses, names of companies, amounts of money, time and date phrases, weather reports, horoscope reports etc. The database was recorded in laboratory conditions, and submitted to several off-line operations necessary for imple- menting TTS, such as phoneme labeling and pitch-marking.

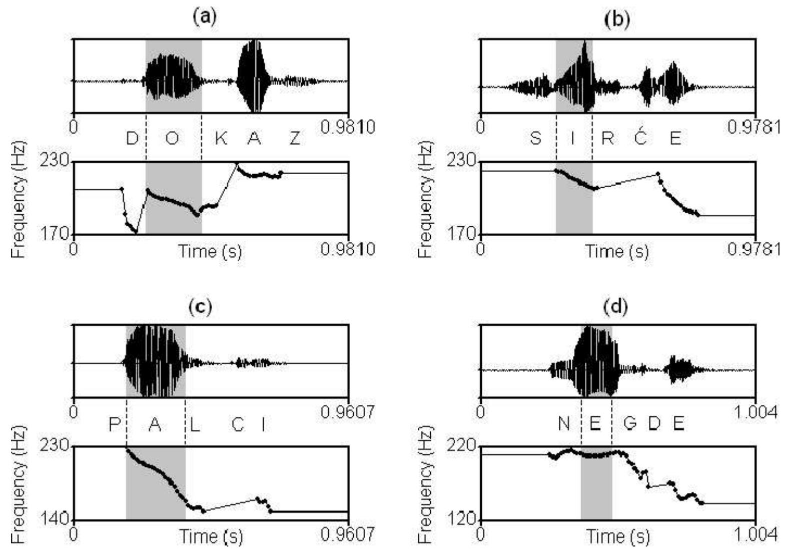

Fig. 5. Stress types in Serbian language and coresponding pitch contours: (a) rise/long, (b) rise/short, (c) fall/long, (d) fall/short; as pronounced by the speaker involved in database recording.

The labeling of the database implies placing boundaries between units belonging to a previously established set of units such as phonemes. It actually implies storing information about units in a separate database. The labeling of the AlfaNum speech database was predominantly phoneme based, although in some cases a better alternative was adopted, due to certain phonetic features of particular phones, as well as certain peculiarities of Serbian language. For instance, some classes of phones, such as plosives and affricates, were considered as pairs of semiphones (including occlusion and explosion). Vowels belonging to classes with significantly different timbres were considered as different vowels altogether, and therefore not interchangeable. Beside unit boundaries, a part of each unit least affected by coarticulation was identified, in order to make further prosody modifications less audible. Since the database contains continuous speech abounding in phones impaired in various degrees, such cases were all noted including the degree of impairment, in order to avoid using these phones in contexts where they are usually well-articulated, for instance, within stressed syllables. This refers to both vowels and consonants. Labeling was performed automatically, using the AlfaNumCASR continuous speech recognizer [4] and verified by human experts. Verifying was based on signal waveform, its spectrogram and its auditory perception. The SpeechVis software available at [10], previously developed within the AlfaNum project, was used. Implementing TD-PSOLA algorithm implies previous pitch-marking of the database, that is, detecting locations within phones most suitable for centering overlapping windows and extracting frames, in case a TTS algorithm which requires pitch-synchronous frame positioning should be used.

As to positioning pitch markers within voiced frames, the process was carried out in two phases. To begin with, preliminary estimations of pitch contours of each segment were made using AMDF pitch-extraction method. Each of the segments was previously low pass filtered with fc = 900 Hz. The next step was locating the frame with the highest degree of voicing and locating the maximum peak within that frame. The initial pitch marker was placed there. Afterwards, the search for other pitch markers was conducted according to preliminary pitch estimations.

Such a procedure is not entirely error-free, because low pass filtering can sometimes modify peak values to such an extent that peaks recognized as maximum in some segments do not coincide with peaks recognized as maximum in the rest. This can happen in case of voiced phones whose waveforms have two prominent peaks of roughly the same height. There is another reason why errors of this kind may occur. At the precise spot of boundaries between phones there is sometimes an irregularity in functioning of the glottis, leading to a discontinuity in positions of glotal impulses. If a phoneme-based pitch-marking is adopted, that is, if pitchmarking is performed independently within phones, and not within speech segments containing more than one phone, most of such errors are eliminated completely.

3.2 Prosody generation

Acoustic parameters such as f_0 contour were calculated in two steps. The first step consisted of analyzing the sentence, word by word, in order to get information such as stress types and locations, part of speech classes and functions of particular words in the sentence. Grammatical information is essential for synthesis of natural-sounding prosody, since words in Serbian language can sometimes be stressed in different ways depending on their morphologic categories, and sometimes even have different meanings if stressed differently. In some cases even syntactic analysis does not help.

3.3 The dictionary

Since stress in Serbian language is fairly unpredictable, a dictionary-based solution was adopted. A special dictionary including information on stress configuration, part of speech class and morphologic categories for each word was created. Furthermore, since stress can vary along with inflections of the same word, and those variations are predictable only to a certain extent, it was necessary to include all word forms as separate entries in the dictionary. Several part of speech classes were identified as having regular behavior when submitted to inflection and they were entered in the dictionary in a form which occupied little space, but was sufficient for correct determination of the stress of every inflected form. In such a way the dictionary contains more than two million entries (including inflected forms). Such a solution is not entirely error-free, since it does not include syntactic analysis, nor does it solve cases when syntactic ambiguities arise, and semantic analysis, however primitive, must be performed. It was decided to leave these two problems for later stages of the project. Another problem that occurs is that some words may not be found in the dictionary. In that case, strategies for determining the correct way of stressing must be defined. Strategies currently being developed within our projects include making analogies based on standard prefixes and suffixes and rhyming.

3.4 f_0 codebook

Using information from the dictionary, the system is able to reconstruct a particular stress configuration of a group of words which form a metrical unit. In this phase of the project, several f0 contours are assigned to each metrical unit, depending on its position in the sentence (beginning, neutral, before comma, ending), and the resulting f0 contour is smoothed in order to avoid audible pitch discontinuities and tilted towards the end of the sentence [8], [9]. Such a method does not take into account syntactical information, but relies only on punctuation marks. However, results are still significantly better than in case of synthesizers with constant f0, available in Serbian until now.

3.5 On-line selection of segments

Halfphones are considered as basic units which cannot be further segmented, but it is desirable to extract segments as large as possible, in order to preserve intelligibility. According to differences between existing and required values of parameters previously defined, each speech segment which can be extracted and used for synthesis is assigned target cost, and according to differences at the boundaries between two segments, each pair of segments which can be concatenated is assigned concatenation cost [6]. Target cost is the measure of dissimilarity between existing and required prosodic features of segments. Concatenation cost is the measure of mismatch of the same features across unit boundaries. Various phoneme groups are treated in different fashion. Some types of phonemes, such as unvoiced plosives, are more suitable for segmentation than the others, and have lower concatenation costs. The degree of impairment of phones is also taken into account.

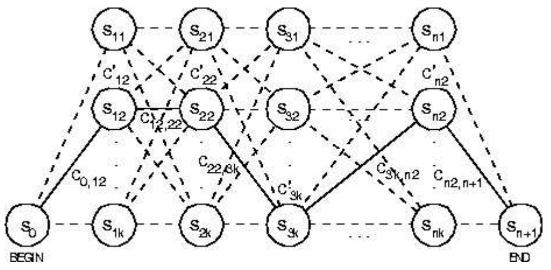

Fig. 6. Finding a best path through a trellis representing a sentence.

The task of the synthesizer is to find the best path through a trellis which represents the sentence, that is, the path along which the least overall cost is accumulated. The chosen path determines which segments are to be used for concatenation, as shown on Figure 6.

4 Conclusion

ASR and TTS are language dependent and R& D projects in these fields require multidisciplinary teams of experts: acoustic, linguistic, programming, mathematic, as well as signal processing. Such an R& D group at Faculty of Engineering, named AlfaNum, has developed ASR and TTS engines for Serbian language. The authors of this paper are leaders of the AlfaNum group. In this paper a review of speech signal processing in our ASR and TTS algorithms has been made. Many standard techniques were used, and some new features introduced, especially in case of prosody generation for Serbian language. More than 3MB of source code in C++ has been written. A stress dictionary for Serbian language with over two million entries was made for the purposes of TTS.

References

[1] D. Pekar and R. Obradović, "C++ library for digital signal processing - slib," in *Proc. TELFOR*, (Belgrade), November 2001.

[2] R. Obradović, D. Pekar, S. Krčo, V. Delić, and V. Šenk, "A robust speaker independent cpu based speech recognition system," in *Eurospeech, Vol.6*, (Budaperst), pp. 2881–2884, September 1999.

[3] D. Pekar, R. Obradović, and V. Delić, "Connected words recognition," in *Proc. TELFOR*, (Belgrade), November 2001.

[4] D. Pekar, R. Obradović, and V. Delić, "Programski paket AlfaNumCASR sistem za prepoznavanje kontinualnog govora," in *Proc. DOGS*, (Bečej), pp. 49–56, 2002.

[5] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland, *The HTK Book.* Microsoft Corporation, 1995-1999.

[6] M. Sečujski, R. Obradović, D. Pekar, and L. Jovanov, "Sinteza govora na srpskom jeziku povezivanjem segmenata odabranih u realnom vremenu," in *Proc. ETRAN*, (Teslić), 2002.

[7] M. Sečujski, "AlfaNum system for speech synthesis in serbian language," in *Proc. TSD*, (Brno), pp. 237–244, 2002.

[8] M. Sečujski, "Prosody in synthesized speech in serbian language," Master's thesis, FTN, Novi Sad, 2002.

[9] T. Dutoit, *An Introduction to Text-to-Speech Synthesis.* Dordrecht Boston London: Kluwer Academic Publishers, 1997.