INDIVIDUAL SOUNDING SPEECH SYNTHESIS BY RULE USING THE MICRophonemic METHOD

Lauri Lehtinen and Matti Karjalainen
Helsinki University of Technology
Acoustics Laboratory
Otakaari 5 A
SF-02150 Espoo, Finland

ABSTRACT

This paper describes the microphonemic speech synthesis method and its implementation in the form of a text-to-speech synthesizer. The synthesis is based on combining waveforms that are collected from real speech, thus retaining the most part of the individual features of the speaker. The rule-based text-to-speech synthesis is based on object-oriented representations of the hierarchical structure of the units in speech. Syllable classes play an important role as a unit for the prosodic structure.

INTRODUCTION

Speech synthesis based on vocal-tract models is considered to be superior in speech quality to time-domain waveform sampling methods. The main problems when concatenating real-speech samples have been the realization of smooth formant transitions and the control of pitch frequency. By proper concatenation and overlap-addition of pitch period prototypes including intermediate samples taken from the transitions we have earlier demonstrated the extremely high quality of reconstructed speech [1]. Similar principles are published also by other researchers e.g. Hamon [2].

Microphonemic synthesis is a method to synthesize speech using a database of pitch periods, transients, noise segments and their positions extracted from uttered speech examples [4]. The microphonemic method to concatenate these speech units yields high-quality and individual sounding speech [1], [3], [4]. Together with a text analyzer and language dependent set of rules to handle prosodic features, microphonemic synthesis forms a complete text-to-speech synthesizer (fig. 1). Our text-to-speech synthesis prototype based on the microphonemic method is implemented on the basis of the object oriented signal processing system QuickSig [5] and it runs on the Symbolics Lisp machines. The present synthesizer is not yet a full scale implementation; only some of the phoneme classes in the Finnish language are realized.

Although the prototype is a large and complex software system not running in real time the method as such is well suited to real-time syntheses on normal microprocessors. It is possible to realize it as a software synthesis method without any extra hardware e.g. for the Apple Macintosh.

DATABASE OF SPEECH SAMPLES

The most laborious task during the development of the synthesizer is the extraction of speech units from real speech even if our purpose was only to model the voice of one speaker. The extraction process is also so critical to the success of the whole project that it is difficult to believe that one could manage without first developing at least a semiautomatic extraction algorithm [1].

When speech units have been extracted a database is formed so that each speech unit belongs to a data object called a prototype which in addition contains also other useful data about the unit. Those prototypes which are extracted from the same diphone are gathered in a diphone object. In addition to prototypes a diphone contains the knowledge about the extraction context as well as duration, frequency and amplitude values of the transition.

TEXT ANALYSIS

The highest level of the text-to-speech synthesizer divides the text into parts and creates a hierarchical object structure (fig. 2) down to syllables. Text consists of sentences, which again consist of clauses and so on. Finally words are divided into syllables, which consist of graphemes. The hierarchical structure is represented by using the so
called relation-mixins that support relations like parts, part of, next, previous etc.

The usual phase in text-to-speech synthesis of converting graphemes to phonemes is omitted, because in Finnish the mapping is almost one-to-one and can be done with only a few rules. In addition to the links parts, part of, next and previous, objects have of course other properties too. For instance, objects down to syllables in structure contain default values or coefficients of gain and frequency contours.

The synthesizer starts from the top of the object structure. From the text-object the synthesizer finds the reference level of gain and fundamental frequency and continues to process the parts of the text. The main task for the synthesizer as it moves down the structure to syllables is to define gain and fundamental frequency contours to which every object has its contribution depending on the context.

The synthesizer creates objects we call speech-syllables which contain essentially a list of diphones that are needed to synthesize the syllable. The search for the right diphones is done symbolically by building the diphone symbols from the graphemes belonging to the syllable. Next the synthesizer applies some rules to check the context and to decide which of the pitch periods are actually needed to synthesize the syllable. It has turned out that this can be done by using only the information about the syllable class and about the syllable classes of the neighboring syllables instead of the syllables themselves. As the process goes on the synthesizer creates an object called a speech-syllable-segment which contains those prototypes that are needed from each diphone. Up to this point the synthesizer operates mainly with discrete symbols and units opposite to vocal-tract synthesizers which operate with continuous-time control parameters.

By using only the knowledge of the syllable class we have managed to tabulate the default durations and both relative gain and relative F0 contours for speech-syllables and speech-syllable-segments for several contexts and to create context dependent selection rules.

At the last phase the location in time of each prototype and prosodic parameters are computed. The last thing to do is to create prototype-pairs which form the input to the microphonemic synthesizer. Prototype-pairs contain the knowledge derived from the types of the prototypes of how the prototypes are to be modified and concatenated and how those pitch periods which had not been extracted should be synthesized.

**MICROPHONEMIC SYNTHESIS**

Since only prototypes, not all pitch periods, were picked up from the real speech it is necessary to have means to approximate those pitch periods between prototypes. Fig. 3. illustrates the microphonemic idea to synthesize intermediate pitch periods in formant transitions. In addition it must be possible to adjust the pitch and to control the loudness of the synthetic speech signal.

The amplitude control is carried out by defining the gain with the help of precalculated peak-to-peak values at the beginning of each prototype. The gain factors for the pitch periods between two prototypes are adjusted automatically by the interpolation process (fig. 4). The fundamental frequency is determined by how frequently pitch periods are placed on the time axis. When F0 is decreased gaps between pitch periods will be left. Because the spectral envelope of a pitch period doesn't on range by inserting zero samples at the end of a phoneme quality is retained in the process. To increase the fundamental frequency is in theory more problematic.
than to decrease it because of some phase cancellation in overlapping regions. In practice, however, the phase cancellation is insignificant because the extraction algorithm [1] takes care that in normal cases both pitch periods have only small sample values in the overlapping region. Plus or minus one octave pitch scaling from the original one works well for the male voice (Fig. 4).

Some phoneme classes need special processing. Short units like transients were stored as direct waveform segments in several vowel contexts. The synthesis of transients is very easy, because only gain must be adjusted before they can be added to the output signal. Fricatives are a bit harder to synthesize because in addition to gain also duration must be adjusted. This is realized by concatenating a time-randomized selection of 10 ms signal segments extracted from real fricatives in all vowel contexts [3].

A more serious phase cancellation problem arising from interpolation and occurring in the middle of the interpolation period can be corrected to be unperceivable by adding pitch periods to the time axis so that their maximum points are always time aligned (Fig. 4).

The microphonemic method approximates transitions by interpolating between two prototypes. In reality the interpolation should also be done between the resonant frequencies. This basic restriction on the method is shown in Fig. 5. The importance of the frequency interpolation decreases as the difference between the initial and final states decreases and becomes unperceivable when the difference between the resonant frequencies is less than 2 Barks [3].

CONCLUSION

The experiments on the microphonemic method we have conducted have given us encouraging impression that an inexpensive text-to-speech synthesizer with natural, high-quality and individual sounding voice is possible.

REFERENCES