Dialog planning in VoiceXML

Csapó Tamás Gábor <csapot AT tmit.bme.hu>
Tarján Balázs

4 January 2011

1. Theoretical introduction

The measurement is designed to familiarize students with the opportunities offered by the VoiceXML programming language. VoiceXML is a standard markup language, which is designed to facilitate and accelerate development of speech-driven human-computer dialogues. VoiceXML is based on similar principles than the HTML language, but the former describes speech and the latter defines the display of visual contents.
This introduction briefly describes the basics of dialog planning, speech synthesis and speech recognition, which are required to carry out the measurement.

1.1 Dialog systems

The goal of the dialogue systems is to assist the proper functioning of human-machine interfaces using speech technology devices. To create a dialogue-based system, a development environment is needed with which the task best suited to the requirements of the dialogue system can be created. The system supports the integration of speech generator (also known as speech synthesis) and robust speech recognition engines [1].

1.1 Speech synthesis

Speech synthesis is nothing more than production of human-like speech in an artificial manner, typically using a computer. If the input is written text, it is called Text-To-Speech (or TTS, briefly). This text is converted through various steps to human-like speech, as shown in Fig 1. In general, these steps are the text-to-text processing of the input, preparation of the synthesis and the creation of speech [2]. An intermediate step in this process is the design of prosody, which means that the melody, rhythm, emphasis, type and position of stress are assigned to the text. To determine these, only the input text is available, which makes this step difficult. After the preparation, the speech synthesizer generates the output speech from the marked input data.

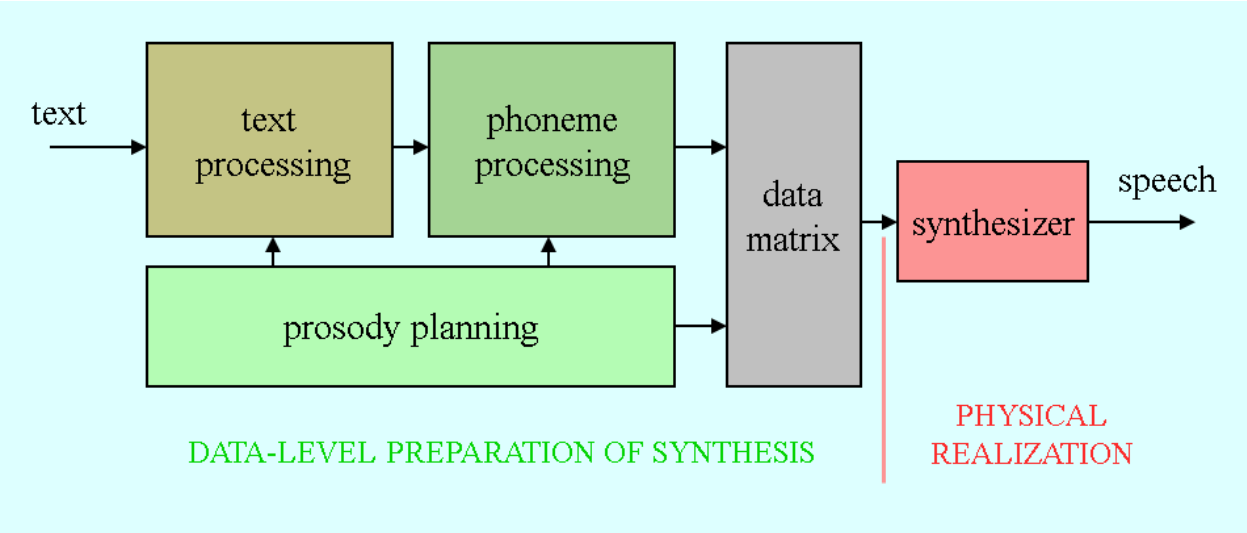

Fig 1. Scheme of a general Text-To-Speech system. The operation consists of two main steps: creation of symbolic information based on the input text (left side), which is the basis of synthesizing the waveform (right side). Source: [2, page 303].

Next, the different generations of speech synthesizers are introduced [3, 4].

1.1.1 Formant synthesis

Formant synthesis was the first technology that allowed automatic conversion of text into intelligible speech. The system tries to model human speech formants to create phones. The sound of such systems is rather "robotic", so they are rarely used today.

1.1.2 Concatenative synthesis

In concatenative speech synthesis, speech waveform elements cut-out from natural speech are concatenated. Previous experiments have shown that the intelligibility of sound transitions (not the speech sounds themselves) are responsible for the perceived naturalness of synthetic speech. Therefore the proper modeling of sound transitions is extremely important. Concatenative systems are distinguished depending on the size of elements used. Of course, this also affects the number of elements needed: while for the Hungarian diphone synthesis 38^2 = 1444 elements required, from triphone elements 38^3 = 54,872 elements would be needed. In practice, using the complete diphone coverage and the 1000-2000 most common triphone elements good quality can be achieved.

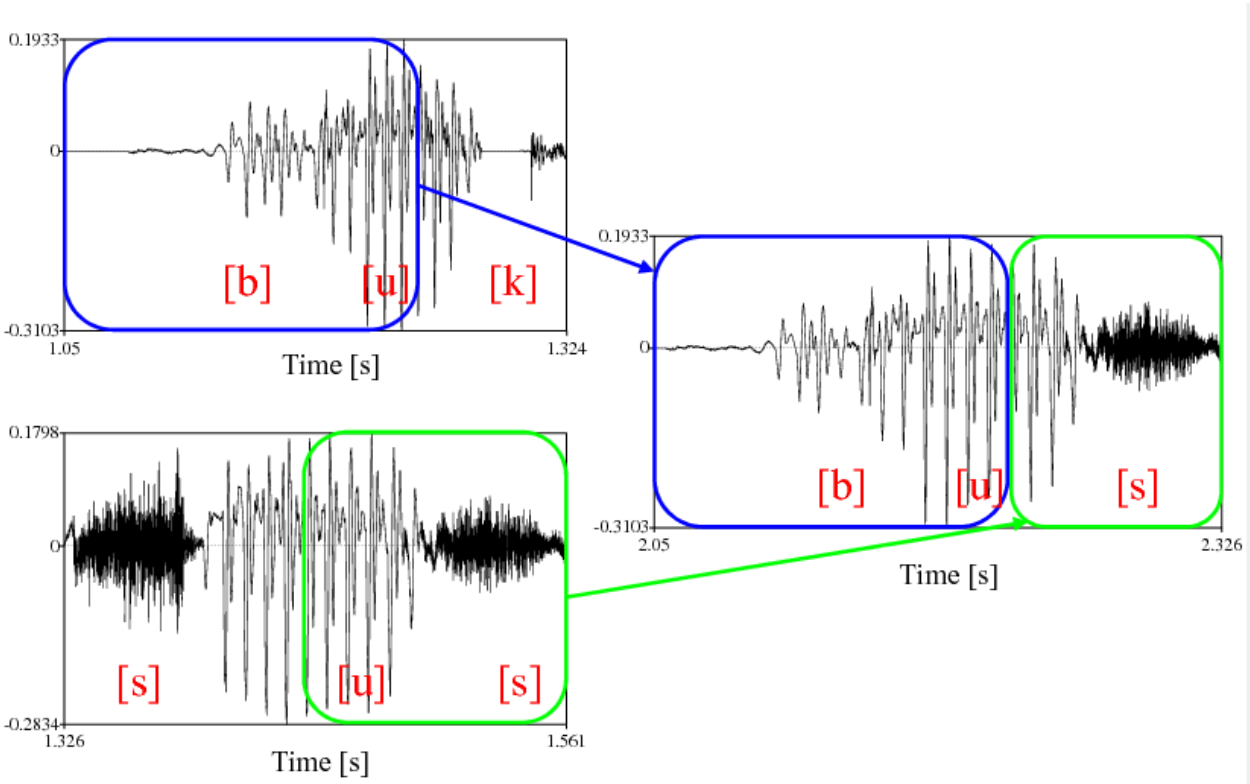

Fig 2. Concatenation of diphones (sound transitions): from „bu" and „us" the word „bus" arises.
Source: based on [5].

1.1.3 Corpus based, unit selection synthesis

The improvement of the concatenation is the unit selection speech synthesis technology. The novelty here is first, that a larger corpus (speech database) is available, in which an item can occur several times, in different forms. On the other hand, these elements are longer: words or phrases are possible as well.
While creating the speech output, the system is looking for longer elements of the corpus which match the input text. The elements are longer compared to the diphone and triphone systems, reducing the number of concatenation points in the produced speech. Since a given speech unit may occur in different forms within the corpus (different melody, intensity), the quality of synthesized speech can be improved by choosing the most natural ones. However, the quality of the system is also affected by how close the input text and the topic of the speech corpus are to each other.

1.1.4 Hidden Markov Model based speech synthesis

The statistical, Hidden Markov Model (HMM) based speech synthesis systems have become increasingly popular in recent years (e.g. HTS [6]). The main limit of the unit selection systems is that they use natural speech samples. Thus, for different voices (people, speaking styles) it is necessary to record a huge database, the production of which is quite costly.

In contrast, to this new technology a sort of training corpus is enough, of which the system generates context-sensitive HMM output, and the output waveform is generated based on these. The training is similar to the speech recognition (since HMM-s were originally used for this technology), and the waveform is the result of the synthesis. This method allows the modeling of different people, emotions with appropriate modification of HMM parameters.
This technology is not yet fully mature, and a strong research and development is taking place in the field of statistical-based TTS-s.

1.1.5 Comparison of speech synthesis technologies

	Pros	Cons
Formant synthesis	small footprint	"robotic" voice, lots of parameters
Concatenation	small footprint, prosody easily modifiable	distortion caused by signal processing
Unit selection	nearly natural	large storage requirements, prosody hardly modifiable
Hidden Markov model	technology used in speech recognition	slow training

Table 1: Comparison of speech synthesis technologies

Speech synthesizers have gradually changed over the past 25 years. From the simplest models we reached the technologies applying complex models, as summarized in Table 1. The formant synthesizers can create "robotic" voice, while using little resources. While the diphone-triphone systems use small databases, they are able to produce human-like speech. Using the corpus-based unit selection speech synthesis almost entirely natural speech can be produced. The latest, hidden Markov model-based systems have small memory requirements and even so good quality can be synthesized.

During the measurement, the Profivox Text-To-Speech synthesizer developed in BME TMIT is used [7]. The Profivox is a Hungarian speech synthesizer, which has 1444 diphone and 6000 CVC-triphone elements. The system has several speech styles, of which we use a male voice.

1.2 Speech recognition

The purpose of the Automatic Speech Recognition (ASR) is to convert the acoustic speech signal to text, doing essentially the inverse process of speech synthesis. The recognition is usually divided into two distinct phases. The first phase is a signal processing step called **feature extraction**, in which the feature parameters that characterize the content of the speech are extracted. In the second so-called **pattern matching** phase the previously received parameter vectors are fitted with a stored model of the language. As a result of the process, the word or word sequence that best matches the input speech is the output of speech recognition [8, 9].

1.2.1 Feature extraction

Human speech signal is very complex, so a complex processing is required for the extraction of the parameters that characterize the contents of speech.

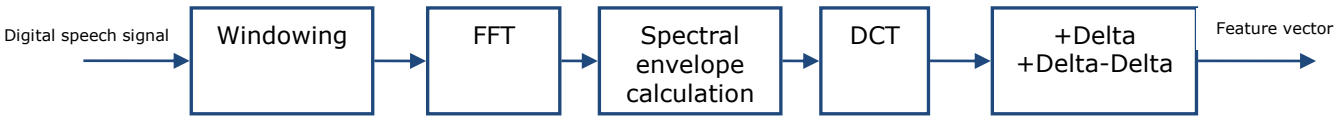

Fig 3. Process of feature extraction.

The signal processing is done on digital speech signal, for which the time function of speech is sampled (~ 8-22 kHz) and quantized (8 bits, 16 bits) depending on the task. In the first step, the digitized signal is split into sections that fit to the duration of human speech sounds (10-30ms), by interlaced window functions (e.g. Hamming). According to our knowledge, the human ear carries out harmonic vibration analysis, so it is obvious to treat the problem in frequency domain. In windowed blocks the signal is considered as periodic, and spectra can be obtained by FFT (Fast Fourier Transform) algorithm. Several methods exist for the calculation of feature vectors from spectral components. Here, the calculation of the most widely used Mel Frequency Cepstral Coefficients (MFCC) is shown briefly.

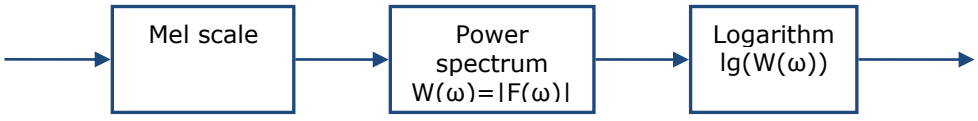

Fig 4. Calculation of MFCC spectral envelope.

To extract more concise features, the FFT spectrum components are averaged. An important characteristic of the human hearing is that the frequency resolution decreases exponentially with increasing the frequency. As a consequence, in the mel-scale averaging that is applied on the spectral components the width of the summarizing window is exponentially increased above 1 kHz, thus compensating the smaller information density. From the mel-sum of the components power spectrum is calculated from noise suppression consideration, and the resulting values are logarithmized according to the often observed relation between the stimulus and feeling. The following step is the DCT (Discrete Cosine Transform), a process that serves to reduce the dimension of the feature vector. The final step is assigning the linear regression estimated time derivatives (Delta and Delta-Delta) to the static elements.

The latter greatly improves the efficiency of feature extraction. In addition, the feature extractor may contain a number of items (noise and distortion reduction), but these are not discussed here in detail.
The feature extraction is a signal processing step, which creates standardized, discrete-time signal from continuous speech.

1.2.2 Pattern matching

The task of pattern matching is to map a feature vector sequence to a dictionary item or sequence of dictionary items (isolated word vs. continuous recognition). Today, most widely **statistical** based recognizers are used where the vector sequences are fitted to a HMM-based probabilistic model structure that is estimated based on training data. This model can be divided into several hierarchical levels (acoustic models, language models), each of which can be interpreted as Weighted Finite State Transducers (WFST) [10]. The WFST-s can be derived from the Finite State Automatons (FSA) with weights placed on the edges (transition probabilities) and extended with output symbols. So they are suitable to process the labels which come from different levels of hierarchy from the bottom to up. The outcomes of the process of pattern matching are the output labels of the top model (language model) and their timestamps, which belong to the best fitting route.

Acoustic model

In the full recognition network usually three acoustic layers are distinguished, however, these can be seen together as a WFST converting only one feature vector sequence to phoneme sequence. The training of the acoustic model is done on labeled speech database. Based on the available training speech and its textual transcript we estimate the conditional distribution of the feature vectors for the spoken phonemes. With the resulting statistical model we can give an estimate to the measure of acoustical match between a feature vectore sequence and phoneme sequence.

Language model

The language model defines how and to what probability the dictionary elements of the recognizer can be linked. The structure of a language model is mainly determined by the intended task of the recognition system. In isolated word recognition usually a parallel structure is used, where we can pass through each lexical item with the same probability during the pattern matching. In such cases, the weight provided by acoustic model is the basis of our decision (Fig. 5). In contrast when continuous speech is recognized, the probability of connecting lexical items is estimated based on training text. This much more complex model structure leads to an opportunity to prevent the uncertainties in the estimation of acoustic model. The WFST created by combining language and acoustic models is called the recognition network.

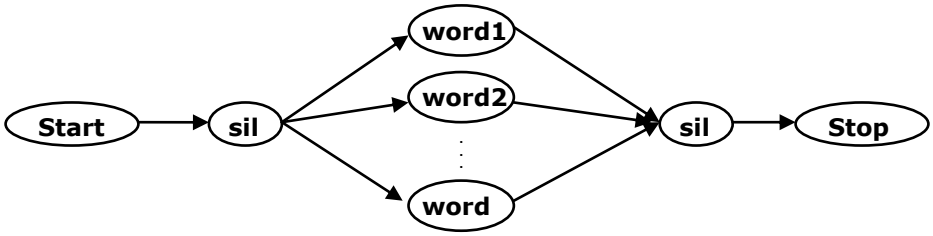

Fig 5. Scheme of the language model of an isolated word recognizer ("sil" is the pause model)

Decoding

The process of pattern matching can be interpreted as searching for an optimal route in the recognition network, driven by the feature vectors. Since the exhaustive search would be too complex computationally, in practice the dynamic programming-based Viterbi algorithm is widely used, which can determine the best route to each moment. In addition, to further accelerate the process; from time to time less probably routes can be pruned.

During the measurement, the VOXerver WFST-based speech recognizer developed at BME TMIT will be used for the speech recognition tasks.

1.3 References

[1] Chetan Sharma & Jeff Kunins, VoiceXML: Strategies and Techniques for Effective Voice Application Development with VoiceXML 2.0, Wiley 2002

[2] Olaszy Gábor –Kovács Magdolna – Nikléczy Péter – Gósy Mária: Magyar nyelvi beszédtechnológiai alapismeretek. (600 oldal CD-ROM-on). http://alpha.tmit.bme.hu/pub/beszinf/start.html, 2002.

[3] Csapó Tamás Gábor, „Változatos prozódia megvalósítása szövegfelolvasó rendszerekben", BME TMIT, Master's thesis, 2008.

[4] Fék Márk – Pesti Péter – Németh Géza – Zainkó Csaba: Generációváltás a beszédszintézisben. Vol. LXI. (2006) No. 3., Híradástechnika, pp. 21–30.

[5] Sprachsynthese. Technische Universität Dresden, Institut für Akustik und Sprachkommunikation, http://www.ias.et.tu-dresden.de/sprache, 2008.

[6] Heiga Zen – Takashi Nose – Junichi Yamagishi – Shinji Sako – Takashi Masu ko – Alan W. Black –Keiichi Tokuda: The HMM-based speech synthesis system (HTS) version 2.0. In SSW6-2007, pp. 294–299.

[7] Gábor Olaszy – Géza Németh – Péter Olaszi – Géza Kiss – Géza Gordos: PROFIVOX – a Hungarian professional TTS system for telecommunications applications. Vol. 3 (December 2000) No. 3/4., International Journal of Speech Technology, pp. 201–216.

[8] Mihajlik Péter, „Spontán magyar nyelvű beszéd gépi felismerése nyelvspecifikus szabályok nélkül", PhD thesis, 2010.

[9] Fegyó Tibor, Mihajlik Péter, „Gépi beszédfelismerés", lecture material, 2009.

[10] M. Mohri, F. Pereira and M. Riley, "Weighted Finite-State Transducers in Speech Recognition," *Computer Speech and Language*, 16(1):69-88, 2002