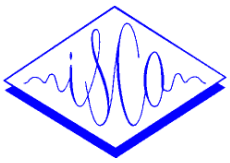

International Symposium on Chinese Spoken Language Processing (ISCSLP 2002)
Taipei, Taiwan
August 23-24, 2002

DESIGN OF EMBEDDED APPLICATION ORIENTED DISTRIBUTED SPEECH SYNTHESIS SYSTEM WITH HIGH NATURALNESS

TANG Hao, YIN Bo and Ren-Hua WANG

Department of Electronic Engineering and Information Science,
University of Science and Technology of China, Hefei
tangh@ustc.edu.cn, boyin@iflytek.com and rhw@ustc.edu.cn

ABSTRACT

In this paper, a unique design scheme of embedded application oriented distributed speech synthesis system with high naturalness is presented in detail. Based on the client/server model, text is firstly converted into parameter sequence (PS) by the front-end tool at the server side. To complete the text-to-speech process, the back-end speech synthesizer at the client side converts the PS into speech upon receival of it through a certain data transmission channel. This design scheme is distinctive in that it is able to obtain highly natural speech with extremely low cost of computing and storage at the client side. Therefore, it is ideally suited for embedded devices with limited performance and storage. The feasible realization of this design scheme on the TI MSP50C614 device with capability of adjusting the speaking speed dynamically in real time is discussed in the end of the paper.

1. INTRODUCTION

With the advent of the new era of embedded applications, the advances of technology have significantly changed the life style and quality of life of the society. People are now able to acquire information as well as perform intelligent computations anywhere, anytime with mobile or handheld embedded devices. Embedded devices are those computers or microchip-based control systems dedicated to perform specific tasks or groups of tasks. While general-purpose computers usually come with fast processors and a wealth of primary and secondary storage, embedded devices are often small, inexpensive and low power [1]. They have been applied to a variety of industrial and consumer products (e.g., assembly lines, medical equipments, in-car GPS navigation systems, information appliances, intelligent toys for children, etc.). However, because embedded devices are always equipped with small screens or there is even no screen at all, it is rather inconvenient for the users to read information often available as text from embedded devices. Thus, the human-machine interface of embedded devices confronts enormous challenges.

A great reform for the human-machine interface is the emergence of the voice (or speech) portal, in which the speech recognition and speech synthesis technologies play a central role. The speech synthesis technology, which is aiming to convert arbitrary text into natural speech, remedies perfectly the deficiency of embedded devices. The text is converted into speech and then the speech is outputted to the speaker instead of the text being displayed on the screen. This is quite useful for the users of embedded devices especially for those who are mobile. However, speech synthesis is a very complicated process. It consumes a lot of computing and storage resource for text-to-speech conversion. Especially when the corpus-based concatenative speech synthesis technique is widely employed today to obtain highly natural synthesized speech, the computing complexity and storage requirement of speech synthesis systems are getting increasingly higher. It is obvious that most corpus-based speech synthesis systems are specifically designed for PCs with high performance and storage capacity. As for embedded devices with low performance and storage capacity, tradeoffs must be seriously considered to resolve the contradiction between the quality of speech and the cost of computing and storage. Many a researcher have implemented some kinds of speech synthesis systems on certain embedded devices by means of simplifying linguistic and prosodic models as well as reducing algorithm complexity and the amount of candidate units in the speech corpus to meet the restricted conditions of those embedded devices [2][3]. Unfortunately, however, because those kinds of traditional lightweight standalone speech synthesis systems mainly focus on reducing the consumption of computing and storage and therefore are extremely simplistic, they are hardly able to produce high-quality speech. In order to obtain high-quality speech on embedded devices, new approaches need to be explored.

This paper presents in detail a unique design scheme of embedded application oriented distributed speech synthesis system with high naturalness. Based on the client/server model, text is firstly converted into parameter sequence (PS) by the front-end tool on a PC (the server). To complete the text-to-speech process, the back-end speech synthesizer on an embedded device (the client) continues to process the PS upon receival of it through a certain data transmission channel, e.g., an RS-232 line or a USB line, converting it into final speech output. The advantages of this client/server based design scheme are evident:

- While the front-end tool running on the server performs the most complicated and computing consuming tasks of the speech synthesis process such as linguistic processing and prosodic processing, the back-end speech synthesizer running on the client performs the waveform concatenation, a task that is relatively simple. Thus, the client reduces its computing complexity down to as low as 5 MIPS. This ultra-low cost of computing at the client side offers open opportunities to various embedded devices ranging from high-end to low-end to exploit speech synthesis.

- The PS, the representation of computing results of the front-end tool transmitted to the back-end speech synthesizer, can be of a very small data size if carefully defined. As a matter of fact, an ultra-low bandwidth of 512Bps meets the requirement.
- Most significantly, the client is able to obtain highly natural speech comparable to the speech obtained by large-scale server class speech synthesis systems on PCs.

This paper is organized as follows. Section 2 illustrates the architecture of the unique design scheme. Section 3 describes the definition of the PS. Section 4 gives a comprehensive consideration of the data transmission channels between the server and the client. Section 5 discusses the feasible realization of this design scheme on the TI MSP50C614 device.

2. ARCHITECTURE

Speech synthesis technologies can be classified into three categories: articulatory synthesis, formant synthesis and concatenative synthesis [4]. The concatenative synthesis, especially the corpus-based concatenative synthesis, is currently the most popular technique that is able to generate high-quality synthesized speech. Whatever synthesis technique a speech synthesis system relies on, the processing procedures are similar to a considerable extent. In this case, we favor the corpus-based concatenative speech synthesis technique.

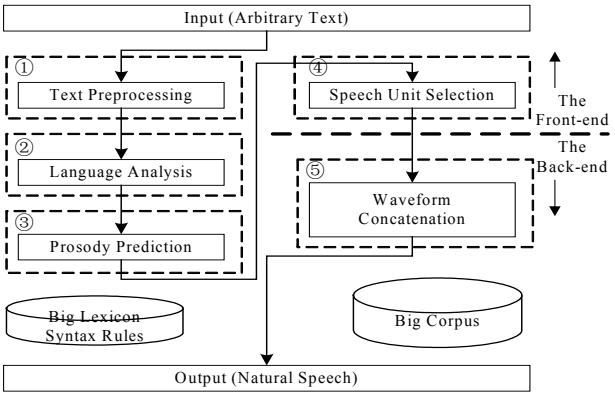

Fig. 1 General processing procedures of corpus-based concatenative speech synthesis systems

The general processing procedures of corpus-based concatenative speech synthesis systems are illustrated in Fig. 1. As shown, the front-end consists of text preprocessing①, language analysis②, prosody prediction③ and speech unit selection④ portions based on big lexicons, syntax rules and big corpuses (in the case of multi-lingual speech synthesis) and the back-end consists of a speech synthesizer, the portion of waveform concatenation⑤. It is known that processing steps of the front-end consume a lot of computing time. Moreover, there is an increasingly demanding trend in order to get more accurate results. This is not a problem at all for PCs but not the case for embedded devices. Compared to the front-end, the back-end, however, is much simpler. It concatenates the speech units selected from the corpus on a one-by-one basis, and if necessary, manipulates the pitch and duration of the concatenated waveform. Results of our experiments indicate that the difference in computing consumption between the front-end and the back-end can be up to an order of magnitude.

From the above analysis, there is no doubt that the proposal of the client/server based design scheme of distributed speech synthesis system is realistic and feasible. In the distributed speech synthesis system, the server deals with the front-end and the client deals with the back-end respectively. The intermediate exchanging data between the front-end and the back-end, called in this paper the Parameter Sequence (PS), is produced by the server and then sent to the client. The architecture of the distributed speech synthesis system is shown in Fig. 2.

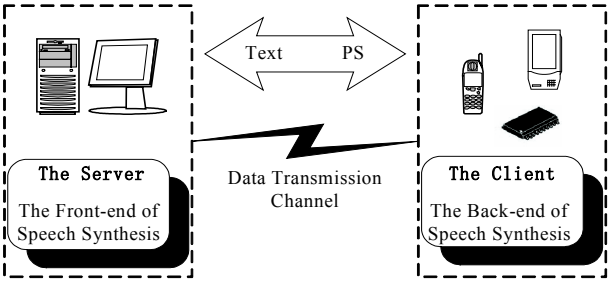

Fig. 2 Architecture of distributed speech synthesis system

3. PARAMETER SEQUENCE

As described in section 2, the PS is generated by the server and transmitted to the client, on which it is converted into speech. Because the server performs most of the tasks of the speech synthesis process including speech unit selection, the parameters transmitted to the client can be as simple as merely consisting of the corpus ID, the index of the speech unit in the corpus, the duration of the speech unit and the pause between the current speech unit and the next one. The size of a speech unit is language dependent. It varies from system to system. In general, a speech unit can be a phoneme, a diphone, a triphone, a half-diphone, a syllable or one of their combinations [5][6], depending on the language concerned and other particular considerations. Nevertheless, this issue does not influence the PS and is beyond the scope of this paper.

A carefully designed organization of the PS will ease both transmission and manipulation. As shown in Fig. 3, the PS consists of a sequence (1~n) of 32-bit frames, each of which consists of four consecutive parts. The first part is the 3-bit "corpus ID" field indicating the corpus in which a speech unit is located. This field is useful for multi-lingual speech synthesis where there are two or more language-specific corpuses. Maximum number of corpuses supported is 2^3=8. The second part is the 20-bit "index" field indicating the location of a speech unit in the corpus specified by the "corpus ID". It provides up to 2^{20}=1048576 indexes for a single corpus. The third part is the 6-bit "duration" field supporting quantization scale 0 to 2^6-1=63 to which the duration of a speech unit originally a floating-point value is quantized. Similarly, the fourth part is the 3-bit "pause" field supporting quantization scale 0 to 2^3-1=7 to which the floating-point pause is quantized.

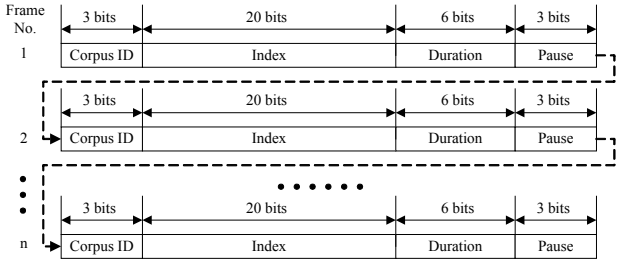

Fig. 3 Definition of PS

In addition to the normal speech units subject to the rules defined above, there is a special kind of "speech units" called the punctuation units. The punctuation units are those units that represent the punctuation marks occurring in the original text. They differ from the normal speech units in that 1) the only valid parameter in the PS frame for a punctuation unit is the pause and 2) the pause of a punctuation unit has a wider quantization range and a more precise quantization scale than those of a normal speech unit. Although the punctuation units share the same PS format with the normal speech units, the meaning is totally different. In the PS frame for a punctuation unit, the "corpus ID" field and the "index" field both take their respective maximum values, i.e., 2^3-1=7 and 2^{20}-1=1048575, respectively, indicating that this particular frame is a frame for a punctuation unit rather than a normal speech unit. The "duration" and "pause" fields merge to form a 9-bit "new pause" field indicating the pause of a punctuation unit. It supports quantization scale 0 to $2^{(6+3)}$-1=511 to which the floating-point pause of a punctuation unit is quantized.

After the text being sent from the client to the server, the PS is constructed by the server upon the computing results of the front-end and sent back to the client. The client extracts the "corpus ID", "index", "duration" and "pause" parameters on a per frame basis from the PS, selects corresponding speech units from corresponding corpuses by the "corpus ID" and "index" parameters and modifies the prosodic features of the concatenated waveform by the rest parameters to produce continuous speech output.

4. DATA TRANSMISSION CHANNELS

The physical data transmission channels between the server and the client are diversified, ranging from a wireless LAN with support of high-layer protocols like HTTP to a telephone channel and to a static media such as CD or tape archive. What transmission scheme the distributed speech synthesis system relies on is an implementation issue and does not make any difference in functionality except that there are two types of usage models. The online usage model requires real-time cooperation between the server and the client, thus demanding support of a certain kind of network and appropriate application protocols. In this usage model, the end user faces a logically standalone system rather than a distributed system. The offline usage model, on the contrary, does not require real-time cooperation between the server and the client. Data exchange between the server and the client can be performed through channels not necessarily restricted to a network, e.g., an RS-232 line, a USB line or even a piece of CD storing the PS taken to the client by a person. This usage model is mainly applied in content-related areas such as Web information services. The ICP (Internet Content Provider) passes a group of web pages to the server and places the generated PS files on the website instead of HTML files. A user who browses the website at any time downloads the PS files and passes them to the client for speech synthesis. In this case, the client can be implemented as a plug-in for prevailing web browsers such as Microsoft Internet Explorer and Netscape Navigator as long as they are supported under the RTOS on embedded devices, if the RTOS is available.

5. FEASIBLE REALIZATION ON TI MSP50C614 DEVICE

In this section, we discuss the feasible realization of the proposed design scheme on the TI MSP50C614 device based on a corpus-based speech synthesis prototype KB2000[1].

5.1 The server

There is no significant distinction between the prototype used and the server except that the prototype generates the speech while the server generates the PS. However, an additional communication module is needed for the server to interact with the client. The communication module can be of different kinds depending on the data transmission channel involved and the underlying protocols, if applicable. The server should be multithreaded if it is required to provide concurrent access from multiple clients, and if necessary, should support features of distributed systems such as load balance, error tolerance and scalability.

5.2 The client

The back-end speech synthesizer, or the waveform concatenator, whatever name we give it, is executed on the client. In the theory of corpus-based speech synthesis, if the corpus contains sufficient contextual and prosodic information, the back-end speech synthesizer does not need to manipulate the pitch and duration of the concatenated waveform, although it always does so. Therefore, computing complexity of the back-end speech synthesizer can be very low provided that the corpus is carefully designed.

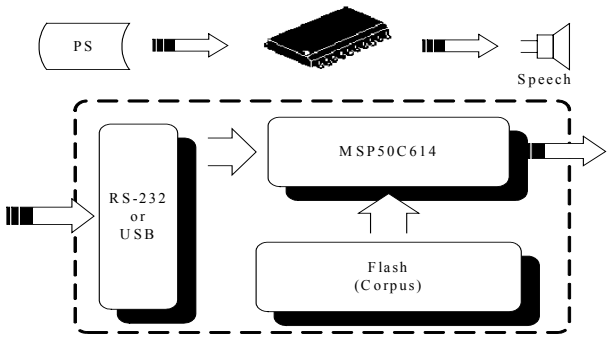

Fig. 4 Framework of client on TI MSP50C614 device

[1] KB2000 is a corpus-based speech synthesis system developed by Anhui USTC iFlyTEK Co., Ltd.

The framework of the client on the TI MSP50C614 device is illustrated in Fig. 4. The TI MSP50C614 device is a low cost, mixed signal controller that combines general-purpose input/output (I/O), on-board ROM and direct speak-drive in a single package. The computational unit utilizes a power DSP which gives the device unprecedented speed and computational flexibility. The device supports a variety of speech and audio coding algorithms such as MELP and CELP, providing a range of options with respect to speech duration and quality. In addiction, the built-in storages on the device are 32K words onboard ROM (2K words reserved), up to 2.36M bits internal data ROM for speech storage and 640 words RAM [7].

Considering the size of the corpus, an 8M bytes additional flash memory is required to extend the internal data storage. Besides, either an RS-232 module or a USB module is needed for establishing a communication link between the client and a PC through which the PS generated on the PC is transmitted to the client. No extra D/A circuit is needed due to that the MSP50C614 device is designed with built-in direct speak-drive. However, because the RAM available on the device is limited, in order to generate continuous speech output, multi-buffer (usually dual-buffer) playback mechanism must be used.

The back-end speech synthesizer on embedded devices expects consideration of two aspects: corpus compression and concatenative algorithm.

5.2.1 Corpus compression

In corpus-based speech synthesis systems, a state-of-the-art corpus is of vital importance. This is also true of the distributed speech synthesis system. Due to limited storage of embedded devices, the big corpus on PC cannot be used directly. At least two steps must be taken. One is to extract pure speech signal from the big corpus. The other is to compress the speech signal using a speech coding algorithm such as G.723.1, G.729, CELP and MELP.

Besides the speech signal, an important kind of data in the corpus used for the back-end speech synthesizer to adjust duration is the marks of fundamental tone. The marks are a series of relative values indicating key positions in the voiced segments of a sound utterance that can be labeled automatically by machine. Usually, the marks in the big corpus cannot be used in the compressed corpus. They must be re-labeled. Since most speech compressions are lossy compressions, after being compressed and decompressed, the speech signal is somewhat different, which leads to inaccuracy of the marks. The re-labeling can be performed on the compressed-then-decompressed speech signal.

5.2.2 Concatenative algorithm

Responsible for concatenating selected speech units into continuous speech, the concatenative algorithm needs to perform three tasks: 1) decompressing the target speech units before they are appended to the output speech, 2) processing the decompressed speech units by means of fading-in and fading-out the tailing edge of the left unit and the leading edge of the right unit to make the speech signal smoother [5], [6] and 3) passing the speech signal to the PSOLA with duration converted to a floating-point value [5], [8].

The PSOLA does the transformation of duration. The duration of a certain speech unit is co-determined by the duration parameter extracted from the PS frame and a user-specifying value indicating the speaking speed. The acceptable range of duration is 0.1 to 2.0. Note that exceeding this range will result in distortion of speech.

6. CONCLUSION

Based on the client/server model, a unique design scheme of embedded application oriented distributed speech synthesis system with high naturalness is proposed in this paper. This design scheme, which is described in detail in the previous sections, is a distinctive solution to speech synthesis on embedded devices. It has many advantages over traditional speech synthesis systems as mentioned earlier. Due to its low computing cost, low storage consumption and high quality of speech produced, this design scheme is suitable for a variety of embedded applications incorporating speech synthesis functionality. Our future work will focus on how to maintain the computing complexity and the corpus size while increasing the naturalness of the speech.

7. REFERENCES

[1] Robert E. Filman, “Embedded Internet Systems Come Home,” *IEEE Internet Computing*, Volume: 5 Issue: 1, Jan.-Feb. 2001, Page(s): 52 –53.

[2] Yasushi Ishikawa, Yasuhisa Kisuki, and Tadashi Sakamoto, “Speech Synthesis Software for a 32-bit Microprocessor,” *IEEE Transactions on Consumer Electronics 44*, 3, pp. 1173-82, August 1998.

[3] Hataoka, N., Kokubo, H., Nukaga, N., Obuchi, Y., Amano, A., Kitahara, Y., “Sophisticated Speech Processing Middleware on Microprocessor,” *IEEE 3rd Workshop on Multimedia Signal Processing*, pp. 691–696, 1999.

[4] Jun Huang, Levinson, S., Davis, D., Slimo, S., “Articulatory Speech Synthesis Based upon Fluid Dynamic Principles,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Volume: 1, pp. 445–448, 2002.

[5] Fu-Chiang Chou, Chiu-Yu Tseng, “Corpus-based Mandarin Speech Synthesis with Contextual Syllabic Units Based on Phonetic Properties,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Volume: 2, pp. 893 –896, 1998.

[6] Rutten, P., Coorman, G., Fackrell, J., Van Coile, B., “Issues in Corpus Based Speech Synthesis,” *IEEE Seminar on State of the Art in Speech Synthesis (Ref. No. 2000/058)*, pp. 16/1 -16/7, 2000.

[7] *MSP50C6xx Mixed-Signal Processor User’s Guide*, Texas Instruments Incorporated, 2001.

[8] Ren-Hua Wang, Qingfeng Liu, Difei Tang, “A New Chinese Text-to-Speech System with High Naturalness,” *Proceedings of Fourth International Conference on Spoken Language, ICSLP 96*, Volume: 3, pp. 1441–44, 1996.