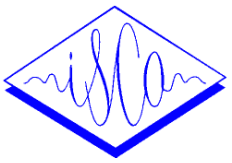

6th International Conference on Spoken Language Processing (ICSLP 2000)
Beijing, China
October 16-20, 2000

A METHOD OF CREATING A NEW SPEAKER' S VOICEFONT IN A TEXT-TO-SPEECH SYSTEM

Takashi Saito, Masaharu Sakamoto

IBM Research, Tokyo Research Laboratory, IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi, Kanagawa-ken, 242-8502, Japan
saito@jp.ibm.com, sakamoto@jp.ibm.com

ABSTRACT

This paper presents a method of creating a new speaker' s voice database (*VoiceFont*) by which the voice of the donor speaker can be synthesized for mimicking in a text-to-speech system. A VoiceFont creation system, "***VoiceFont Builder***", is developed to make the creation process easier and more effective than current systems. The voice feature extraction applied in the system is a simple but powerful method that makes the most of the target speech synthesizer. Using a VoiceFont obtained, we conducted experiments on F0 contour generation in view of reproducing that of the donor speaker' s voice.

1. INTRODUCTION

Creating a voice database, we call it *VoiceFont* here, is fundamental, but still time-consuming and laborious task in a text-to-speech system, although the variety of synthetic voices is of great importance to augment the applicability of the system. In current text-to-speech systems, there is an apparent restriction on the variety on available output voices; the only way in which users can customize the voices is to choose a predefined voice, or to change the speaking attributes such as pitch, speed, and amplitude. Hence, registration of new voices in text-to-speech systems, if possible by users or by application developers, would be a new paradigm and an attractive function for speech synthesis, which would broaden the range of current text-to-speech applications.

This paper presents a method of creating a new speaker' s VoiceFont by which the voice of the donor speaker can be synthesized for mimicking in a text-to-speech system. We present here a VoiceFont creation system, "***VoiceFont Builder***", as an efficient tool for database creation. It is designed to make the creation process easier and more effective than current environments. It provides three kinds of functions: voice feature extraction, data checking/editing, and inventory generation.

The voice feature extraction is a powerful function of *VoiceFont Builder*, which we propose in this paper. The voice characteristics we intend to extract are of two kinds: segmental (phonetic) and suprasegmental (prosodic). Several methods for extraction of prosodic features from a speech corpus have been reported [1], [2]. In the method proposed here, the segmental features as well as the prosodic features are simultaneously extracted for direct use in a speech synthesizer.

The approach we take here to the automatic extraction of these features is based on a simple phonemic alignment of input speech utterance with reference templates generated by a text-to-speech synthesizer. Our strategy is very simple, but has several notable merits. First, phonetic and prosodic boundaries obtained by the alignment are basically well suited and consistent for use as synthesis control parameters, since the reference templates are generated by the same criteria for speech synthesis. Second, the accuracy of phonetic segmentation is very high, mainly because the spectral context dependency expressed by the synthetic templates is fairly appropriate, and as a result, a fundamental dynamic time warping can align the input segments with the synthetic templates. This capability is provided by the text-to-speech module [3] that we are using in this study. Lastly, the segmentation process does not intrinsically require any training procedure.

The VoiceFont creation tool is integrated into the target text-to-speech system. Through the database creation procedure, both segmental and suprasegmental features are simultaneously extracted from a speech corpus, and are directly stored in unit inventories for synthesizing the donor speaker' s voice.

This paper is organized as follows. In section 2, we describe the outline of the database creation tool, *VoiceFont Builder*. In section 3, we present an automatic method of voice feature extraction that makes the most of the target speech synthesizer. In section 4, we also report here the experimental results on F0 contour generation in view of reproducing that of the donor speaker' s voice. In the last section, we give a brief summary on this study.

2. VOICEFONT BUILDER

In this section, we describe the design concepts of *VoiceFont Builder* and the basic outline of the prototype system.

2.1 Design Concept

We take into account the following viewpoints in designing *VoiceFont Builder*.

- Provided along with Synthesizer

Customization functions like registering application' s vocabulary are essential for a text-to-speech system to make it adaptable to a wide range of applications. Ideally, VoiceFont should be regarded as one of customization functions. Hence, it is desirable that the

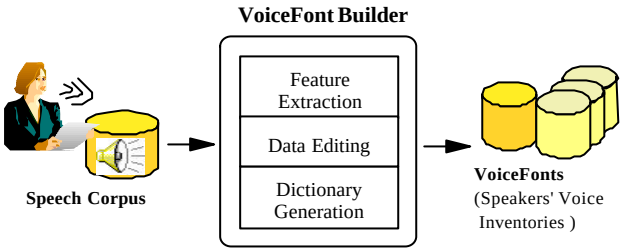

Fig. 1 VoiceFont Creation System

building tool can be provided to users or application developers along with a speech synthesizer, not staying at TTS developers.

- Framework of Mimicking Voice Synthesis

Mimicking voice synthesis, that is to reproduce the voice of a donor speaker as a synthetic voice, is an interesting issue on speech synthesis. In order to improve the level of mimicry, the speech synthesizer should be able to utilize VoiceFont information in an appropriate manner. Therefore, the feature extraction in *VoiceFont Builder* is designed to carry out the processes so as to be as consistent as possible with the target synthesizer.

- Easy Data Checking/Editing

The automatic voice feature extraction could not be perfect, even if it shows extremely successful performance. Hence, a checking/editing function is essential for now in *VoiceFont Builder*, but it should provide easy-to-use operations.

2.2 System Outline

Figure 1 shows the configuration of the system. The input of the system is a speech corpus of a new speaker, which consists of speech data and its Japanese orthographic text in mixed kanji/kana form. The output is the speaker's VoiceFont, that is, voice dictionary for the speaker. *VoiceFont Builder* has three functions: voice feature extraction, data checking/editing, and dictionary generation. The feature extraction function carries out the procedure to automatically extract voice features needed for synthesizing the donor speaker's voice. The data checking/editing function provides a GUI-based editor, with which users can check or modify the extracted voice information such as pitch markers, phonetic segments, accent positions, and so on. The dictionary generation function compiles the speaker's voice data in the form of VoiceFont, which can be directly fed into the speech synthesizer.

The prototype of *VoiceFont Builder* runs on Windows 98/NT. VoiceFonts obtained can be used with the target text-to-speech synthesizer immediately after the creation process is completed.

A typically procedure of VoiceFont creation using the system is shown as follows;

1. Script Checking

First, text-to-phoneme conversion of the utterance scripts of a speech corpus is carried out to check reading errors of the text-to-speech synthesizer. The error correction is done by registering unknown words in the word dictionary of the synthesizer. This checking process is aimed to run the following feature extraction successfully which uses reference templates provided by the synthesizer in phonemic alignment. This procedure has only to be carried out once for a set of utterance scripts.

2. Voice Feature Extraction

Then, an automatic voice feature extraction procedure is carried out for the speech corpus using the word dictionary prepared for the script set. Through this procedure, input speech utterances are decomposed into multi-layered segments: breath groups, accent phrases, phonemes, and pitch marks.

3. Data Checking

If segmentation errors or badly uttered speech data are found, these portions can be checked, and corrected or replaced by using the data editor. Figure 2 shows the output of extracted features by the data editor. All the parameters needed for VoiceFont creation can be checked in a GUI-based environment.

Finally, a VoiceFont for the given corpus is generated by compiling the segmented speech data to three kinds of VoiceFont information: acoustic unit inventory, F0 unit inventory, and duration parameter set.

3. AUTOMATIC FEATURE EXTRACTION

Figure 3 shows the procedure of the automatic voice feature extraction. As explained earlier, the extraction procedure makes the most of internal modules in the very text-to-speech synthesizer that later uses the extracted features for synthesis. Multi-layered segments, which include breath groups, accent phrases, phonemes, and pitch marks, are all efficiently obtained through the extraction procedure. It can be said that the intermediate information provided by the synthesizer contributes greatly to decomposing the utterance into the multi-layered segments, which are all required for speech synthesis.

The processes in all the stages except the first two are completely automatic. (Currently, parts of the outputs of stage (1) and stage (2) are checked manually in order to obtain highly accurate final results.) The stages of the procedure are described as follows:

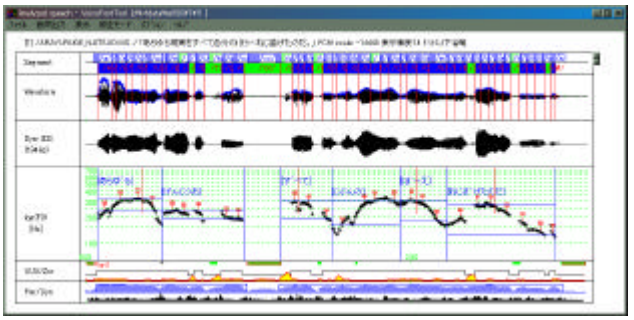

Fig.2 Output of Voice Features using the Data Editor

(1) Silence Detection

Silences in speech data are detected by using the log power of speech signals. As a result, utterances are divided into breath groups. Currently, in this stage, manual checking is carried out to exclude Japanese choked sounds (/Q/), which are incorrectly detected as silences.

(2) Text Analysis

The text analysis module of the text-to-speech synthesizer is used here to obtain phonetic transcription and accentual phrases. As in stage (1), manual checking is carried out to correct transcription errors by registering unknown words in the word dictionary. This check is important for eliminating transcription errors, which are not desirable in the subsequent processes.

(3) Breath Group Segmentation

Pause positions in phonetic transcription are automatically determined by an algorithm that uses the phoneme duration estimated by the synthesizer from the phonetic transcription and the silence positions obtained in stage (1). As a result of it, the input utterance is segmented into breath groups.

(4) Pitch Detection

Pitch mark information is extracted by a wavelet-based pitch marking method [7], and is applied here to obtain precise F0 contours, which are used for refining phoneme boundaries in stage (5) and extracting F0 phrasal units in stage (6).

(5) Phonemic Segmentation

In this stage, several constraints based on top-down information are imposed on the alignment, to minimize alignment errors. First, the alignment is performed for breath groups that are divided by pauses, not for sentences, since pause positioning is fairly sensitive to several factors such as context, speaker and speaking style, and consequently may cause gross errors in segmentation. Another constraint is on the length of the synthetic templates, which are generated with the same length as those of input utterances to ensure effective dynamic time warping. Reference templates for alignment are prepared by synthesizing the input text on the basis of breath groups. Input utterances already divided into breath groups are segmented further into phonemic segments by alignment with synthesized templates.

(6) Phrase Segmentation

Finally, input utterances are segmented into accentual phrases by using the outputs of stage (2) and stage (5). F0 contour fragments for accentual phrases are extracted as phrasal F0 units for F0 contour generation in speech synthesis.

We investigated the performance of the feature extraction using three speech corpora of different types. The method worked quite successfully for read-out speech, although there is still room for improvement in its handling of emotional speech. The details of the experiments were described in [6].

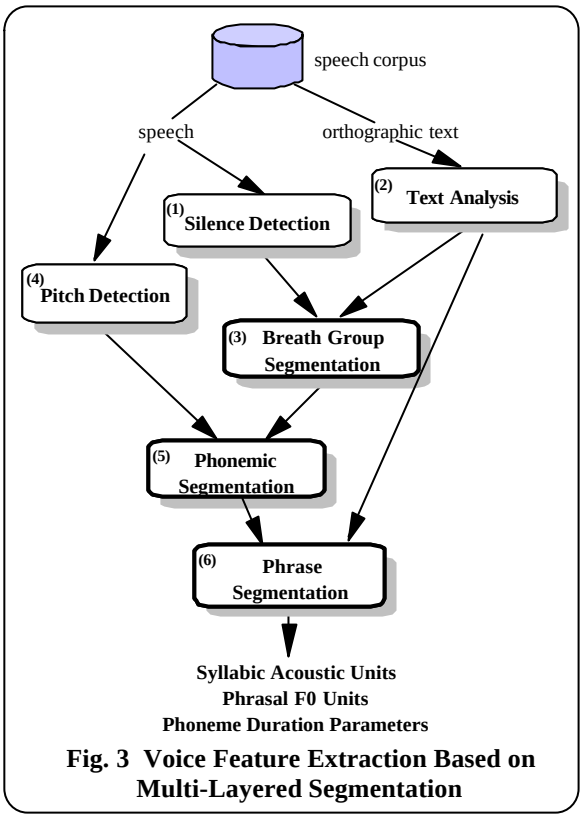

Fig. 3 Voice Feature Extraction Based on Multi-Layered Segmentation

4. F0 CONTOUR GENERATION USING A CREATED VOICEFONT

In the speech synthesizer, we are trying to implement a concatenation-based F0 contour generation method. In this section, we conducted an experimental evaluation to investigate the performance of the F0 contour generation using a created VoiceFont, in view of reproducing F0 contours of the donor speaker.

4.1 F0 Contour Generation Method

In VoiceFont data, the F0 contour information of a donor speaker is stored in the form of a phrasal unit with its prosodic context. A phrasal F0 unit is defined as an F0 contour fragment nearly equal to one accentual phrase. For F0 contour generation, F0 phrasal units appropriate for synthesis environment are retrieved from the inventory, according to their unit attributes (accent type, phrase length, phoneme categories) and their prosodic contexts (currently, we use accent type, phrase length, and pauses between phrases as contextual information). The retrieved F0 units are concatenated just like acoustic units, basically without modification. Exceptionally, two kinds of modifications are applied to the F0 units; one is time axis adjustment according to the phoneme constituents of the units in synthesis environment, the other is F0-unit level adjustment in order to smooth the levels between concatenated F0 units. An F0-unit level prediction model is applied to the level adjustment. The model parameter is statistically estimated by using the quantification method type-I.

Here, F0-unit level is defined as the average of the log F0 values at vowel center points in F0 units.

4.2 Speech Corpus

For this experiment, we used a set of 503 sentences from the ATR continuous speech database, uttered by a professional female speaker. Out of 503 sentences, 450 sentences (3037 units) were used for VoiceFont creation (inventory and model parameter estimation), and 53 sentences (284 units) are used for open test. The accent position errors were corrected before dictionary generation for this evaluation.

4.3 F0-Unit Level Prediction

In the F0-unit level prediction model based on the quantification method type-I, the following factors are taken for the model parameter estimation: unit accent type (current 6, preceding 7, following 7), unit length (current 7, preceding 8, following 8), unit boundary condition (preceding 4, following 4), unit position in a breath group (4), number of units in breath group (5). The number in parenthesis means the number of categories for each subject. Table 1 shows the results of F0-unit level prediction. Since the correlation coefficient is fairly high, and the prediction errors for open and close data are small and not so different, it can be said that the modeling worked successfully.

Coefficient of correlation	Prediction error for closed data	Prediciton error for open data
0.84	0.15 [oct]	0.16 [oct]

Table 1. Results of F0-Unit Level Prediction By Quantification Method Type-I

4.4 F0 Contour: Generated vs. Original

For the open data of 53 sentences, the root mean square error between the generated F0 contours and the original ones is 0.21 [oct]. Figure 4 shows an example of the generated F0 contour and the original one for a same sentence. In a preliminary listening test, synthetic speech samples using the generated F0 contours are very close, in terms of intonation contour, to the natural speech utterances of the donor speaker.

powerful method that makes the most of the target speech synthesizer. The data consistency with the speech synthesizer contributes, not only to the performance in the multi-layered segmentation, but also to mimicking in synthesis. Using a created VoiceFont, we conducted experiments on F0 contour generation in view of reproducing that of the donor speaker. The F0 generation method was successful in synthesizing the F0 contour of the donor speaker with the VoiceFont created quickly and easily through the proposed procedure.

6. REFERENCES

1. A. Sakurai et al., "A Linguistic and Prosodic Database for Data-Driven Japanese TTS Synthesis," Proc. of ICSLP '98, pp. 1048-1051, 1998.
2. N. Campbell et al., "Auto-ToBI: a Prosody-Labeling Workbench for Japanese Read Speech," Proc. of ASJ Autumn Meeting, pp.247-248, 1996.
3. T. Saito et al., "High-Quality Speech Synthesis Using Context-Dependent Syllabic Units," Proc. of ICASSP '96, pp. 381-384, 1996.
4. T. Kagoshima et al., "An F0 Contour Control Model for Totally Speaker Driven Text-To-Speech System," Proc. of ICSLP '98, pp. 1975-1978, 1998.
5. T. Saito, "A Method for Registering New Voices in a Text-to-Speech Synthesizer," Proc. of 3rd ASA & ASJ Joint Meeting, pp. 1057-1060, 1996.
6. T. Saito et al., "Voice Feature Extraction Method for a Mimicking Voice Synthesizer," Proc. of ICICS' 99, 2D1.3, 1999.
7. M. Sakamoto et al., "An Automatic Pitch Marking Method Using Wavelet-Based Features," Proc. of ICICS '99, 1999.

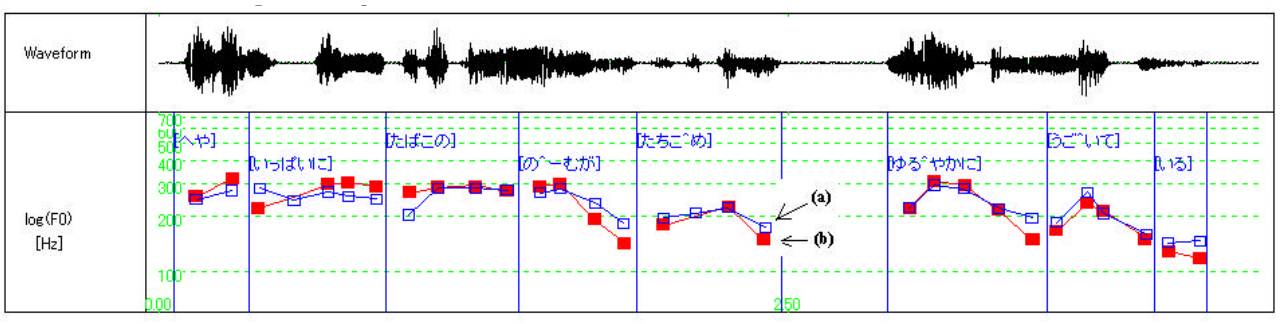

Fig.4 Comparison of (a) Generated F0 Contour and (b) F0 Contour of Natural Speech