Yanhui (宴会), a Software Based High Performance Mandarin Text-To-Speech System

John Choi*, Hsiao-Wuen Hon, Jean-Luc Lebrun, Sun-Pin Lee,
Gareth Loudon, Viet-Hoang Phan, and Yogananthan S.

Apple-ISS Research Center
Institute of Systems Science
National University of Singapore
Republic of Singapore

E-mail: sunpin@apple-iss.iss.nus.sg

ABSTRACT

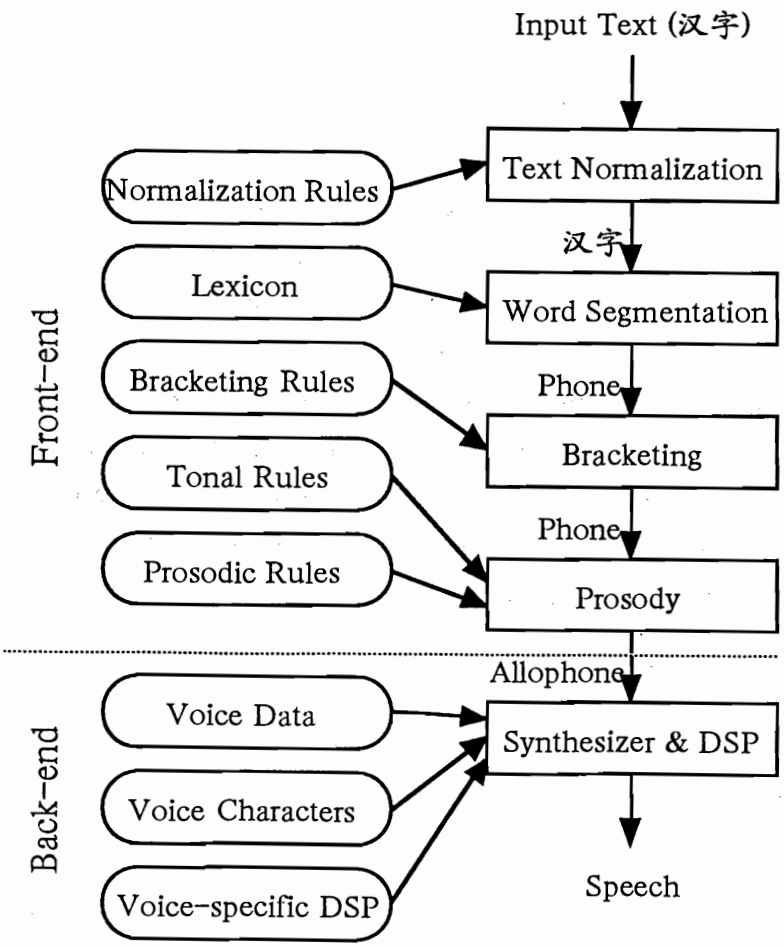
This paper presents Yanhui (宴会), a software based, high performance Mandarin text-to-speech system, developed at Apple-ISS Research Center, Singapore. This system uses diphone concatenative synthesis method to produce Mandarin speech and applies extensive prosody control in pitch and duration to achieve natural sounding speech. The system takes a free-form Chinese text and generates the corresponding natural speech sound. Yanhui (宴会) is running real-time and software-only on commercial personal computers. This paper describes the various algorithms and techniques used in the front-end and back-end.

* John Choi is currently with Berkeley Speech Technologies Inc.

1. INTRODUCTION

People have been enjoying various real-world speech synthesis applications, including automated telephone systems and automobile alert systems. Because of its ability to utter any unexpected conversation and its small memory footprint, its great potential is believed to be explored widely. Its current usage is often constrained by the special hardware for signal processing and its unnaturalness. This paper describes Yanhui (宴会), the first software based, high performance, low memory footprint (1.5MB) Mandarin text-to-speech system running real-time on commercial Macintosh personal computers.

Speech synthesis has been a very interesting computational linguistics research field for years because it involves almost the entire spectrum of computational linguistics. For Chinese, a good Mandarin speech synthesis system often requires extensive work on lexicon, segmentation, bracketing, language modeling, part-of-speech, parsing and prosody to achieve natural sounding quality. Yanhui (宴会) developed at Apple-ISS research center in Singapore is based on Apple's Macintalk Pro architecture [1]. It consists of two major parts: front-end and back-end components. The technology modules and data resources of Yanhui (宴会) are shown in the figure on the left. The front-end component is responsible for taking an arbitrary Chinese text input and transferring it into a sequence of phonemes along with its duration and pitch attributes. The back-end component is then responsible for converting this parametered list into a concatenative and smooth speech waveform which is ready for D/A device to playback.



The emphasis of this paper is on the front-end of Yanhui (宴会). The paper is arranged as follows. Section 1 gives the general introduction of Yanhui (宴会). Section 2 describes in detail the front-end component, including text normalization, word segmentation, bracketing, tonal rules, duration and pitch modification rules. Section 3 describes how the back-end generates synthesized speech waveform from the phoneme list passed in by the frond-end component. Finally, Section 4 describes our future direction to improve the quality and memory footprint of Yanhui (宴会).

2. FRONT-END

The responsibility of the front-end components of Yanhui (宴会) is to convert the given Chinese text to be spoken into phoneme units with pitch and duration information properly assigned. The communication between the front-end and back-end components is done on a statement basis. A statement is an arbitrary length of text in GB2312-80 (the standard character set for Chinese adopted in Mainland China) characters separated by any punctuation mark or new-line character. Thus the system iterates for each statement until all the text are processed. The entire process performed by the front-end components on a statement can be further divided into the following steps, which are discussed in more details in the next few sections.

- Text normalization
- Word segmentation
- Bracketing (segmentation post-processing)
- Tonal rules
- Duration assignment and manipulation
- Pitch assignment and manipulation

2.1 Text Normalization

In Chinese text, we often come across non-Chinese characters that need to be spoken. The most common of these are the numeric characters. A text-to-speech system must be able to translate these characters to the equivalent Chinese characters to be spoken. This is the main task of the Text Normalization module.

There are two ways in which a number can be read, either read it as a number or read it digit by digit. For example, 123 can be read as either 一百二十三 or 一二三. Numbers with many digits are more likely to be read digit by digit, while numbers with

few digits are usually read as a number. In our system, numbers having four digits or more will be read digit by digit, unless they are correctly separated by commas after every three digits. Commas which are not placed at the correct positions are treated as separators. The following examples illustrate how certain numbers will be read.

Number	Normalized Text
123	一百二十三
123.45	一百二十三点四五
1234567	一二三四五六七
1,234,567	一百二十三万四千五百六十七
1,23,4567	一、二十三、四千五百六十七

Symbols that are frequently used together with numbers also need to be correctly read. Our system currently handles the dollar sign, percentage sign, fraction symbol (forward slash) and dash. For dollar amounts, if there are exactly two digits after the decimal point, the system interprets them as cents. Otherwise, it will be read as a decimal of the dollar. The forward slash is considered as a fraction symbol only when there is a number each immediately before and after the slash, both numbers are integers, and there is no other slash before the first number or after the second number. For example, 2/5 is a fraction, while 2.2/7 or 3/5/8 are not fractions, and the slashes will be treated as separators. All symbols which are treated as separators (comma, slash or dash) are translated into 200ms pauses later on in the duration module. The following examples illustrate how symbols are processed by the system.

Number	Normalized Text
$1.50	一元五角
$1.57	一元五角七分
$1.5	一点五元
$1.578	一点五七八元
40%	百分之四十
68.25%	百分之六十八点二五
1/3	三分之一
6/4/7	六、四、七
776-4005	七七六、四零零五

2.2 Word Segmentation

In Chinese, the definition of word (a sequence of characters that represent a

complete, unambiguous syntactic and semantic unit) is very vague. Words do affect the prosody and intonation in continuous Mandarin speech. For example, people might tend to put a small pause between words. Moreover, some characters have different pronunciations when they appear in different words. For example, 行 can be pronounced as either *xing2* or *hang2*. If the word 行动 is identified, it is pronounced as *xing2*. If the word is 银行, then it should be pronounced as *hang2*.

Languages such as Chinese that use ideographic characters do not have spaces in the text to indicate word boundaries, unlike an alphabetical language such as English. Thus it is necessary to explicitly segment a string of Chinese characters into its constituent words. This segmentation process is a difficult task and researchers have been investigating on this topic for years [2, 3, 4, 5]. Because of its vagueness, people have not reached consensus on automatic word segmentation. To make it even worse, most Chinese characters can stand alone to form monosyllabic words, and they can also partner with other characters to form other words. The word segmentation module takes the output of the text normalization module and attempts to insert the word boundaries.

2.2.1 Lexicon and Compression

Most automatic segmentation techniques are lexicon based. Our lexicon contains words, their pronunciations, as well as their frequency information. The lexicon contains about 82,000 entries, including every monosyllabic Chinese character in the GB2312-80 character set. We generated the frequency data using a 40MB Chinese corpus, mostly newspaper material. The following is an example of a lexicon entry.

Chinese characters	Pronunciation	Frequency
新加坡	xin1 jia1 po1	386

The 82,000 entry lexicon is about 1.7MB in size. To reduce its memory footprint, we adopt some compression technique. Firstly, all Chinese characters and pronunciations are stored in an indexed table. Each lexicon entry is encoded in reference to this table. This reduces significantly the repeated occurrences of characters. Moreover, most Chinese characters are pronounced in one particular way most of the time. In our lexicon, we only store the pronunciation of a Chinese character when it differs from its default pronunciation. With a fast search algorithm, an entry can be retrieved almost as quickly as in an uncompressed lexicon. This technique achieved a 4:1 compression ratio on the lexicon.

2.2.2 Segmentation Algorithm

Yanhui (宴会) currently uses a dynamic programming approach to segment a statement into words. First of all, those text generated in the text normalization module to substitute the non-Chinese characters are already words by themselves and will not be processed in the word segmentation module.

Suppose we want to segment a string of n Chinese characters, c_1, c_2, ... , c_n, into words. Let $o(i, j)$ denote the optimal segmentation of the segment c_i, c_{i+1}, ... , c_j. We also define $w(i, j)$ if c_i, c_{i+1}, ... , c_j is a word in the lexicon. Otherwise, $w(i, j)$ is not defined. Since the lexicon contains every monosyllabic Chinese character, $w(i, i)$ is defined for all i. To determine $o(i, j)$, we first check if $w(i, j)$ is defined. If so, then $o(i, j) = 0$, meaning that this segment is itself a word in the lexicon. If $w(i, j)$ is not defined, then $o(i, j)$ is made up of two optimal segmentations $o(i, k-1)$ and $o(k, j)$ where k is between $i+1$ and j. The value of k that gives the best segmentation for c_i, c_{i+1}, ... , c_j is assigned to $o(i, j)$. We also define $f(i, j)$ to be the frequency value of c_i, c_{i+1}, ... , c_j. The use of $f(i, j)$ will be described in the next paragraph. Finally, we define $s(i, j)$ to indicate the score value of c_i, c_{i+1}, ... , c_j. The score value is related to the lengths of words contained in the segment. Naturally, longer words get higher scores than shorter words. Having the above definitions, it is quite clear now that our objective is to obtain $o(1, n)$.

First of all, we need to know which $w(i, j)$ are defined, and which are not. This is done by simply looking up the lexicon. If a particular $w(i, j)$ is defined, then the corresponding $f(i, j)$ is set to the frequency data obtained from the lexicon. Otherwise, $f(i, j)$ is set to 0. Now we are ready to obtain all the $o(i, j)$. Begin with $o(i, i)$ for i ranging from 1 to n. This is trivial because all $w(i, i)$ are defined. Hence $o(i, i) = 0$ for all i. Whenever we set $o(i, j)$ to 0, we also assign $j - i + 0.75$ to $s(i, j)$. Thus $s(i, i) = 0.75$ for all i.

Next, we compute $o(i, i+1)$ for i ranging from 1 to $n-1$, For each case, if $w(i, i+1)$ is defined, we perform the same operations as before. Assign 0 to $o(i, i+1)$ and 1.75 to $s(i, i+1)$. If $w(i, i+1)$ is not defined, $o(i, i+1)$ must be made up of two optimal segmentations $o(i, k-1)$ and $o(k, i+1)$ where k must be $i+1$. In this case, this must be the optimal segmentation for $o(i, i+1)$, and $i+1$ is assigned to $o(i, i+1)$. The product of $s(i, k-1)$ and $s(k, i+1)$ is then stored as the value of $s(i, i+1)$. As for $f(i, i+1)$, we store the average frequency of the two optimal segmentations. To obtain the average frequency, we multiply each frequency value by the number of Chinese characters in the sub-

segment, add the two products, and divide by the number of Chinese characters in the entire segment. Thus, the following formula is used to compute *f(i, j)*.

$$f(i, j) = \frac{f(i, k-1) * (k-i) + f(k, j) * (j-k+1)}{j - i + 1}$$

Then, the algorithm computes *o(i, i+2)* for *i* ranging between *1* and *n-2*. The same method is used, except that when *w(i, i+2)* is not defined, the optimal segmentation is *o(i, k-1)* followed by *o(k, i+2)*, where *k* is either *i+1* or *i+2*. For each value of *k*, compute what *s(i, i+2)* would be. If there is a highest value of *s(i, i+2)* for a particular *k*, then take that as the optimal segmentation. If there is more than one *k* that gives the highest value of *s(i, i+2)*, compute *f(i, i+2)* for these cases and select the *k* that gives the highest *f(i, i+2)* as the optimal segmentation. Assign values to *o(i, i+2)*, *s(i, i+2)* and *f(i, i+2)* using the same method as before.

The algorithm repeats for all *o(i, i+3)*, *o(i, i+4)*, ... , until *o(1, n)* is obtained. To retrieve the segmentation result, simply reverse the process. We now know that the optimal segmentation for $c_1 \ldots c_n$ is $c_1 \ldots c_{o(1, n)-1}$ followed by $c_{o(1, n)} \ldots c_n$. The same process is performed recursively for each of the two sub-segments. The recursion stops whenever a 0 is found in *o(i, j)*, which indicates that $c_i \ldots c_j$ is a word in the optimal segmentation of $c_1 \ldots c_n$.

When choosing the optimal segmentation for any particular $c_i \ldots c_j$, the best choice is of course when $c_i \ldots c_j$ is a word by itself, ie. *w(i, j)* is defined. Otherwise, the algorithm looks for combination of words and try to choose long words as far as possible. Therefore, *s(i, k-1)* and *s(k, j)* where *k* is between *i+1* and *j* are used. It is clear that a segment that is made up of long words has a large score value. Also, we try to avoid choosing segments with too many monosyllabic words. That is why the score assigned to monosyllabic words is less than 1, which serves the effect of decreasing the scores of segments that contain them.

We attempted another approach to perform automatic word segmentation using the relaxation technique [5]. This technique is meant to be used as a first pass in a two-pass word-identification system. This first-pass system is used to try and resolve much of the word-formation ambiguities and to eliminate those which are redundant. The second-pass would utilize syntactic and semantic information in the final resolution. We found that this technique is more computationally intensive compared to the dynamic

programming approach, while achieving similar segmentation results.

2.3 Bracketing (Segmentation Post-Processing)

The purpose of bracketing in Yanhui (宴会) is to group a number of adjacent words so that they can be treated as a more syntactically and semantically complete unit. This grouping of words is particularly used for the application of tonal rules, which is described in the next section. The unit after bracketing is called phrase. Therefore, a phrase can be a word or a group of words according to bracketing rules.

The bracketing algorithm first looks for adjacent numeric Chinese characters and group them into a phrase. For example, if 九, 百, 五, 十 and 八 are found in the text, they would be segmented as individual characters using the lexicon. The bracketing algorithm then groups them into a single phrase, 九百五十八.

The next category of grouping is for quantifiers (量词). If a quantifier is preceded by a numeric phrase, then the numeric phase, the quantifier and the next word are grouped together. If the quantifier is at the end of the statement, then only the numeric phrase and the quantifier are grouped. About 200 quantifiers are checked by the algorithm. For example, suppose that 五, 把 and 椅子 are found in the text. 把 is a quantifier and it is preceded by a numeric phrase, 五. Thus the algorithm groups them together with the following word into a single phrase, 五把椅子.

Finally, the algorithm looks for the pronoun 我 or 你, and groups it together with the next word. For example, in the statement 我跑回家, 我跑 is grouped into a single phrase. If the pronoun occurs at the end of the statement, then no grouping is done.

2.4 Tonal Rules

In Mandarin speech, there are occasions when the tone of a Chinese character needs to be changed due to the context of the character. The most common is tone sandhi, whereby if there are two consecutive tone 3 characters, the first character changes to tone 2 [6]. For example, both 老 and 板 are tone 3 characters, but in the word 老板, 老 is changed to tone 2. This rule usually applies only within the same word. For cases where tone sandhi is applied across word boundaries, many of them are included in the categories of phrases bracketed in the previous section. That is the reason why we do bracketing, so that tone sandhi can be applied across word boundaries under certain circumstances.

In Yanhui (宴会), we make use of tonal rules to manipulate the tones of each syllable. There are two types of tonal rules. The syntax of the first type of rule states that tone α changes into tone β if it occurs before/after a tone γ, and whether the rule can be applied across phrase boundaries. Tone sandhi is an example of this type of rule. It states that tone 3 changes into tone 2 if it occurs before a tone 3, and this rule does not apply across phrase boundaries. We use two other rules of this type to change tone 5 into slightly modified tones when it occurs after tones 1 and 4, and these rules can be applied across phrase boundaries. These are to improve the naturalness of speech.

For the second type of rule, it states that a particular Chinese character is to be read in tone α if it occurs before/after a tone β, and whether the rule can be applied across phrase boundaries. One of such rules states that the Chinese character 不 is to be read in tone 2 if it occurs before a tone 4, as in 不对. Three other rules state that 一 is to be read in tone 4 if it occurs before tones 1, 2 or 3. Finally, three rules change the tone of 一, 七 or 八 to tone 2 when it occurs before a tone 4. All the rules belonging to the second type can apply across phrase boundaries.

2.5 Duration Assignment and Manipulation

Yanhui (宴会) sets a duration for each phoneme in a statement. There are altogether 61 different phonemes defined in the system. A phone table lists all the phonemes, their default duration value in milliseconds, and an attribute that indicates whether it is voiced or not. All consonant phonemes are unvoiced except for *m*, *n*, *l* and *r*, and all vowel phonemes are voiced. This voicing attribute is used in the pitch manipulation.

First of all, the default duration value is assigned to all the phonemes [7]. Then, a few sets of rules are applied to modify the duration values. The first set of rules modifies the duration according to the tone of the syllable. This is because for different tones, different amounts of time are used to pronounce them. For example, for a tone 3 syllable, the durations of all its phonemes are increased by 10%. The following list shows all the duration modification rules by tone of syllable.

Tone	Modification
1	No change
2	No change
3	+10%

4	−10%
5	−50%

The second set of rules takes into account the number of syllables in the word. For longer words, we usually read them quicker. The effect of these rules spread across all the syllables of the word. The following list show all the rules belonging to this category.

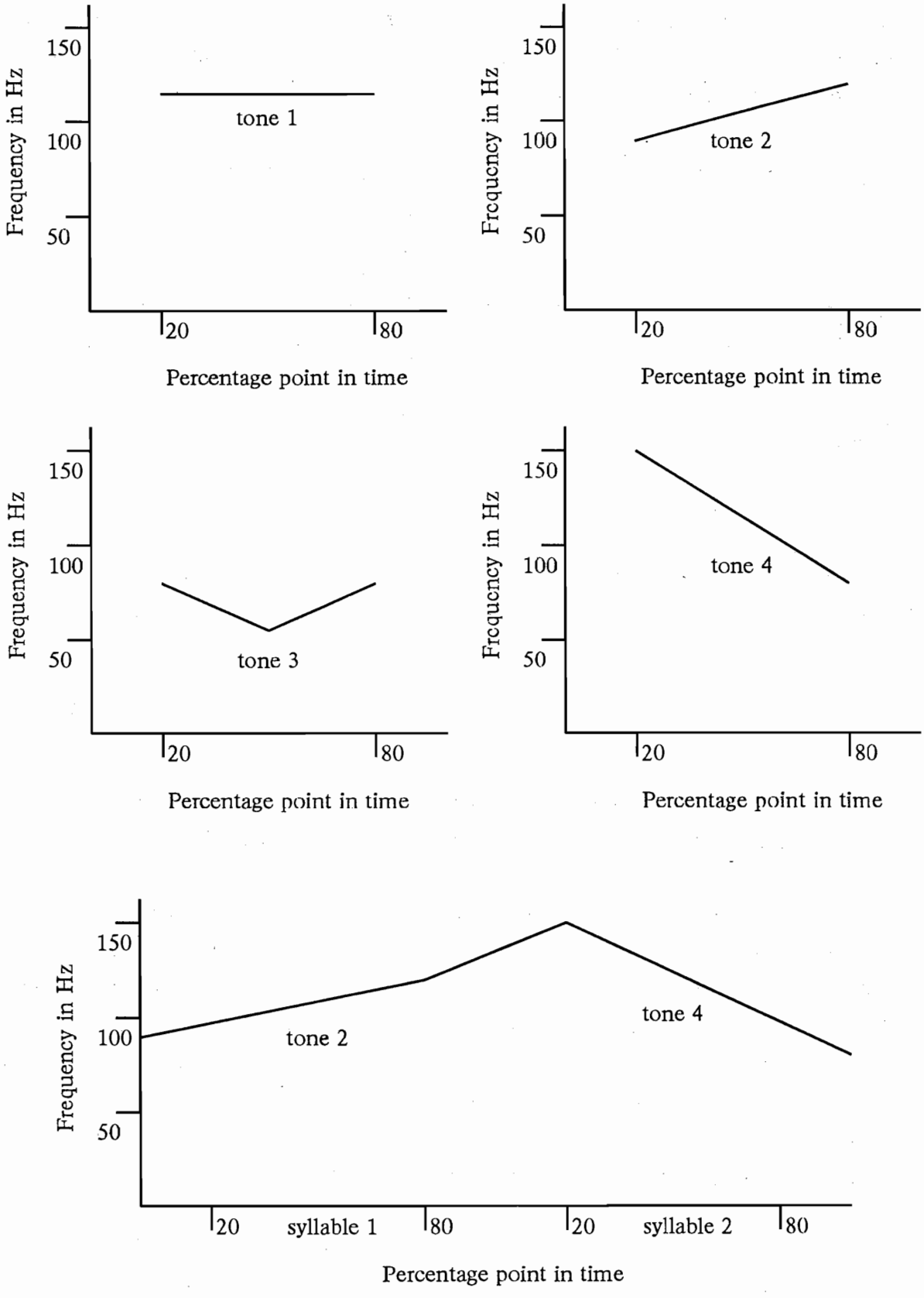
Number of syllables	Modification
1	+20%
2	+10%
3	No change
4	−20%
> 4	−30%

All punctuation marks, such as 。 、 : ! , including the separators from the text normalization module, are translated into 200ms pauses. Moreover, prepausal syllables are pronounced with longer durations. Thus, the duration values of all phonemes belonging to prepausal syllables are increased by 20%.

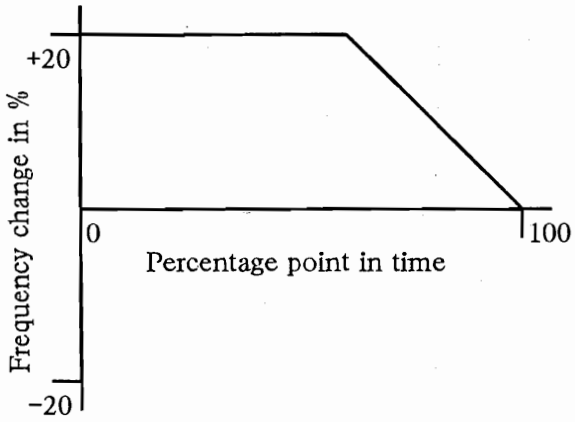
2.6 Pitch Assignment and Manipulation

The pitch contour for each syllable is very important in Yanhui (宴会) because Mandarin is a tonal language. Eight different tonal templates are defined in our system. The default pitch contours are first assigned to only the voiced phonemes for each syllable based on the tonal templates. If a syllable has more than one voiced phoneme, the pitch contour is distributed among them according to their duration ratios. The diagrams on the next page show the tonal templates used in our system for the four lexical tones.

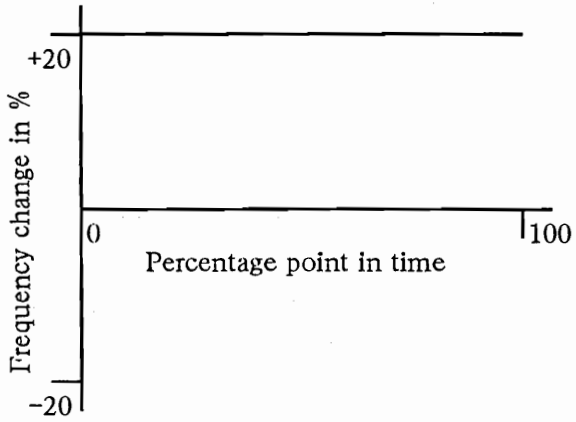
Notice that the tonal template only specifies the pitch contour between 20% and 80% points in time for the syllable. This is to facilitate co-articulation between two syllables. The pitch contour from the 80% point of a syllable to the 20% point of the next syllable is linearly interpolated between the two points. For the first and last syllable, the first 20% point and the last 80% point are extended to 0% and 100% respectively. The bottom diagram on the next page shows the pitch contour of a tone 2 syllable followed by a tone 4 syllable.


Frequency in Hz
150
100
50
tone 1
20
80
Percentage point in time
Frequency in Hz
150
100
50
tone 2
20
80
Percentage point in time
Frequency in Hz
150
100
50
tone 3
20
80
Percentage point in time
Frequency in Hz
150
100
50
tone 4
20
80
Percentage point in time
Frequency in Hz
150
100
50
tone 2
tone 4
20
syllable 1
80
20
syllable 2
80
Percentage point in time

Finally, we apply an intonation contour for the entire statement. Currently, intonation contours are defined only for statements that end with a question mark. Two types of intonation contours are defined. The first contour applies to questions that end with 吗 or contains an A不A construct. The second contour applies to all other types of questions. The following diagram illustrates these two intonation contours.



First intonation contour for questions



Second intonation contour for questions

Having set the duration and pitch attributes for each phoneme in the statement, the processing of the front-end components of Yanhui (宴会) is complete.

3. BACK-END

Since the main objective of this paper is to discuss the front-end components of Yanhui (宴会), the back-end components will only be briefly described.

Our back-end is a diphone [8] concatenation synthesis engine. A diphone describes the transition from one phoneme to another. In other words, it is a waveform containing the second half of one phoneme and the first half of the following phoneme. The beginning and end of a speech waveform are denoted by a diphone transition from silence to the first phoneme which consists of only the first half of the phoneme, and a diphone transition from the last phoneme to silence which consists of only the second half of the phoneme. For example, the pinyin *ba* is made up of 2 phonemes, *b* and *A*. To pronounce this syllable, 3 diphones are required, namely, *sil-b*, *b-A*, and *A-sil*.

The back-end receives from the front-end described above a list of phonemes with the duration and pitch attributes. Each phoneme is constructed by retrieving its component diphones from the voice table, which contains voice characteristics, DSP

routines, and voice data. The diphone boundaries are refined to eliminate phase mismatch at the juncture and blended to ensure smoothness. The pitch periods in the speech waveform are modified in the time-domain to correspond to the values specified by the front-end. The segment duration of each phoneme is adjusted to the specified values by inserting or deleting speech samples in a pitch synchronous manner. Finally, the resulting speech is passed through a post-filter to reduce the artifacts resulting from the manipulation of speech waveform for pitch and duration modification, and to improve smoothness in the synthesized speech. The computational complexity of the signal processing algorithms is kept to the minimum to ensure real-time implementation of the synthesizer.

Yanhui (宴会) currently has only one male voice. Two voice tables were generated from real speech data as described below:

1. An optimal recording script is constructed with complete diphone coverage for Mandarin. The words in the script are chosen from a large phonetic dictionary, filtered by several criteria, including phonological context for the target diphone, length/ambiguity/difficulty of the word's pronunciation, etc.

2. A high-quality speaker is selected and the script is recorded under professional conditions.

3. The speech is digitized and phoneme/diphone boundaries are determined by semi-automatic methods.

4. Amplitude normalization is performed to ensure smooth blending of diphone boundaries.

5. The optimal candidate for each diphone unit is chosen.

6. Several cycles of test listening and inventory adjustment occur.

7. The voice table is compressed using vector quantization. 8:1 and 24:1 compression ratios are used and the sizes of the compressed voice tables are 1.7MB and 650KB respectively.

4. FUTURE DIRECTIONS

Although Yanhui (宴会) has been receiving good ratings from several native Mandarin speakers, it is still far away from human voice quality. In this section, we would like to present some future directions to improve the quality and memory footprint of Yanhui (宴会).

A close look of the word segmentation module reveals some errors, especially in names. Continuous improvement on word segmentation will not only improve correctness of pronunciation, but also improve intonation of the whole sentence. We would also like to incorporate deeper syntactic and semantic analysis to enhance our bracketing algorithm to improve the grouping of words and therefore better intonation. These segmentation and bracketing improvements will also improve the correctness of sanhdi rules because those rules rely on correct word and phrasal boundaries.

For back-end component, we would like to consider syllable unit, instead of phoneme unit. With bigger units such as syllables, we could have more natural speech because of far fewer junction points. It will also reduce the requirement of smoothing on pitch contours and waveforms. Of course, we need to pay the price of larger memory footprint for more natural speech by switching to syllable units.

One of the tasks of the present back-end module is to convert stored compressed speech, recorded with a certain pitch contour and duration, into speech with the desired pitch contour and duration. If the desired pitch and duration is significantly different from the pitch of the stored waveform the quality of the synthesized speech is affected. This is because modifications have to be made to the stored speech signal to convert its underlying pitch and duration to the desired pitch and duration. The larger the pitch and duration change the more modifications are required. Using this approach it is therefore not possible to change the pitch and duration significantly without adversely affecting speech data describing the vocal tract information and hence the synthesized speech quality. One way to allow more flexibility in the Mandarin system is to use LPC based synthesis methods [9].

One approach is to use Residual Excited Linear Predictive coding (RELP). A speech signal comprises vocal tract shape and excitation signal. One could use the whitening filter to split the speech signal into its two components. The vocal tract shape is then represented by the characteristics of the whitening filter (known as a Linear Predictive model). The output of the filter produces a signal known as the residual signal

and represents the excitation signal. For speech synthesis the speech signal can be recreated by passing the residual (or excitation) signal through the Linear Predictive (Vocal Tract) model, hence Residual Excited Linear Prediction (RELP). The main advantage of this approach is that the pitch can be changed by modifying the residual signal alone and thereby not affecting the Vocal Tract model. The result is that the quality of the synthesized speech is more robust to pitch and duration changes.

The other approach uses the Linear Predictive (Vocal Tract) model parameters and a model of the excitation signal to produce the synthesized speech directly. The excitation signal is modeled as a train of impulses separated by the pitch period. The advantage of this approach is that any pitch and duration can be requested and the quality of the synthesized speech will not be affected. The disadvantage of this approach in the past has been that the general quality of the synthesized speech is poor. One other big advantage of this approach is that only the LPC model parameters need to be stored. Therefore the memory footprint of the system with this method is much smaller than with the other two approaches.

REFERENCE

[1] Kai-Fu Lee, "The Conversational Computer: An Apple Perspective", The Proceedings of EUROSPEECH, pp. 1377-1384, 1993.

[2] L. S. Lee, C. Y. Tseng, H. Y. Gu, F. H. Liu, C. H. Chang, Y. H. Lin, Y. Lee, S. L. Tu, S. H. Hsieh, and C. H. Chen, "Golden Mandarin (I) - A Real-time Mandarin Speech Dictation Machine for Chinese Language With Very Large Vocabulary", IEEE Transactions on Speech and Audio Processing, Vol. 1, No. 2, pp. 158-179, 1993.

[3] J. S. Chang, C. D. Chen and S. D. Chen. "Chinese Word Segmentation through Constraint Satisfaction and Statistical Optimization", Proceedings of ROCLING-IV, ROC Computational Linguistics Conference, pp. 147-165, Kenting, Taiwan, ROC, 1991.

[4] T. H. Chiang, M. Y. Lin and K. Y. Su, "Statistical Models for Word Segmentation and Unknown Word Resolution", Proceedings of ROCLING-V, ROC Computational Linguistics Conference, pp. 121-146. Taipei, Taiwan, ROC, 1992.

[5] C. K. Fan and Tsai W. H., "Automatic Word Identification in Chinese Sentences by the Relaxation Technique", Computer Proceedings of Chinese and Oriental Languages, Vol. 4, No. 1, pp. 33-56, 1988.

[6] Xiao-nan Susan Shen, "The Prosody of Mandarin Chinese", 1989.

[7] Yueh-Chin Chang, Bang-Er Shia, Yi-Fan Lee, Hsiao-Chuan Wang, "A Study on the Prosodic Rules for Chinese Speech Synthesis", Prroceedings of International Conference on Computer Processing of Chinese and Oriental Languages, pp. 210-215, 1991.

[8] L. S. Lee, C. Y. Tseng, M. Ouh-young, "The Synthesis Rules in a Chinese Text-to-Speech System", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 37, No. 9, pp. 1309-1319, 1989.

[9] Chi-Shi Liu, et al., "A Chineswe Text-to-Speech based on LPC Synthesizer", Telecommunication Laboratories Technical Journal, Vol. 19, No. 3, 1989.