RECENT IMPROVEMENTS ON MICROSOFT'S TRAINABLE TEXT-TO-SPEECH SYSTEM - WHISTLER

X. Huang, A. Acero, H. Hon, Y. Ju, J. Liu, S. Meredith, M. Plumpe

Microsoft Research
One Microsoft Way
Redmond, Washington 98052, USA

ABSTRACT

Whistler Text-to-Speech engine was designed so that we can automatically construct the model parameters from training data. This paper will focus on recent improvements on prosody and acoustic modeling, which are all derived through the use of probabilistic learning methods. Whistler can produce synthetic speech that sounds very natural and resembles the acoustic and prosodic characteristics of the original speaker. The underlying technologies used in Whistler can significantly facilitate the process of creating generic TTS systems for a new language, a new voice, or a new speech style. Whisper TTS engine supports Microsoft Speech API and requires less than 3 MB of working memory.

1. INTRODUCTION

Although Text-to-Speech (TTS) systems today have achieved a high level of intelligibility, their unnatural prosody and synthesis voice quality still prevent them from being widely deployed in man-machine communication. In addition, the process of building a new synthesis voice often is highly labor-intensive.

For prosody modeling, most traditional TTS systems use linguistic rules to define the prosody parameters [[9]][[12]]. Only limited natural language processing is generally used prior to prosody parameter generation. These rule-based prosody models tend to sound robotic. Moreover, while these rules may have been derived from speech of a donor speaker, the resulting synthetic prosody typically does not resemble the prosody of the original speaker. To increase naturalness, stochastic learning techniques such as decision trees [[2]][[4]][[13]][[14]] have been recently proposed to learn the prosody from a hand-labeled prosody corpus. Nonetheless, the creation of a prosody-labeled corpus remains a labor-intensive process.

For speech generation, there are two main methods used: formant synthesis [[1]] and concatenative synthesis [[2]][[14]][[15]]. Formant synthesizers use a simple model of speech production and a set of rules to generate speech. While these systems can achieve high intelligibility, their naturalness is typically low, since it is very difficult to accurately describe the process of speech generation in a set of rules. In recent years, data-driven approaches such as concatenative synthesis have achieved a higher degree of naturalness. Nevertheless, these speech units are still tediously extracted by human experts. As there are thousands of possible co-articulation contextual variations, the process of creating a good quality TTS system often takes years. Formant synthesizers may sound smoother than concatenative synthesizers because they do not suffer from the distortion encountered at the concatenation point. To reduce this distortion, concatenative synthesizers often select their units from carrier sentences, or monotone speech, and/or perform spectral smoothing, all of which can lead to a decrease of naturalness. The resulting synthetic speech may not resemble the donor speaker in the training database.

Another data-driven approach used to minimize the number of concatenation points is to select large units, such as syllables or words. While this approach allows for excellent voice quality, it results in a large non-scaleable system, and it does not generalize well to new acoustic contexts.

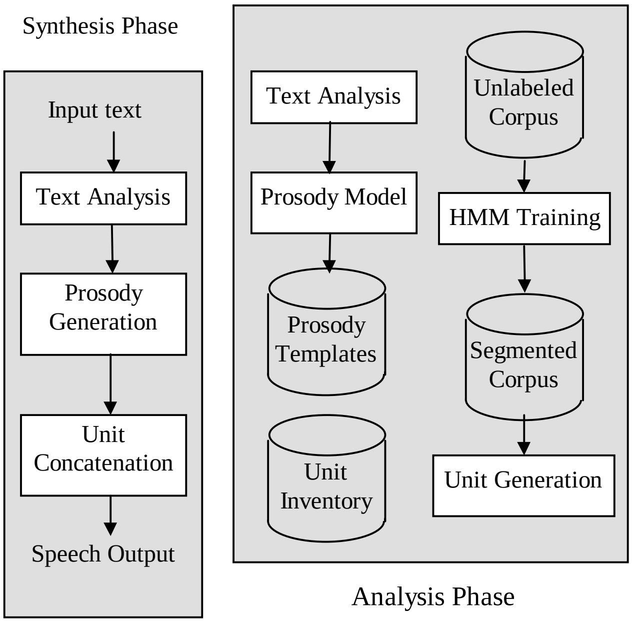

Figure 1. Block diagram of the Whistler TTS system. The left part represents the run-time synthesis, while the right part represents the analysis phase.

The objective of Microsoft's *Whistler* (Whisper Highly Intelligent Stochastic TaLkER) TTS system [[6]][[10]] is to make the system completely trainable. We will discuss the underlying technology used in Whistler and its most recent improvements. Our goal is to leverage our work in the *Whisper* speech recognition system [[5]] to make Whistler trainable, scaleable and natural. Our current implementation used the text analysis component derived from Lernout & Hauspie's commercial TTS engine [[16]]. A block diagram of Whistler can be seen in Fig. 1. The left part of the diagram corresponds to the run-time part of the system, whereas the right part corresponds to the analysis module. The analysis phase is required to obtain the inventory of acoustic units and to determine the parameters of the prosody model.

Our data-driven approach takes advantage of the analysis of large amounts of natural speech and avoids the over-generation problem found in traditional hand-tuned systems. Our methods not only improved naturalness but also decreased the time required to create a new voice, and made the synthetic speech similar to the original donor speaker. We were also able to build a highly scaleable system, which can tradeoff voice quality and memory size; generality and specificity.

This paper is organized as follows. In Section 2 we discuss Whistler's front-end, and our corpus-based prosody model. In Section 3 we then describe Whistler's back-end, and how we extract acoustic units from the corpus. Finally we summarize our major findings and outline our future work.

2. WHISTLER'S FRONT-END

2.1 Text Analysis Model

We have used a text analysis component derived from Lernout & Hauspie's commercial TTS system [[16]]. This component performs efficient text analysis functions, including text normalization, grapheme-to-phoneme conversion, part-of-speech tagging, moderate grammatical analysis, and intonational specifier generation. The Whistler prosody model assigns pitch contours to input sentences based on the analysis this component provides. Alternative text analysis modules may be easily substituted in future versions.

2.2 Prosody Model

The prosodic patterns of speech can be modeled by predicting several accentual classes of syllables. These classes are characterized by their phonological, grammatical, and contextual properties, as well as by tonal markings or tones that occur on most syllables: a high tone (H), a low tone (L), combinations of H and L, or unmarked (*) [[12]]. Further additional special markings can be used for syllables in words ending a phrase or sentence.

For clarity, we define:

- ***Clause*** as a linguistic phrase of *N* syllables, forming a natural pause-bounded unit;
- ***Clause Specification Vector S*** as an *N*-dimensional vector of the tones and other context features for each of the clause's syllables; and
- ***Clause Pitch Vector P*** as an *N*-dimensional vector of the F0 values for each syllable of the corresponding vector ***S***.

The runtime model is scaleable. In its most comprehensive form, it consists of a very large set of ***S:P*** pairs derived from a professional speaker's reading of a variety of text genres in a uniform style. These are supplemented by a phrase specification table, which captures general characteristics of representative very short phrases of a wide variety of types and contexts. The phrase specification table is used to generate high-quality defaults as described below. In its most minimal form, the runtime system relies on the phrase specification table alone for contour generation.

The prosody generation module has the following components:

- *Clause Specification.* Given an input clause analysis of an input ***i*** of *N* syllables, consisting of lexical stress marks and the annotations produced by the text analysis module, this component generates an *N*-dimensional clause specification vector ***S(i)***.
- *Prosody Contour Generation.* We then select the specifier ***S*** from the donor's database which is most similar to the input vector ***S(i)***. We use its corresponding prosody vector ***P*** to generate the pitch contour anchor points. If no clause pattern in the database ***S:P*** is sufficiently close by the matching metric, every individual syllable in the input string will be assigned a high quality default pitch, based on the phrase context of each as indexed by the phrase specification table. The phrase specification table will also be used when the system is running in minimized mode, without the ***S:P*** database present.
- *Stochastic Variation.* When an exact specification vector match is not present, or when running in minimized mode with only the phrase specification table, stochastic variation is applied to the anchor values obtained from either the (partially mismatched) vector ***P*** for ***S(i)*** or from the phrase specification table.
- *Contour Interpolation and Smoothing.* Interpolation across unspecified phonemes and smoothing over a short window across the clause are then applied for a more natural effect.

While prosody parameters generally refer to pitch, duration and amplitude, we found that with our rich context-dependent units (see Section 3.1), the use of the unit-based amplitude and duration resulted in fairly natural speech.

To obtain the various clause specification vectors, we need to assign tones and other contextual feature annotations to each syllable in the training database. Because doing this by human inspection is a labor intensive process, and because in the synthesis process the prosody module has to derive the specification vector ***S(i)*** directly from input text, we have experimented with consistent use of the text analysis module for annotation of the training data as well. In combination with a carefully constructed recording script, and a well-trained speaker, this has resulted in acceptable levels of consistency between linguistic feature annotation and the speaker's recording, in representative samples. Where variance has been found, it appears to be systematic, providing a consistent effect of individual style. The pitch vector ***P*** for each clause is derived by representative sampling of salient points from each syllable of the speaker's recorded utterances.

The assignment of the template closest to the input specification vector is straightforward when there is an exact match of annotation features. Otherwise, a dynamic programming algorithm is used in conjunction with a cost function (distortion measure) to align tonal vectors of different length or type. The cost function values are derived under plausible assumptions regarding intonational structure. For instance, the cost of mismatching a right-edge boundary tone (say the extra-high H at the end of certain yes-no questions) is much higher than having to interpolate over a 'missing' unaccented syllable earlier in the clause.

The sentences in the training script are selected to correspond to statistically frequent tonal patterns in natural speech. We have derived approximately 10,000 different clause specifiers from our 6,000-sentence corpus. Coverage of these for test texts ranges from 40% to 80%, depending on the cost threshold set in the matching process, with the remainder being made up as high-quality defaults from the phrase specification table, described above.

Whistler's data-driven model has resulted in a relatively natural prosody. The prosodic speaking style can be readily identified when used in conjunction with the units extracted from the same speaker that are described in the following section.

3. WHISTLER'S BACK-END

3.1 Unit Generation

Concatenative synthesizers accomplish unit generation by cutting speech segments from a database recorded by a target speaker [[11]]. There are three phases in the process of building a unit inventory:

- Conversion between a phoneme string and a unit string.
- Segmentation of each unit from spoken speech.
- Selection of a good unit instance when many are available in the corpus.

Traditionally, the conversion between a phoneme string and a unit string has been handled by defining the unit as a *diphone*, which contains the transition between two phones. In English there are about 1500 to 2000 diphones, and given this choice of unit the mapping is straightforward. While the choice of the diphone as the basic unit retains the transitional information, there can be large distortions due to the difference in spectra between the stationary parts of two units obtained from different contexts. As evidenced in today's diphone-based systems, naturalness of synthetic speech can be sometimes significantly hampered by the context mismatch of diphone units. Once the set of units has been decided, the database is traditionally manually segmented into those diphone units, a labor-intensive process. In addition, selection of the representative diphone units can only be done on a trial and error basis, which doesn't usually address the potential distortion at any concatenation point.

To achieve a more natural voice quality, one must take more contexts into account, going beyond diphones. However, simply modeling *triphones* (a phone with a specific left and right context) already requires more than 10,000 units for English. Fortunately, effective clustering of similar contexts modeled in a sub-phonetic level, to allow flexible memory-quality compromise, has been well studied in the speech recognition community [[5]]. Whistler uses decision tree based senones [[3]][[7]] as the synthesis units. A senone is a context-dependent sub-phonetic unit which is equivalent to a HMM state in a triphone (which can be easily extended to more detailed context-dependent phones, like quinphone). A senone could represent an entire triphone if a 1-state HMM is used to model each phoneme. The senone decision trees are generated automatically from the analysis database to obtain minimum within-unit distortion (or entropy). As widely used in speech recognition, the use of decision trees will generalize to contexts not seen in the training data based on phonetic categories of neighboring contexts, yet will provide detailed models for contexts that are represented in the database.

To segment the speech corpus we used the speech features developed in Whisper [[5]] to align the input waveform with phonetic symbols that are associated with HMMs states. HMMs are trained from the speaker-dependent data of the target speaker. Our training database for unit selection contains about 6,000 phonetically balanced sentences, recorded in natural style. Both the decision tree and hidden Markov models are trained with the speaker-dependent data. With our current training database, we noticed that 4% of the training sentences contain some gross segmentation errors (larger than 20 ms) when compared to hand labeled data, which were mostly caused by incorrect transcriptions. Nevertheless, good context coverage and consistent segmentation by HMMs typically overcomes the drawback of an imperfect automatic segmentation when compared to manual segmentation.

To select good unit instances when many are available, we first compute unit statistics for amplitude, pitch and duration, and remove those instances far away from the unit mean. Of the remaining unit instances, a small number can be selected through the use of an objective function. In our current implementation, the objective function is based on HMM scores. During runtime, the synthesizer could either concatenate the best units pre-selected in the off-line analysis or dynamically select the senone instance sequence that minimizes a joint distortion function. The joint distortion function is a combination of HMM score, unit concatenation distortion and prosody mismatch distortion. Experiments indicate that our multiple instance based synthesizer significantly improve the naturalness and overall quality over traditional single instance diphone synthesizer because of its rich context modeling, including phonetic, spectral and prosodic contexts. The Whistler system is highly scaleable because the number of senones and instances per senone can be determined based on a balance of quality versus memory resources.

One interesting note is that we conducted some experiments comparing data recorded with natural and monotone pitch. On the contrary to the common belief that monotone recording is critical to derive acoustic units, we have not observed any significant improvement using monotone recording. This is probably because of our detailed context models.

3.2 Speech Synthesis

For synthesis we employ a source-filter model. This allows a more efficient coding of the acoustic inventory, as well as more flexibility in modifying the parameters at unit transitions by doing interpolation of the filter parameters.

The filter is implemented as a time-varying sequence of LPC vectors. Each LPC vector is quantized to reduce storage (At 11kHz sampling rate, 14 LPC coefficients are transformed to LSF and quantized with 4 codebooks of 256 entries each). For unvoiced segments the LPC vectors are spaced uniformly and for voiced regions the frames are centered at each pitch epoch (obtained with the aid of a laryngograph signal [[6]]). In both cases a Hanning window of fixed length is used.

Source modeling is done differently for voiced and unvoiced regions. For unvoiced segments, the source is white Gaussian noise. For voiced segments, the source is the convolution of a train of impulses with a time-varying excitation. Pitch can be changed by controlling the separation between impulses. The time-varying excitation is described in the frequency domain as the sum of a purely voiced component and colored random noise. The voiced component is the sum of independent frequency sub-bands that have been vector-quantized separately. The excitation codebooks are designed to minimize the error when reconstructing the original units. The use of mixed excitation can improve naturalness for voiced fricatives and other voiced sounds with relatively large aspiration noise, and its energy is derived from the quantization error. By using this integrated synthesis/coding framework, we can achieve naturalness and keep the acoustic inventory to less than 1MB.

4. SUMMARY

For ongoing research to further improve Whistler, we are experimenting with our Natural Language Understanding System [[8]] to improve our text analysis and prosody models. We are also experimenting with longer units with senonic baseforms for difficult contexts (like vowel-vowel transitions) and/or frequent contexts (like the most frequent triphones or words). Each senone becomes essentially our basic building block and we can use it to construct syllable/word/phrase dependent triphone sequences. These specific senone sequences can be used to cover these most needed acoustic units while each individual senone can still be shared for other generic contexts in our multiple instance stochastic unit framework.

Our preliminary work indicated that we can effectively leverage speech recognition technology to significantly improve the naturalness of TTS systems. Whistler benefited substantially from stochastic learning techniques that have been widely used in speech recognition. The data-driven approach could help to create speech output that has a paradigm-shift impact. We think that factors such as speaking style, utterance situation and the speaker's mental states could all be modeled in Whistler's stochastic data-driven framework in the future.

5. REFERENCES

[1] Allen J., Hunnicutt S., and Klatt D. *From text to speech: the MITalk system*. MIT Press, Cambridge, MA, 1987.

[2] Bailly G. and Benoit C., editors. *Talking Machines: Theories, Models, and Designs*. Elsevier Science, 1992.

[3] Donovan R.E. and Woodland P.C. "Improvements in an HMM-Based Speech Synthesizer". *Proceedings of Eurospeech Conference*, Madrid, Spain, 1995, pages 573-576.

[4] Hirschberg J. "Pitch accent in context: Predicting intonational prominence from text". *Artificial Intelligence*, 63:305-340, 1993.

[5] Huang X., Acero A., Alleva F., Hwang M.Y., Jiang L. and Mahajan M. "Microsoft Windows Highly Intelligent Speech Recognizer: Whisper". *IEEE International Conference on Acoustics, Speech, and Signal Processing*. Detroit, May 1995.

[6] Huang X., Acero A., Adcock J., Hon H., Goldsmith J., Liu J., and Plumpe M. "Whistler: A Trainable Text-to-Speech System". *International Conference on Spoken Language Processing*. Philadelphia, Oct, 1996.

[7] Hwang, M.Y. and Huang, X. and Alleva, F. "Predicting Unseen Triphone with Senones". *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Minneapolis, MN, pages 311-314. April, 1993.

[8] Jensen K., Heidorn G., and Richardson S. *Natural Language Processing: The PLNLP Approach*, Kluwer Academic Publishers, 1993.

[9] Klatt D. "Review of text-to-speech conversion for English". *Journal of the Acoustical Society of America*, 82(3):737-793, 1987.

[10] Microsoft Research's Speech Technology Group web page: http://www.research.microsoft.com/research/srg/.

[11] Nakajima S. and Hamada H. "Automatic generation of synthesis units based on context oriented clustering". *IEEE International Conference on Acoustics, Speech, and Signal Processing*. New York, April 1988, pages 659-662.

[12] Pierrehumbert J. "Synthesizing intonation". *Journal of the Acoustical Society of America*, 70:985-995, 1981.

[13] Ross K. N. "Modeling of Intonation for Speech Synthesis". *Ph.D. thesis*, Boston University, 1995.

[14] Sagisaka Y., Kaiki N., Iwahashi N. and Mimura. K. "ATR v-Talk speech synthesis system". *International Conference on Spoken Language Systems*, Banff, Canada, 1992, pages 483-486.

[15] Sproat R., Hirschberg J., and Yarowsky D. "A corpus-based synthesizer". *International Conference on Spoken Language Systems*, Banff, Canada, 1992, pages 563-566.

[16] Van Coile B. "On the Development of Pronunciation Rules for Text-to-Speech Synthesis". *Proceedings of Eurospeech Conference*, Berlin, Sep 1993, pages 1455-1458