WHISTLER: A TRAINABLE TEXT-TO-SPEECH SYSTEM

Xuedong Huang, Alex Acero, Jim Adcock,
Hsiao-Wuen Hon, John Goldsmith, Jingsong Liu and Mike Plumpe

Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052, USA

ABSTRACT

We introduce Whistler, a trainable Text-to-Speech (TTS) system, that automatically learns the model parameters from a corpus. Both prosody parameters and concatenative speech units are derived through the use of probabilistic learning methods that have been successfully used for speech recognition. 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.

1. INTRODUCTION

Traditionally, Text-to-Speech (TTS) systems convert input text into voice by using a set of manually derived rules for prosody generation and/or voice synthesis [1]. While these systems can achieve a high level of intelligibility, they typically sound unnatural. The process of deriving these rules is not only labor intensive but also difficult to generalize to a new language, a new voice, or a new speech style.

For prosody modeling, most TTS systems use linguistic rules to define the prosody parameters [5,11]. 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] have been recently proposed to learn the prosody from a hand-labeled prosody corpus. 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 [2,13,14]. 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 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.

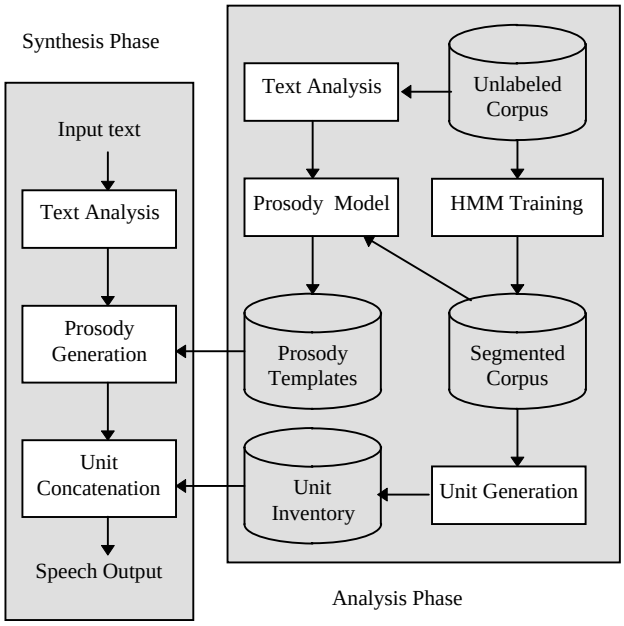

Figure 1. Block diagram of the Whistler TTS system. The left part represents the run-time synthesis, while the right part represents the analysis phase.

In this paper we introduce the *Whistler* (Whisper Highly Intelligent Stochastic TaLkER) TTS system being developed at Microsoft, and describe its underlying technologies. Our goal is to leverage our work in the *Whisper* [6] speech recognition system and Microsoft's Natural Language Processing (NLP) system [9] to make Whistler trainable, scaleable and natural. 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. In designing the system, we were also able to build a highly scaleable system which can tradeoff voice quality and memory size, generality and specificity.

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.

This paper is organized as follows. In Section 2 we discuss Whistler's front-end, 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

Whistler's text analysis leveraged Microsoft NLP engine [9] for basic text understanding. The NLP system performs a flexible parse of domain-independent sentences, producing as output a complete grammatical parse tree, including part of speech tagging for each word. From this parse tree, sentences are broken into phrases and pauses. The input text is then converted to a phonetic string with lexical stress marks and pause marks by the use of the parse tree, the phrasing information, the pronunciation dictionary and the letter-to-sound (LTS) module. The phonetic string and the word list augmented with the phrasing information are then passed to the prosody generation module.

2.2 Prosody Model

Much of the human prosodic patterns of speech can be captured by predicting one of three *tonal markings* or *tones* that occur on most syllables: a high tone (H), a low tone (L), or no special emphasis (*) [8,11,12]. Further additional special markings can be used for syllables in words ending a phrase, or ending a sentence.

Before we proceed it is useful to define the following terms:

- A *Phrase* is defined as a linguistic phrase of *N* syllables identified by NLP parsing.
- A *Phrase Tonal Vector* ***T*** is defined for each phrase as an *N*-dimensional vector of the tones for each of the corresponding syllables.
- A *Phrase Prosody Vector* ***P*** is defined for each phrase as an *N*-dimensional vector of the pitch values for each of the corresponding syllables.
- A *Phrase Prosody Contour C(t)* is a continuous function of time for pitch and amplitude that represents a phrase.

Following this model, the prosody generation module in Whistler has the following components:

1. *Tone Marking*. Given a phrase of *N* syllables, the corresponding lexical stress marks and the parse tree produced by the text analysis module, this component generates an *N*-dimensional phrase tonal vector ***T***. In our current implementation, tone marking is done automatically by rules that operate on the parse tree, the phone string and the lexical stress marks.
2. *Prosody Contour Generation*. Given a tonal vector ***T*** we select the template, extracted from the donor's database, most similar to the input tonal vector. Given the chosen tonal template, we use its corresponding prosody vector ***P***, to generate the prosody contour *C(t)* by fitting a curve through those prosody values.

While prosody parameters generally refer to pitch, duration and amplitude, the work described in this section refers only to pitch contours. We found that with our rich context-dependent units (see Section 3.2), the use of the default unit amplitude and duration resulted in fairly natural speech. Some authors report quality improvements by refining the duration of the units [14], but so far we have not obtained any significant improvement by modeling duration beyond the default value of each context-dependent unit.

To obtain the templates of different phrasal tonal vectors, we need to assign a tone (H, L or *) to each syllable in the training database. Not only is this a labor intensive process, but also there is typically no consensus in the labeling done by two different human experts. The disagreement between tonal vectors labeled by experts from speech waveforms and tonal vectors labeled automatically from text alone is even greater. Since in the synthesis process the prosody module has to derive the tonal vector ***T*** directly from input text, we decided to use the automatic labeling from text for the training data as well to achieve consistent labeling.

We clustered all tonal vectors in the training data into a set of templates. Each template is characterized by a tonal vector ***T*** and its corresponding prosody vector ***P,*** which is the one closest to the prosody mean vector. Even though the predicted tonal vector from text parsing does not necessarily match the true tonal vector of the speaker's training utterance, we observed that the selected prosody template reflected fairly well the speaking style of the donor speaker. This is probably because the tone marking is consistent across different sentence patterns in the training data, even if they differ from the tone marks generated from true speech by human experts. We can view the tone vector as the hidden variable used to index the prosodic template database.

The sentences in the training script are selected to statistically correspond to frequent tonal patterns in natural speech. We have derived about 1500 different templates for our 1000-sentence corpus. Once again, the number of templates is scaleable to balance the resource requirements and perceived naturalness.

The assignment of the template closest to the input tonal vector is straightforward when there is an exact match. A dynamic programming is used in conjunction with a cost function (distortion measure) to align tonal vectors of different length. This cost function is the sum of the individual costs associated with the modification needed to turn the tonal template into the

desired tonal vector. The individual costs are derived under the assumption that, for instance, the cost of turning a L-tone syllable into a H-tone syllable is higher than that of turning such L-tone syllable into a neutral one, because it will likely change prosody realization significantly by converting an unstressed syllable into a stressed one.

Whistler's data-driven prosody modeling resulted in a relatively natural sounding prosody in our synthesis speech. 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 Speech Analysis

In order to derive synthesis units and modify prosody, we need to analyze the speech of our corpus, including pitch tracking and speech segmentation. The acoustic speech units are represented by pitch-synchronous LPC parameters and their residual waveforms.

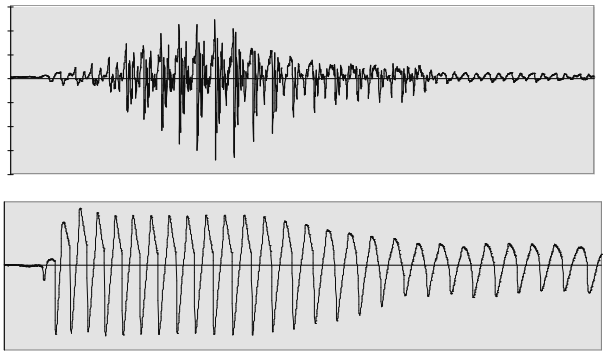

Figure 2. Comparison of original speech signal (above) and laryngograph signal (below) for the word "man" uttered by a male speaker.

To perform the pitch-synchronous LPC analysis, it is necessary to determine voicing information and the epochs of the speech signals accurately. We implemented several automatic methods that estimate this information from the speech signal, but none resulted in perfect voiced/unvoiced decision and perfect epoch accuracy. Error-free epoch determination is difficult in regions with low-amplitude, or under sudden spectral changes, or because of the all-pole inaccuracy of the LPC model on nasals, etc. Voiced/unvoiced errors can also occur in regions of low-amplitude and transitions. We have seen that even if the error rate is only 5%, the few errors that occur can lead to audible distortions that dramatically reduce the quality of the synthetic voice.

While these errors could be fixed manually, we preferred to use a laryngograph signal that measures the movement of the vocal cords directly, and leads to essentially perfect voicing and epoch detection without manual intervention. To achieve this, a stereo database was recorded with the original speech on one channel and the laryngograph signal on the other. As can be seen in Fig 2, voicing and epochs are considerably simpler to extract from the laryngograph signal than from the speech signal itself. The algorithm used in determining epochs is based on peak-picking on the pre-emphasized laryngograph signal.

Prosody modification implemented this way does generally not result in audible distortions unless the pitch change is larger than a factor of 2.

3.2 Unit Generation

Concatenative synthesizers accomplish unit generation by cutting speech segments from a database recorded by a target speaker [10]. There are three phases in the process of building a unit inventory:

1. Conversion between a phoneme string and a unit string.
2. Segmentation of each unit from spoken speech.
3. 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 solved by defining the unit as a *diphone*, that contains the transition between two phones. In English there are about 1500 to 2000 diphones, and given this choice of unit the mapping is straight forward. 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 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 the concatenation points.

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 [6]. 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). The senone decision trees are generated automatically from the analysis database to obtain minimum within-unit distortion (or entropy). 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 [6]. We used Whisper 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. We observed that 4% of the sentences in the

training set 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 dynamically selects the best senone instance sequence which minimizes the spectral distortion at the junctures. Since severe prosody modification yields audible distortion, it is possible to keep several unit instances with different pitch and duration values, and select one of them at run-time depending on the target pitch and duration. The objective function can be extended to cover sufficient prosodic variation in the unit inventory for each senone unit.

The other dimension to improve synthesis quality is to extract corresponding senone units to form longer units for difficult contexts (like vowel-vowel transitions, etc.) and/or frequent contexts (like the most frequent triphones or words). Each senone is essentially our basic building block and we can use it to construct triphone/syllable/word/phrase dependent senone sequences. These specific senone sequences can be used to cover these most needed acoustic units while each individual senone can still be shared for any generic senone in the multiple instance framework.

Experiments indicate that our multiple instance based senone 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. Finally, the Whistler system is also highly scaleable because the number of senones and instances per senone can be determined based on a balance of quality versus memory resources.

4. CONCLUSIONS

Our preliminary work indicated that we can effectively leverage speech recognition and NLP technologies to significantly improve the naturalness of TTS systems. We have shown that TTS could benefit 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 the Whistler's probabilistic data-driven framework in the future.

5. ACKNOWLEDGMENTS

The authors would like to express their gratitude to Dennis Adler, Fil Alleva, Mei-Yuh Hwang, Karen Jensen, Yun-Cheng Ju, Li Jiang, Dan Ling, Joseph Pentheroudakis, Rick Rashid and Jiang Zixin for their assistance in the development of Whistler.

6. REFERENCES

1. Allen J., Hunnicutt S., and Klatt D. *From text to speech: the MITalk system*. MIT Press, Cambridge, Massachusetts, 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. Klatt D. "Review of text-to-speech conversion for English". *Journal of the Acoustical Society of America*, 82(3):737-793, 1987.
6. 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.
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. Goldsmith J. "English as a tonal language" in *Phonology in the 1980s*. Edited by Goyvaerts D., Story-Scientia, 1980.
9. Jensen K., Heidorn G., and Richardson S. *Natural Language Processing: The PLNLP Approach*, Kluwer Academic Publishers, 1993.
10. 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.
11. Pierrehumbert J. "Synthesizing intonation". *Journal of the Acoustical Society of America*, 70:985-995, 1981.
12. Ross K. N. "Modeling of Intonation for Speech Synthesis". *Ph.D. thesis*, Boston University, 1995.
13. 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.
14. Sproat R., Hirschberg J., and Yarowsky D. "A corpus-based synthesizer". *International Conference on Spoken Language Systems*, Banff, Canada, 1992, pages 563-566.