An Overview of Text-To-Speech Synthesis Techniques

M. Z. Rashad

Department of Computer Science, Faculty of Computer and Information Systems, Mansoura University, Egypt
magdi_12003@yahoo.com

Hazem M. El-Bakry, Islam R. Isma'il

Department of Information Systems, Faculty of Computer and Information Systems, Mansoura University, Egypt
helbakry20@yahoo.com, islamcis@yahoo.com

Nikos Mastorakis

Technical University of Sofia, Bulgaria

Abstract- The goal of this paper is to provide a short but comprehensive overview of text-to-speech synthesis by highlighting its natural language processing (NLP) and digital signal processing (DSP) components. First, the front-end or the NLP component comprised of text analysis, phonetic analysis, and prosodic analysis is introduced then two rule-based synthesis techniques (formant synthesis and articulatory synthesis) are explained. After that concatenative synthesis is explored. Compared to rule-based synthesis, concatenative synthesis is simpler since there is no need to determine speech production rules. However, concatenative synthesis introduces the challenges of prosodic modification to speech units and resolving discontinuities at unit boundaries. Prosodic modification results in artifacts in the speech that make the speech sound unnatural. Unit selection synthesis, which is a kind of concatenative synthesis, solves this problem by storing numerous instances of each unit with varying prosodies. The unit that best matches the target prosody is selected and concatenated. Finally, hidden Markov model (HMM) synthesis is introduced.

Keywords: Speech Synthesis, Grapheme-to-Phoneme (G2P) Conversion, Concatenative Synthesis, Hidden Markov Model (HMM)

1. Introduction

Speech is the primary means of communication between people. The goal of speech synthesis or text-to-speech (TTS) is to automatically generate speech (acoustic waveforms) from text [1]. In other words, a text-to-speech synthesizer is a computer-based system that should be able to read *any* text aloud. There is a fundamental difference between text-to-speech synthesizer and any other talking machine (such as a cassette-player) in the sense that we are interested in the automatic production of new sentences [2].
Speech synthesis performs this mapping in two phases. The first one is text analysis, where the input text is transcribed into a phonetic representation, and the second one is the generation of speech waveforms, where the acoustic output is produced from this phonetic and prosodic information. These two phases are usually called as high- and low-level synthesis.
There are three main approaches to speech synthesis: articulatory synthesis, formant synthesis, and concatenative synthesis. Articulatory synthesis generates speech by direct modeling of human articulator behavior. Formant synthesis models the pole frequencies of speech signal. Formants are the resonance frequencies of the vocal tract. Since the formants constitute the main frequencies that make sounds distinct, speech is synthesized using these estimated frequencies. On the other hand, concatenative speech synthesis produces speech by concatenating small, prerecorded units of speech, such as phonemes, diphones, and triphones to construct the utterance. The following figure gives a high-level block diagram of the concatenative TTS synthesis process.

2. Text Analysis

2.1 Text normalization

The first task of all text-to-speech systems is to preprocess or normalize the input text in a variety of ways. We will need to break the input text into sentences. For each sentence, we divide it into a sequence of tokens (such as words, numbers, dates and other types). Non-

natural language tokens such as acronyms and abbreviations must be converted to natural language tokens. In the following subsections, the steps of text normalization are explained in more details.

2.1.1 Sentence Tokenization

The first task in text normalization is sentence tokenization. This step has some difficulties because sentence boundaries are not always indicated by periods and can sometimes be indicated by other punctuations like colons. To determine sentence boundaries, the input text is divided into tokens separated by whitespaces and then any token containing one of these characters ! , . , or ? is selected and a machine learning classifier can be used to determine whether each of these characters inside these tokens indicate an end-of-sentence or not.

2.1.2 Non-Standard words

The second task in text normalization is normalizing non-standard words such as numbers, abbreviations or acronyms. These tokens need to be converted to a sequence of natural words so that a synthesizer can pronounce them correctly. The difficulty with non-standard words is that they are often ambiguous. For example, a number like 1773 can be spoken in a variety of ways and the The final task in text normalization is homograph disambiguation. Homographs are words that have the same spelling but differ in their pronunciation. For example, the two forms of the word use in the following sentence "It's no use to ask to use the telephone." have different pronunciations. The correct pronunciation of each of these forms can easily be determined if the part-of-speech is known. The first form of the word use is a noun whereas the second one is a verb. Indeed, Liberman and Church (1992) showed that the part-of-speech can disambiguate many of the most frequent homographs in 44 million words [4]. For situations in which homographs cannot be resolved by the part-of-speech, a word sense disambiguation algorithm can be used to resolve them.

Formant synthesis is based on the source-filter model of speech production. In this model, speech is generated by a basic sound source, and then modified by the vocal tract. The sound source for vowels is a periodic signal with a fundamental frequency. For unvoiced consonants a random noise generator is used. Voiced fricatives use both sources. To produce intelligible speech three formants are needed

correct way is determined from the context. The previous number is read as *seventeen seventy three* if it is a part of a date. It is read as *one thousand, seven hundred, and seventy three* if it is a measure. Or it can be read as *one seven seven three* if it is a part of a ZIP code or if it is a part of a password. Acronyms can be pronounced letter by letter such as CD or they can be pronounced as if they were words such as RAM. For the dollar sign ($) symbol, the word order must be changed. For example, an expression like $10 million must be read out as ten million dollars. To resolve the ambiguity of non-standard words, each non-standard word must be assigned a type. For example, the NSW classifier of Sproat et al. (2001) uses 136 features, such as "all-upper-case", "has-two-vowels", and "contain-slash" to determine the correct type of each non-standard word [3]. After that these non-standard words need to be expanded into natural words. For some types like LSEQ or ASWD, this expansion is trivial. For some other types, this expansion may be quite complex. For example, a type such as EXPN which includes abbreviations and acronyms must be expanded with the help of an abbreviation dictionary. The following table lists some of these NSW types.

2.1.3 Homograph Disambiguation

and up to five formants are needed to produce high quality speech.
The two basic structures of formant synthesizers are cascade and parallel formant synthesizers. A cascade formant synthesizer consists of resonators connected in series and the output of each resonator is fed into the next one. The cascade configuration is simpler than the parallel configuration and the formant amplitudes do not need individual control. The cascade structure has been found to be better for non-nasal voiced sounds. In a parallel formant synthesizer, each formant is modeled in isolation and the source signal is fed through each separately. Then the outputs of all the formants are summed. The parallel configuration has an amplitude control of each formant. The parallel structure has been found to be better for nasals and fricatives.

2.2 Pronunciation

The next stage after normalizing the input text is to find a pronunciation for each word. The main component in this stage is a large pronunciation lexicon. The pronunciation lexicon alone is not enough, because the input text can contain words such as names that cannot be found in the lexicon. For this reason, many text-to-speech systems use a name-

pronunciation lexicon in addition to the principal pronunciation lexicon. The name-pronunciation lexicon needn't be very large, since the pronunciation of many names can be produced by analogy. For example, if the name-pronunciation lexicon contains the pronunciation of the name Trotsky, but not the name Plotsky, the initial /tr/ from Trotsky can be replaced with the initial /pl/ to generate a pronunciation for Plotsky. The pronunciation of unknown words that are not found in the pronunciation lexicon can be produced via the grapheme-to-phoneme conversion methods.

2.2.1 Grapheme-to-Phoneme Conversion

A grapheme-to-phoneme (G2P) algorithm generates a sequence of phones from a sequence of characters. The earliest of such algorithms were rule based techniques. These are called letter-to-sound or LTS rules. LTS rules produce quite reasonable results for languages with a shallow orthography such as Spanish, but LTS rules produce poor results for languages like English and French. So that most modern text-to-speech systems apply data driven and statistical techniques [5].

In general there is no one-to-one correspondence between letters and phonemes. A letter can align to multiple phonemes (e.g., x often aligns to k s), two letters can align to a single phoneme (e.g., gh in rough align to f), or a letter may align to no phonemes at all (e.g., the e in cake). So a prerequisite for the data driven G2P algorithms is to align the characters with the phonemes. The first data driven G2P algorithm was the NetTalk algorithm developed by Sejnowski and Rosenberg. NetTalk uses a feed-forward neural network [6]. Pronunciation by analogy was first introduced by Dedina and Nusbaum [7]. Pagel et al introduced the use of decision trees for this purpose [8].

2.3 Prosodic Analysis

The final stage of text analysis is prosodic analysis. Prosody refers to the features that make sentences flow naturally. Without these features, speech would sound like a reading of a list of words. The three main components of prosody are phrasing, prominence, and intonation. For unit selection synthesis, an abstract representation of these features is all what is needed. For diphone and Hidden Markov Model (HMM) synthesis, a further step is needed which is to predict the fundamental frequency (F0) and the duration values.

Phrasing has many effects on speech synthesis; the final vowel of a phrase is longer than the previous vowels and there is often a drop in the fundamental frequency from the start of a phrase to its end. Phrasing prediction can be based on deterministic rules. Modern techniques for phrasing prediction are data-driven techniques. Wang and Hirschberg introduced the use of decision trees for phrase break prediction [9]. A wide variety of machine learning algorithms have been applied for phrasing prediction such as memory based learning [10] and neural networks [11].

Prominence is used to indicate the strength of a word, syllable or phrase when it is used in a sentence. A word is made more prominent by saying it louder, saying it slower, or by varying the fundamental frequency during the word. Prominent words are generally associated with pitch accent.

Intonation is the pattern of fundamental frequency variation over an utterance. An obvious example of intonation is the difference between sentences and yes-no questions in English. A sentence can be said with a final rise in F0 to indicate a yes-no question.

In the following sections, DSP component is explored. Two rule-based synthesis techniques (formant synthesis and articulatory synthesis) are explained, and then concatenative synthesis is introduced, after that unit selection synthesis is explored and finally, HMM synthesis is introduced.

3. Formant Synthesis

For vowels that can not be modeled with any of these structures, a combination of them is used. For example, In 1980 Dennis Klatt developed one of the most sophisticated formant synthesizers, it included both parallel and cascade resonators. The Klatt synthesizer was controlled by 39 parameters which were updated every 5 ms [12]. The general assessment of formant synthesis is that it can produce intelligible speech but the produced speech is far from natural.

4. Articulatory Synthesis

Articulatory synthesis generates speech by direct modeling of the human articulator behavior, so in principle it is the most satisfying method to produce high-quality speech. In practice, it is one of the most difficult methods to implement. The articulatory control parameters include lip aperture, lip protrusion, tongue tip position, tongue tip height, tongue position and tongue height [13]. There are two difficulties in articulatory synthesis. The first difficulty is acquiring data for articulatory model. This data

is usually derived from x-ray photography. X-ray data do not characterize the masses or degrees of freedom of the articulators [1]. The second difficulty is to find a balance between a highly accurate model and a model that is easy to design and control. In general, the results of articulatory synthesis are not good as the results of formant synthesis or the results of concatenative synthesis.

5. Concatenative Synthesis

The main limitation of formant synthesis and articulatory synthesis is not so much in generating speech from parametric representation, but the difficulty is in finding these parameters from the input specification that was created by the text analysis process. To overcome this limitation, concatenative synthesis follows a data driven approach. Concatenative synthesis generates speech by connecting natural, prerecorded speech units. These units can be words, syllables, demisyllables, phonemes, diphones or triphones. The unit length affects the quality of the synthesized speech. With longer units, the naturalness increases, less concatenation points are needed, but more memory is needed and the number of units stored in the database becomes very numerous. With shorter units, less memory is needed, but the sample collecting and labeling techniques become more complex.

The most widely used units in concatenative synthesis are diphones. A diphone is a unit that starts at the middle of one phone and extends to the middle of the following one. Diphones have the advantage of modeling coarticulation by including the transition to the next phone inside the diphone itself. The full list of diphones is called diphone inventory, and once determined, they need to be found in real speech. To build the diphone inventory, natural speech must be recorded such that all phonemes within all possible contexts (allophones) are included, then diphones must be labeled and segmented. Once the diphone inventory is built, the pitch and duration of each diphone need to be modified to match the prosodic part of the specification.

6. Unit Selection Synthesis

In concatenative synthesis, diphones must be modified by signal processing methods such as PSOLA to produce the desired prosody. This modification results in artifacts in the speech that can make the speech sound unnatural. Unit selection synthesis (also, called corpus-based concatenative synthesis) solves this problem by storing in the unit inventory multiple instances of each unit with varying prosodies. The unit that matches closest to the target prosody is selected and concatenated so that prosodic modifications needed on the selected unit is either minimized or not necessary at all.

Since multiple instances of each unit are stored in the unit inventory, a unit selection algorithm is needed to choose the units that best match the target specification. This selection is based on minimizing two types of cost functions, which are target cost and join cost. The target cost function, C^t, is a measure of the differences between the features of the candidate unit and the target unit. The join cost function, C^c, is a measure of the differences between the features of the candidate unit and its pervious neighbor unit. The target cost function given in equation (1) is the weighted sum of target sub-costs

$$C^t(t_i, u_i) = \sum_{j=1}^{p} w_j^t C_j^t(t_i, u_i) \quad (1)$$

where i indexes the target and candidate units and j indexes the feature vector. Also, the join cost function given in equation (2) is the weighted sum of the join sub-costs.

$$C^c(u_{i-1}, u_i) = \sum_{j=1}^{p} w_j^c C_j^c(u_{i-1}, u_i) \quad (2)$$

The unit selection is applied by finding the set of units that minimizes the total cost of an utterance of n units

$$\overline{u}_1^n = \min_{u_1,\ldots,u_n} [\sum_{i=1}^{n} C^t(t_i, u_i) + \sum_{i=1}^{n} C^c(u_{i-1}, u_i)] \quad (3)$$

Unit selection synthesis technique was first introduced by Sagisaka et al., in ATR v-Talk speech synthesis system [14]. To choose the best unit sequence, prosodic features such as duration and intonation have been added to the target specification in CHATR system [15]. The AT&T Next-Gen speech synthesis system combines the unit-selection method of CHATR with the HNM model to resolve mismatches [16].

7. Hidden Markov Model Synthesis

In unit selection synthesis, multiple instances of each phone in different contexts are stored in the database. To build such a database is a time consuming task and the database size

increases in an enormous way. Another limitation of the concatenative approach is that it limits us to recreate what we have recorded. An alternative is to use statistical parametric synthesis techniques to infer specification-to-parametric mapping from data. These techniques have two advantages: firstly, less memory is needed to store the parameters of the models than to store the data itself. Secondly, more variations are allowable for example; the original voice can be converted into another voice.

One of the most usable statistical parametric synthesis techniques is the hidden Markov model (HMM) synthesis. It consists of two main phases, the training phase and the synthesis phase. At the training phase, it should be decided which features the models should be trained for. Mel frequency cepstral coefficients (MFCC) and their first and second derivatives are the most common types of features used. The feature are extracted per frame and put in a feature vector. The Baum-Welch algorithm is used with the feature vectors to produce models for each phone. A model usually consists of three states that represent the beginning, the middle and the end of the phone. The synthesis phase consists of two steps: firstly, the feature vectors for a given phone sequence have to be estimated. Secondly, a filter is implemented to transform those feature vectors into audio signals.

The quality of the HMM generated speech is not as good as the quality of the speech generated from unit selection synthesis. The modeling accuracy can be improved by using hidden semi-Markov models (HSMMs) [17], trajectory HMMs [18], and stochastic Markov graphs [19].

8. Conclusion

Formant synthesis and articulatory synthesis are used less today but these techniques can be suitable for applications that require less memory and low processing costs. The focus nowadays is on the unit selection synthesis as it generates speech that is closest to natural than any other technique but the main limitation of the unit selection is that fewer variations are allowable on the recorded data. HMM and other statistical synthesis techniques allow more variations on the recorded data and these techniques will become dominant.

References

[1] Klatt D. H., "Review of text-to-speech conversion for English." Journal of the Acoustical Society of America, vol. 82(3), 1987.

[2] Thierry Dutoit, "High-Quality Text-to-Speech Synthesis: an Overview." Journal of Electrical & Electronics Engineering, Australia: Special Issue on Speech Recognition and Synthesis, vol. 17, pp. 25-37, 1999.

[3] Sproat R. et al., "Normalization of non-standard words." Computer Speech & Language, vol. 15(3), pp. 287-333, 2001.

[4] Liberman M. Y. and Church K. W., "Text analysis and word pronunciation in text-to-speech synthesis.", Advances in speech signal processing, pp. 791-832.

[5] Marchand Y. and Damper R., "A multistrategy approach to improving pronunciation by analogy." Computational Linguistics, vol. 26(2), pp.195–219, 2000.

[6] Rosenberg C. and Sejnowski T., "NETtalk: A parallel network that learns to read aloud.", EE & CS technical report no JHU-EECS-86/01, Johns Hopkins University, Baltimore, MD, 1986.

[7] Dedina M. and Nusbaum H., "Pronounce: a program for pronunciation by analogy." Computer speech & language, vol. 5(1), pp. 55–64, 1991.

[8] Pagel V., Lenzo K. and Black A., "Letter to Sound Rules for Accented Lexicon Compression." in Proceedings ICSLP, 1998.

[9] Wang M. Q. and Hirschberg J., "Automatic classification of intonational phrase boundaries." Computer Speech and Language, vol. 6, pp. 175–196, 1992.

[10] Busser B. et al., "Predicting phrase breaks with memory-based learning.", in 4th ISCA Tutorial and Research Workshop on Speech Synthesis, 2001.

[11] Fackrell J. W. A. et al., "Multilingual prosody modeling using cascades of regression trees and neural networks." in Proceedings of Eurospeech, 1999.

[12] Klatt D. H., "Software for a cascade/parallel formant synthesizer." Journal of the Acoustical Society of America, vol. 67, pp. 971–995, 1980.

[13] Kroger B., "Minimal Rules for Articulatory Speech Synthesis.", Proceedings of EUSIPCO92, pp.331-334, 1992.

[14] Sagisaka Y. et al., "ATR-TALK speech synthesis system." in International Conference on Spoken Language Processing, pp. 483–486, 1992.

[15] Black A. W. et al., "CHATR: A Generic Speech Synthesis System." Proceedings of the International Conference on

Computational Linguistics, pp. 983–986, 1994.
[16] Beutnagel M. et al., "The AT&T Next-Gen TTS System." in 137th meeting of the Acoustical Society of America, 1999.
[17] Tokuda K. et al., "Hidden semi-Markov model based speech synthesis." in *Interspeech*, pp. 1185–1180, 2004.
[18] Tokuda K. et al., "An introduction of trajectory model into HMM-based speech synthesis." in ISCA SSW5, 2004.
[19] Eichner M. et al., "Speech synthesis using stochastic Markov graphs." in ICASSP, pp. 829–832, 2001.

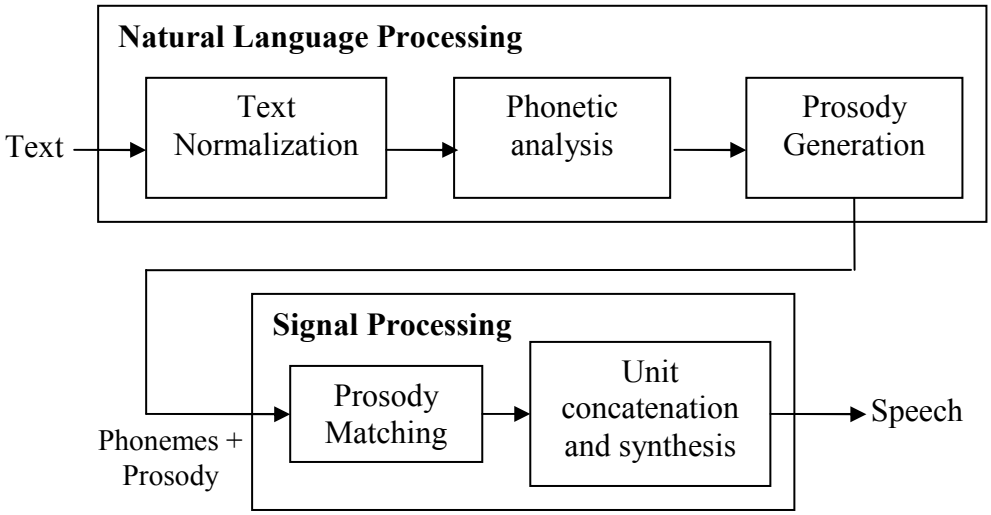

Figure 1: General block diagram of concatenative text-to-speech synthesis.

Table 1: Some non-standard words types, selected from Table1of Sproat et al. (2001)

ALPHA	**EXPN**	abbreviation	*adv, N.Y., mph*
	LSEQ	letter sequence	*DVD, D.C., PC, UN*
	ASWD	read as word	*IKEA, unknown words/names*
NUMBERS	**NUM**	number (cardinal)	*12, 45, 1/2, 0.6*
	NORD	number (ordinal)	*May 7, 3rd, Bill Gates III*
	NTEL	telephone	*212-555-4523*
	NDIG	number as digits	*Room 101*
	MONEY	money (US or other)	*$3.45, HK$300, Y20,200, $200K*
	BMONEY	money tr/m/billions	*$3.45 billion*
	PRCT	percentage	*75%, 3.4%*