International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Text to Speech Conversion Using FLITE Algorithm

Tejashree M. Shinde[1], V. U. Deshmukh[2], P. K. Kadbe[3]

[1]P.G.Scholar EnTC Department, VPCOE,Baramati, Pune, Maharashtra, India

[2] VPCOE, Baramati, Pune, Maharashtra, India

[3]VPCOE,Baramati, Pune, Maharashtra, India

Abstract*: Speech synthesis is the artificial production of human voice. A computer system used for this task is called a speech synthesizer. Anyone can use this synthesizer in software or hardware products. The main aim of text-to-speech (TTS) system is to convert normal language text into speech. Synthesized speech can be produced by concatenating pieces of recorded speech that are stored in a database. TTS Systems differ in size of the stored speech units. A system which stores phones or diphones provides the largest output range, but this may give low clarity. For specific application domains, the storage of entire words or sentences allows for high-quality output. Alternatively, a synthesizer can constitute a model of the vocal tract and other human voice characteristics to create a fully synthetic voice output. The quality of a speech synthesizer is decided by its naturalness or simillarity to the human voice and by its ability to be understood clearly. This paper summarizes the published literatures on Text to Speech (TTS), with discussing about the efforts taken in each paper. This system will be more helpful for an illiterate and visually impaired people to hear and understand the text.*

Keywords: Text To Speech (TTS).

1. Introduction

Today many efforts are taken for improving of the human interface to the computer. Because no longer people want to sit in front of monitor to read data. Since it needs too much effort to be taken, which involves strain to their eyes. In this point of view Speech Synthesis is becoming one of the most important steps towards improving the human interface to the computer. Speech synthesis is the artificial production of speech. This system is called a speech synthesizer, and it can be implemented in software or hardware products. A text-to-speech (TTS) system converts normal language text into speech; other systems render symbolic linguistic representations like phonetic transcription into speech. Voice is one of the best alternatives for hours of eye strain involved in reading any document. In case of illiterate people Voice is a better interface rather than Graphic User Interface in English. For that reason research is being done throughout the world for improving the Human Interface to the computer and one of the best options is the ability of a computer to speak to humans. Text-To-Speech is a process in which input text is first analyzed, then processed and understood, and then the text is converted in digital audio and then "spoken". It is a small piece of software, which will speak out the text given to it, such as reading from a newspaper.

2. Brief Introduction about Contributed Papers

2.1 The syllabification algorithm

Indian languages are syllable-centered so the choice of syllable as a unit for Indian languages is appropriate. In one approach,The syllabification algorithm breaks a word such that there are minimum number of breaks in the word. This algorithm dynamically looks for polysyllable units making up the word, and cross checks the database for units availability,and then breaks the word accordingly. If the polysyllable units are not there, then the algorithm naturally picks up smaller units. For example, For breaking a word .satara. algorithm looks for unit ./satara/. in database, if not found it looks for unit combinations such as ./sata/, /ra/., ./sa/, /tara/. etc.. And at the end it falls back on phone sequence ./s/, /a/, /t/, /a/, /r/, /a/.. In the next approach, the syllabification algorithm breaks a word into monosyllables without checking for its availability in the database.In this case syllabification is done based on standard linguistic rules.If a unit is not found in database it can be substituted by a nearest unit or by silence. Although, syllable based synthesis does not require significant prosodic modification, the prosodic modification that needs to be performed in the context of syllable is significantly different from that of conventional diphone based synthesis.[1]

2.2 Corpus-based speech synthesis

Corpus Method is very popular for its high quality and natural speech output. The basic idea of corpus based speech synthesis or unit selection is that to use the entire speech corpus as the acoustic inventory and to select the run time from this corpus the longest available strings of phonetic segments that match a sequence of target speech sounds in the utterance to be synthesized thereby minimizing the number of concatenations and reducing the need for signal processing.One main drawback of Courpus method is relative weighting of acoustic distance measures.It needs large speech database with optimal coverage of target domain which is often the whole language.The word or syllable based approaches give better results only in strictly closed application domain.[2].

2.3Adaptive Speech Rate Control Technology

This paper developed a new speech rate conversion method to efficiently achieve information from audio sources with very fast replay. This algorithm will help sighted people to enjoy audio books and also for visually impaired people because almost all of their information is obtained from speech.

Licensed Under Creative Commons Attribution CC BY

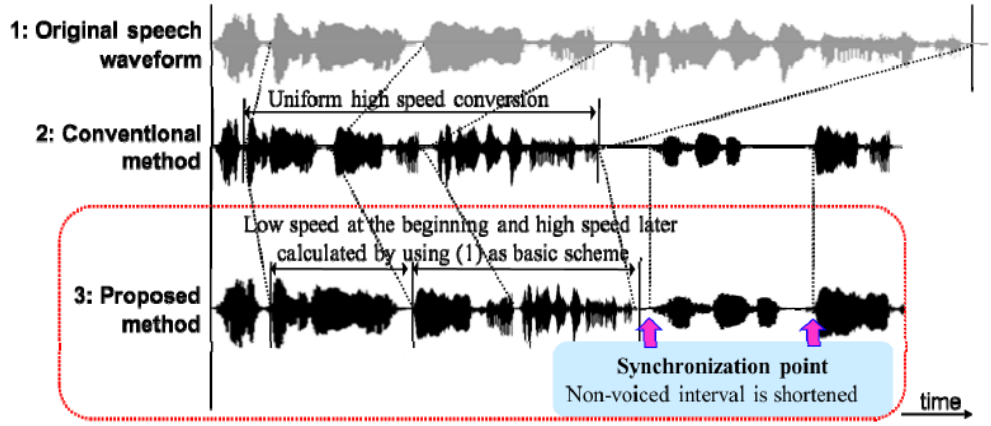


Figure 1: Example of basic operation of adaptive SRC technology

When someone is saying something important, he raise the pitch or power of his voice or both. The basic principle of SRC algorithm is to keep parts of speech that are likely to have a significant effect on listening comprehension. For instance, if both the power and the pitch of a voice are relatively low in a particular part of a continuous utterance, it can be removed from utterance.because it is expected that it may be difficult to hear in the original recording, even if played back at normal speeds.Thus it is deleted and assign the time to another section of the utterance.[3]

2.4 Synthesis of fast speech with interpolation of adapted HSMMs.

This Paper evaluate a method for generating synthetic speech at high speaking rates based on the interpolation of hidden semi-Markov models (HSMMs) which is trained on speech data recorded at normal and fast speaking rates.

With this method a better intelligibility rate and higher voice quality can be achieved as compared to standard HSMM-based duration modeling.

All synthetic voices used are built using the framework of a speaker adaptive HMM-based speech synthesis system. Model adaptation is a two-step method: The first adaptation is for the speaker transformation and the second is for speaking rate adaptation. First average voice model is trained using several background speakers from speech data at normal speaking rate.Then the average voice model adapted to two Austrian German male speakers (SPO, HPO) using speech data having the normal speaking rates.[4]

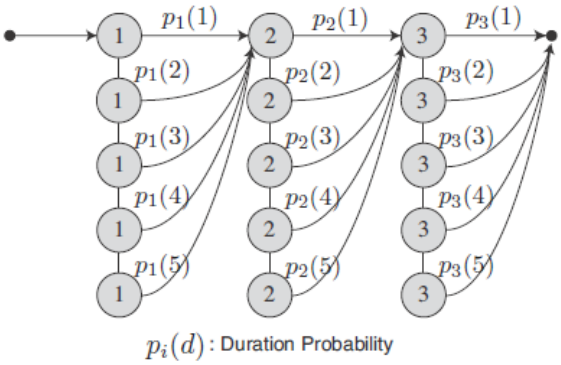


Figure 2: Syllabification Procedure

A HSMM-based method for the synthesis of fast speech that outperforms other standard methods on understanding and overall quality. Especially for blind users it is very important to have high quality synthesis techniques for fast speech. The adaptive method that is presented can be used with limited amounts of fast speech adaptation data

.

2.5 Data-driven formant synthesis.

This paper describes a new effort is taken to combine a rule-based formant synthesis approach and a corpus-driven approach. The new approach takes advantage of the fact that a unit library can better model detailed gestures then the current general rules. The rule system and the unit library is clearly separated compared to our earlier work (e.g. Högberg, 1997). The rule based model keeps the flexibility to make modifications and the possibility to include both linguistic and extra linguistic knowledge sources. Figure 3 shows the approach from a technical point of view.

A database used for createing a unit library. Each unit is described by a selection of extracted synthesis parameters with the linguistic information about the unit's original context and linguistic features. The parameters can be extracted automatically. In our traditional text-to-speech system the synthesiser is controlled by rule-generated parameters from the text-to-parameter module (Carlson et al., 1982). These parameters are represented by time and values pairs including labels and prosodic features such as duration and intonation. In this approach some of the rule-generated parameter values are replaced by values from the unit library. The process is controlled by the unit selection module that takes into account not only parameter information but also linguistic features supplied by the text-to-parameter module. The parameters are normalized and concatenated before being sent to the synthesizer.[5]

Licensed Under Creative Commons Attribution CC BY

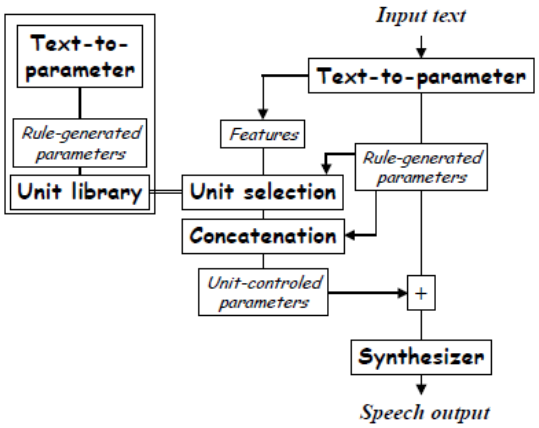


Figure 3: Rule-based synthesis system using a rule-generated unit library

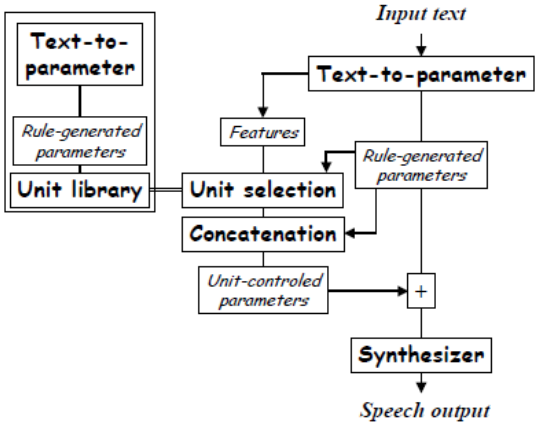


Figure 4: Rule-based synthesis system using a rule-generated unit library

2.6 Flite: a small fast run-time synthesis engine.

Flite is a small, fast run-time synthesis library suitable for embedded systems and servers. Flite is designed as an alternative for Festival in applications where speed and size are important. Voices are built using the FestVox process may be compiled into efficient representations that can be linked against Flite to produce complete text-to-speech synthesizers. The Flite algorithm is much faster and much smaller than the equivalent Festival system. Flite is the core library. For synthesis, this library require three further three parts to make a complete synthesizer. **language model** : providing phoneset, tokenization rules, text analysis, prosodic structures etc. **lexicon** : a pronunciation model including a lexicon and letter to sound rules for out of vocabulary words. **Voice-** A voice depends upon the primitives provided by the language model.[6]

The first two of these can be shared across voices of the same language. Each of these subsection are compiled into separate libraries. Each voice definition in Fest Vox include specific tokenization rules and prosody rules. However we can provide the basic tools. For basic diphone voices for known languages and simple generic limited domain voices built using the FestVox build model.Each word in the lexicon is converted to a list of letters (plus part of speech) and held in a sorted table, and each entry has an index into a list of phones (with lexical stress marked on vowels). We exclude these entries from the list that our letter to sound rules get perfectly correct. The letter-to-sound rules are built using CART techniques we further minimize the generated CART trees as follows: each tree is treated as a finite state transducer whose arcs are labeled with the questions on the nodes of the CART tree plus their answers. We then use a standard FST minimization algorithm to reduce the FST, for merging much of the later states in the tree.

3. Conclusion:

A Flite-based synthesizer has been thoroughly tested, and it runs on multiple platforms. The example voice distributed is an 8KHz diphone voice; this is the same voice as kallpc8K as distributed with Festival. That voice is rather old and not very good, but we deliberated wanted to use a stable voice as our first example so we could properly ensure the quality in Flite was the same as it is under Festival. Run-time memory requirements for Flite are less than twice the size of the largest waveform built. The current Flite system with an 8KHz diphone voice has a full footprint of 5M, 4M of code and data and 1M of RAM. The equivalent for Festival is about 30-40M.For a twenty word utterance, Flite starts writing to the audio device in 45ms, for a 40 word utterance it is about 75ms. The startup time before synthesis function is called is about 23ms. For Festival running from the command line the equivalent is about 4-5 seconds. When running as a server and using the client access method and thus exclude the start up time, we still can't make the time less that 1 second for the 20 word utterance and nearer 2 seconds for the 40 word utterance.[6]

References

[1] Venugopalakrishna Y R, Sree Hari Krishnan P, Samuel Thomasy, Kartik Bommepall, Karthik Jayanthiz, Hemant Raghavan, Suket Murarka and Hema A Murthy "Design and Development of a Text-To-Speech Synthesizer for Indian Languages" Indian Institute of Technology Madras, Chennai, India 600 036.
[2] Bernd mobius,"Corpus-based speech synthesis method and challenges"Institute of Natural Language Processing
[3] Atsushi Imai, Naoyuki Tazawa, Tohru Takagi, Toshiaki Tanaka and Tohru Ifukube "A New Touchscreen Application to Retrieve Speech Information Efficiently" IEEE Transactions on Consumer Electronics, Vol. 59, No. 1, February 2013
[4] Michael Pucher, Dietmar Schabus, Junichi Yamagishi "Synthesis of fast speech with interpolation of adapted HSMMs and its evaluation by blind and sighted listeners" Telecommunications Research Center Vienna (FTW), Austria ,The Centre for Speech Technology Research (CSTR), University of Edinburgh, UK *INTERSPEECH 2010.*
[5] Rolf Carlson, Tor Sigvardson and Arvid Sjolander "Data-driven formant synthesis" CTT, Department of Speech, Music and Hearing, KTH, TMH-QPSR Vol. 44 - Fonetik 2002.
[6] Alan W Black and Kevin A. Lenzo "Flite: a small fast run-time synthesis engine" Carnegie Mellon University.

Licensed Under Creative Commons Attribution CC BY