Building a Prototype Text to Speech for Sanskrit

Baiju Mahananda[1], C.M.S. Raju[2], Ramalinga Reddy Patil[2], Narayana Jha[1], Shrinivasa Varakhedi[3], and Prahallad Kishore[2]

[1] SCSVMV University, Kanchipuram
[2] International Institute of Information Technology, Hyderabad
[3] Sanskrit Acadamy, Osmania University, Hyderabad
{baijunanda,mouli.raju,patilrlreddy53,shrivara}@gmail.com,
kishore@iiit.ac.in
http://www.iiit.ac.in

Abstract. This paper describes about the work done in building a prototype text to speech system for Sanskrit. A basic prototype text-to-speech is built using a simplified Sanskrit phone set, and employing a unit selection technique, where prerecorded sub-word units are concatenated to synthesize a sentence. We also discuss the issues involved in building a full-fledged text-to-speech for Sanskrit.

Keywords: Text to Speech for Sanskrit, Festvox.

1 Introduction

Developing a text to speech (TTS) system for Sanskrit primarily involves studying its phonology and phonetics [2]. Sanskrit has a highly developed system of phonemic description, developed in ancient times mainly for preserving Vedic texts[3].

1.1 Goal

We try to develop a text to speech system for simple classical Sanskrit, using simplified phone set of Sanskrit. This uses Direct synthesis, in which the speech signal is generated by direct manipulation of its wave form representation. Wave form concatenation, is representative of this synthesis category. In this approach, several fundamental periods of pre-recorded phonemes are simply concatenated. The phonemes are then connected to form words and sentences [1].

1.2 Previous Work

Making a computer to talk in different languages has been attempted by many scholars from many years and they are successful. We are making an attempt to take forward the work done on "Making a Computer to talk Sanskit". The following are few studies in this path:

1. Vani - An Indian Language Text to speech Synthesizer for Sanskrit [15], is a speech synthesizer which employs formant synthesis, in which the basic

G.N. Jha (Ed.): Sanskrit Computational Linguistics, LNCS 6465, pp. 39–47, 2010.

assumption is that the vocal tract transfer function can be satisfactorily modeled by simulating formant frequencies and formant amplitudes. The synthesis thus consists of the artificial reconstruction of the formant characteristics to be produced. This is done by exciting a set of resonators by a voicing source or noise generator to achieve the desired speech spectrum, and by controlling the excitation source to simulate either voicing and voicelessness. The addition of a set of anti-resonators furthermore allows the simulation of nasal tract effects, fricatives and plosives [1].

2. Text to speech synthesis for Indian languages (Acharya), is a syllable level representation of the text and each syllable directly translates into a sound that can be synthesized or simply played from a prerecorded piece of audio [2].
3. Text to Speech conversion systems developed by C-DAC (Centre for Development of Advanced Computing) for various Indian languages supplementing the GIST Card [3].

Speech synthesis by concatenation of sub-word units (e.g. diphones) has become basic technology. It produces reliable clear speech and is the basis for a number of commercial systems. However with simple diphones, although the speech is clear, it does not have the naturalness of real speech[16].

In this work we are going to present you the general issues in building a TTS for Sanskrit, which could help in building an efficient Speaking system for Sanskrit.

2 Sanskrit Orthography

2.1 Nature of Sanskrit Scripts

The basic units of the writing system in Sanskrit are characters which are an orthographic representation of speech sounds [3]. A character in Indian language scripts is close to a syllable and can be typically of the form: C*VN, where C is a consonant, V is a vowel and N is anusvAra, visarha, jivhAmUllya e.tc. . There is fairly good correspondence between what is written and what is spoken [5]. Sanskrit has richer set of characters than many of the other Indian languages.

2.2 Phone Set

Most institutions and scholars propose "Pāṇinīya - Śikṣa" as a reference for written form of Sanskrit[8], in this book it is said that "The speech-sounds in Prakrit and Sanskrit are 63 or 64, according to their origin, has been said by Brahman (Svyambhū) himself." among which vowels are 21, and consonants are (included stop, approximent, sibilant, nasal, palatels) 43 [4]. We are using UTF-8 format to represent the text. In our system we use 8 vowels and 40 consonants including other symbols, which are given in the below picture (Fig. 2) with respective IT3 representations [1].

[1] Since the present computers can only process ASCII characters at machine level, we use a codding mechanism to represent each character (may belong to any language) in an intermediate format. This format was formulated by CMU.

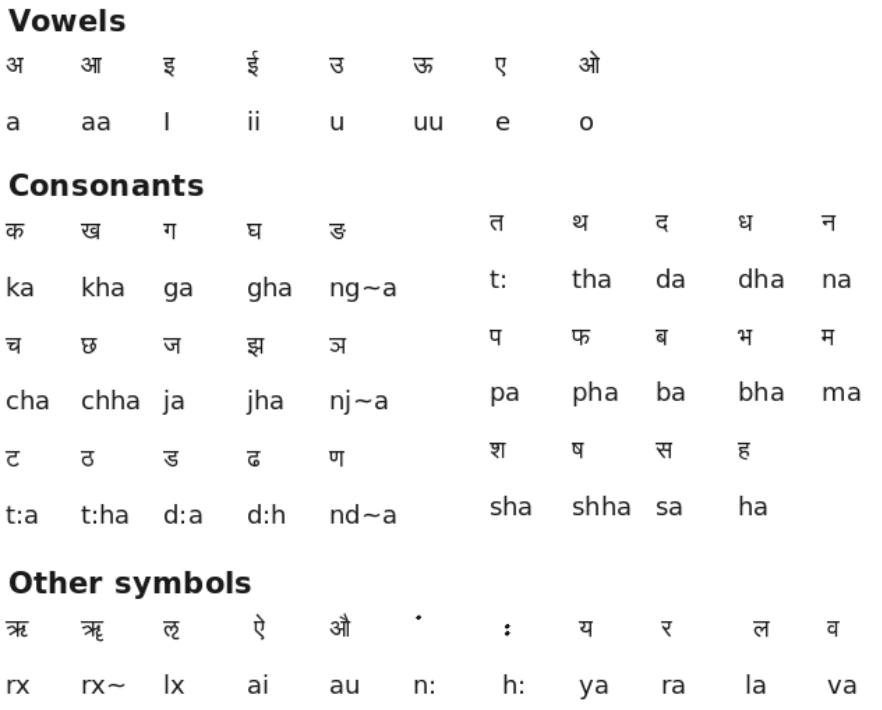

Fig. 1. Set of phones and their respective IT3 Format of Devanagari Script used in TTS

There has been many discussions on the character set of Sanskrit, but some say devanagari alphabets are developed for writing Sanskrit (which are descended from brahmi scripts) [9], however as we all can see Sanskrit can be written in multiple languages.

Letters not used in TTS - ऽ, जिह्वामूलीयः, उपध्मानीय, प्लुत, कुं, खुं, गुं, घुं

2.3 Suitability of Sikshaa Granthas for Building a TTS

Shiksha (Phonetics) explains the proper articulation and pronunciation of vedic texts. There are six parts of Shiksha letters (varnas), accent (swara), time consumed in articulating vowel (matra), effort (bala) Melodius chanting of mantras (sama) and conjugation of letters (sandhi). If some mistake is committed in any of the above six, instead of giving the desired result it can prove to be disastrous as well [8].

In order to make an efficient talking system all the parts of shiksha are to be met, however the present day systems can only add some parts of siksha, research has to be done to add the above six to the synthesized speech.

3 System Architecture

Sanskrit TTS system is implemented within the Festival/Festvox framework. Segmentation, unit selection and synthesis are done using Festvox and Festival framework.

3.1 Why Festvox?

Frameworks are built to ease the software development. But the frameworks which can be customized for user need are mostly preferred. Festvox offers such a facility and it is an open software.

4 Letter to Sound Rule

Letter-to-sound rules are almost straight forward in Indian languages, as they are phonetic in nature. We write what we speak and we speak what we write, and hence generally the necessity of a pronunciation dictionary does not arise in our case. The pronunciation for a Sanskrit word such as chatura (clever) (the word chatura is the IT3 representation of the word "चतुर" using the chart specified above)in terms of phones marked with syllable boundaries can be written as ((ch a) 1) ((t u) 0) ((r a) 0). As the characters in Indian language are close to a syllable, clustering C*VN can be done easily taking into account a few exceptions [6].

In this work, simple syllabification rules are followed. Syllable boundaries are marked at the vowel position. If the number of consonants between two vowels is more than one, then first consonant is treated as coda of the previous syllable and the rest of the consonant cluster as the onset of the next syllable. For stress assignment, the primary stress is associated with the first syllable and secondary stress with the remaining syllables in the word. The integer 1 assigned to first syllable in the word chatura indicates the primary stress associated with it. Letter to sound rules, syllabification rules and assignment of stress patterns are different for different languages, which has to be specified accordingly. The architecture of Festival synthesis engine allows these rules to be written in Scheme, so that they get loaded at the run time, essentially avoiding recompilation of the core code for every new language.

5 Creation of Speech Database

A common trend in concatenative approach for TTS system is to use large database of phonetically and prosodically varied speech.

5.1 Text Selection for Sanskrit

The quality of data-driven synthesis approaches is inherently bound to speech database from which the units are selected. It is important to have an optimal text corpus balanced in terms of phonetic coverage and the diversity in the

realizations of the units. Sanskrit has a vast literature starting from Vedic text to latest classical literature. In this work, the test corpus was generated from a set of simple Sanskrit sentences selected from daily life conversations . Some of the example sentences are given below. A common trend in concatenative approach for TTS system is to use large database of phonetically and prosodically varied speech.

हरिः ओम्! अस्ति किं गृहे ?

किं भोः, मन्त्री इव विलम्बेन आगच्छति?

मम नाम श्रीधरः।

कार्यक्रमः कथम् आसीत् ?

आगच्छतु! किम् आवश्यकम्?

5.2 Speech Data Collection

The number of Sentences recorded to generate Sanskrit database is 852. In these sentences we tried to cover the possible conversations related to our daily life. As it is not possible to complete the recording in a single session, to ensure consistency, the recordings were made at the same time every day.

6 Speech Segmentation

One of the most important tasks in building speech databases is the annotation of speech data with its contents (labeling) and the time alignment between labeling and speech (segmentation). Phoneme segmentation and labeling are highly desirable and useful for TTS as this information is used for classifying the speech units that helps to select and concatenate the right units in terms of linguistic and acoustic features.

The most precise way to annotate speech data is manually by linguistic experts. However, manual phoneme labeling and segmentation are very costly and require much time and effort. Even well trained, experienced phoneme labelers using efficient speech display and editing tools require about 200 times real time to segment and align speech utterances. To reduce this effort considerably and aid the phoneme labelers, an automated segmentation tool is required. For automatic phoneme segmentation, we have been using the most frequently used Tool EHMM based phoneme recognizer.

Spoken Sentence (UTF-8): आगच्छतु! किम् आवश्यकम्?
Spoken Sentence (IT3): aagachhatu! kim aavashyakam?

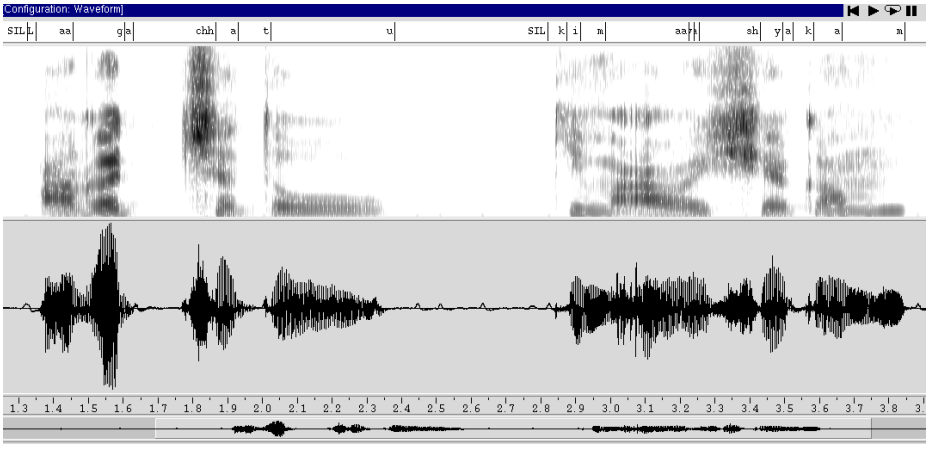

Fig. 2. Sample speech used for training with its labels, spectrogram and wave form

7 Unit Clustering and Synthesis

Given these segments, the unit selection algorithm (a Statistical Parametric Synthesizer) in the framework clustered the phones based on their acoustic differences. These clusters are then indexed based on higher level features such as phonetic and prosodic features. During synthesis, the appropriate clusters are sought using phonetic and prosodic features of the sentence. A search is then made to find a best path through the candidates of these clusters. Though the units used here are phones, the acoustic frame of previous unit is used during clustering as well as for concatenation.

Clustergen is a statistical parametric synthesizer released as part of the Festival distribution [11]. It predicts frame-based MFCCs clustered using phonetic, metrical, and prosodic contexts. Unlike CLUNITS, the unit size is one frame (5ms by default), and the signal is partitioned at the HMM-state size level (3 states per phone). The clustering, done via CART, optimizes the standard deviation of the frames in the cluster. The frames are 24 coefficient MFCCs plus F0. Clustergen offers a number of options for clusters which can be single frames, trajectories, or trajectories with overlap and add. We used the simplest model for our TTS. Synthesis is done by predicting the HMM-state duration, then predicting each frame with the appropriate CART tree. The track of MFCC plus F0 vectors is re-synthesized with the MLSA algorithm [13], as implemented in the HTS system and already implemented within Festival.

8 Evaluation

In order to evaluate we conducted subjective and objective evaluations. As part of subjective evaluation 10 samples were extracted from the database. These

Table 1. MOS Test Results

Sentence Name	Least Score	Best Score	Average Score
1	3	4	3.4
2	3	5	4.2
3	2	4	3.1
4	4	5	4.5
5	1	3	2.2
6	2	4	3.3
7	4	4	4
8	5	5	5
9	3	4	3.5
10	2	5	3.4
Total Average			3.67

samples were played to 10 Sanskrit speakers for obtaining mean opinion scores (MOS), i.e., score between 1 (worst) to 5 (best) [7]. The results were given in the Table 1.

For objective evaluation we have choose MCD. Mel cepstral distortion (MCD) is an objective error measure used to compute cepstral distortion between original and the synthesized MCEPs. Lesser the MCD value the better is synthesized speech. MCD is essentially a Euclidean Distance defined as

$$MCD = \frac{10}{\ln 10} * \sqrt{2 * \sum_{i=1}^{25} (mc_i^t * mc_i^e)^2} \tag{1}$$

where mcti and mcei denote the target and the estimated MCEPs, respectively. MCD is used as an objective evaluation of synthesized speech. Informally it is observed in [11] that an absolute difference of 0.2 in MCD values makes a difference in the perceptual quality of the synthesized signal and typical values for synthesized speech are in the range of 5 to 8.

Table 2. MCD Test Results

	Synthesis with Diphones
MCD	6.474

To compute MCD, we have taken 20 sentences from Sanskrit database and synthesized using diphone as unit.

9 Discussion and Future work

So far, we have discussed the build process of a prototype text-to-speech for Sanskrit. While we have demonstrated the usefulness of this prototype text-to-speech for Sanskrit by conducting listening tests, it should be noted that there

exists several research issues that need to be addressed in building a full-fledged text-to-speech for Sanskrit. The following are a few research issues that need to be addressed in building a complete voice for Sanskrit.

1. Choice of Phoneset: In this work, we have used a simplified phoneset (mostly borrowed from Hindi) to develop this prototype text-to-speech system. A careful acoustic-phonetic study needs to be done to build a phone set for Sanskrit. Also, the relationship between Akshara and the sound is assumed to one-to-one in Sanskrit. However, it is often may not be the case. Typically it is known that the Sanskrit scholars in the Northern part of India drop Schwas at the end. For example, /raama/ is pronounced as /raam/. Hence a carefully analysis need to be done to derive a Akshara (written symbol) to phone (spoken sound) correspondence.
2. Choice of Unit: In this work, we have used diphone as a unit for concatenation. This type of unit is found to be useful for English. It is also well known that syllable is a better unit for Indian languages which are mostly derived from Sanskrit. Hence, it is important to study various levels of units (diphone, syllable, polysyllable) for the case of Sanskrit TTS.
3. Nature of Sanskrit Language: In this work, the prototype TTS is built for spoken on conversational form of Sanskrit. It is known that the Sanskrit comes in various flavors such as Vedic. Poetry prose, classical etc. and hence the nature of the Sanskrit language has to be studied and considered in developing a complete text-to-speech for Sanskrit.
4. Prosody: Prosody is often found to be manifested in intonation (rhythm), stress (energy), and duration patterns. Sanskrit is known for its richness in prosody. Hence, a study on prosodic aspects of Sanskrit TTS has to be conducted to identify suitable acoustic properties that can incorporated in TTS.
5. Incorporation of Sandhi Rules: Sanskrit is also known for Sandhi rules. The effect of these Sandhi rules has to be studied and understood in order to implement these rules in Sanskrit TTS.

Acknowledgments

We would like to thank all the persons who participated in conducting the subjective evaluation.

References

1. Formant Synthesis by Thomas Styger and Eric Keller, Laboratoire danalyse informatique de la parole (LAIP), Université de Lausanne, Switzerland
2. Acharya Project by IIT, Madras. Multilingual computing fo r Literacy and Education, `http://acharya.iitm.ac.in/disabilities/tts.php`
3. Ramani, S., Chandrasekar, R., Anjaneyulu, K.S.R. (eds.): KBCS 1989. LNCS, vol. 444. Springer, Heidelberg (1990)

4. पाणिनीय शिक्षा verse 3 and 4.
5. Sarkar, T., Keri, V., Yuvaraj, S., Prahalad, K.: Building Bengali Voice Using Festival. In: Proceedings of ICLSI 2005, Hyderabad, India (2005)
6. Kishore, S.P., Sangal, R., Srinvas, M.: Building Hindi and Telugu Voices using Festvox. In: Proceedings of International Conference on Natural Language Processing, ICON (2002)
7. Raghavendra, E.V., Desai, S., Yegnanarayana, B., Black, A.W., Prahallad, K.: Global Syllable Set for Building Speech Synthesis in Indian Languages. In: Proceedings of IEEE workshop on Spoken Language Technologies, Goa, India (December 2008)
8. About Sanskrit, http://www.sanskrit.nic.in/ABOUTSANSKRIT1.htm
9. Ager, S.: Omniglot writing systems and languages of the world, http://www.omniglot.com/writing/devanagari.htm
10. Veera Raghavendra, E., Prahallad, K.S.: Database Pruning for Indian Language Unit Selection Synthesizers. In: ICON-2009, Hyderabad, India (December 2009)
11. Black, A.W.: CLUSTERGEN: a statistical parametric synthesizer using trajectory modeling. In: Proceedings of Interspeech, pp. 1762–1765 (2006)
12. Black, A.W., Lenzo, K.: Building voices in the festival speech synthesis system (2000), www.festvox.org/festvox/index.html
13. Tokuda, K., Yoshimura, T., Masuko, T., Kobayashi, T., Kitamura, T.: Speech parameter generation algorithms for HMM-based speech. In: Proceedings of ICASSP, Istanbul, Turkey, pp. 1315–1318 (June 2000)
14. Black, A.W., Bennett, C.L., Blanchard, B.C., Kominek, J., Langner, B., Prahallad, K., Toth, A.R.: CMU Blizzard 2007: A Hybrid Acoustic Unit Selection System from Statistically Predicted Parameters. In: Blizzard Challenge 2007, Bonn, Germany (2007)
15. Jain, H., Kande, V., Desikan, K.: Vani - An India Language Text to speech Synthesizer. IIT, Mumbai
16. Black, A.W., Taylor, P.: Automatically Clustering Similar Units for Unit Selection in Speech Synthesis (1997)