# Dzongkha Text-to-Speech Synthesis

*Uden Sherpa[1], Dechen Chhoden, Dawa Pemo*

*Department of Information and Technology, Bhutan*

## Abstract

This paper describes the Dzongkha Text-to-Speech Synthesis (TTS) prototype developed by Bhutan team in collaboration with the HLT team at NECTEC. The Dzongkha TTS is based on Hidden Markov Model-based Speech Synthesis System (HTS). The report discusses two major steps involved in the process of building the prototype: text-processing and speech synthesis. Our experimental results based on subjective evaluation shows that recorded speech had an average rating of 3.93 as compared to the synthesized speech, which was lower at 3.19.

## 1. Introduction

There are ongoing activities and efforts in the world to improve Natural Language Processing. In Bhutan, Dzongkha is the national language. Recent years have seen increasing effort to develop Dzongkha computing tools and natural language processing (NLP) based applications. The Dzongkha Linux Operating system has already been launched under the first phase of the PAN Localization project. Dzongkha Linux is based on the Debian GNU/Linux and has a complete set of localized applications like Desktop (Gnome), office suite (open office), web browser (Firefox), mail client (Thunderbird), chat application (Gaim) etc. The localization team is also currently working on a Dzongkha Lexicon and Optical Character Recognition for Dzongkha.

Dzongkha TTS system is an outcome of research study conducted for creating NLP applications. The HTS system [1] based on Hidden Markov Model was used as TTS engine for Dzongkha. In that the HTS is trained with Dzongkha to output the resulting synthesized speech. TTS systems are used in applications like reading machines, talking web browser and talking dictionary by people with disabilities. It is also used for education systems like interactive learning systems and spell checkers etc.

It is known that speech signals can be represented as waveforms. Taking a sample of waveform representing a portion with initial consonant, vowel and final consonant, we notice that all vowels are periodic and repeatable. Consonants can be either periodic (voiced) or non periodic (unvoiced). The waveforms are converted from time domain to frequency domain by means of Fourier transformation. These representations are called spectrums. Spectrums can be represented by the Mel-Cepstral Coefficients (MCEP), also known as the spectral parameter. This parameter along with some other components like the excitation parameter log F0 (associated with the emotion related to speech), are used to model speech.

There are two ways for speech synthesis: Unit concatenation method and HMM based. For Dzongkha

[1] *Corresponding author, contact at uden.sherpa@gmail.com*

TTS the later was chosen. The advantage of this HMM based TTS system over unit concatenation is that it produces smooth and stable speech and that the speech can be modified to include various voices just by transforming the HMM parameters. However its disadvantage is that since it is vocoder-based unlike in unit concatenation which uses direct recorded voice, it produces buzz noise (robotic) in the background.

A series of syllables make up a speech signal. And each syllable consists of an arrangement of phonemes or phones. In order to model phones using HMMs we have a 3 state machine to represent that phone. So each phone in a word or syllable is in the order: initial consonant, vowel and final consonant. The sequence of phones in a speech signal are represented as vectors of speech parameters which are modeled as three state machines and are simply concatenated to produce back speech. Therefore for each speech utterance only the HMM with maximum probability is selected.

## 2. Methods

This TTS has two modules, text analyzer and speech synthesizer. The structure of a TTS is given in Figure 1 below. The text analyzer that we have used here only looks up transcriptions in dictionary and generates phoneme sequence for the corresponding text.

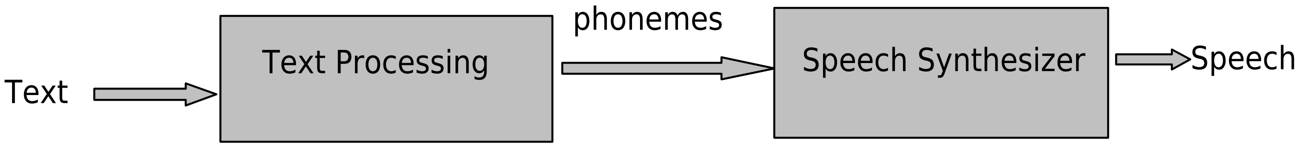


*Figure 1: TTS Structural Diagram*

### *2.1 Design*

### 2.1.1 Sentence Corpus

First step was to collect a sentence corpus. About 40,000 sentences were collected approximately. These sentences were collected from National Digital Library [2] of Bhutan, Dzongkha Dictionary [3] and Dzongkha Development Commission (DDC) website. The sentences would be later divided into syllables to create a syllable dictionary. Since Dzongkha script already has syllable markers, syllables were used as units for text processing. From these sentences a script was run to create a list with unique syllables and their frequency alongside. As a result only around the first 4000 unique syllables were used.

### 2.1.2 Phoneme Set

The written form of Dzongkha has thirty consonants and five vowels. The preferred word order for Dzongkha is subject (s), object (o) and verb (v). The basic unit of morpheme is separated by 'tsheg' (.‾). Each syllable contains a root letter (Ming-Zhi) and may contain any or all of the following parts: prefix, head-letter, subjoined-letter, vowel, suffix and post-fix. There are no inter-word spaces in Dzongkha. Each sentence is ended by a sentence end marker called Shed ( | ).

For instance, to show how it is an SOV language..

ངགིས་ | ཁོ་ལུ་ | སླབ་ཅི།

nga-gi | kho-lu | lab-ci

s o v

"I told him"

Figure 2 shows syllable formation in Dzongkha word.

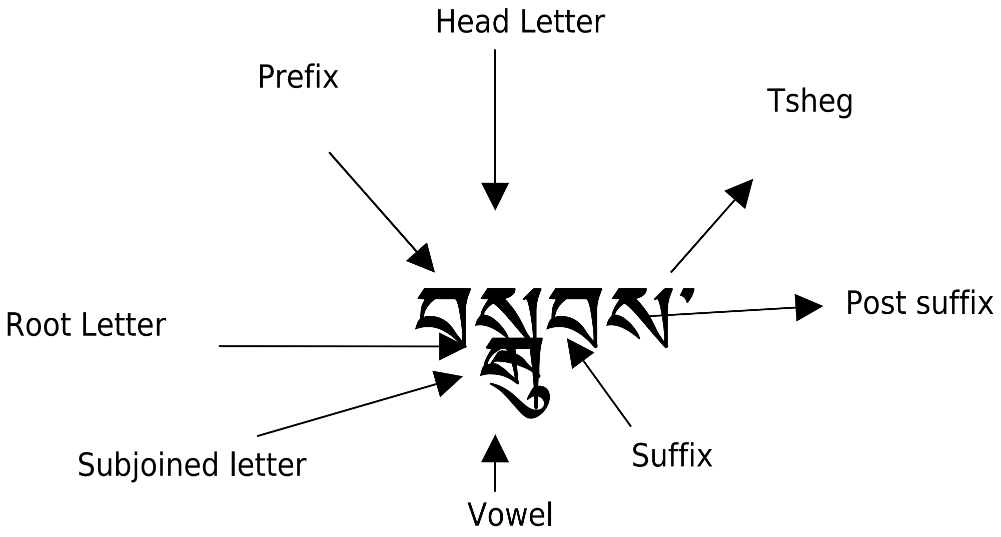


*Figure 2: Syllable Formation*

Next step was to develop a phoneme set for Dzongkha language. It was noted that Dzongkha can be represented by 30 initial consonants, 4 clusters, 10 vowels and 9 diphthongs. These phonemes were looked up in the Dzongkha IPA [4] table that was worked on by DIT with the help from Dr. Sarmad Hussain, NUCES, Pakistan. Also initial consonants, final consonants, clusters and vowels were looked up from the Dzongkha grammar book developed by the DDC.

Table 1 depicts the phonetic table that was used as a basis to form phonemes to represent Dzongkha language. This table is also used in question files to create cluster of similar phones using context trees.

*Table 1: Dzongkha Phonetic Table*

| Place Manner | Labio | | Lab io- | Dental/alve oler | Retroflex | Palatal | Velar | Laryngal/ Glottis |
|---|---|---|---|---|---|---|---|---|
| Stops | Voiceless | P(p) (པ) | | T(t) (ཏ) | Tr (ཀྲ) | | K(k) (ཀ) | A(?) (ཨ) |
| | Aspirated | Ph(p$^h$) | | Th(t$^h$) (ཐ) | Thr (ཁྲ) | | Kh(k$^h$) | |
| | voiced | B(b) (བ) | | D(d) (ད) | Dr (གྲ) | | G(g) | |
| Fricatives | Voiceless | | | Sa(s) (ས) | | Sh(ɕ) (ཤ) | | Ha(h) (ཧ) |
| | Voiced | | | z(z) (ཟ) | | Zh(ʑ) (ཞ) | | 'A(ɦ) (འ) |
| Affricatives | Voiceless | | | Ts(ts) (ཙ) | | C(tɕ) (ཅ) | | |
| | Aspirated | | | Tsh(ts$^h$) | | Ch((tɕ$^h$) | | |
| | Voiced | | | Dz(dz) (ཛ) | | J(dʑ) (ཇ) | | |
| Trill | | | | R(r) (ར) | | | | |
| Lateral | | | | L(l) (ལ) | | | | |
| Approximant | W(w) (ཝ) | | | Y(j) (ཡ) | | | | |
| Nasals | M(m) (མ) | | | N(n) ( ན) | | Ny(ɲ) (ཉ) | Ng(ŋ) (ང) | |

While creating the syllable dictionary with transcription, new diphthongs were encountered, which were important to make proper transcription and to represent at best the spoken form of Dzongkha. So while transcribing, new diphthongs and clusters were added as needed, see Table 2.

*Table 2: Computer representations:*

| Initial consonants | Vowels | Clusters | Diphthongs | Final Consonants |
|---|---|---|---|---|
| P | A | Dr | ai | g |
| Ph | i | Tr | ui | ng |
| B | u | Thr | oi | n |
| T | e | Lhh | au | b |
| Th | o | | ae | m |
| D | ue | | eu | r |
| K | oe | | ou | l |

| | | | | |
|---|---|---|---|---|
| Kh | aa | | iu | p |
| G | ii | | ei | |
| @ | uu | | eo | |
| M | | | | |
| N | | h | | |
| Ny | | | | |
| Ngs | | | | |
| S | | | | |
| Z | | | | |
| Sh | | | | |
| Zh | | | | |
| H | | | | |
| Hh | | | | |
| W | | | | |
| R | | | | |
| L | | | | |
| Y | | | | |
| Ts | | | | |
| Tsh | | | | |
| Dz | | | | |
| C | | | | |
| Ch | | | | |
| J | | | | |

- *Two tones: zero (0) for normal and one (1) for modified one are used.*

  * With some suffixes one vowel is modified to another:

- For example vowel 'a' with n is modified to vowel 'en'.

  Likewise as depicted in the Table 3 below:

*Table 3: Vowel Modification*

| Suffixes | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| vowel | G | ng | n | b | m | r | l | p | d | s | hh |
| a | | | e n | | | | e l | | e | | |
| i | | | l n | | | | i l | | | | |
| e | | | ue n | | | | ue l | | u e | | |
| o | | | | | | | | | | | |
| u | | | en | | | | e l | | e | | |

- *Also keeping in mind that the suffixes d, h and hh are not pronounced when used.*

Here it was decided that two tones would be used, zero (0) for normal speaking and one (1) for modifications in pitch while speaking some words. Most of the time, it was found that tone one (1) is used with prefix, g, m and d. During discussions it was also found that with some suffixes, one vowel is changed to another in spoken Dzongkha. It means that when a word is written with one vowel it may happen that while pronouncing, it is changed to another vowel. This is depicted in the table above. Additionally it was noted that some suffixes in the writing system are not pronounced while speaking.

## 2. 1.3 Transcription and syllable dictionary

After designing the phoneme set, the task for transcribing the 40,000 syllables to create a syllable dictionary began. At the end of this process, syllable dictionary in excel sheet format resulted, consisting of a list of syllables with their corresponding transcription.

## *2. 2.Text Processing*

A simple text processor that takes in a Dzongkha text and generates its equivalent phonemes is used for text processing. Here the syllable dictionary as a corpus was used to look up the equivalent phonemes for the input text.

The text is segmented into syllables and the sentence marker is removed from the sentence. These syllables are then looked up in the syllable dictionary to get its equivalent transcription after which corresponding phoneme sequence for the syllables are generated for each sentence.

Figure 3 below depicts the process of text analysis.

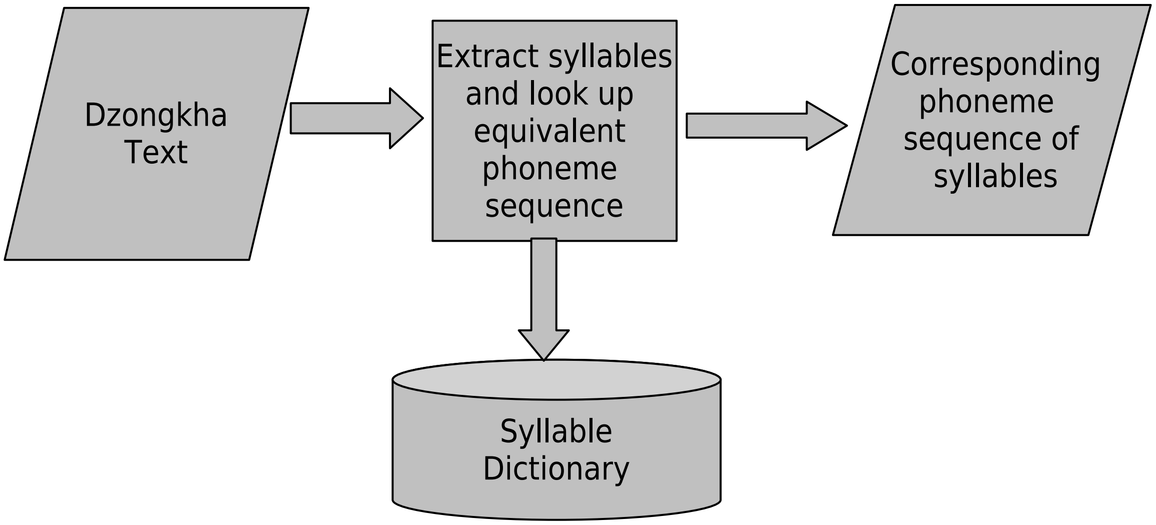

Dzongkha Text
Extract syllables and look up equivalent phoneme sequence
Corresponding phoneme sequence of syllables
Syllable Dictionary

*Figure 3: Overview of Text-Processing*

## *2.3 Speech Synthesis*

### 2.3.1 System Overview

For speech synthesis, there is a speech database from which speech parameters like Mel-Cepstrum (MCEP) and Log Fundamental frequency (F0) are extracted. These parameters are used to train the HMM for generating synthesized speech.

There are two parts or modules to speech synthesis using HMM. The first being the training of the HMMs and the second, the actual generation of synthesized speech.

In the training part, a speech database of wave files was formed. From there label files were created. These are files with phones, their context and duration. Speech signals are passed into the HTS along with labels. In the process, excitation parameter (LF0) and spectral parameter (MCEP) are generated from statistics of data and HMMs are trained with this data. After the training process we have a database of trained context-dependent HMMs and duration modules.

In the synthesis part, text is passed on to text analyzer which generates phonemes and full label files. These files are then called with HTS-engine where spectral parameter (MCEP) and excitation parameter (LF0) are extracted from trained HMMs and passed onto MLSA synthesis filter to produce synthesized speech.

Figure 4 depicts the process of speech synthesis system:

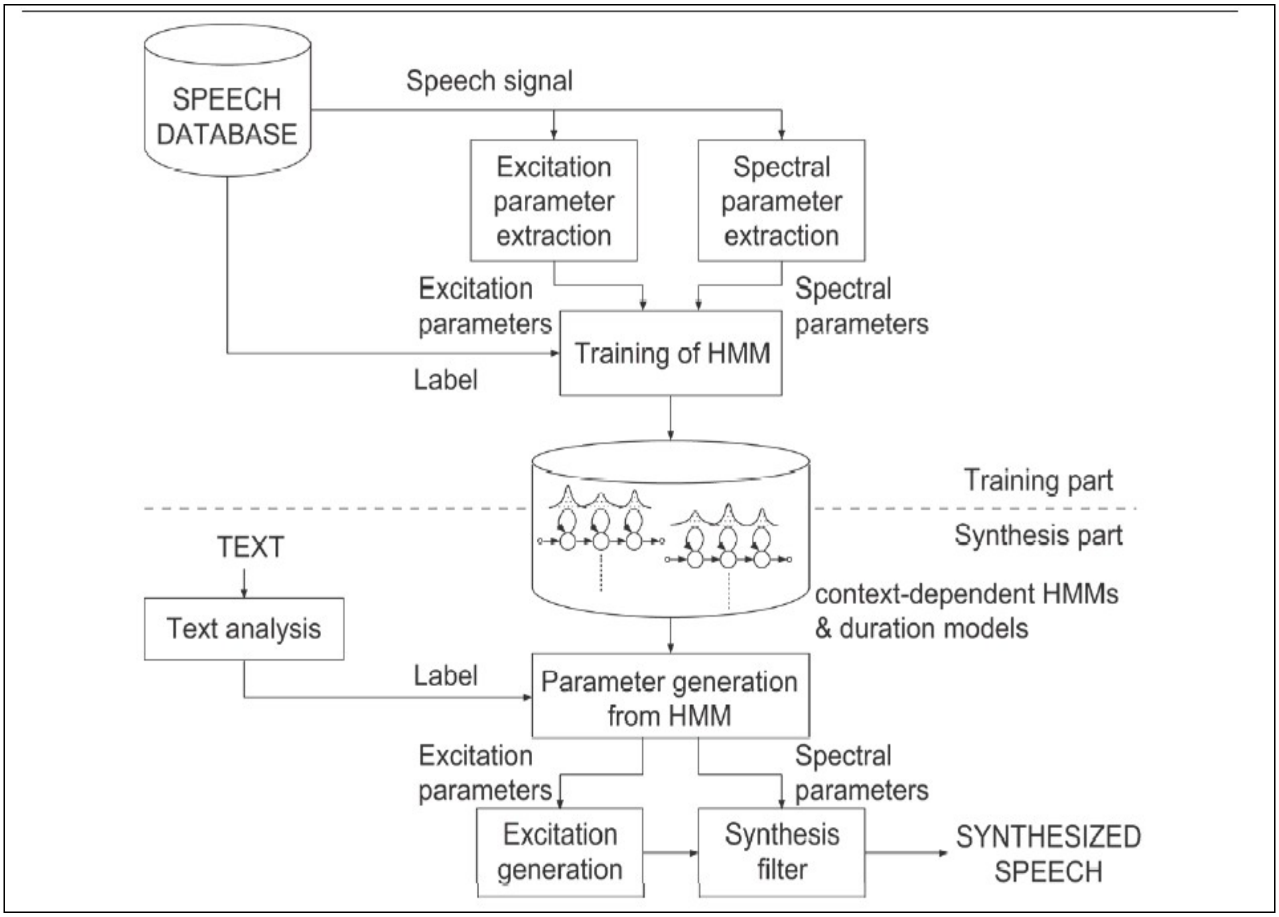


*Figure 4: Speech Synthesis System Overview*

## *2.4 Step by step workings of HTS training*

### 2.4.1 Map back to sentence from syllables

The syllables are mapped back to create sentences. For that purpose a script was run in which the original 40,0000 sentences were fed along with the syllable dictionary, to generate a list of 23,000 sentences along with their corresponding transcriptions.

### 2.4.2 Create label files and wav files for training the HTS

For training the HTS some files need to be fed into the HTS engine. From the above generated 23,000 sentences, a corpus of .syl files were created. It is a collection of 23,000 files, each file containing transcription of each sentence. Also a file containing all of the .syll files concatenated with the filenames following the transcription was created. For example, 1.txt is the file name for the first transcribed sentence, 2.txt is the file name for the second transcribed sentence and so on till 23000.txt.

Figure 5 depicts the process for creating label files:

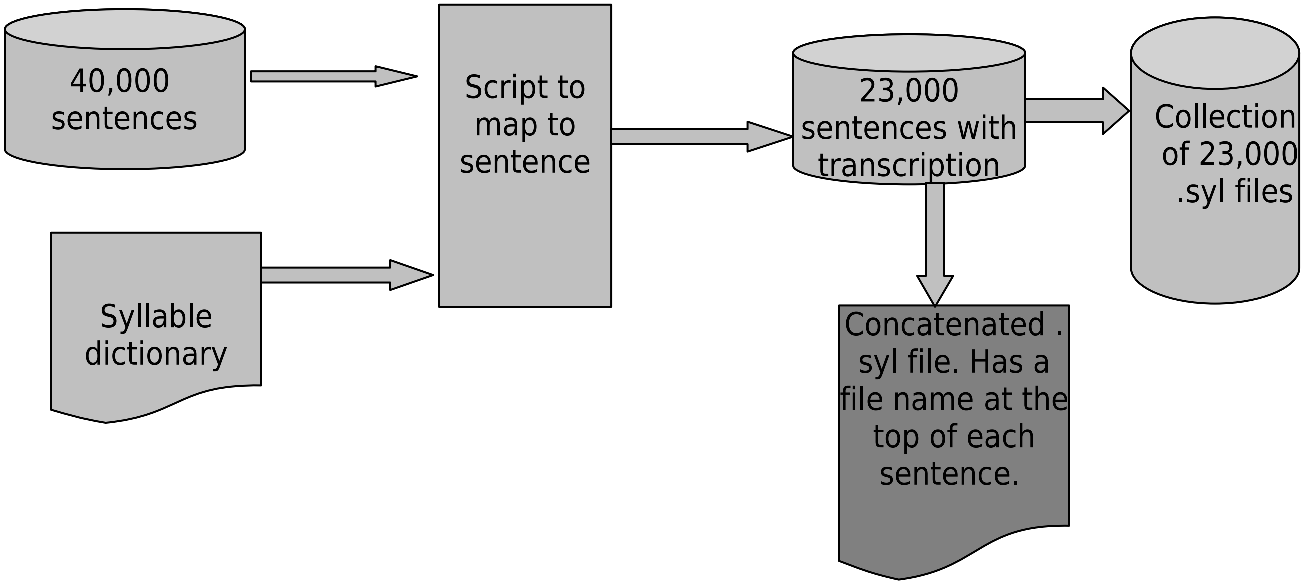

40,000 sentences
Syllable dictionary
Script to map to sentence
23,000 sentences with transcription
Collection of 23,000 .syl files
Concatenated . syl file. Has a file name at the top of each sentence.

*Figure 5: Creating label files*

Now from the collection of .syl files only 509 essential sentences were generated for recording. These 509 sentences contain all the phones that make up the Dzongkha language. These sentences were generated by passing the earlier 20,000 sentences into NECTEC's sentence selector. This tool considers diphones while generating the sentences.

Then these 509 sentences were recorded. The resulting wave files are edited to a 44 Khz and 16 bits format. At the end of these procedures we have 509 wav files named 001.wav, 002.wav and so on till 509.wav. The wav files are needed to feed into the HTS for training.

They make up the speech database that will be used for training process. So now our speech database has 509 wav files. Next we need to create labels.

Before the files like mono.dic, word.mlf and mono.mlf were required.

Mono.dic is of the format:

```
!ENTER      $0
!EXIT $0
(དང)    d0 a0 ng0
```

(ལུ)    l0 u0

(མི)    m0 i0    ..................................

Word.mlf is of the format:

```
#!MLF!#
"*/001.lab"
!ENTER
(མི)
(འཕེལ)
(འཕེལཕ)
(ནོར)
(སེམས)
(ཅན)
(འཕེལ)
(འཕེལཕ)
!EXIT
.
```

The file mono.dic was generated from the syllable dictionary and word.mlf from the 509 sentences. Hence mono.dic would contain all 4000 syllables along with their corresponding transcription and tone.

Word.mlf would contain all 509 sentences with their filenames, syllables of each sentence bounded within a bracket.

A tool called HLEd was used, which is an HTK tool to generate mono.mlf from the two files (mono.dic and word.mlf).

Mono.mlf is in the format:

```
#!MLF!#
"\*/1.lab"
$
m0
i0
x0
ph0
e0
l0
ph0
e0
```

```
w0
n0
o0
r0
s0
e0
m0
c0
e0
n0
ph0
e0
l0
ph0
e0
w0
$
.
```

Then these wav files along with .mlf and .dic files were passed into HTK to generate label (.lab) files which is in the format of time start, time end and phone.

After that there were question files, which are files that have phones with their context. We use flat start to do labelling process which generates full and mono label files.

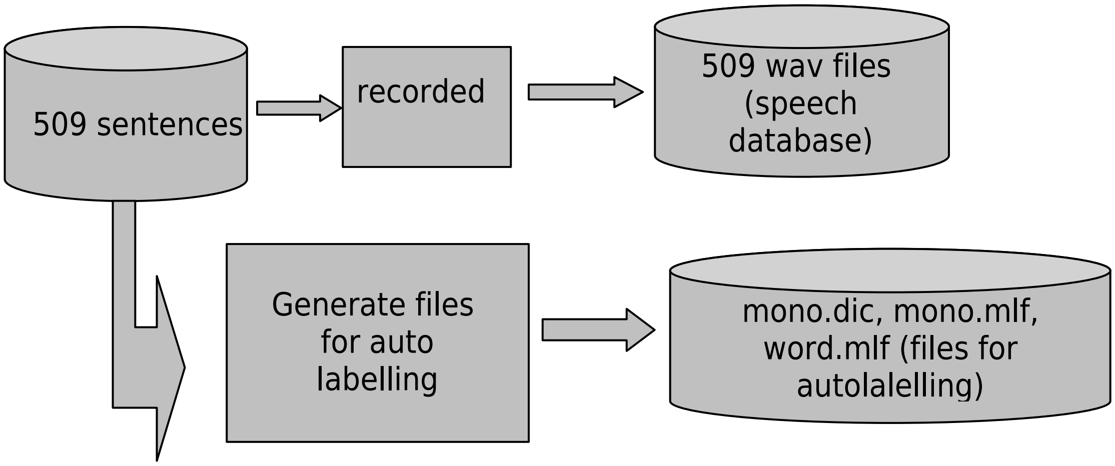

509 sentences
recorded
509 wav files (speech database)
Generate files for auto labelling
mono.dic, mono.mlf, word.mlf (files for autolalelling)

*Figure 6: Labeling files*

### 2.4.3 Train HTS

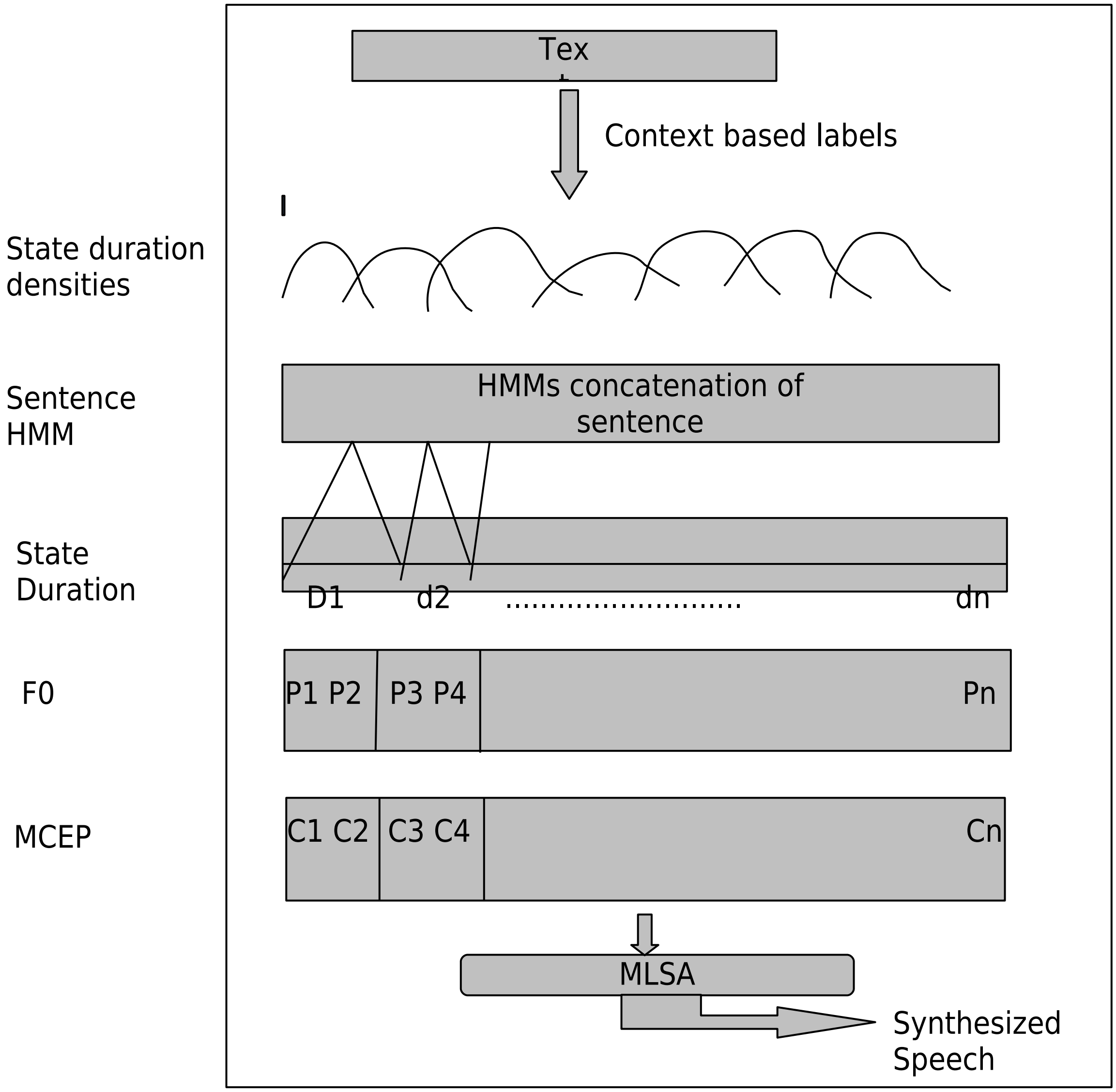
By referring to the system overview diagram we now have speech database and labelling is also done. So in the training process, MCEP and LF0 parameters are extracted from this data and trained HMMs are generated.

Therefore the end of the training, results in a collection of trained HMMs which are used to generate speech parameters (MCEP, LF0 and duration). These are passed onto the MLSA filter to generate synthesized speech.

### 2.4.4 Speech synthesis module

Now to elaborate on HMM-Based speech synthesis, there is a collection of trained HMMs. When text is passed to text analyzer, label sequences are passed to the HTS-engine. Speech parameters are generated from the HMM and the best HMMs with the highest probability are selected and concatenated to form sentence HMM from the syllable HMMs. These are then passed to MLSA filter to finally generate the speech.

Figure 7 below depicts the process.



After training the HTS, text can be passed into the text processor to generate phoneme sequence. This phoneme sequences are then passed into the speech synthesizer to generate its equivalent speech.

## 3. Evaluation and Result

An evaluation form was prepared to allow people to rate the synthesized speech keeping the original recorded speech as the basis for comparison in the evaluation. It had a 1 to 5 rating, 1 being the worst and 5 being the best. After evaluation, recorded speech had an average rating of 3.93 as compared to the synthesized speech, which was lower at 3.19.

## 4. Conclusion

The Dzongkha TTS prototype has been developed using HTS based on HMM. Then it was evaluated based on mean opinion score using subjective evaluation method. The mean score of synthesized speech is rated at 3.19 as compared to recorded speech with mean opinion core at 3.93. The synthesized speech output is somewhat flat, robotic and sounds like it was spoken by a non native person.

The synthesized speech quality may be improved by using words and phrase detection algorithm, and integrating more prosodic features.

## 5. References

[1] http://hts.sp.nitech.ac.jp, HTS-based Speech Synthesis.

[2] www.library.gov.bt, National Digital Library, DIT, Bhutan

[3] Dzongkha-Dzongkha Dictionary, Dzongkha Development Commission, Bhutan, 2005.

[4] Dzongkha IPA Table, DIT, Bhutan, Dr. Sarmad, NUCES, Pakistan.

[5] Keiichi Tokuda, “HMM Based approach to flexible speech synthesis”, Nagoya Institute of technology, Japan.

## Acknowledgement

This research study has been supported by the PAN Localization Project (www.panl10n.net) grant from the International Development Research Center (IDRC), Ottawa, Canada, administered through Center for Research in Urdu Language Processing, National University of Computer and Emerging Sciences (NUCES), Pakistan.

We would like to thank the Human Language Technology (HLT) of National Electronics and Computer Technology Centre (NECTEC), Thailand, particularly Dr. Chai Wutiwiwatchai and his team members for their technical guidance and for supporting us to design and develop the Dzongkha TTS.