*International Journal of Computer Applications (0975 – 8887)*
*Volume 86 – No 8, January 2014*

# A Technical Guide to Concatenative Speech Synthesis for Hindi using Festival

Somnath Roy
Jawaharlal Nehru University
319, Jhelum Hostel
JNU, New Delhi-110067

## ABSTRACT
Speech is the most natural and effective medium of communication among the human beings. Speech has played a great role in the evolution of human civilization. Speech synthesis is an artificial way of producing speech. The native speakers of any language use their knowledge base of various prosodic features during the speech production. These features they acquire unconsciously in childhood. With the help of these features, they are capable of expressing the meaning of any utterance and emotional states. It is still a challenging task to bring similar naturalness in artificial speech production (speech synthesis).This paper covers the details of how to develop a speech synthesizer using Festival tool. Two approaches have been discussed in details: Limited domain synthesis technique and Unit selection synthesis for Hindi. Apart from that how to configure Festival tool on Linux so that one can start working for a TTS also has been discussed. The purpose this paper is to give the full insights of technicalities involved while manipulating festival for a new language to the naive speech researchers.

**General Terms**
Speech Synthesis, Unit selection method, Grapheme to Phoneme, Formant

## Keywords
Speech synthesis; Prosody; Speech production; Linguistic module; Phonemic; Phonetic; F0; Pitch, Festival

## 1. INTRODUCTION
Festival[1] is a framework for developing speech synthesis systems. Festival is a language independent platform and voices can be developed for many languages. Voice for many languages has been developed including English, Welsh and Spanish. The system has been developed by using C++ and has a Scheme (SIOD) based command interpreter for control. Festival has been designed in such a way that itcan work as a whole system in which one may simply work on his small part to improve the whole. Without a complete system like Festival, to test a new module one need to spend significant effort to build a whole system, or adapt an existing one. Festival allows the addition of new modules, easily and efficiently. Festival is a simple environment for researching new synthesis techniques. It is a completely usable text-to-speech system suitable for embedding in other projects that require speech output. For developing a speech synthesizer for any arbitrary language using Festival, it is necessary to have the knowledge of phoneset, rules of syllabification and stress marking in that language. These are the prerequisite that need to be modified in the existing Festival environment for developing new voice. Speech synthesis technique and types of Speech synthesis has been described with their advantage and disadvantages below.

A speech synthesizer is an engine which takes input as a sequence of words (strings) and converts it into speech that resembles as close as native speaker reading that text. A TTS generally contains two modules: Text Analysis/Linguistic Analysis and Digital Signal Processing. The Linguistic Analysis module takes set of strings (words) as an input and gives a normalized phonetic sentence. These phonetic sentences are the input for DSP module [5][9] which is responsible for generating the corresponding possible natural speech. Speech Synthesizer can be used for various purposes like: i) can be used by visually impaired ii) can be used by vocally impaired iii) in language pedagogy iv) talking books and talking toys etc. Current area of research is speech prosody for all languages, which is essential in both speech synthesis and speech recognition. Synthesized speech should contain prosodic cues for clear perception of words and the construction of meaning of the utterance for listeners.

There are two main types of speech synthesis methods: 1. Rule Based 2. Corpus Based

In Rule Based method no pre-recorded speech sound is required because each sound is evaluated by specific set of parameters. There exist two main Rule based techniques: Formant Synthesis and Articulatory Synthesis.

### 1.1 Formant Synthesis
Formant Synthesis gives a set of rules which describes how to modify pitch, formant frequencies and other parameters from one sound to other [2][4][10][19]. These rules are based on source-filter model of speech production [5]. Klatt 1980, described formant synthesis model very clearly, which later evolved as MITalk[20]. By modifying the filter parameters i.e. formants, one can bring prosody in formant synthesis. “The observed irregularities in the spectrum between the formant peaks are of little perceptual importance; only the strong harmonics near a formant peak and below F1 must be synthesized with the correct amplitude in order to mimic an utterance with a high degree of perceptual fidelity” [21].

### 1.2 Articulatory Synthesis
Speech synthesis is based on mechanical and acoustic models of speech production. It used to model the physiological effects, such as the movement of the lips, tongue, jaw, and the dynamics of the vocal tract and glottis [15][16][17] . This method of synthesis is still in its infantry stage, hence no need to take care of prosody. It is one of the most complex methods because it very difficult to model the dynamics involved in the physiological speech production.

### 1.3 Concatenative Synthesis
Concatenative Synthesis is a type of Corpus based synthesis technique. Commercially, concatenative synthesis [3] is most popular and commonly used. It got reason for being popular: storage devices became so cheap that storing pre-recorded wave file for processing is no more any costly affair. In this method, utterance is synthesized by concatenating several natural speech segments. Speech database is created

by storing the speech samples in form of sentences, intonational phrases, phonological words, syllables, diphone or phoneme. If all segments are of same length then it is called fixed inventory otherwise unit selection (variable length segments are stored and system makes decision to the best match). Naturalness in concatenative synthesis is then increased by using PSOLA (Pitch Synchronous Overlap Add) algorithm.

### 1.4 Statistical Parametric Synthesis

HMM based approach of synthesis is the best example of statistical parametric synthesis. In this approach from the speech database it extracts a set of parameters. By statistical analysis of these extracted parameters, parameters are clustered according to their prosodic features and context. According to [7], the contrast between concatenative synthesis and statistical parametric synthesis is that latter might be mostly described as generating average of some sets of similarly sounding speech segments while the former used to concatenate the original most similar segment of speech to the target.

## 2. INSTALLING FESTIVAL

I have installed in Fedora 14, the following versions: Festival-1.9.6, Speech_tools-1.2.9.6, Festvox-2.0. Although, the latest version of festival and speech_tools is available on the following link:

http://www.speech.cs.cmu.edu/awb/fftest/speech_tool s-1.2.96-beta.tar.gz

http://www.speech.cs.cmu.edu/awb/fftest/festival-1.96-beta.tar.gz

Different voices for English are available and can be installed. These voices can be found on the link shown above in box in red color. Here is the following list of voices:

festvoxcallpc16k.tar.gz
festlex_POSLEX.tar.gz
festlex_CMU.tar.gz
festvox_cmu_us_slt_arctic_hts_tar.gz
festvox_cmu_us_jmk_arctic_hts_tar.gz
festvox_cmu_us_bwl_arctic_hts_tar.gz
festvox_cmu_us_awb_arctic_hts_tar.gz

### 2.1 Working Steps

Extract festival, speech_tools, and festvox to your home directory.Extract all tar file by using the command: For example type on the terminal following command:

```
$ tar –zxvf followed by [file name]

$cd speech_tools

./configure

$make

$cd ../festival

 ../configure

$make and then $make install
```

If everything works properly then we have to testfestival, whether it is working properly or not.$festival/bin/festival

Or

```
$ cd festival/bin

$festival

festival>(SayText “Hello World”).
```

We presume here the system will produce sound. More information is available in festival document.

## 3. LIMITED DOMAIN SYNTHESIS

Here we assume festival works perfectly. Extract festvox by

```
$tar –zxvf festvox-2.0-release.tar

$cd festvox

$make
```

Now, one needs to set the environment variable using following command:

```
export FESTVOXDIR=$HOME/festvox export ESTDIR=$HOME/speech_tools
FESTIVAL=$HOME/festival/bin/festival
exportLD_LIBRARY_PATH=$HOME/speech_tools/lib:$L
D_LIBRARY_PATH

$mkdir project1

$cd project1

$FESTVOXDIR/src/ldom/setup_ldom cdac hindi som
```

Here user can write any arbitrary name of the voice being built up. Hence, the above name signifies: cdac-organization name, hindi- language name, som(somnath)-speaker name or developer name. After executing the above command we will get the whole branch of directory.

The next step is to make a file with extension as “.data” for example say (hindinew_out.data) with the following format and put it in project1/etc/ directory:

```
( P001 "kyA Apa Kuxa ko kama karake AMka sakawe hEM ") ( P002 "merI bahana unake haswAkRara lene ke lie badZI uwsuka WI")

( P003 "agara vidiyo Ora AVdiyo vebasAita para upalabXa ho wo ACA rahegA ")
```

The number of sentences depends upon the choice of user. The above sentences in double quotes are in wx notation, because Festival is not compatible with Unicode characters. The format is "(" followed by a filename, root followed by the text for that sentence, followed by ")" each on separate lines. This text when converted to a phone sequence by Festival should match (as closely as possible) the phone sequence of the speech. With this in mind we should probably ensure all words are in your lexicon (if we are using one) and it is probably best to write numbers and dates out in full as they were spoken. (e.g "the ninth of May" rather than "9 May" etc...).

Next step is to make the utts.data file in project1/etc/ directory in following format:

```
(P001 P001 )

(P00  P002 )

(P003 P003 )
```

And then run this script:

```
$FESTIVAL -b      festvox/build_ldom.scm
'(build_prompts"etc/hindinew_out.data")
```

The next command (prompt_them) is used for voice recording. We can record our voice in linux system by the following command. Advantage is that we do not have to label of our voice. But we have to careful about our voice recording. Recording voice will be automatically saved in /home/demo/wav directory. After recording of our voice, we have to test that, our recording voice and hindinew_out.data sentences matches perfectly.

```
$bin/prompt_them etc/utts.data
```

All the instructions given above are important, read carefully all of them and go accordingly. If one will try to escape any step or if one will try to ignore the sequence of execution error will occur. While doing these things for first time you might see some errors but don’t quit because these errors could be due to some of the dependency errors. Although in this paper, I have explained all the minute details related to error information which might occur during synthesis process, but this cannot guarantee that error won’t occur.

Now, Execute the following command sequentially.

```
$bin/make_labs prompt-wav/*.wav
```

After this step replace the wave file generated in prompt-wave directory by the recorded wave files for the sentences in hindinew_out.data.

```
$FESTIVAL -b festvox/build_ldom.scm '(build_utts

"etc/hindinew_out.data")'

$bin/make_pm_wave wav/*.wav

$bin/make_pm_fix pm/*.pm

$bin/simple_powernormalize wav/*.wav

$bin/make_mcep wav/*.wav

$FESTIVAL -b festvox/build_ldom.scm '(build_clunits

"etc/hindinew_ut.data")'

$FESTIVAL festvox/cdac_hindi_som_ldom.scm

'(voice_cdac_hindi_som_ldom)'
```

Now your voice has been loaded and you can write on the prompt as shown below to check the voice.

```
festival> (SayText " kyA Apa Kuxa ko kama karake AMka sakawe hEM.")
```

This techniques has certain limitation written below.

a. In larger context it is not better.

b. We can’t add our lexicon, phonesets, and LTS rule.

c. If segment mismatch occur it generate speech by its default voice.

## 4. UNIT SELECTION SYNTHESIS

Unit selection synthesis is a type of concatenative synthesis in which the largest matching sound file available in the speech corpus is concatenated for synthesis of target speech. It is capable of managing large number of units, also imparts prosody beyond the role of F0. It is quite necessary to make a clear distinction between role of F0 and Pitch: F0 is the actual frequency generated by the vocal cord or vocal fold, while Pitch is the perception of that frequency by the listener. Hence it not necessary that both are equal.This synthesis technique also retains the naturalness in the speech sounds being generated. Two types of costs are calculated for synthesis purpose: i. Target cost,$T(u_i,s_t)$ ii. and Join cost, $J(u_t,u_t+1)$[4][8]. Target cost is the distance between the specification $s_t$ and the unit $u_i$ in the database, while Join cost is measure of how well fit two units. The total combined cost for a sentence is given by $C(U,S) =\sum_{i=1}^{N} T(u_i+ s_t)+\sum_{i=1}^{N-1} J(u_t+u_{t+1})$. The goal of the search is to find the single sequence of units $U^*=\text{argmin } u\{\sum_{i=1}^{N} T(u_i+ s_t)+\sum_{i=1}^{N-1}J(u_t+u_{t+1})\}$. For Hindi speech synthesis, one needs to go through the details shown below in step by step fashion. Also for clear understanding where to go and what changes to do figures has been given.

```
$mkdir unitsel
```

After executing the above script a directory naming unitsel will be generated in home directory.

```
cd unitsel
```

Now enter the directory unitsel by writing the above script.

```
$FESTVOXDIR/src/unitsel/setup_clunits hindi acad som
```

hindi referring to the language name, acad- stands for academic, som(somnath)- speaker name. After executing the above command we will get the whole branch of directory. Now we need to put all sentences in wx-form in “uniphone.data” file in unitsel/etc directory in the same form as shown below.

( Part-111 "apne tikat ko tikat kalektar ko xiKAxo" )

( Part-112 "SrIlafkAI rARtrapawi ne Awafkvax ke KZilAPZ kadZe kZaxam uTAne kA vAxA kiyA" )

( Part-113 "PZijUl KarchI par niyanwraN karnA AsAn hojAegA" )

The more data will be taken as the training data for alignment purposes with the sound file that much accuracy can be granted. The reason is very simple, is that with large number of sound file festival can find larger segments while generating the speech. Now, open the emu speech tool and click on load database and give the path of wav files and select all recorded wave files for manual labeling.Then click on the template operation and after that click on the Labfiles. The type of it should be SEGMENT then select the path where lab files are stored and its extension should be .esps [18]. Now click on each lab file and do the labelling by hearing the segments. Labelling here is meant for marking the boundary (start and end) of each segment (phoneme). Click on any lab file to open it for labelling by hearing the segments. Segments here for me is any consonant or vowel. After labelling do the following steps.

i. Put all the wav files in unitsel/wav and unitsel/prompt-wav directory. Also, put all the lab files in unitsel/lab and unitsel/prompt-lab directory.

ii. Create a file hindi_acad_lex.out file, because in the festvox/hindi_acad_som_lexicon.scm file there is a dependency for this file without compiling it we cannot proceed further. This file (hindi_acad_lex.out) contains the lexemes of the sentences with their stress pattern at the syllable level. For example pattern is given below:

("karake" V (((ka) 1) ((ra) 0) ((ke)

1)))

("lene" V (((le) 0) ((ne) 1)))

The syntax for making the above file is the following: (“word” pos ((syl1) stress mark) ((syl2) stress mark))) 0 is for lower stress and 1 is for the high stress.

Finally, do the following changes in file hindi_acad_som_lexicon.scm in Lexicon definition part to resolve the dependencies. This is one of most difficult thing that I found during my work. The last two lines have been added to compile the lex.out file.

(lex.create "hindi_acad") (lex.set.phoneset "hindi_acad_som") (lex.set.lts.method'hi_cdac_lts_function)

(if(probe_file(path-appendhindi_acad_som::dir "festvox/hi_cdac_lex.out")) (lex.compile"festvox/hindi_acad_lex.out")

(lex.set.compile.file(path-append hindi_acad_som::dir"festvox/hindi_acad_lex.out")))

In the last step, Insert these phones in the same format in festvox/hindi_acad_som_phoneset.scm file. To know about value written after ‘a’ see the documentation of phoneset.scm file. These values are features related to sound production of this phone. For ex:

(a + s 3 3 - 0 0 0 ) ;hindi letter अ

Now run the following scripts in sequential fashion:

./make_utts festival/relations/Segment/*.Segment

After executing the above script, you will get the .Segment files in the relations/Segment directory. An extra file “sil.Segment” you have to put by yourself, because this silence file is not automatically generated. After this execute the following scripts:.

/bin/make_pm_wave wav/*.wav

./bin/make_mcep wav/*.wav

This script generates the files of mel cepstrum in the mcep directory.

festival –b festvox/build_clunits.scm ‘(build_clunits

“etc/uniphone.data”)’

After running the above scripts a file in unitsel/festival/hindi_acad_som_clunits is created. This file contains each phoneme in indexed form. Also it contains other required details of all the phonemic units available in lexicon as well as their alignment with corresponding acoustic part. For example:

```
EST_File index DataType ascii NumEntries1561

IndexName hindi_acad_som

EST_Header_End

H#_37 sil 0.000000 0.222927 0.445854

Q_46 sil 0.445854 0.498544 0.551234

sil_32 sil 0.551234 1.401792 2.252350

H#_36 Part-126 0.000000 0.009144 0.018289

Q_45 Part-126 0.018289 0.227270 0.436250

m_47 Part-126 0.436250 0.448125 0.46000
```

## 5. CONCLUSION

Limited domain synthesis is used when we are sure that only some set of sounds need to be produced artificially. It is not used when we want a generalized output in form of speech. But Unit Selection Synthesis is good for producing sound for any generalized text as an input. In my case I have experimented only on a speech corpus of containing training database of sound files of one hour duration recorded by professional speakers. 10 Delhi Hindi speakers were asked for perception test. We instructed them to give 4- very good, 3-good, 2-average and 1-bad for naturalness and intelligibility both. Analysis shows that the intelligibility of speech is more than 98% but naturalness is very low around 40%. Naturalness could be improved if there had any pre-processor which is capable of generating automatically the labeled input file ( a text file with prosodic labeling) at training time; these labeled text file (prosodic labeling) can easily be aligned with sound file in festival and the prosody of speech (synthesized) could be enhanced.

## 6. ACKNOWLEDGEMENT

This paper is a revised part of my M.Tech (CSE)dissertation C-DAC, Noida in the year of 2010-12. I am very thankful to Prof. PKS Pandey (JNU) for valuable guidance and inspiration. Also his suggestion inspired me to revise my work. I am also thankful to my colleague Mr. Dhirendra Kumar (JNU) for the discussion related to linguistic essence in TTS.

## 7. REFERENCES

[1] Black & Lenzo. (2003). Building synthetic voices.http://festvox.org/bsv/.

[2] Anderson, M., Pierrehumbert, J. & Liberman, M.Y. (1984). Synthesis by rule of English intonation patterns. *IEEE Congress on Acoustics, Speech, and Signal Processing*: pp 77-80.

[3] Ashwin Bellur, K Badri Narayan, Raghava Krishnan K, Hema A Murthy. Prosody Modeling for Syllable-Based Concatenative Speech Synthesis of Hindi and Tamil. *IEEE* 2011.

[4] Anumanchipalli K. Gopala, Cheng Ying-Chang, Fernandez Joseph, Huang Xiaohan, Mao Qi, Black W. Alan, (n.d.). Klattstat: Knowledge based parametric speech synthesis.

[5] Atal B. S and Hanauer Suzanne L. (1971). Speech analysis and synthesis by linear prediction of the speech wave. *The journal of acoustic society of America*: pp 637-655.

[6] Bagshaw Christopher Paul. (1994). Automatic prosodic analysis for computer aided pronunciation teaching, Ph.d thesis.

[7] Black & Kominek. (2009). Optimizing Segment Label Boundaries forStatistical Speech Synthesis. *IEEE*: pp. 3785-3788.

[8] Black W Alan, Hunt J Andrew. (1996). Unit Selection Synthesis in a Concatenative Speech Synthesis Using a Large Speech Database. IEEE: pp.373-376.

[9] Buric M. R, Kohut J & Olive J. P. (1981). Digital Signal Processor: Speech Synthesis, The Bell system

technical journal, vol. 60. pp.1621-1631.

[10] Buza Ovidiu, Gavril Toderean, Jozsef Domokos . (2010). A rule based approach to build a text to speech system for Romanian, in proceedings of international Conference on communications. pp. 33-36.

[11] Chomsky & Halle. (1968). The sound pattern of English. New York: Harper & Row Publishers.

[12] Cutler, A., Dahan D.& Donselaar. (1997),Prosody in the comprehension of spoken language: A literature review, Language and Speech,141-201.

[13] Dutoit Theirry. (1993). High quality text-to-speech synthesis of the french language; Ph.d thesis.

[14] Fabio Tamburini. (2003). Automatic prosodic prominence detection in speech usingacoustic Features: an unsupervised system. Eurospeech 2003: pp.129-132.

[15] Fujisaki, Hirose & Kawai (1986). Generation of prosodic symbols for rule synthesis of connected speech of japenese. *IEEE*: pp. 2415-2418.

[16] Fujisaki,Ohno. (2002) A preliminary study on the modelling of fundamental frequency of Thai utterances. *IEEE*: pp. 516-519.

[17] Fujisaki, Ljungqvisi, Murata(1993). Analysis and modelling of word accent and sentence intonation in Swedish. *IEEE:* pp.211-214.

[18] Harrington Jonathan (2010). *The Phonetic Analysis of Speech Corpora*. Delhi: Jhon & Wiley.

[19] Huang X., Acero A., Hon H., (2001). Spoken Language Processing.Prentice-Hall, PTR, Upper Saddle River, NJ.

[20] Jonathan Allen, M Sharron Hunnicutt & Klaat Dennis. (1987). From Text to Speech: The MITalk System.

[21] Klatt H. Dennis. (1987). Review of text-to-speech conversion for English, *JASA, vol.82(3):* pp.737-793.