2. LITERATURE SURVEY

2.1 Speech Synthesis

Speech synthesis has progressed remarkably in recent years, and it is no longer the case that state-of-the-art systems utter sounds like mechanical and robotic. It is normally fair easy to tell that, it is a computer talking rather than a human and so substantial progress is still to be made. When assessing a computer’s ability to speak, one fluctuates in mind i.e. talking computer is like a dog’s walking on his hind legs [1]. It is not done successfully well; but it is surprised it perform on many of the extent more than expectation. An experience text-to-speech researcher has made substantial efforts on TTS for international languages. And hence now days somebody genuinely surprised and thrilled in a naive way that having talking computer: “like wow! it talks!”, more naturally and intelligent. On the other hand it is also possible to have the impression that TTS are quite dreadful at the job of speaking; they make frequent mistakes, drone on, and just sound plain wrong in many cases. All these impressions are a part of the mysteries and complexities of speech.

2.1.1 History of Speech synthesis

Human fascination with talking machines is not new. For centuries, people have tried to empower machines with the ability to speak; prior to the machine age humans even hoped to create speech for inanimate objects. The early men attempted to show that their idols could speak, usually by hiding a person behind the figure or channeling voices through air tubes. Producing artificial speech with mechanical devices, (i.e., talking machines) has been the goal of scientists for centuries. The first attempts to produce human speech by machine were made in the 18th century [45]. The first talking machines were mechanical speech devices that imitated the human vocal tract. A brief history of these machines is given below.

1) In 1773 Christian Kratzenstein in St. Petersburg explained the physiological differences between five long vowels (/a/, /e/, /i/, /o/, and /u/) by an apparatus that could produce them artificially. A set of acoustic resonator tubes connected to organ pipes imitated the human vocal tract.
2) Wolfgang von Kempelen (the first experimental phonetician), wrote *Mechanismus der menschlichen Sprache nebst Beschreibung einer sprechenden Maschine* that appeared in 1791 in Vienna. The 1791 book had a detailed description of a talking machine shown in Fig. 2.1 (a). Kempelen’s studies led to the theory that the vocal tract, a cavity between the vocal cords and the lips, is the main site of acoustic articulation. The essential parts of the machine were a pressure chamber for the lungs, a vibrating reed to act as vocal cords, and a leather tube for the vocal tract action. The machine was hand operable and could produce not only single sounds but also whole words (and even short phrases).
3) **Electrical synthesizers:** The first device to be considered as a speech synthesizer was an electric synthesizer called VODER (Voice Operating Demonstrator) which was presented by Homer Dudley at the World Fair in New York in 1939. It had a voicing/noise source with a foot pedal for fundamental frequency control. Signals, in this synthesizer, routed through ten band pass filters. The device was played like a musical instrument (Fig. 2.2).
4) **Formant synthesizers**: The first formant synthesizer, PAT (Parametric Artificial Talker), was introduced by Walter Lawrence in 1953 [55]. PAT consisted of three electronic formant resonators connected in parallel. The input signal was either a buzz or noise.

5) At about the same time when PAT was introduced, Gunnar Fant introduced the first cascade formant synthesizer, OVE (Orator Verbis Electric) which consisted of formant resonators connected in cascade.

6) **Articulatory synthesizer**: First articulatory synthesizer was introduced in 1958 by George Rosen at the Massachusetts Institute of Technology [55]. The DAVO (Dynamic Analog of the Vocal tract) was controlled by tape recording of control signals created by hand.

7) The first full TTS synthesizer for English was developed in the Electro technical Laboratory, Japan in 1968 by Noriko Umeda and his companions [55]. It was based on an articulatory model and included a syntactic analysis module with sophisticated heuristics. The speech was quite intelligible but monotonous and far away from the quality of present synthesizers.

8) The first reading aid with optical scanner was introduced by Kurzweil in 1976. The Kurzweil Reading Machines for the Blind were capable to read quite well the multi font written text. However, the synthesizer was far too expensive for average customers, but was used in libraries and service centers for visually impaired people [55].

9) In 1979 Allen, Hunnicutt, and Klatt demonstrated the MITalk laboratory TTS synthesizer developed at M.I.T. [Allen]. More details on the history of Speech Synthesis, in a chronological order are given in [55]. Some milestones of speech synthesis development are shown in Fig. 2.3.

![Fig. 2.3: Some milestones in speech synthesis](image-url)
2.2 Text–to-Speech systems

Text-to-speech systems have an enormous range of applications. Their first real use was in reading systems for the blind, where a system would read some text from a book and convert it into speech. These early systems of course sounded very mechanical, but their adoption by blind people was hardly surprising as the other options of reading Braille or having a real person do the reading were often not possible. Today, quite sophisticated systems exist that facilitate human computer interaction for the blind, in which the TTS can help the user, navigate around a windows system.

The mainstream adoption of TTS has been severely limited by its quality. Apart from users who have little choice (as in the case with blind people), people’s reaction to old style TTS is not particularly positive. While people may be somewhat impressed and quite happy to listen to a few sentences, in general the novelty of this soon wears off. In recent years, the considerable advances in quality have changed the situation such that TTS systems are more common in a number of applications. Probably the main use of TTS today is in call-centre automation, where a user calls to pay an electricity bill or book some travel and conducts the entire transaction through an automatic dialogue system beyond this, TTS systems have been used for reading news stories, weather reports, travel directions and a wide variety of other applications.

While this work concentrates on development of text-to-speech system, it is worth commenting that research in this field has contributed an enormous amount to our general understanding of languages.

2.2.1 Goals of TTS System Development

One can legitimately ask, regardless of what application talking computer want for, is it really necessary that the quality needs to be high and that the voice needs to sound like a human. Wouldn’t a mechanical sounding voice suffice? Experience has shown that people are in fact very sensitive, not just to the words that are spoken, but to the way they are spoken. After only a short while, most people find highly mechanical voices irritating and discomforting to listen. Furthermore tests have shown that user satisfaction increases dramatically the more “natural” sounding the voice. Experience particularly commercial experience shows that users clearly want natural sounding that is human-like systems [47].
Hence goals should be in building a computer system capable of speaking are to first build a system that clearly gets across the message, and secondly does this using a human-like voice. Within the research community, these goals are referred to as **intelligibility** and **naturalness**.

A further goal is that the system should be able to take any written input; if it is build an English text-to-speech system, it should be capable of reading any English sentence given to it. It is a need to develop the TTS system in other languages in country like India. There are several official languages speaking in various states. In few smart TTS are developed, as per as naturalness and intelligences are concern. TTS system available in regional languages are very less. With this in mind, it is worth making a few distinctions about computer speech in general. It is of course possible to simply record some speech, store it on a computer and play it back. This can be done by answer machine, which replays a message recorded in it; the radio plays interviews that were previously recorded and so on. This is of course simply a process of playing back what was originally recorded. **The idea behind text-to-speech is to “play back” messages.** One step away from simple playback is to record a number of common words or phrases and recombine them, and this technique is frequently used in telephone dialogue services. Sometimes the result is acceptable, sometimes not, as often the artificially joined speech sounded stilted and jumpy. This allows a certain degree of flexibility, but falls short of open ended flexibility. Text-to-speech on the other hand, has the goal of being able to speak anything, regardless of whether the desired message was originally spoken or not [52].

There are a number of techniques for actually generating the speech. These generally fall into two camps, which can call bottom-up and concatenative. In the bottom-up approach, speech signal generate “from scratch”, artificially create a basic signal and then modify it. It is based on formant frequencies referred as **formant Synthesis**. In the concatenative approach, there is no bottom-up signal creation purse; rather record some real speech, cut this up into small pieces, and then recombine these to form “new” speech. In a concatenative Speech synthesis signals are not generating from the scratch. At present concatenative techniques far out perform than other techniques, and for this reason concatenative techniques currently dominate [53].

The present work uses concatenative synthesis with phoneme, syllable-like units and maximum of words for the development of Marathi TTS synthesizer. In the next section, details are given only about concatenative synthesis.
2.3 Concatenative Synthesis

In the last decade, there has been a significant trend for development of speech synthesizers using Concatenative Synthesis techniques. There are a number of different methodologies for Concatenative Synthesis such as TDPSOLA, PSOLA, MBROLA and Epoch Synchronous Non over Lapping Add (ESNOLA) [2].

There are three main subtypes of concatenative synthesis:

1. **Unit selection synthesis**: This type of synthesis uses large speech databases (more than one hour of recorded speech). During database creation, each recorded utterance is segmented into some or all of the following: individual phones, syllables, morphemes, words, phrases, and sentences. Typically, the division into segments is done using a specially modified speech recognizer set to a "forced alignment" mode with some hand correction afterward, using visual representations such as the waveform and spectrogram. An index of the units in the speech database is then created based on the segmentation and acoustic parameters like the fundamental frequency (pitch), duration, position in the syllable, and neighboring phones [20].

   At runtime, the desired target utterance is created by determining the best chain of candidate units from the database (unit selection). This process is typically achieved using a specially-weighted decision tree. Unit selection gives the greatest naturalness due to the fact that it does not apply a large amount of digital signal processing to the recorded speech, which often makes recorded speech sound less natural, although some synthesizers may use a small amount of signal processing at the point of concatenation to smooth the waveform.

2. **Diphone synthesis**: It uses a minimal speech database containing all the Diphones (sound-to-sound transitions) occurring in a given language. The number of diphones depends on the phonotactics of the language: In diphone synthesis, only one example of each diphone is contained in the speech database. At runtime, the target prosody of a sentence is superimposed on these minimal units by means of digital signal processing techniques such as Linear predictive coding, PSOLA or MBROLA. The quality of the resulting speech is generally not as good as that from unit selection but more natural-sounding than the output of formant synthesizers. Diphone synthesis suffers from the sonic glitches of concatenative synthesis and the robotic-sounding nature of formant synthesis, and has few of the advantages of either approach other than small
size. As such, its use in commercial applications is declining, although it continues to be used in research because there are a number of freely available implementations [48].

3. **Domain-specific synthesis:** It concatenates pre-recorded words and phrases to create complete utterances. It is used in applications where the synthesizer output is limited to a particular domain, like trains schedule announcements or weather reports. This technology is very simple to implement, and has been in commercial use for a long time: this is the technology used by gadgets like talking clocks and calculators. The naturalness of these synthesizers can potentially be very high because the variety of sentence types is limited and closely matches the prosody and intonation of the original recordings. However, since these synthesizers are limited by the words and phrases in their database, these are not general-purpose and can only synthesize the combinations of words and phrases they have been pre-programmed with. Fig. 2.4 gives the sub-modules of a general concatenative synthesizer, as proposed by [36].

![Diagram of a general concatenation-based synthesizer](image)

**Fig. 2.4:** General concatenation-based synthesizer
2.4 Some Generic TTS framework

This section gives some of the generic frameworks available in public domain for the development of a TTS synthesizer. Some of these acts as back-end engines and others are full-featured commercial TTS frameworks [57].

2.4.1 MBROLA Synthesizer

MBROLA is a high-quality, diaphone-based speech synthesizer that is available in public domain. It is provided by the TCTS Lab of the Faculty Polytechnique de Mons (Belgium) which aims to obtain a set of speech synthesizers for as many languages as possible. The MBROLA speech synthesizer is free of charge for non-commercial, non-military applications. Anyone can send in his or her own speech recordings and an MBROLA database for synthesis is prepared. There are presently diaphone databases existing for several languages: American English, Brazilian Portuguese, Breton, British English, Dutch, French, German, Greek, Romanian, Spanish and Swedish. TCTS also provides speech database labeling software: MBROLIGN, a fast MBROLA-based TTS aligner. MBROLIGN can also be used to produce input files for the MBROLA v 2.05 speech synthesizers. More information and demos of the different voices and languages and also comparisons between MBROLA and other synthesis methods can be found on the MBROLA project home page [67]

2.4.2 FESTIVAL

The Festival TS synthesizer was developed in CSTR at the University of Edinburgh by Alan Black and Paul Taylor and in co-operation with CHATR, Japan [Black]. It is a freely available complete diaphone concatenation and unit selection TTS synthesizer. Festival is the most complete freeware synthesis system and it includes a comprehensive manual. Festival offers a general framework for building speech synthesis systems as well as including examples of various modules. As a whole, it offers full TTS synthesizer through a number of APIs. Festival is multi-lingual (currently English, Spanish and Welsh). The English version is most advanced and the developments for this version are very fast. The synthesizer is written in C++ and uses the Edinburgh Speech Tools for low-level architecture and has a Scheme (SIOD)-based command interpreter for control. The latest details and a full software distribution of the Festival Speech Synthesis System are available through the Festival home page [73].
2.4.3 FLITE

Flite (Festival-lite) is a smaller, faster alternative version of Festival designed for embedded systems and high volume servers. More information is available at:
http://www.speech.cs.cmu.edu/flite/

2.5 Tools for Development of TTS

The tools available for developing a TTS synthesizer include speech API’s provided by different vendors, and different markup languages. There exist many different APIs for speech output but there is a trend towards using the Microsoft API for synthesizers running on Windows. Another API that is not so frequently used is the Sun-Java Speech API [45]. These two are described below.

2.5.1 Sun-Java Speech API

The Java Speech API is being developed to allow Java applications and applets to incorporate speech technology. The API defines a cross-platform API to support command and control recognizers, dictation systems and speech synthesizers. Java Speech Grammar Format provides a cross-platform control of speech recognizers. Java Speech Markup Language provides a cross-platform control of speech synthesizers. Text is provided to a speech synthesizer as a Java String object. The Java Platform uses the Unicode character set for all strings. Unicode provides excellent multi-lingual support and also includes the full International Phonetic Alphabet (IPA), which can be used to accurately define the pronunciations of words and phrases. More information can be found on the Java homepage: http://java.sun.com/products/java-media/speech/

2.5.2 Sapi Microsoft Speech API

The leading vendors are beginning to support Microsoft's Speech API, or SAPI, which is based on the COM specification and is being adopted as the industry standard. The motive of SAPI is to eventually allow interoperability between the speech engines. The Microsoft Speech API provides applications with the ability to incorporate speech recognition (command & control dictation) or TTS, using either C/C++ or Visual Basic. SAPI follows the OLE Component Object Model (COM) architecture. It is supported by many major speech technology vendors [70]. The major interfaces are:
- Voice Commands: high-level speech recognition API for command and control.
- Voice Text: simple high-level TTS API. The Voice Text object is available in two forms: a standard COM interface IVoice Text and companion interfaces, and also an ActiveX COM object, VtxtAuto.dll
- Multimedia Audio Objects: audio I/O for microphones, headphones, speakers, telephone lines, files etc

2.5.3 Markup Language

The input to a TTS synthesizer is often a string of words but sometimes it also contains information in the form of markers to indicate emphasis, stress placement, speech rate, voice etc. System providers normally have their own markup codes but there is some co-operation between providers to develop standards for markups. A number of markup languages have been established for rendition of text as speech in an XML compliant format.

SSML-SSML is the most recent markup language, proposed by the W3C.

JSML-The Java Synthesis Markup Language (JSML), an SGML-based mark-up language, is being specified for formatting text input to speech synthesizers. JSML allows applications to control important characteristics of the speech produced by a synthesizer. Pronunciations can be specified for words, phrases, acronyms and abbreviations to ensure comprehension and naturalness. [71].

SABLE-Sable is consortium aimed at providing a single standard for speech synthesis markup. The consortium's principle aim is to merge the two existing proposed standard, namely SSML developed by Bell Labs and Edinburgh, and JSML, developed by Sun [54].

2.6 TTS Synthesizers for Indian Languages

2.6.1 Studies in India

India is a country with more than 17 official languages and each language is spoken in different forms across different places. According to experts, Speech recognition and Speech synthesis technology can be very useful in the Indian context as it provides an easy interface for interacting with computers. Using such a convenient means of rendering information to or from the machine would mean that the end-user need not be computer literate and still can use the power of the IT industry.
There are several language problems in India which researcher can solve in the context of text & speech processing is concern. Till 1990s, Indian speech synthesizers were research synthesizers, generating small segments of speech in non-real time. Several factors boosted the activity in the 1990s: (i) Government of India had funded Indian language projects generously, through Technology Development for Indian Languages (TDIL) and other schemes. (ii) Fast PCs, with decent quality sound boards, were available at affordable prices. (iii) Concatenation synthesis technology has evolved and development environments were available. In this regard Government of India has funded several projects, which were taken up by various institutions all over the country [22].

One of the program i.e. **TDIL Programme**, Department of Information Technology under Ministry of Communications & Information Technology, New Delhi. Website: http://tdil.mit.gov continuously providing the infrastructure, research opportunity to the researcher and developer. Many system they have developed some are in developing stage as per as language processing and TTS is concern. Their successful development contains TTS in Tamil, Bangali, Oriya and Malayalam.

As per TTS development, the most helpful aspect of Indian scripts is that they are basically phonetic in nature, and there is one-to-one correspondence between the written and spoken forms of most of the Indian languages. This makes the task of automatic Phonetization simpler [13]. Further section gives the survey of the work done in the field of TTS for Indian languages.

The goal of this survey is to study the techniques used for Indian language TTS development and identify suitable techniques for development of Marathi TTS synthesizer. Even among Indian Languages, focus is given more to the work done for Hindi TTS synthesizer, keeping in mind that Hindi text and speech resembles with Marathi and Devnagari text and speech.

Several institutions in India are working in the field of TTS for Indian languages. The work done in the following institutions is discussed in detail.

- C-DAC Bangalore - Matrubhasha AP[68]
- IIT Mumbai - Vani Framework
- HP Labs - Hindi TTS
- IIT Kharagpur - SHRUTI TTS
- Simputer Trust - Dhvani TTS [Dhvani ], [76]
- IIIT Hyderabad- Telgu Text to Speech Synthesis System
- C-DAC Kolkata – Bengali TTS
- C-DAC, Thiruvananthapuram- Malayalam TTS System (SUBHASHIN™)
- RCILTS Tamil, Anna University Chennai- Tamil TTS (Ethiroli)

Some other institutions where the TTS development is going on, include – IIT Madras, IIIT Hyderabad, HCU, IISc Bangalore, Utkal University Bhubaneswar, TIFR Mumbai, CDAC Noida, Pune [34]

2.6.2 Country Report –India: National Status

As per the Country Report –India, O-COCOSDA 2007 Meeting Hanoi, Vietnam Dec. 4-6, 2007, out of 25 top institutes in India, only at TIFR Mumbai, Speech Synthesis System for Marathi has been developed using formant synthesis strategy with phonemes as a database [4].

Table 2.1 gives the comparative study of speech synthesis at various institutes of India

<table>
<thead>
<tr>
<th>Institute</th>
<th>Language Cover</th>
<th>Synthesis Strategy</th>
<th>Unit/Data base</th>
<th>Text/Speech Segment processing/ Tools</th>
<th>Prosody</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIFR Mumbai</td>
<td>Hindi, Bengali, Marathi, Indian English (Partly)</td>
<td>Format (Klatt – type) Synthesis</td>
<td>Phonemes and other units</td>
<td>Automatic parsing rules for Phonetization, Rules for smoothing prosody</td>
<td>Prosody rules (limited)</td>
<td>Unlimited TTS – More than Average</td>
</tr>
<tr>
<td>IIIT (Hyd)</td>
<td>Hindi, Telugu, Other languages</td>
<td>Concatenative</td>
<td>Data base in required languages as per requirements of Festival System</td>
<td>For unit as per language&amp; implemented</td>
<td>Prosody studies in required language &amp; implemented</td>
<td>Un-limited TTS- better than Average</td>
</tr>
<tr>
<td>Institution</td>
<td>Language(s)</td>
<td>Concatenative/Phonetic</td>
<td>Syllabus/Units</td>
<td>Automatic Processing</td>
<td>Pitch Tracks/Implementation</td>
<td>TTS Quality</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------</td>
<td>------------------------</td>
<td>----------------</td>
<td>----------------------</td>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>IIT, Chennai</td>
<td>Hindi, Tamil</td>
<td>Concatenative diphones</td>
<td>Syllabus/Mainly</td>
<td>Group delay functions</td>
<td>Festival System determination</td>
<td>Unlimited</td>
</tr>
<tr>
<td>CDAC, Pune</td>
<td>Hindi, Indian English</td>
<td>Concatenative phonemes</td>
<td>Phonemes, other units</td>
<td>Automatic parsing rules for Phonetization, Rules</td>
<td>Some Prosody rules</td>
<td>Un-limited</td>
</tr>
<tr>
<td>CDAC, Noida</td>
<td>Hindi</td>
<td>Concatenative</td>
<td>Multi-form units, Syllables, frequent words, phrases etc.</td>
<td>Parsing for syllables, Statistical processing of text formation of phonetically rich sentences and other units (Vishleshika)</td>
<td>Study of intonation patterns,</td>
<td>Domain Specific-Excellent, Unlimited TTS-Average</td>
</tr>
<tr>
<td>CDAC Kolkata</td>
<td>Bengali</td>
<td>Concatenation</td>
<td>Phonemes &amp; Sub-Phonemes (Size 1 MB)</td>
<td>Cool – edit Phonemic/Segmentation</td>
<td>TDPSOLA/ESNOLA</td>
<td>Unlimited</td>
</tr>
<tr>
<td>Bhrigus Software Ltd. Hyd</td>
<td>Hindi, Telugu &amp; Others</td>
<td>Concatenative phonemes</td>
<td>Phonemes, Using Festival requirements</td>
<td>Fest VOX tools Festival</td>
<td>Intonation using (CART for Prosody modeling)</td>
<td>Unlimited TTS-Average</td>
</tr>
<tr>
<td>Prologix Software, Lucknow</td>
<td>Hindi</td>
<td>Concatenative</td>
<td>Di-phone data base</td>
<td>Festival based- Fest VOX tools</td>
<td>Not applied</td>
<td>Unlimited TTS-better than Average</td>
</tr>
<tr>
<td>Webel Mediatronics, Kolkata</td>
<td>Bengali, Hindi</td>
<td>Formant type</td>
<td>Phonemes (Parameter of phonemes)</td>
<td>Rules for concatenation and smoothing of parameters Text processing rules</td>
<td>Intonation rules being implemented</td>
<td>Unlimited TTS-Less than Average</td>
</tr>
</tbody>
</table>

Table 2.1: Comparative study of Speech synthesis at various institutes of India
2.6.3 Details of some TTS System

A. C-DAC Bangalore– Matrubhasha API

Matrubhasha is a project carried out at C-DAC Bangalore, as a part of the digital divide bridging activities [68]. It was undertaken with the intention of making end user applications speak and listen to the masses in any Indian language that they are comfortable to communicate in. Matrubhasha is a Unicode and MBROLA based software solution for TTS synthesis and, CMU Sphinx based Speech Recognizer for Indian languages. It is visualized with the objective of building a framework, which can be used by any software developer to incorporate speech capabilities (in Indian languages) into his/her software thus increasing its usability across different sections of society. The Matrubhasha project is an activity of the ICT Research and Training Centre (India) of Development Gateway Foundation. The Government of India is a member of the Development Gateway Foundation, a World Bank initiative. Information and Communication Technologies (ICT)

B. IIT Mumbai– VANI Framework

Vani is a TTS synthesizer proposed by IIT Mumbai. It is primarily developed for Hindi, but with minor modification could directly be modified for any language which is phonetic in nature. The approach is similar to concatenation synthesis, with phonemes as the basic unit. However, the phonemes are not selected from a database, but are generated from a more basic unit, which they call fract-phoneme in the case of vowels. The basic observation is that vowels look like a continuous repetition of very small segment phonemes called fract-phonemes. Since fract-phonemes are very small in size, they are a good choice for acting as a basic unit. The aim of Vani was to allow complete specification of speech, i.e., one can get the complete software for TTS. In order to give complete freedom of expression to the user Vani represents the speech by a new encoding scheme called vTrans which is an extension of iTrans encoding scheme. Vani generates speech from a given vTrans file [62].

C. HP Labs – Hindi TTS

HP Labs, India have developed a Hindi TTS synthesizer based on Festival framework. This effort is a part of the Local Language Speech Technology Initiative (LLSTI), which facilitates collaboration between motivated groups around the world, by enabling sharing of tools,
expertise, support and training for TTS development in local languages. It aims to develop a TTS framework around Festival that will allow for rapid development of TTS synthesizers in any language. Since Festival does not provide complete language processing support specific to various languages, it needs to be augmented to facilitate the development of TTS synthesizers in certain new languages. Because of this, a generic G2P converter has been developed at HP Labs India as part of the LLSTI initiative. The system consists of a rule processing engine which is language independent. Language specific information is fed into the system in the form of lexicon, rules and mapping [66].

D. IIT Kharagpur- SHRUTI TTS

An Indian language TTS synthesizer (named SHRUTI) that accepts text inputs in two Indian languages namely Hindi and Bengali and produces near natural audio output. It has been developed at IIT Kharagpur [38]. The synthesizer runs on a Compaq iPaq PDA built around the Intel Strong Arm-1110 processor running Microsoft Pocket PC, a customized version of Microsoft’s operating system WinCE for mobiles and other handheld devices. The synthesizer has also been ported to a Casio Cassiopeia built around a MIPS 124 processor running Pocket PC. Two versions of the synthesizer have been built, one which resides on the system memory and another which runs from a storage card.

E. IIT Madras

Systems Development Laboratory, IIT Madras has come up with a speech enabled multilingual editor, which supports text processing and speech generation for several Indian languages including Telugu. The speech is produced using the MBROLA speech engine. Since the databases required for Indian languages are not yet available for use with MBROLA, the IITM team had experimented with other available data bases where the phonemes are close to the phonemes of Indian languages. 65].

The Telecommunication and Computer Networking (TeNeT) Group, a coalition of group of Faculty members from the Electrical Engineering and Computer Science & Engineering Departments of IIT-Madras, has taken initiative for developing local language speech interface system. In this regard, they have worked with Festival to perform speech synthesis. Their system has following features:
• Common phoneset for Hindi and Telugu, also usable for other Indian languages.
• Diphone unit selection for synthesis.
• Data-driven prosody modeling using Classification and Regression Tree (CART).
• Concatenative synthesis technique is used to produce natural sounding speech.

F. IIIT Hyderabad

Language Technologies Research Centre (LTRC), IIIT Hyderabad has as its goal the development of technologies dealing with language. It includes technologies pertaining to translation and other NLP areas, speech processing, optical character recognition, etc. Their research and development focus is to develop speech interfaces for Indian Languages. Currently, they are working on TTS synthesizers for Indian languages. They already have some restricted domain TTS synthesizers developed for Telugu and Hindi working in a number of domains. LTRC has used data-driven (Corpus-based / Example-based) approach using festvox for the development of TTS synthesizers. They have used Unicode as the input to the front end for the text processing. They performed various experiments with different choices of units: syllable, diphone, phone and half phone, for unit selection speech synthesis for Hindi. The observation they have come up with is that the syllable unit performs better than diphone, phone and half phone, and seems to be a better representation for languages such as Hindi. Also, the half phone synthesizer performs better than diphone and phone synthesizers. They have also proposed a data-driven synthesis method for Indian languages using syllables as basic units for concatenation [14]. Some applications developed by LTRC based on TTS technology i.e. Samachar Vani, Talking Tourist aid, Screen Reader [34].

They are also working on a Hindi speech synthesizer that fits in 1.45 MB memory area. Such a small speech synthesizer can be put on small systems like PDAs and embedded devices including mobiles. The only other speech synthesizer available at comparable sizes for Indian languages is the Dhvani speech synthesizer which is more than 2 MB in size. LTRC has also come up with an implementation of an API for the TTS synthesizer in Windows environments.

G. IISC Bangalore -Thirukkural & Vaachaka

IISc Bangalore has developed a complete Tamil and Kannada TTS synthesizer named Thirukkural and Vaachaka respectively. Similar techniques are applied in the development of
both the TTS synthesizers. Syllables of different lengths have been selected as units. Automatic segmentation algorithm has been devised for segmenting syllables into consonant and vowel. Thirukkural is designed in VC++ [27].

**H. Tata Institute of Fundamental Research (TIFR), Mumbai**

Work has been done in TIFR, Mumbai for TTS synthesis in Indian English. The important components of the language processor used are the parser to categorize words, an Indian English phonetic dictionary, morphological analyzer, letter-to-sound rules, phonological rules, prosody rules and Indian name detector. The relevant rules are formulated with the aid of a large CMU pronunciation dictionary and a language tool GENEX, developed in TIFR, which can generate a sub-dictionary, a set of specified constraints [69].

**I. C-DAC, Noida**

The TTS project at C-DAC, Noida, is based on the concatenative approach and aims at developing a TTS synthesizer for Indian languages primarily, Hindi. The input in Unicode is processed by the Text processing unit and the speech-processing unit resulting in synthesized speech. The input text is also normalized and converted to their orthographic form, before breaking into the basic units [22].

**2.7 Extract of some Research Papers**

A thorough review of earlier work carried out and published in the field is taken; near about the fifty papers on the Text –to- speech were referred. Papers referred mostly on the state of art clearly focused on regional and standard language speech synthesis, their strategies, experimentation, tools used, evolution procedure and techniques. Following are the few research paper and their extract which referred for Development of TTS for Marathi language.

S.P. Kishore and Alan W Black [20] focused on theoretical background of TTS system, role of language technologies, how the words are mapping? He proposed the system for Hindi and Telgu voices. System was evaluated on the basis of subjective test on different size of database. Results of his system were taken for the comparison. He has taken non uniform data which may lead wrong perception by the subjects to decide correct NSN words.
Aniruddha Sen [22] presented the detail review of the TTS systems for Indian languages; various test methodology, important issues to take up for development of an intelligent TTS system.

In one of the paper of S.P.Kishore [16] detail study of Unit Size selection of words has taken, techniques for the selection of words was very useful for cutting and concatenation ,various experimentation helps to evolute the MATTS. ITRANS-Transliteration scheme for TTS was discussed in detail.

Rohit Kumar and Rajeev Sangal [14] proposed the various data for the concatenation of words for the TTS system. They had taken the basic unit as syllables to form the words and evolute the system for the naturalness. Method helps to create the data base on selection of syllables.

For formation of specific words from the database, it is important to check the word if not then deriving from the syllables is one of the way concatenative synthesis. S. P. Kawachale and J. S. Chitode [19] focused on the issue syllable concatenations, but here the naturalness hampers.

Novel technique of approximate matching of the syllables as back-off technique for building voices was proposed by E. Veera Raghavendra [23] where three different voices English full voice, English ARCTIC voice and Mandarin was used for TTS. Approach has been used to experiment and exploit syllable-like large units for concatenative synthesis. This helps for appropriate selection of syllable as unit for concatenation.

Hiroyuki Segi and Tohru Takagi [24] investigated the new concatenative speech synthesis method using context dependent phoneme sequences with variable length as search units for Japanese language. They performed various subjective evolution of synthesized speech like run time test, comparison with conventional methods, mean opinion score (MOS) test .They got the MOS score at 3.94. This helps process for conduction of various test and selection of appropriate test.

Eric Lewis and Mark Tatham [11] developed the TTS system on simple word and syllable database where they got the maximum naturalness .Process of syllable reconstruction was thoroughly explained and implemented by K Morton [12].

Javier Latorre, Koji Iwano and Sadaoki Furui [17] presented TTS with mixed-lingual texts using a polyglot system. This is for English portions of mixed English/German texts using a German-based polyglot text-to-speech (TTS) synthesis system. The polyglot system is based on
a monolingual German TTS system, which uses a phone mapping from English to German. Two systems with varying degrees of assimilation to English are compared. The naturalness and overall intelligibility and acceptability of the polyglot systems are assessed by native bi-lingual speakers of both English and German. Study of this work is useful for the evolution of intelligibility of MATTS.

Kalika Bali, Partha and Pratim Talukdar [25] demonstrated generic Grapheme –to-phoneme converter for Hindi (TTS), where they used the festival framework. Since Festival does not provide complete language processing support specific to various languages, it needs to be augmented to facilitate the development of TTS systems in certain new languages. They handled the schwa deletion and compound word extraction. Experimentation carried out to test the Hindi G2P on a text segment of 3485 words, good accuracy of word Phonetization was obtained. This Hindi G2P has been used for phonetising large text corpora, used in designing an inventory of phonetically rich sentences. This presentation helps to develop inventory of good coverage of the phonetics.

Jerneja Zganec Gros and Mario Zganec [26] focused on the development of TTS by using unit selection method of concatenative synthesis. He performs the various subjective analysis. This work helps specially to test system with different opinion score versus selective samples with various size of database.

G. L. Jayavardhana Rama and A. G. Ramakrishna [27] worked on Tamil TTS. They developed the work in two phases namely, offline phase and the online phase. Offline phase includes pre-processing, segmentation and pitch marking. Online phase includes text analysis and synthesis. Work proposed the BME algorithm to identify the word from database. ‘Thirukkural’ synthesizes intelligible Tamil speech. It has a male voice. Efforts are not made to reduce the size of the large database. Attempts are less to make natural sounding.

P. Prathibha and A.G. Ramakrishnan [28] proposed the Tamil TTS. Thirukkural-I is the basic Tamil speech synthesis system which can pronounce only pure Tamil words. Thirukkural II (improved version of Thirukkural-I) handles pronunciation of words of foreign origin. Due to the non-standardization of the language, Tamil Speech synthesis system cannot perform adequately well, if it strictly follows the Tamil pronunciation rules. Thirukkural II generates intelligible and acceptably natural speech. It can synthesize many of the proper nouns derived from certain other
languages. Better naturalness due to incorporation of pitch modification and emulation of voices of different speaker. Work helps to improve the Naturalness in synthesized Speech.

Branislav Gerazov and Goce Shutinoski [29] presented novel approach to develop synthesis of Macedonian language. The system uses a unique optimized mixed-rank inventory, based on a modification of the classical diphone concept. A new unit type is introduced in the work, dubbed the quasi-diphone unit. A set of these units is designed to cover all the critical transitions between phones and phone-length units for concatenation. This allows for inventory optimization in respect to its size and quality of the generated speech. A result shows TTS achieved full intelligibility and reasonable naturalness while maintaining small inventory. Work also carried out the spectrogram analysis of the various words and sentences.

Spectrogram analysis for synthesized speech is thoroughly presented in the PDF presented by Kishore Prahallad [31] to evolve the performance of concatenated sentences, words, syllables and phonemes.

2.8 Summary

In this chapter, exhaustive survey of the work done in the field of TTS for Indian languages is presented. The goal of this survey is to study the techniques used for Indian Language TTS development. A lot of work is done and a lot of work is in progress in the field of TTS for Indian languages. Most of the TTS synthesizers for Indian languages were able to generate intelligent speech; however getting speech close to natural speech is still a challenge. The Hindi Telgu, Panjabi, Bengali and Kannada TTS synthesizer, developed by many of the institutions in India. It was found to be producing a quality speech which was close to natural speech. There is scope in these languages also for improvement in naturalness and data optimization.

It has been found during the literature survey that there is no any TTS synthesizer available for the Marathi language that uses concatenative synthesis strategy. After the study of various TTS works by Indian institutions, attempts were made to develop a TTS synthesizer for the Marathi language.