Text to Speech Synthesis for Numerals into Punjabi language

Dissertation submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Maninder Singh
(Roll No. 801132015)

Under the supervision of:
Mr. Karun Verma
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

July 2013

Certificate

I hereby certify that the work which is being presented in the thesis entitled, **"Text to Speech Synthesis for Numerals into Punjabi language"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Computer Science and Engineering* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr. Karun Verma* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Maninder Singh
(Maninder Singh)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

15/7/13
(Mr. Karun Verma)
Assistant Professor
Computer Science and
Engineering Department
Thapar University
Patiala

Countersigned by

(Dr. Maninder Singh)
Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

Page | i

Acknowledgement

I would like to express my deep sense of gratitude to my supervisor, **Mr. Karun Verma**, Assistant Professor, Computer Science and Engineering Department, Thapar University, Patiala, for his invaluable help and guidance during the course of thesis. I am highly indebted to him for constantly encouraging me by giving his critics on my work. I am grateful to him for giving me the support and confidence that helped me a lot in carrying out the research work in the present form. And for me, it's an honor to work under him.

I also take the opportunity to thank **Dr. Maninder Singh**, Associate Professor and Head, Computer Science and Engineering Department, Thapar University, Patiala, for providing us with the adequate infrastructure in carrying the research work.

I would also like to thank my parents and friends for their inspiration and ever encouraging moral support, which went a long way in successful completion of my thesis.

Above all, I would like to thank the almighty God for His blessings and for driving me with faith, hope and courage in the thinnest of the times.

Maninder Singh

Abstract

Text To Speech synthesis (TTS) is an online application to convert the text written in a language into speech. The text to speech conversion system enables user to enter text in Punjabi and as output it gets sound.

This work presents the steps followed for converting text to speech for Punjabi (Gurumukhi) language and the algorithm used for it. This application helps user to type text and as output it gets the speech.

The text inputted may be written by the operator or it may be scanned paper that is converted to speech. Though the user gets its output but still conversion of text to speech is not an easy task. It involves a great deal of work to be performed perfectly. It is difficult to create database of number of phonemes, syllables, words for a particular language.

This thesis represents the work done on text to speech synthesis for numerals in which speech waveform is generated and applies it to Punjabi speech synthesis using the general speech synthesis architecture.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1
Introduction

In today's life every person wants that the computer systems should behave like humans and proved to be user friendly. Even many of the great researchers have dreamed of involving the machines in every face of human life. With the growth in the power of computing machines, their applications in modern daily life are also rising. Speech and spoken words have always played a big role in the individual and collective lives of the people. Speech represents the spoken form of a language and is also one of the important means of communication. So the researchers have been trying their hands to make the computer speak and to make the task easy.

Over the past few decades, many researchers have been done in the field of converting text to speech. This research has resulted in important advances with many systems being able to generate a close to a real natural speech. These advances in speech synthesis also pave the way for many speech related new applications. This will also help physically challenged people and even partially blind people who find it difficult to read from a monitor.

Natural Language Processing (NLP) techniques is very useful in the production of voice from an input text as known as Text-To-Speech synthesis, and the inverse process, which is the production of a written text transcription from an input voice utterance as known as Automatic Speech Recognition.

1.1 Natural Language Processing

Natural Language Processing (NLP) is the science most directly associated to processing human (natural) language. It derives from computer science since computers, or any other processing unit, are the target devices used to accomplish such processing. This description responds basically to the "Processing" particle in NLP. What makes NLP different from any other processing-related activity is the field of application: the human languages. They deal with more knowledge-related aspects thus requiring the support of learning capabilities by the processors of text.

It could be stated that most NLP or NLP-related computerized tasks can be wrapped by the more general concept of Machine Learning, clearly related to computer science, which contemplates any subject relating to the use of computers for the

inference of the relevant features of the systems of study. Since the particular field of study is natural language, these learning techniques are of vital interest, because in some way, we humans make use of this kind of language as our basic means of communication and reasoning, inherently. If otherwise a formal language was to be studied (e.g., a programming language), there would be no reason to make use of such learning approaches because the construction and logic issues bound to the formalism of that kind of language would already be known or predefined.

Common applications of sheer high-level NLP would deal solely with text data (at the input and output of the system) such as text classification, text summarization, question answering or machine translation. When approaching speech technologies other domains should be considered, despite NLP techniques refer exclusively to the textual analysis or synthesis of the applications. Either Text To Speech (TTS) synthesis or Automatic Speech Recognition (ASR) need a trustful module of NLP because text data always appears somewhere in the processing chain. TTS produces a speech utterance from an input text while ASR produces such text from an input speech utterance. Although these two objectives look very same but their approach of working differs significantly.

1.2 Speech Technology

When we talk about speech-based interfaces for computer system, we refer to two basic technologies: one is Speech Recognition and other one is Speech Synthesis.

Speech Recognition is Speech-to-Text system and Speech Synthesis is Text-to-Speech system and, together forms a speech interface.

1.3 Speech Recognition

Speech recognition (SR) is the translation of spoken words into text. It is also called as “computer speech recognition”, “automatic speech recognition” abbreviated as ASR [1], “speech to text” abbreviated as STT.

Some SR systems have “speaker independent speech recognition” that do need of training while others use “training” where an individual speaker manually reads sections of text into the SR system. First the person's specific voice is analyzed by systems and to produce more accurate result, tunes the recognition of that person's speech. The system is called “speaker independent” systems that do not need training. The “speaker dependent” systems use training.

Speech recognition applications include voice user interfaces such as Home appliance control [2] [3], voice dialing for example “Call Maninder”, search a podcast like to find audio where particular words were spoken, simple data entry like entering a credit card number, preparation of structured documents for example a radiology report, speech-to-text processing like a word processors or emails, and aircraft that usually termed Direct Voice Input.

The voice recognition system mainly focuses on finding the identity of “who” is speaking, rather than more emphases on what they are saying. Recognizing the speaker can easier the task of translating spoken words in systems that have been already trained on specific speaker's voices or it is useful to verify or authenticate the identity of a person as part of a security process.

The goal of speech recognition system is to accurately and efficiently convert a speech signal into a text message independent of the device, speaker or the environment. ASR may be viewed as working in four stages namely, preprocessing, feature extraction, modeling and testing [4] as shown in Figure 1.1.

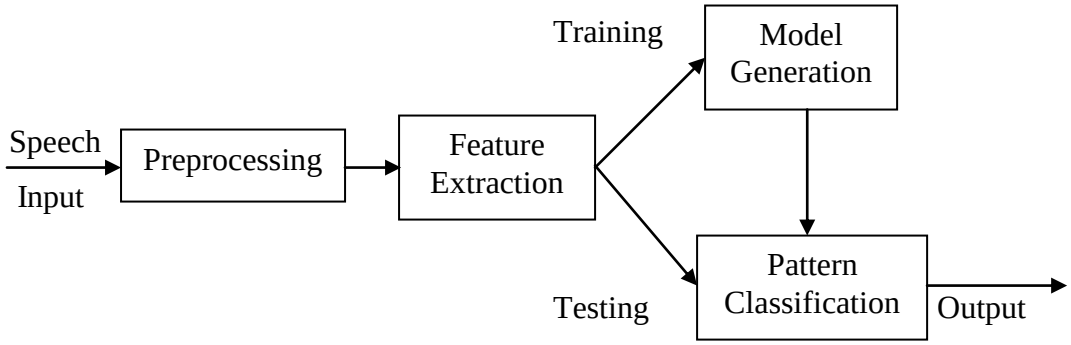

Figure 1.1 Speech recognition system components

Automatic Speech Recognition systems make use of NLP techniques in a fairly restricted way: they are based on grammars. A grammar as a set of rules that determine the structure of texts written in a given language by defining its morphology and syntax. ASR takes for granted that the incoming speech utterances must be produced according to this predetermined set of rules established by the grammar of a language, as it happens for a formal language. In that case, Context-Free Grammars (CFG) plays an important role since they are well capable of representing the syntax of that language while being efficient at the analysis (parsing)

of the sentences [5]. For this reason/restriction, such language cannot be considered natural. ASR systems assume though that a large enough grammar rule set enable any (strictly formal) language to be taken for natural.

NLP techniques are of use in ASR when modeling the language or domain of interaction in question. Through the production of an accurate set of rules for the grammar, the structures for the language are defined. These rules can either be hand crafted or derived from the statistical analyses performed on a labeled corpus of data. The former implies a great deal of hard-work since this process is not simple and not brief because it has to represent the whole set of grammatical rules for the application. The latter is generally the chosen one because of its programming flexibility at the expense of a tradeoff between the complexity of the process, the accuracy of the models and the volume of training and test data available (notice that the corpus has to be labeled, which implies a considerably hard workload). Since hand-crafted grammars depend solely on linguistics for a particular language and application they have little interest in machine learning research in general.

1.4 Speech Synthesis

The aim of speech synthesis is to provide the spoken output to the users by generating speech from text. Thus it can read out the textual contents from the screen. A speech recognizer has the ability to understand the spoken words and convert it into text. So, it converts the speech input into its corresponding text output. Thus Speech synthesis is used in spoken dialog systems, applications for blind and visually-impaired persons, telecommunication applications, eyes and hands free applications.

The function of text-to-speech (TTS) system is to convert an arbitrary text to a spoken waveform. This task generally consist two steps, *i.e.*, text processing and speech generation. Text processing is the conversion of the given text into a sequence of synthesis units, whereas speech generation is generation of an acoustic wave form corresponding to each of these units in the sequence. A general view of the system is depicted in the Figure1.2

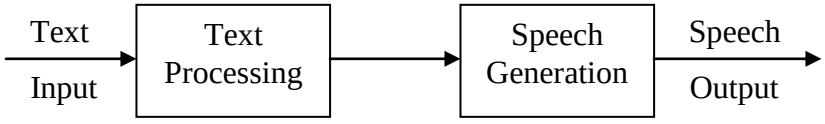

Figure 1.2 Text to speech system

This thesis work is mainly for Punjabi language in which TTS synthesizer has been developed that converts a numeral value into Punjabi speech.

1.4.1 Components of a Text to speech Synthesizer

As depicted in Figure 1.3, a TTS synthesizer is composed of two parts:

• A front end is high-level synthesis phase that receives the text as input and outputs a symbolic linguistic representation.

• A back end is low-level synthesis phase that receives input in the form of symbolic linguistic representation and gives the synthesized speech in a waveform as outputs.

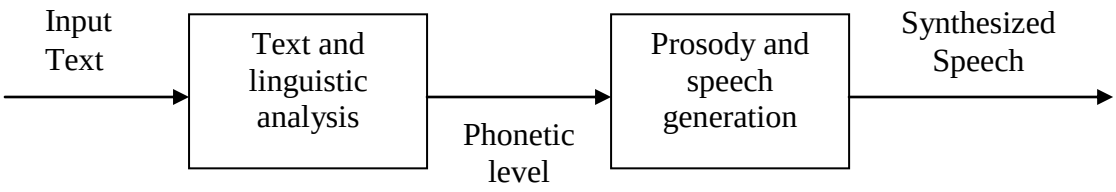

Figure 1.3 Black-box view of a TTS synthesizer

1.4.2 TTS Synthesizer v/s Talking Machine/Voice Response Systems

A TTS synthesizer will be different from any other type of "talking machine" in the sense that it should be able to automatically produce "new sentences"; hence, it should be also capable to read unrestricted text. It is also important to note that a TTS synthesizer differs from any of the so-called voice response systems (such as announcements made in a train etc.) in a sense that the vocabulary in a TTS synthesizer is not limited to an extent. The set consisting of utterance types is also not limited in TTS synthesizer.

1.4.3 Quality of a TTS Synthesizer

Commonly, two quality criteria are proposed for deciding the quality of a TTS synthesizer.

- Intelligibility – It mainly concerns to how easily the output can be realized and understood. It may be determined by taking into account several kinds of units (phrases, words, phonemes, syllables, etc.).
- Naturalness or pleasantness - it concerns to how much the output sounds like the speech of a real speaker (person).

Naturalness may be associated to the concept of realism. Hence, listening to a synthetic voice should necessarily allow the listener to attribute this voice to some pseudo-speaker and to realize some kind of expressivities and also some indices differentiate the speaking style and the specific situation of elocution. For this goal, the corresponding extra-linguistic information essentially is supplied to the synthesizer.

Most of the existing text to speech synthesizers are more concerned about to produce a standard level of intelligibility with the naturalness dimension, the stability to control expressions; speech style and pseudo-speaker/not genuine identity are still now the areas of concern and need improvements.

However, users' interests vary to a large extent according to the field of application: general public applications for example telephonic information retrieval need maximal realism and naturalness, whereas some applications involving professionals (process or vehicle control) or highly motivated persons (visually weakened, applications in unfriendly environments) demand intelligibility with the highest priority.

1.4.4 Application of Speech Synthesis

Synthetic speech may be used in several applications. Revolution in communication aids have found from low quality talking calculators to modern 3D applications, for example talking toys or heads.

The implementation method of synthetic speech depends mostly on used application like announcement or warning systems, unrestricted vocabulary is not important and the best result can be usually achieved with some simple messaging system.

With good development some investments may also be saved. The other applications, such as reading machines for the blind or electronic-mail readers, must have unlimited vocabulary needed in a TTS system. The application field of synthetic speech is expanding fast whilst the quality of TTS systems is also improved.

Speech synthesis systems seem more adaptable for common customers, which makes these systems more affordable for everyday use. Such as better availability of TTS systems may improve employing possibilities for people with communication difficulties.

- **Aid to Vocally Handicapped**

A hand-held, battery-powered synthetic speech aid can be used by vocally handicapped person to express their words. The device will have especially designed keyboard, which accepts the input, and converts into the required speech within blink of eyes.

- **Source of Learning for Visually Impaired**

Listening is an important skill for person who is visually blind. Blind people depend on their ability to hear or listen to understand information quickly and efficiently. Students are gained information from books on tape or CD by using their sense of hearing, but also to judge/measure what is happening around them.

- **Talking Books and Toys**

Talking book not only teaches how to read but also has more impact on students than text reading. It makes their study more enjoyable and easy. In the same way talking toys are a great source of fun and entertainment for children.

- **Games and Education**

Synthesized speech can also be used in various educational institutions in field of study as well as sports. A teacher can be tired at a point of time but a computer with speech synthesizer can teach whole day with same efficiency and accuracy.

- **Telecommunication and Multimedia**

TTS systems provide facility to access textual information over the telephone. Texts can be large databases so problem that it can hardly be read and stored as digitized speech. User queries to such information retrieval systems and that could be put through the user's voice with the help of a speech recognizer, or through the telephone keyboard. Synthesized speech may also be used to speak out short text messages in mobile phones.

- **Man-Machine Communication**

Speech synthesis may be used in several kinds of human-machine interactions like in warning alarm systems, clocks and washing machines. Synthesized speech can be used to give more accurate information of the current situation. Speech signals are far better than that of warning lights or buzzers as it enables to react to the signal more fast if the person is unable to get light due some obstacles.

- **Voice Enabled E-mail**

Voice-enabled e-mail uses voice recognition and speech synthesis technologies to enable users to access their e-mail from any telephone. The subscriber dials a phone number to access a voice portal, then, to gather their e-mail messages, they press a couple of keys and system sense as like "Get my e-mail." Speech synthesis software translates e-mail text message to a voice message that is getting played back over the phone. Voice-enabled e-mail is especially useful for mobile workers, because it make it possible for them to access their messages easily from virtually anywhere (as long as they can get to a phone), without having a need to invest in expensive equipment like laptop computers or personal digital assistants (PDAs).

1.5 Challenges in Text-to-Speech System

To perform the work perfectly in every scenario, a system should be programmed accordingly. The programmer should find the cases that the program can face. While it is easy for some tasks, it can be very difficult to manage for some tasks like natural language processing. Since it is difficult to determine every possible input to the system, some techniques which are different from the classical programming approach should be used. However, these techniques usually offer some heuristics that give correct result in most cases but not in every scenario. A TTS system works with a natural language text, therefore a TTS system also meet such problems.

1.5.1 Text pre-processing

Text pre-processing is usually a complex work and includes many language dependent problems. Numerals and digits must be converted into full words. For example in English, numeral 345 would be expanded as *three hundred and forty-five* and 1630 as *sixteen thirty* (if year) or *one-thousand six-hundred and thirty* (if measure). Related cases include the difference between *the 747 people* and *747 pilots*. Fractions and dates are also problematic. 5/18 can be expanded as *five-eighteens* (if fraction) or *May eighteenth* (if date). Expansion ordinal numbers also being problematic. The first three ordinals must be expanded different from others, 1st as *first*, 2nd as *second*, and 3rd as *third*. Similar kind of contextual problems are also faced with roman numerals. Chapter IV should be expanded as *Chapter four* and Henry IV as *Henry the four* and *I* may be either a pronoun or number. Roman numerals may have confusion with some common abbreviations, such as MCM.

Numbers may also have several special forms of expression, such as 33 as *double three* in telephone numbers and 1-0 as *one love* in sports.

Abbreviations can be translated into full words, pronounced as written or pronounced letter by letter. There are also many contextual difficulties. For example kg can be read as either *kilogram* or *kilograms* depending on preceding number, St. can be read as *saint* or *street*, Dr. as *doctor* or *drive* and ft. as *Fort*, *foot* or *feet*. In some cases, the adjacent information may be efficient to find the correct conversion, but to prevent bad conversions the best solution in some cases may be the use of letter-to-letter interpretation. Innumerable abbreviations for company names and other associated things exist and they can be pronounced in different ways. For example, N.A.T.O. or RAM are always pronounced as written and SAS or ADP letter-by-letter. Some abbreviations like MPEG as *empeg* are pronounced irregularly.

Special characters and symbols, such as '&', '%', '$', '/', '+', '-', also cause special kind of problems. In some cases the word order must be changed. For example, *$81.60* must be expanded as *eighty-one dollars and sixty cents* and *$200 million* as *two hundred million dollars*, not as *two hundred dollars million*. The expression '1-3' may be expanded as *one minus three* or *one three*, and character '&' as *et* or *and*. Also special type of characters and character strings in for example web-sites or e-mail messages necessarily is expanded with some special rules. For example, character '@ ' is commonly converted as *at* and e-mail messages can contain character strings, like as some information about header, which may be omitted. Several languages may also include special non ASCII characters, like accent markers or special symbols. The written text can also be constructed in many ways, like in many columns and pages such as in a normal newspaper article. This may cause unconquerable problems especially with optical reading machines.

1.5.2 Pronunciation

The second job is to find out the correct pronunciation for different contexts in the text. Some words, called *homographs*, can cause the most difficult problems in TTS systems. Homographs are having the same spellings but they are different in meaning and usually in pronunciation (e.g. fair, lives). For example the word *lives* is pronounced differently in sentences "Two *lives* were lost" and "One *lives* to eat". And also there are other many words for example *lead*, has different pronunciations when they are used as a verb or noun, and between two noun senses (He followed his *lead* /

He covered the hull with *lead*). With these kinds of words some semantically information is required to achieve correct pronunciation.

The pronunciation of a word can also be different due to contextual effects. This is very easy to find when we compare the phrases '*the end'* and '*the beginning'*. The pronunciation of word '*the'* depends on the initial phoneme in the following word. Compound words are also being problematic. For example the characters 'th' in father and penthouse is pronounced differently. Some sounds can also act as either voiced or unvoiced in different context. For example, phoneme /s/ in word *dogs* is voiced, but it is unvoiced in word *cats*.

To find correct pronunciation for proper names, especially when they are borrowed from some other languages, is usually one of the difficult task for any text to speech system. Some common words, such as Nice and Begin, are ambiguous in capital letter context, including sentence initial position, headings and single text. For example, the sentence '*Nice is a nice place'* is very problematical because the word *Nice* can be pronounced as either /niis/ or /nais/. Some names and places also have specialized pronunciations, for example Leicester and Arkansas. For correct pronunciation, these types of names/words may be included in a particular exception dictionary but it is bitter fact that there is no way to create such a database of all proper names in the world.

1.5.3 Prosody

To find correct intonation, stress, and duration from the written text is probably the most challenging and difficult problem for years to come. These features together are known as prosodic or suprasegmental features and can be considered as the rhythm, melody and emphasis of the speech at the perceptual level. The intonation emphasizes how the pitch pattern or fundamental frequency varies during speech.

The prosody of continuous speech depends on several different aspects, such as the meaning of the sentence and the speaker characteristics and emotions. Unfortunately, written text commonly contains a little information of these features and some of them change dynamically during speech. However, with some particular control characters this information may be provided to a speech synthesizer.

Timing at sentence level or grouping of words into phrases correctly is difficult because prosodic phrasing is not always marked in text by punctuation, and phrasal

accentuation is not almost marked. If there is not any breath pause in speech or if they are at wrong places, the speech may be very unnatural or even the meaning of the sentence may be misunderstood. For example, the input string "John says David is a liar" can be spoken as two different ways giving two different meanings as "John says: David is a liar" or "John, says David, is a liar". In the first sentence, David is a liar, and in the second one the liar is John.

1.6 Structure of the Thesis

The rest of thesis is organized in the following order:

Chapter-2: The brief history in evaluation of text to speech and technique has been used in past in explained in this chapter.

Chapter-3: In this chapter, the gap has been found in the past text to speech system is mention. The objective of this thesis a methodology to improve the text to speech system for numerals is described in this chapter.

Chapter-4: Details of our proposed number to speech system with a flow chart and algorithm is explained in this chapter.

Chapter-5: In this chapter, the snapshots of proposed system and test cases related to system are shown.

Chapter-6: This chapter concludes our thesis work and mention the future scope of our work.

Chapter 2

Literature Review

2.1 Indian Languages

Language is not only a rule-governed system of communication but also a phenomenon that to a great extent structures our thought and defines our social relationships in terms of both power and equality. India is rich in languages. There are a quite a number of languages spoken in India. Some of these languages are accepted nationally while others are accepted as dialects of that particular region.

The Indian languages belong to four language families namely *Indo-European, Dravidian, Austro-Asiatic (Austric) and Sino-Tibetan.* Majority of India's population are using Indo-European and Dravidian languages. The former are spoken mainly in northern and central regions and the latter in southern India. Some ethnic groups in Assam and other parts of eastern India speak Austric languages. People in the northern region speak Punjabi language. The written forms of language or scripts come from an ancient Indian script called Gurmukhi.

2.2 Evaluation of TTS

Let us start, with the understanding of the progression of text to speech (TTS) system [6]. In 1779, the Danish scientist Christian Kratzenstein, working at the Russian Academy of Sciences, built models of the human vocal tract that could produce the five long vowel sounds they are [a], [e], [i], [o] and [u].In 1791, an Austrian scientist developed a system based on the previous one included tongue, lips and "mouth" made of rubber and a "nose" with two nostrils which was able to pronounce consonants. In 1837, Joseph Faber developed a system which implemented Pharyngeal Cavity, used for singing. It was controlled by keyboard. Bell Labs Developed VOCODER, a clearly intelligible. Keyboard-operated electronic speech analyzer and synthesizer. In 1939, Homer Dudely developed VODER which was an improvement over VOCODER. The Pattern Playback was built by Dr. Franklin S. Cooper and his colleagues at Haskins Laboratories. First Electronic based TTS system was designed in 1968. Concatenation Technique was developed by 1970's. Many computer operating systems have embedded speech synthesizers since the early 1980's. From 1990's, there was a progress in Unit Selection and Diphone Synthesis.

2.3 Text to Speech Systems for Indian Language

Speech synthesis involves algorithmically converting an input text into speech waveforms and some previously coded speech data. Speech synthesizers can be classified by the size of the small speech units they concatenate and also by the method apply to code, store and synthesize the speech. The choice of synthesis method is influenced by the size of the vocabulary because they must model all possible utterances. Unlimited-text systems are generally more complex, and yield lower quality speech than limited-text systems. Thus it is seen that various different techniques exist to convert a given text to speech and that each such system has its own advantages and disadvantages [7].

In India, to help the visually impaired, vocally disabled and day to day increasing applications of speech synthesis has necessitated the development of more and more innovative text-to-speech (TTS) system. Some of the already developed TTS are described below.

2.3.1 Dhvani- Indian Language Text To Speech System

Dhvani [8] [9] system has won FOSS India award in 2008. The main characteristics of this system are as follows.

- Dhvani system has been designed by Simputer trust headed by Dr. Ramesh Hariharan at Indian Institute of Science Bangalore in year 2000.
- Dhvani is a Text To Speech System specially developed for Indian languages.
- It uses diphone concatenation algorithm.
- Currently this system has Hindi, Malayalam, Kannada, Bengali, Oriya,
- Punjabi, Gujarati, Telugu and Marathi modules.
- All sound files are *'gsm'* compressed files that stored in the database.
- It has different modules for every language.
- It is based on the observation that a direct grapheme to phoneme mapping exits for all Indian languages in general.
- It is an attempt in India to cover all Indian languages under a single framework.
- In this system each language requires a Unicode parser.

2.3.1.1 Architecture of *Dhvani* System

The architecture of ***Dhvani*** system is shown in figure 2.1. It includes the following components.

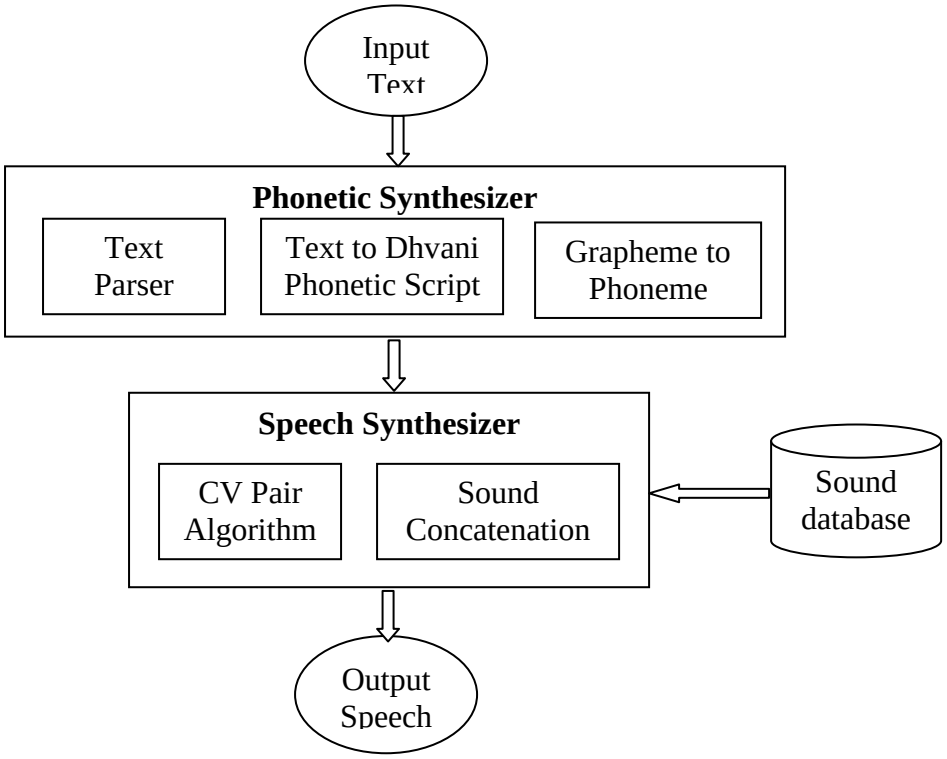

Figure 2.1 Dhvani- Indian language Text to Speech System [8]

- **Text Parser:** Each language requires a Unicode parser. It simply parses the input text.
- **Text to Dhvani Phonetic Script:** This makes ***Dhvani***, language independent. In this phase any Unicode text is converted to a common script. This script is further act as the input to the speech synthesizer.
- **Grapheme to Phoneme Conversion:** This converts the grapheme to phoneme. The phonetic description is syllable based. In this eight kinds of sounds are allowed.
- **Sound Database:** All sound files are *'gsm'* compressed files that stored in the database
- **Speech Synthesizer:** Speech synthesizer takes phonetic script and with the help of CV (consonant-vowel) pair algorithm and sound concatenation component concatenates the sound files to produce speech.

- **Limitations of *Dhvani***: This speech engine has simply concatenated basic sound units at pitch periods and plays in output but do not made any attempt to do prosody on the output. Inserting prosody in the current system is a task for the future and it is tested on currently around 1MB size of database.

2.3.2 *Shruti:* An Embedded Text To Speech System for Indian Languages

Mukhopadhyay *et al.* [10] have developed ***Shruti*** system in year 2006 at Indian Institute of Technology Kharagpur. The main characteristics of this system are as follows.

- It is a text-to-speech system, which has been developed using a Concatenative speech synthesis technique.
- This is the first text-to-speech system built specifically for two of the Indian languages, namely Bengali and Hindi.
- The system, however, has been designed in such a manner that it can be very easily extended to other Indian languages as well.
- For ease of use and portability, the system has been ported to two existing handheld platforms running two processor families, namely, the Compaq iPaq and the Casio Cassiopeia.

2.3.2.1 Architecture of *Shruti* system

A schematic diagram of the speech synthesis system is shown in figure 2.2. The system consists of two main blocks, block-A and block-B, where block-A is the language-dependent block and block-B is the Indian language phonetic synthesizer (ILPS). The block-A consists of an input device, a natural language processing (NLP) unit and an intonational and prosodic rule base. The system accepts input text in iTrans, a notation for expressing Indian language scripts through Roman characters. A language-dependent prosodic and intonation rules are included in this block primarily as a knowledge base. The NLP in block-A comprises a phoneme parser, which uses either a phonological dictionary, syllable and word marker or a linguistic rule base to produce an output phonemic string. The NLP simply takes the text and analyses it to perform grapheme to phoneme conversion. The grapheme is the actual text, whereas the phoneme is a token that directly maps to a signal unit in the voice database or in the partneme dictionary.

The block-B is the synthesizer. The output speech is produced by taking the output string of phonemes (in ILPS symbols) and information for intonation and prosody from the basic rule base as input. The concatenation unit reads the phonemes one by one from the output phoneme string. For each read phoneme, the unit fetches the corresponding sound unit that the phoneme maps to from the voice database. To make the speech output more natural, the speech units are algorithmically modified before being concatenated. Finally, intonational and prosodic rules are applied to produce the final output speech [10] [7].

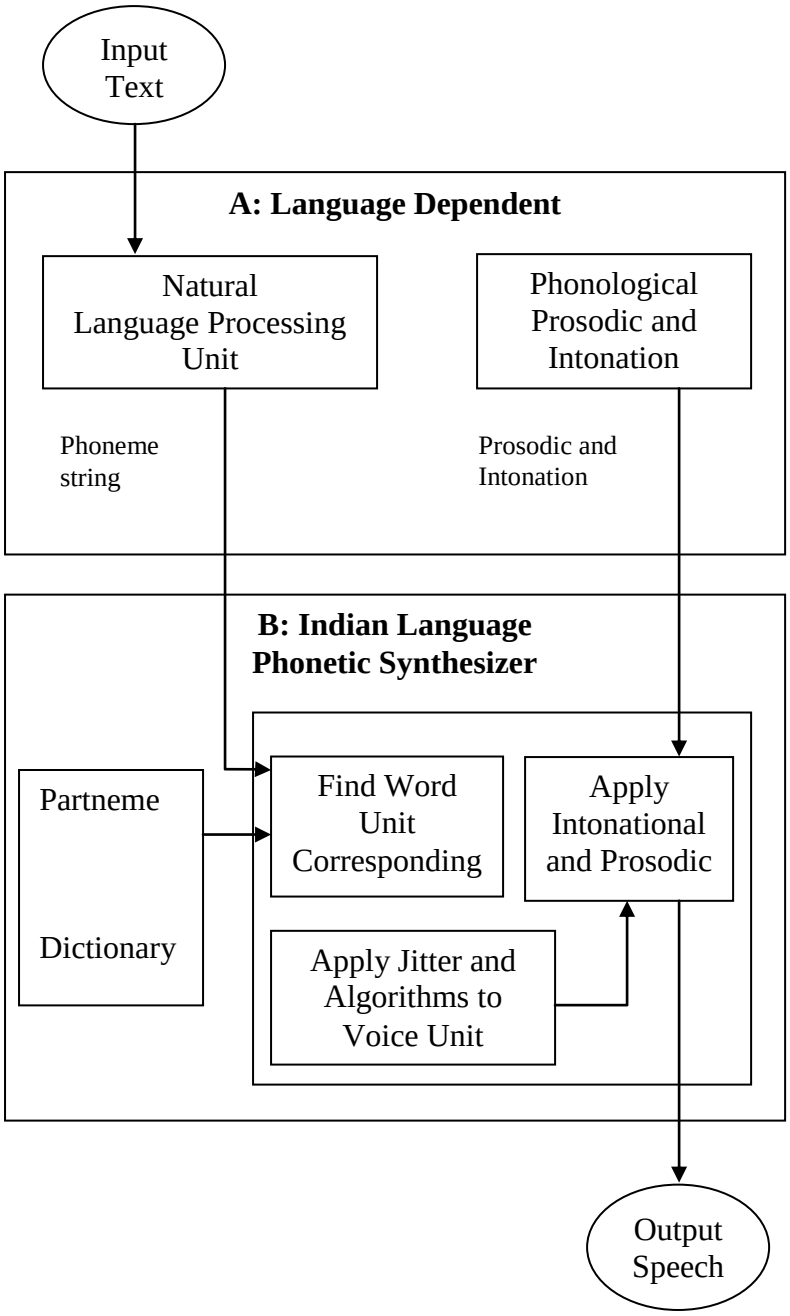

Figure 2.2 Schematic Diagram of Shruti System [10]

2.3.3 HP Labs India TTS System

HP Labs India developed a Hindi TTS system based on the open source TTS framework, Festival. 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 [11].

2.3.3.1 Architecture of HP Lab India System

As part of the LLSTI initiative, a generic grapheme [12] into phoneme conversion system has been developed at HP Labs India. The architecture of the grapheme to phoneme (G2P) is shown in Figure 2.3. This G2P framework has then been customized for Punjabi. It is composed of following units.

- **Grapheme to Phoneme Conversion Engine:** It is a language independent engine, which requires language dependent information in the form of lexicon, rules and mapping.
- **G2P Rule Base:** It gives grapheme to phoneme conversion rule.
- **Character Phone Mapping*:*** It provides the character phone mapping.
- **Exception lexicon:** It contains the phonetic transcription of the exceptions.

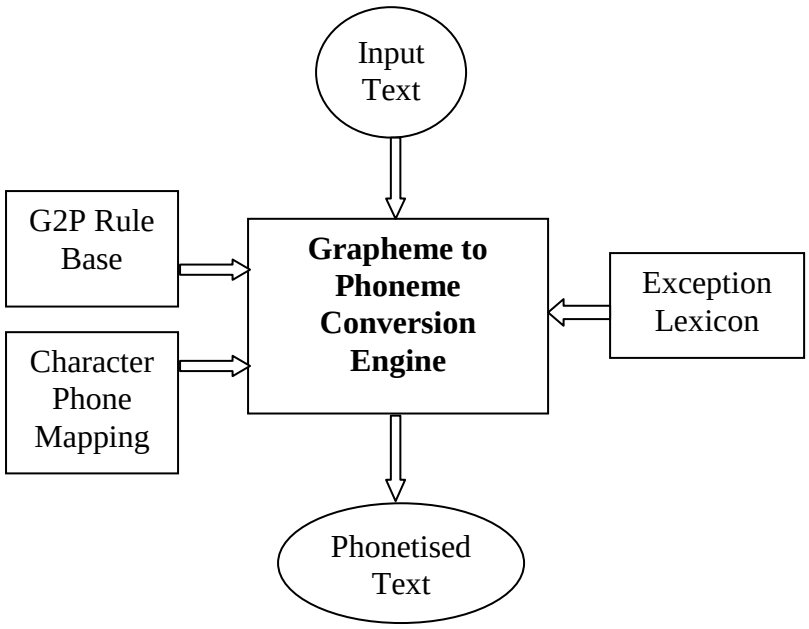

Figure 2.3 HP Lab India TTS System Architecture [12]

2.3.4 *Vani*: An Indian Language Text to Speech Synthesizer

Vani [13] is an Indian Language text to speech synthesizer developed at IIT Bombay, India. Generally, all existing TTS system allows user to specify what is to be spoken but does not give any control on how it has to be spoken. In ***Vani***, a new encoding scheme has been introduced called *vTrans*. A *vTrans* file makes a person to encode what text he wants to be spoken and also the way that text to be spoken. A signal processing module is then used to bring out this speech by making appropriate variations to the sound database. *vTrans* is an XML document that contains a head and a body. In the head part of XML document the parameters and styles are defined. These parameters are pitch, volume and duration. The body section contains several tags. They may be nested, but text is put only in the innermost of the tags. The text within the tags is iTrans encoded. The tags may be any of the parameters defined in the head section. The attributes assigned to these tags determines which style to use and allow the user to scale and translate the function to be used as required.

2.3.4.1 Architecture of *Vani* System

The architecture of ***Vani*** system is given in Figure 2.4. It is based on fact that phonemes can be decomposed into fract-phonemes (a very small segment whose continuous repetition forms a vowel) for all vowels. These fract-phonemes are very small in size, and hence a good choice for a base unit.

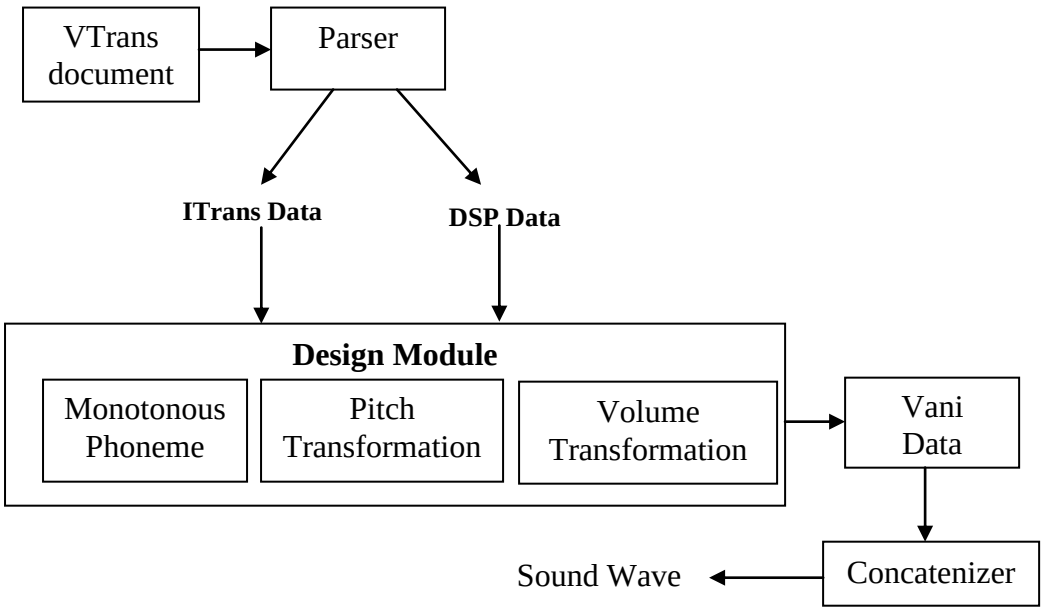

Figure 2.4 Vani Architecture [13]

The duration a phoneme can be generated by taking into account the volume and frequency curves using fract-phoneme as per the following procedure.

- Virtual duration is calculated.
- Monotonous phoneme of virtual duration is generated.
- Pitch transformation is applied.
- Volume transformation is applied.

2.4 Architecture of TTS

The text to speech conversion may be done in different steps: Text pre-processing, text analysis, text phonetization, prosody generation and then the speech synthesis using various algorithms [14]. The steps followed to convert text to speech are described in Figure 2.5

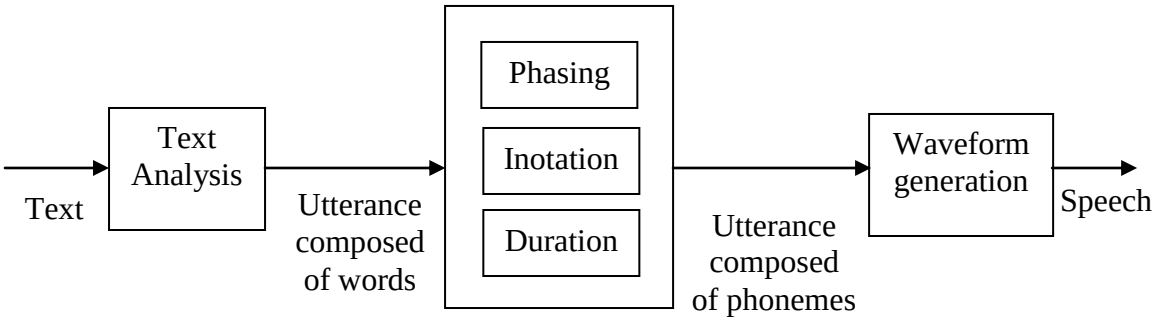

Figure 2.5 Steps in TTS System

2.4.1 Tokenization/Text pre-processing

Text pre-processing also called text normalization. A normal text contains many abbreviations, acronyms, special characters. So they must be processed to convert them to segments called words or standard text format [15].

2.4.2 Syllabification

Combination of phonemes gives rise to next higher unit known as syllables which is one of the most important units of a language [16]. A syllable must have a vowel called its nucleus, whereas presence of consonant is optional. In Punjabi language, seven types of syllables are recognized .These syllable types are: V, CV, VC, VCC, CVC, CCVC and CVCC; where V and C represent vowel and consonant respectively. Out of these seven types of syllable, occurrence of last two syllable types having sound clusters, is rarely found in Punjabi.

2.4.3 Speech synthesis/Waveform production

The speech can be synthesized by concatenating different pieces of recorded speech from the database. Different methods can be employed for synthesized speech. Synthesis by rebuilding speech using parametric models like the source-filter model for the speech production process. Acoustic frequency domain models or articulatory models can perform this process. Speech can be synthesizes using rule based, concatenation methods [17]. During speech synthesizes various parameters must be considered for the speech like frequency, pitch, inotation, stress, noise levels. These parameters vary over time to create a wave of speech.

2.5 Formant Synthesis

Formant synthesis assumes that the vocal tract transfer function should be satisfactorily modeled by simulating formant frequencies and formant amplitudes. The synthesis thus contains the artificial reconstruction of the formant characteristics to be generated. This is performed by exciting a set of resonators by a voicing source or noise generator succeed to get the desired speech spectrum, and by controlling the excitation source to simulate either voicing or voicelessness. The addition of a set of anti-resonators furthermore allows the simulation of nasal tract effects, fricatives and plosives. The specification of about 20 or more such parameters can lead to a satisfactory restitution of the speech signal. The advantage of this technique is that its parameters are highly correlated with the production and propagation of sound in the oral tract. The main current drawback of this approach is that automatic techniques of specifying formant parameters are still largely unsatisfactory, and that consequently, the majority of parameters must still be manually optimized [18].

The formant synthesis doesn't use any human speech samples but relies on rules written by linguists to generate the parameters that will permit the synthesis of speech, and to deal with the transition from one phoneme to another, that is, the co articulation. To write the rules, linguists have studied spectrograms and derived the rules of evolution of formants. However we do not yet know the optimal rule to do this [19]. Moreover, the speech waveform is naturally produced in such a complex process that, currently, rules can only model the features of the speech waveform. Therefore, the synthesized speech has an artificial, robotic sound, and the goal of naturalness is not reached. However, the rule-based synthesized speech is very intelligible, even at high speeds, which is quite useful for visually impaired for fastest

navigating systems using a screen reader. Moreover, when memory and processing costs are limited, such as in embedded systems, these synthesizers are more interesting because they don't have a database of speech samples. The formant synthesis approach has been implemented in MITalk [20] [21], in KlatTalk [22], and in DECTalk [23].

2.6 Articulatory Synthesis

Articulatory synthesis generates speech by direct modeling of the human articulator behavior, so in principle it is the most accurate method to meet the goal, the generation of high quality speech and most difficult techniques to implement. The articulatory control parameters include lip aperture, lip protrusion, tongue tip position, tongue tip height, tongue position and tongue height [24]. There are two difficulties in articulatory synthesis. The first difficulty is acquiring data for articulatory model. This data is usually derived from X-ray photography. X-ray data do not differentiate the masses or degrees of freedom of the articulators [19]. The second difficulty is to find a balance between a highly accurate model and a model that is easy to design and control. In general, the results of articulatory synthesis are not as good as the results of formant synthesis or the results of Concatenative synthesis.

2.7 Concatenative Synthesis

The main limitation of formant synthesis and articulatory synthesis is not so much in generating speech from parametric representation, but the difficulty is in finding these parameters from the input specification that was created by the text analysis process. To overcome this limitation, Concatenative synthesis follows a data driven approach. Concatenative synthesis generates speech by connecting natural, prerecorded speech units. These units can be words, syllables, half-syllables, phonemes, diphones or triphones. The unit length affects the quality of the synthesized speech. With longer units, the naturalness increases, less concatenation points are needed, but more memory is needed and the number of units stored in the database becomes very numerous. Less memory is needed if units are shorter but the sample collecting and labeling techniques become more complex [25].

The most widely used units in Concatenative synthesis are diphones. A diphone is a unit that starts at the middle of one phone and extends to the middle of the following one. Diphones have the advantage of modeling co-articulation by including the

transition to the next phone inside the diphone itself. The full list of diphones is called diphone inventory, and once determined, they need to be found in real speech. To build the diphone inventory, natural speech must be recorded such that it includes all the phonemes within all possible contexts is called allophones, then diphones must be labeled and segmented. Once the diphone inventory is built, the pitch and duration of each diphone need to be modified to match the prosodic part of the specification.

2.8 Unit Selection Synthesis

In Concatenative synthesis, diphones must be modified by signal processing methods to produce the desired prosody. This modification results in artifacts in the speech that can make the speech sound unnatural. Unit selection synthesis (also, called corpus-based Concatenative synthesis) solves this problem by storing in the unit inventory multiple instances of each unit with varying prosodies. The unit that matches closest to the target prosody is selected and concatenated so that prosodic modifications needed on the selected unit is either minimized or not necessary at all. Since multiple instances of each unit are stored in the unit inventory, a unit selection algorithm is needed to choose the units that best match the target specification. This selection is based on minimizing two types of cost functions, which are target cost and join cost. In the case of automatic unit selection, the co-articulatory influence isn't limited to the last phoneme.

The database is much larger (1-10 hours) and comprises several occurrences of each acoustic unit, captured under various contexts (like its neighboring phonemes of course, but also its pitch, its duration, its position in the syllable, etc.). As a result, the sequence of phonemes to synthesize leads to a lattice of acoustic units, in which the best corresponds to the expected contexts (prosody, phonetics, etc) but also minimizes the spectral and prosodic discontinuities. Consequently, automatic unit selection requires much less modification of the speech units, which leads to an overall quality of the synthesized speech much more natural than with diphones based synthesis. Apart from this naturalness, unit selection techniques have several disadvantages. They rely on a very large database, which implies, on the one hand, considerable development time and cost to collect and label the data, and on the other hand, large memory resource requirements to store the data. The second drawback is incorrect labeling and occurrence of unseen target contexts lead to fragments of synthesized speech of extremely poor quality. This phenomenon of unseen contexts may well

never be fully overcome with Concatenative synthesis as [26] suggest that rare events will always occur in language.

2.9 Hidden Markov Model Synthesis

In unit selection synthesis, multiple instances of each phone in different contexts are stored in the database. To build such a database is a time consuming task and the database size increases in an enormous way. Another limitation of the Concatenative approach is that it limits us to recreate what we have recorded. An alternative is to use statistical parametric synthesis techniques to infer specification to parametric mapping from data. These techniques have two advantages: firstly, less memory is needed to store the parameters of the models than to store the data itself. Secondly, more variations are allowable for example; the original voice can be converted into another voice. One of the most usable statistical parametric synthesis techniques is the hidden Markov model (HMM) synthesis. It consists of two main phases, the training phase and the synthesis phase. At the training phase, it should be decided which features the models should be trained for. Mel frequency cepstral coefficients (MFCC) and their first and second derivatives are the most common types of features used. The feature are extracted per frame and put in a feature vector. The synthesis phase consists of two steps:

Firstly, the feature vectors for a given phone sequence have to be estimated. Secondly, a filter is implemented to transform those feature vectors into audio signals. The quality of the HMM generated speech is not as good as the quality of the speech generated from unit selection synthesis. The modeling accuracy can be improved by using hidden semi-Markov models (HSMMs) [29], trajectory HMMs [30], and stochastic Markov graphs [31]. Suggestion in this paper [32] integrates the Harmonic plus Noise Model (HNM) into the Hidden Markov Model-based Speech Synthesis System (HTS). This integration leads to a Text- To-Speech system that requires smaller development time and cost, in comparison with the usual state-of-the-art Text-To-Speech systems typically based on automatic selection and synthesis of sub-words units (e.g., diphones), while also producing a better quality speech output (compared to HTS alone). This quality enhancement is achieved by replacing the source filter modeling approach typically used in HTS with the HNM model, which is known for being able to synthesize natural sounding speech under various prosodic modifications.

Chapter 3

Problem Statement

3.1 Gap and Scope Analysis

Over the last few years there has been a great development of the quality of the speech produced with text to speech. Many people think that synthetic speech as it is also called sounds like robots from older movies. The truth is though that some voices almost sound like recorded speech.

The greatest improvements when it comes to natural speech were during the last 10 years. The first voices used for ReadSpeaker back in 2001 were produced using Diphone synthesis. The voices are sampled from real recorded speech and split into phonemes, a small unit of human speech. This was the first example of Concatenation synthesis. However, they still have an artificial/synthetic sound. We still use diphone voices for some smaller languages and they are widely used to speech-enable handheld computers and mobile phones due to their limited resource consumption, both memory and CPU.

It wasn't until the introduction of a technique called Unit selection, that voices became very naturally sounding. This is still concatenation synthesis but the used units are larger than phonemes, sometimes a complete sentence. We use different providers for different languages to always assure we can offer the best voices available for that language.

Before the development of this system, the text to speech system was not developed for numerals to convert into Punjabi language. In the system or software developed before, to develop this system firstly we have to record the sound like .wave files in human voice then these files were used to convert the number into speech.

3.2 Proposed Objective

- To develop a system for converting numerals to speech without recording sound files.
- A system that uses phonemes to the corresponding numbers; mapping of these phonemes with numbers are being stored either in the database or a file.

3.3 Methodology

This thesis work presents the steps followed for converting text to speech for Punjabi (Gurmukhi) language and the algorithm used for it.

- Text To Speech synthesis (TTS) is an online application to convert the text into speech in Punjabi language.
- The text to speech conversion system enables user to enter numbers as input and as output it gets sound in Punjabi language.
- In proposed system, input first pass into text pre-processing and verify in normalization phase of text to speech system.
- Tokenizer is an essential phase of text to speech system; it breaks the big number into small tokens.
- The phonemes are stored in database. System fetches the phonemes corresponding to tokens.
- We have used a library file of the Microsoft i.e. Speechlib.dll file which produces the speech using the function SpeakVoice defined in the library and make the computer able to speak by itself.
- Speech synthesizer speaks these phonemes.
- Though the user gets its output but still conversion of text to speech is not an easy task. It involves a great deal of work to be performed perfectly. It is difficult to create database of number of phonemes, syllables, words for a particular language.

Chapter 4
Implementation

There are two main approaches for solving the TTS paradigm. The first one uses basic units, which are either recorded speech segments or parameters representing these segments. These units may correspond to words, phonemes or even sub phonemes, as used in this thesis. This speech generation method is called Concatenative TTS (CTTS). In this approach, speech is generated by concatenating the best compatible segments according to certain concatenation rules. By this approach, generated speech inherently possesses natural quality. However its quality depends on the size of the recorded database, as high-quality CTTS needs an extensive database.

In thesis work, A Text to Speech system has been implemented that work only for numerals and convert those written numeral numbers into Punjabi speech.

4.1 Punjabi Phonemes for the corresponding number

Gurmukhi has its own set of numerals that behave exactly as Hindu-Arabic numerals do. These are used extensively in older texts. In modern contexts, they have been replaced by standard Latin numerals. The schwa ("ə"), used in this section, makes a sound like the unstressed "a" in "about".

Numeral	Name	Number
0	ਸਿਫਰ *sifər*	Zero
1	ਇੱਕ *ik*	One
2	ਦੋ *do*	Two
3	ਤਿੰਨ *tin*	Three
4	ਚਾਰ *char*	Four
5	ਪੰਜ *pənj*	Five
6	ਛੇ *sche*	Six
7	ਸੱਤ *sət*	Seven
8	ਅੱਠ *əṭ*	Eight
9	ਨੌਂ *nãũ*	Nine
10	ਦਸ *dəs*	Ten

Table 4.1 Punjabi phoneme for numbers 0 to 10

Numeral	Name	Number
100	ਸੌ *sãũ*	Hundred
1000	ਹਜਾਰ h*əzəər*	Thousand
100000	ਲੱਖ l*əəkh*	Lac
10000000	ਕਰੋੜ crh*ãũũrh*	Crore
1000000000	ਅਰਬ *əũrəbh*	Billion
100000000000	ਖਰਬ kh*ərəb*	Hundred Billion

Table 4.2 Punjabi phoneme for numbers greater than 100

4.2 MSDN Library

This work has been performed in the Microsoft Visual Studio 2010 in Asp.net. We have used the Microsoft's msdn library which will make the computer be able to speak the required speech. This dll library is available at www.msdn.com.

The Microsoft Speech Platform consists of a Software Development Kit (SDK), a Runtime, and Runtime Languages (language packs that enable speech recognition or text-to-speech for a specific language) that you can redistribute with your applications. The Microsoft Speech Platform SDK provides a comprehensive set of development tools for managing the Speech Platform Runtime in voice-enabled applications. Add the ability to recognize spoken words (speech recognition) and to generate synthesized speech (text-to-speech or TTS) to enhance users' interaction with your applications.

The Speech Platform SDK includes the *Microsoft.Speechlib* namespace to support authoring speech applications. The Speech Platform Runtime includes this library at runtime. Also included in the SDK, the Microsoft Grammar Development Tools provide a comprehensive set of command-line applications with which you can validate, analyze, and tune your grammars for speech recognition.

4.3 Flow chart of Designed system

The numeral text is given as input in the textbox; the given input is then pre-processed. The input is then broken up into different parts using the algorithm used in this work. Now the different parts are assigned an indices, when whole data is get processed then the corresponding phonemes are fetched from the database and then after performing all tasks, the Punjabi phonemes generated for the input text are given as output to speech synthesizer engine, then the output speech is get played into Punjabi language.

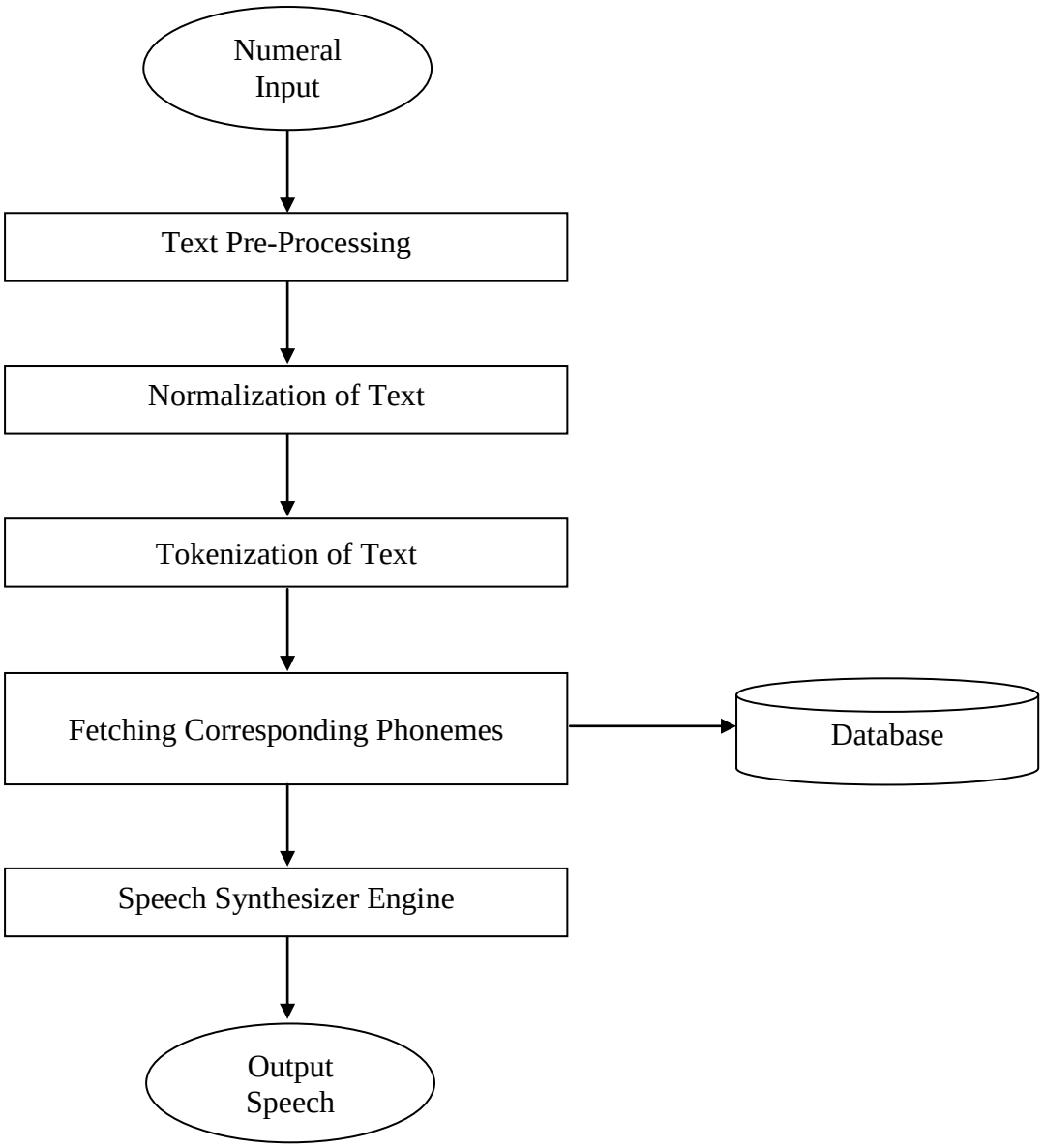

Figure 4.1 Flow Chart of System

4.4 Details of Designed system

Steps performed in the system are explained below.

4.4.1 Text pre-processing

Text pre-processing is an essential part of text to speech system. User enters the text into text area that he wants to get spoken and submit it by clicking speak button. Now system converts the input text into number and now the next step is normalization.

4.4.2 Normalization of text

Normally human has the habit of making mistakes while writing. User can give the input wrong by giving some space or giving a comma sign between the digits or special character like '$', '%', ';'. So this will make a problem for the system. Hence in this step, we will check these errors and make sure that the given input is correct.

4.4.3 Tokenization of text

System cannot speak the complete number instantly because it will not recognize the actual position of individual digit in input number. For example input number is 543, system cannot understand that digit 5 is at hundredth position in number 543. To make the system understand, divide input number into different parts. In this step, the number will be divided into a number of tokens. System will find that which number is at which position i.e. what is present at the tens place, hundred place, thousands place, lakhs place etc. these are assigned a token. These tokens are essential for our output speech.

The algorithm for the tokenization is as follows:

Algorithm 4.1 Tokenization

Procedure Tokenizer (Input_number)

1. Find the digit at tens place
2. Find the digit at hundred place
3. Take an array Num[index]
4. ***While*** ∃ More tokens ***do***
5. num[index] ← Input_number % 100
6. index ← index+1
7. num ← num/100

The algorithm 4.1 takes the numeral as input. If input number is less than 100 then system directly fetches the corresponding phoneme from the database. If number is equal to or larger than 100 then find the position of each digit. In this method we have used the strategy that the unit and tenth place digits are always taken to together.

To find the digits at tenth place, we find the remainder by doing number modulo 100. Now number will be divided by 100 and this resultant number is modulo by 10. The digit got at this position is the hundredth place digit. Now this resultant number is divided by 10. If resultant number is greater than 0 (means there more tokens are available in the system), take an integer array num with an initial index 0.

Now a number is modulo by 100 and place resultant number into array num at current index and increment the index by one. Number is divided by 100 that create the new token. Repeat this process till the new token is found.

4.4.4 Fetching corresponding phonemes

Index of array num gives the actual position of digit in input number. Now the question is that how system will speak h*əzəər,* l*əəkh* and crh*ãũũrh* in Punjabi language. It will be resolved using math function. Now fetch the token number from the array in decreasing order of index. Now phoneme will be searched into the database according to the token present at index and are fetched from the database. Now this phoneme is given as input to the speech synthesizer engine and decrement the index by one. Repeat this process till the index is not 0.

4.4.5 Speech synthesizer engine

Finally the phonemes are played by the speech engine and the required output speech is generated by the system for the user.

Chapter 5

Result and Testing

5.1 Snapshots of Designed TTS

In Figure 5.1, user enters the number in correct form then system speaks the input number into Punjabi language.

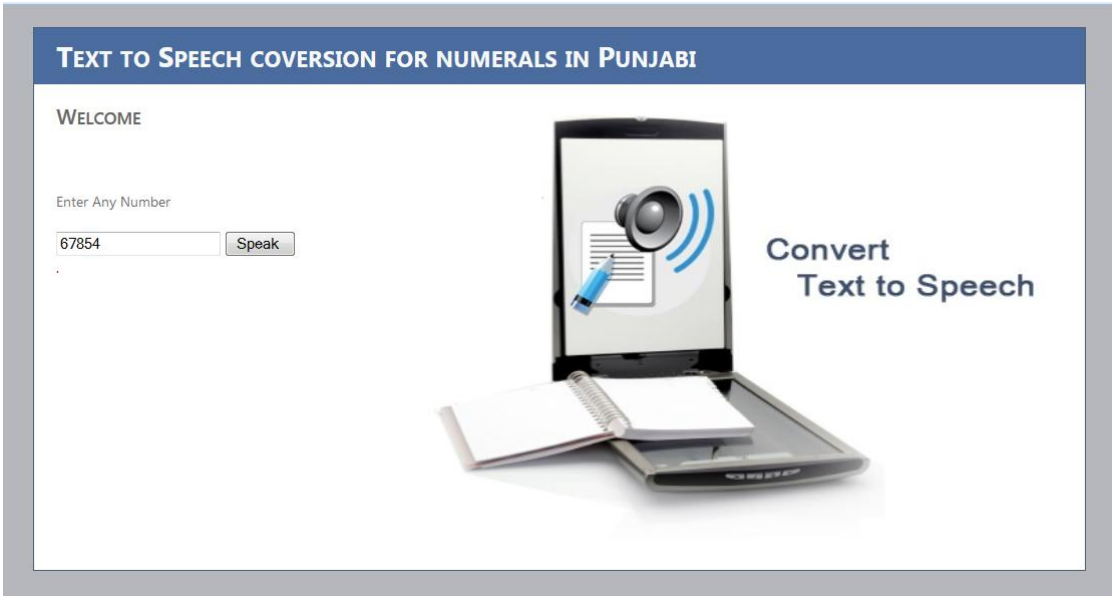

Figure 5.1 Correct output Speech

User enters the number with space in between the digits so system identify the incorrect form of input number and gives an error message *"Enter number without space and comma"* shown in Figure 5.2

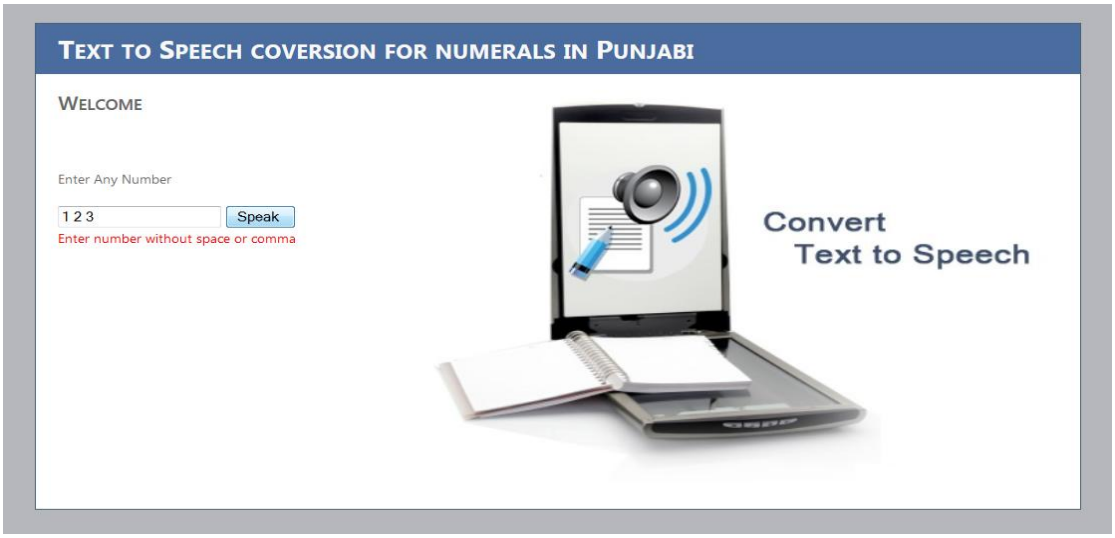

Figure 5.2 Error message due to space in input number

User enters the number with comma in between the digits so system identify the incorrect form of input number and gives the error message *"Enter number without space and comma"* shown in Figure 5.3

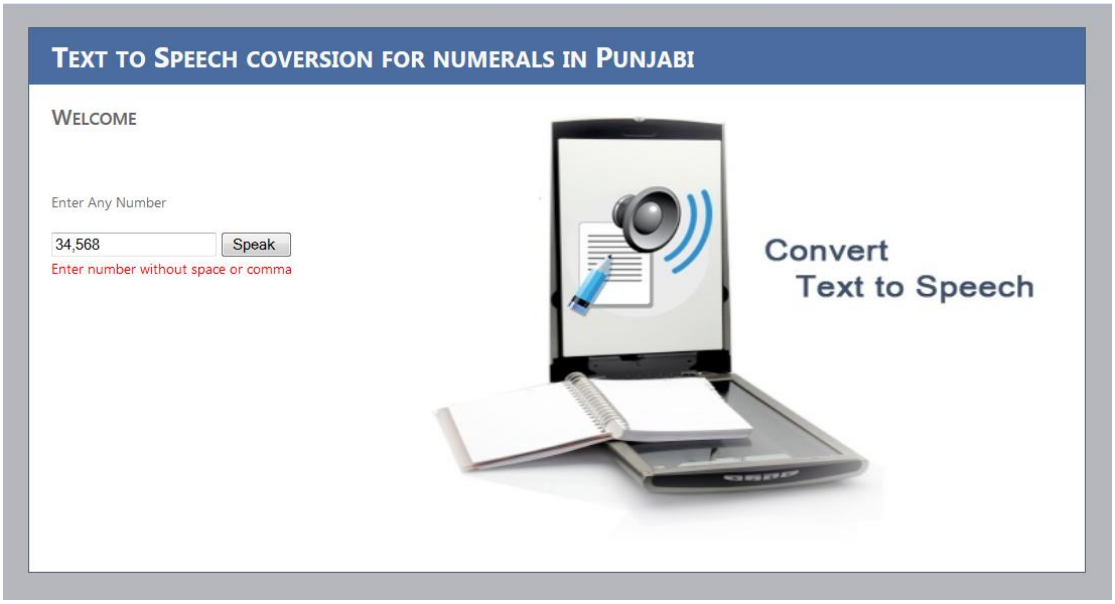

Figure 5.3 Error message due to comma in input number

In Figure 5.4, user enters the number with alphabet by mistake then system recognizes the alphabet and alert user with an error message *"Alphabet is not allowed"*.

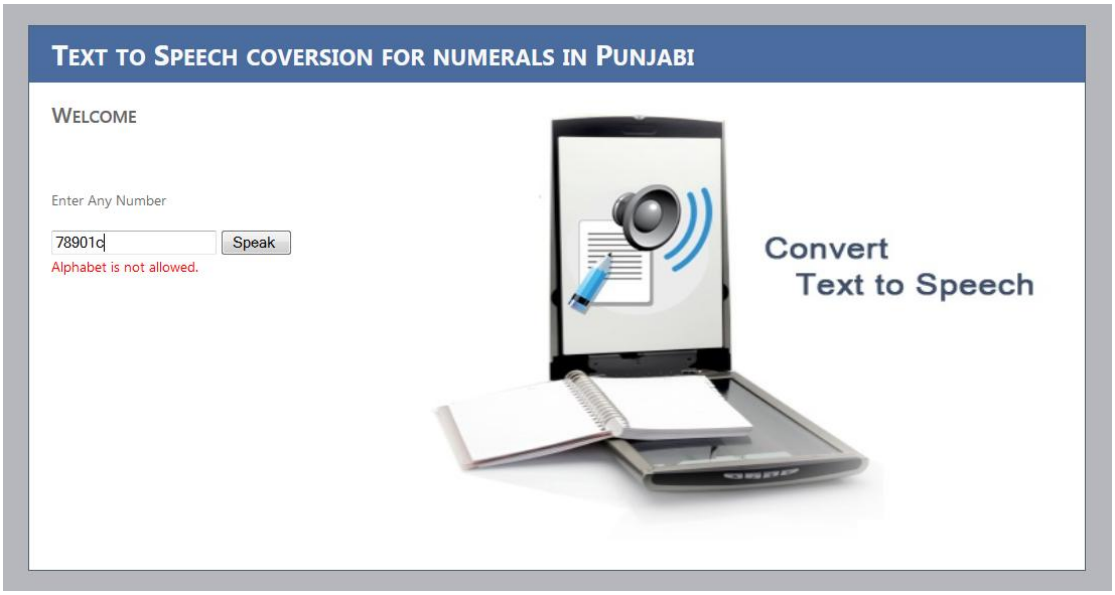

Figure 5.4 Error due to alphabet in input number

5.2 Types of Testing

- **Acceptance Testing:** Formal testing conducted to determine whether or not a system satisfies its acceptance criteria and to enable the customer to determine whether or not to accept the system. It is usually performed by the customer.
- **Alpha Testing:** Type of testing a software product or system conducted at the developer's site. Usually it is performed by the end user.
- **Beta Testing:** Final testing before releasing application for commercial purpose. It is typically done by end-users or others.
- **Black box Testing:** A method of software testing that verifies the functionality of an application without having specific knowledge of the application's code/internal structure. Tests are based on requirements and functionality. It is performed by QA teams.
- **White box Testing:** Testing technique based on knowledge of the internal logic of an application's code and includes tests like coverage of code statements, branches, paths, conditions. It is performed by software developers.

5.3 Test Cases of Designed TTS

Sr. No.	Task	Input	Expected Output	Actual Output	Result
1.	Enter the number	Number with space	Display error message	Display error message	Pass
2.	Enter the number	Alphanumeric input or alphabet only	Display error message	Display error message	Pass
3.	Enter the number	Entered the symbols like (. % & * / ; : =)	Invalid input	Invalid input	Pass
4.	Enter the number	Not entered any number	Display warning	Display warning	Pass
5.	All the functions of application are working well	Check all functions	Should be working well	Working well	pass

Table 5.1 Test cases of TTS

Chapter 6
Conclusion and Future Scope

6.1 Conclusion

This thesis proposed the system in which recorded sound files are not used. Phonemes have been created for corresponding numbers and these phonemes have been stored either in the database or a file. To make the computer able to speak, a library file of the Microsoft i.e. Speechlib.dll file has been included in our system. The algorithm of proposed system uses the function SpeakVoice of library to enable the speech synthesizer.

Text to speech synthesis (TTS) is a critical research and application area in the field of multimedia interfaces. Recent advances in TTS will impact is wide number of disciplines from education, business and entertainment applications to medical aids. Until recently, speech synthesis relied on models and rule-based approaches.

While this had yielded intelligible sounding speech, the voice quality was unacceptable for widespread adoption. Fortunately, there has been a major technological paradigm shift recently in how speech synthesis is done: going from rule-based to explicit data-driven methods. Recent advances in computing and corpus driven methodologies have yielded exciting possibilities for research and development in this domain yielding highly natural sounding speech.

The thesis focuses on recent advances and new paradigms in text to speech synthesis contributed by leading experts from both academia and industry from across the world.

Using the above said points in implementation chapter, we can use our text to speech convertor for the Punjabi users. The Text to speech conversion may seem effective and efficient to its users if it produces natural speech and by making several modifications to it.

6.2 Future Scope

- The proposed TTS work only for integer number but work can be extended for any real number with great precision.
- System can be used for reading an excel sheet.
- The text inputted may be written by the operator or it may be scanned paper that is converted to speech.
- We will produce a system that would be able to solve problems of various individuals in their busy life.
- Specially for the people with low vision or reading disabilities as it would help them to listen to their emails while relaxing, listen eBooks, study for exams by listening to notes.

References

[1] T. Virtanen, R. Singh and B. Raj, Techniques for Noise Robustness in Automatic Speech Recognition, Chichester, UK: John Wiley & Sons Ltd, 2012.

[2] J. Gerhart, Home Automation and Wiring, McGraw Hill Professional, March 31, 1999.

[3] R. Harper, "Inside the Smart Home: Ideas, Possibilities and Methods," in *Inside the Smart Home*, Springer London, 2003, pp. 1-17.

[4] L. Rabiner and B. w. Juang, Fundamentals of Speech Recognition, Pearson Education, April 12, 1993.

[5] Y. Y. Wang, M. Mahajan and X. Huang, "A unified context-free grammar and n-gram model for spoken language processing," in *IEEE International Conference on Acoustics Speech and Signal Processing vol. 3, pp.1639-1642*, Istanbul, Turkey, 2000.

[6] D.Sasirekha and E.Chandra, "Text to Speech : A simple Tutorial," *International Journal of Soft Computing and Engineering*, vol. 2, no. 1, pp. 275-278, March 2012.

[7] B. A, S. D, S. S and C. S, "An Indian Language Speech Synthesizer – Techniques and Applications," in *Proceedings of national systems*, Indian Institute of Technology, Kharagpur, India, 2003.

[8] T. S, "Dhvani Indian Language Text to Speech System," February 2013. [Online]. Available: http://foss.in/2007/register/slides/Dhvani__Indian_Langauge_Text_to_Speech_System-_Demo,_Adding__Language_support,_Usage_370.pdf.

[9] H. R, February 2013. [Online]. Available: http://dhvani.sourceforge.net/.

[10] M. A, C. S, C. M, L. A, D. S and B. A, "Shruti- an Embedded Text-to-speech System for Indian Languages," *IEEE Proceedings on Software Engineering*, vol. 153, no. 2, pp. 75-79, 2006.

[11] R. A. G, B. K and T. P. P, "Tools for the Development of a Hindi Speech Synthesis System," in *in 5th ISCA Speech Synthesis Workshop*, Pittsburgh, 2004.

[12] S. A. K, "A computational phonetic model for indian language scripts," in *In Constraints on Spelling Changes: Fifth International Workshop on Writing Systems*, Nijmegen, The Netherlands, 2006.

[13] J. H, K. V and S. G, "Design of a Text to Speech Synthesizer to Generate Arbitrary

Speech," in *In International Conference on Speech and Language Technology*, Noida , 2004.

[14] S. K. Thakur and K. Satao, "Study of Various kinds of Speech Synthesizer Technologies and Expression for Expressive Text To Speech Conversion System," *International Journal of Advanced Engineering Sciences and Technologies,* vol. 8, no. 2, pp. 301-305, 2011.

[15] M. H. O'Malley, "Text-to-Speech Conversion Technology," *Journal Computer,* vol. 23, no. 8, pp. 17-23, August, 1990 .

[16] M. N. Rao, S. Thomas, T. Nagarajan and H. A. Murthy, "Text-to-Speech Synthesis using syllable-like units".

[17] Y. A. El-Imam and K. Banat, "Text-to-Speech Conversion on a Personal Computer," *Journal IEEE Micro,* vol. 10, no. 4, pp. 62-74, July 1990.

[18] T. .Styger and E. Keller, "Formant synthesis," *Fundamentals of Speech Synthesis and SpeechRecognition: Basic Concepts, State of the Art, and Future Challenges,* pp. 109-128, 1994.

[19] D. Klatt, "Review of text-to-speech conversion for English," *Journal of the Acoustical Society of America,* vol. 82, no. 3, pp. 737-793, 1987.

[20] J. Allen, S. Hunnicutt, R. Carlson and B. Granstro, "MITalk-79: The 1979 MIT text-to-speech system," in *Sppech communications papers presented at the 97th meeting of the acoustical society of america*, Cambridge, USA, 1979.

[21] J. Allen, S. Hunnicutt and D. Klatt, "From Text-to speech:The MITalk System," in *Cambridge University Press,*, Cambridge, 1987.

[22] D. Klatt, "The klattalk text-to-speech conversion system," in *Proceeding on the international conference on acoustic, speech and signal processing*, Paris, 1982.

[23] D. Klatt, "DecTalk user's manual," Digital Equipment Corporation Report, 1990.

[24] B. Kroger, "Minimal Rules for Articulatory Speech Synthesis," in *Proceedings of EUSIPCO92*, 1992.

[25] T. Dutoit, "High-Quality Text-to-Speech Synthesis: an Overview," *Journal of Electrical & Electronics Engineering,* vol. 17, pp. 25-37, 1999.

[26] B. Mobius, "Rare events and closed domains: Two delicate concepts in speech synthesis," in *Proceedings of the 4th ESCA Workshop on Speech Synthesis*, Perthshire,

Scotland, 2001.

[27] Y. Stylianou, "Harmonic plus Noise Models for Speech, combined with Statistical Methods, for Speech and Speaker Modification," PhD thesis, Ecole Nationale Supérieure des Telecommunications, Paris, January 1996.

[28] Y. Stylianou, "Modeling Speech Based on Harmonic Plus Noise Models," in *Nonlinear Speech Modeling and Applications*, Springer Berlin Heidelberg, 2005, pp. 244-260.

[29] H. Zen, K. Tokuda, T. Masuko and T. Kobayashi, "Hidden semi-Markov model based speech synthesis," in *Proceedings of ICSLP*, 2004.

[30] H. Zen, K. Tokuda and T. Kitamura, "An introduction of trajectory model into HMM-based speech synthesis," in *Proceeding of ISCA SSW5*, 2004.

[31] M. Eichner, "Speech synthesis using stochastic Markov graphs," in *ICASSP '01 Proceedings of the Acoustics, Speech, and Signal Processing, 200. on IEEE International Conference*, 2001.

[32] C. Hemptinne, "Integration of the Harmonic plus Noise Model into the Hidden Markov Model-Based Speech Synthesis System (HTS)," Master Thesis:IDIAPLauzane, Suisse, 2006.