TEXT-TO-SPEECH FOR REGIONAL LANGUAGES

A project report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

submitted by

Iftekharuddin Mohammed (16A31A0544)
Saranya Siddi (16A31A0528)
Nagaraju Ganna (16A31A0539)

Under the Guidance of

Mr. M. Raja Kumar, MTech., (Ph.D.)
Department of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PRAGATI ENGINEERING COLLEGE
(AUTONOMOUS)

(Approved by AICTE & Permanently Affiliated to JNTUK, Kakinada & Accredited by NAAC)

1-378, ADB Road, Surampalem, E.G.Dist., A.P, Pin-533437.

2018-2019

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PRAGATI ENGINEERING COLLEGE
(AUTONOMOUS)

(Approved by AICTE & Permanently Affiliated to JNTUK & Accredited by NAAC)

1-378, ADB Road, Surampalem, E.G.Dist., A.P, Pin-533437.

CERTIFICATE

This is to certify that the report entitled **"TEXT-TO-SPEECH GENERATION FOR REGIONAL LANGUAGES"** *that is being submitted by* ***Md. Iftekharuddin, S. Saranya and G. Nagaraju of III Year II Semester bearing (16A31A0544, 16A31A0528, 16A31A0539)****,respectively in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering, Pragati Engineering College is a record of Bonafede work carried out by them.*

Supervisor

Mr. M. Raja Kumar, Associate Professor

Department of CSE

Head of the Department

Dr. M. Radhika Mani, PhD

Head of Department, CSE

ACKNOWLEDGMENT

It gives us an immense pleasure to express deep sense of gratitude to our guide, Assoc. Prof. Mr. M. Raja Kumar sir, Department of Computer Science & Engineering because of his whole hearted and valuable guidance throughout the report. Without his sustained and sincere effort, this report would not have taken this shape. He encouraged and helped us to overcome various difficulties that we have faced at various stages of our report.

We would like to sincerely thank Prof Dr. M. Radhika Mani, HOD, Computer Science & Engineering, for providing all the necessary facilities that led to the successful completion of our report.

We would like to take this opportunity to thank our beloved Principal and Vice Principal Dr. S. Sambhu Prasad, Dr. K. Sathyanarayana for providing a great support to us in completing our project and for giving us the opportunity of doing the project report.

We would like to thank all the faculty members of the Department of Computer Science and Engineering for their direct or indirect support for helping us in completion of this report.

Finally, we would like to thank all our friends and family members for their continuous help and encouragement.

Iftekharuddin Mohammed (16A31A0544)

Saranya Siddi (16A31A0528)

Nagaraju Ganna (16A31A0539)

ABSTRACT

The main problem in communication is language bias between the communicators. This device basically can be used by people who do not know English and want it to be translated to their native language. The novelty component of this research work is the speech output which is available in 74 different languages translated from English. This paper is based on a prototype which helps user to hear the contents of the text in the desired language. It involves the usage of trained language data of various languages, extracting meaning of the sentence, detecting its language and then converting it into speech using Google Text-to-Speech (gTTS), which is the Text to speech engine. This relieves the travelers as they can use this device to hear the English text in their own desired language. It can also be used by the visually impaired. This device helps users to hear the images being read in their desired language.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

TTS	Text-to-Speech
RTTS	Regional Text-to-Speech
HMM	Hidden Markov Model
gTTS	Google Text-to-Speech
API	Application Programming Interface

CHAPTER 1
INTRODUCTION

Text-To-Speech (TTS) is a type of **speech** synthesis application that is used to create a spoken sound version of the **text** in a computer document, such as a help file or a Web page. A computer system used for the purpose is called a speech synthesizer & can be implemented in software or hardware products. A text-to-speech (TTS) system converts normal language text into speech; this problem focuses on TTS for regional languages.

1.1 CONCEPTS

Text-to-Speech convention transforms linguistic information stored as data or text into speech. It is widely used in audio reading devices for blind people. In the last few years however, the use of text to speech conversion technology has grown far beyond disabled community to become a major adjunct to the rapidly grown use of digital voice storage for voice mail and voice response systems. Also, development in Speech synthesis technology for various languages have already taken places.

TTS applications are well known as assistive ideas for people for people who experience dyslexia, reading challenges or visual impairment. The other valuable uses for the application range from reducing eye strain from reading (digital or paper formats), reducing paper use due to printing digital text, foreign language learning, writing and editing, or promoting listening skills.

1.2 BACKGROUND

The initial work was done in MATLAB environment for TTS generation, one of the paper discuses Kannada language TTS generation using MATLAB. Many different Machine learning and deep learning approaches have been made.

Wave Net is a powerful generative model of audio. It works well for TTS but is slow due to its sample-level autoregressive nature. It also requires conditioning on linguistic features from an existing TTS frontend, and thus is not end-to-end: it only replaces the vocoder and acoustic model. Another recently-developed in 2017 in neural model is DeepVoice, which replaces every component in a typical TTS pipeline by a corresponding neural network. However, each component is independently trained, and it's nontrivial to change the system to train in an end-to-end fashion.

In year 2016, the earliest work touching end-to-end TTS was using seq2seq with attention. However, it requires a pre-trained hidden Markov model (HMM) aligner to help the seq2seq model learn the alignment. It's hard to tell how much alignment is learned by the seq2seq per se. Second, a few tricks are used to get the model trained, which the authors note hurts parody.

1.3 INDIC TTS

This is a project on developing text-to-speech (TTS) synthesis systems for Indian languages, improving quality of synthesis, as well as small foot print TTS integrated with disability aids and various other applications. This is a consortium-based project funded by the Department of Electronics and Information Technology (Deity), Ministry of Communication and Information Technology (M CIT), Government of India involving 13 institutions and SMT, IITM being one of them. The project comprises of Phase I and Phase II. Phase I of the project used Festival-based speech synthesis for Bengali, Hindi, Tamil, Telugu, Malayalam and Marathi. Phase II of the project commenced in 2012 employing HTS based statistical speech synthesis for 13 Indian languages.

CHAPTER 2
LITERATURE SURVEY

Fundamentals of speech and speech signal processing are discussed in “Text-to-Speech Synthesis” by Paul Taylor. Lungs, larynx and vocal tract are the speech organs that are responsible for speech production. Lungs are the source of an airflow that passes through the larynx and vocal tract, before leaving the mouth as pressure variations constituting the speech signal. The source of most speech occurs in the larynx where vocal folds can obstruct airflow from the lungs. Humans produce a large number of sounds within the constraints of the vocal tract [3] [5].

Each language has a small set of linguistic units called phonemes to describe its sounds. A phoneme is the smallest meaningful unit in the phonology of a language. The physical sound produced when a phoneme is articulated is called a phone. The importance of vocal organs in speech production is clearly discussed. In addition, this book explains about the concept of how the vocal organs respond when vowels and consonants are uttered.

Basics of speech synthesis and speech synthesis methods are discussed in “An Introduction to Text-to-Speech Synthesis” by Thierry Dutoit [6]. Text to speech system organization, functions of each module and conversion of text which is given as input in to speech is clearly explained in this book.

Different speech synthesis methods, which are used for development of synthesis system, are explained in “Review of methods of Speech Synthesis” [4].

The process of text normalization is understood from “Normalization of non-standard words” by Christopher Richards. It research, conversion of non-standard words (NSWs) in to standard words is explained clearly with examples. Non-standard words are the words which are not found in the dictionary. NSWs are tokens that need to be expanded into an appropriate orthographic form before the text-to-phoneme module.

The concept of prosody is explained briefly in “Text to speech synthesis with prosody feature” by M. B. Chandak. This book explains how the prosody is predicted from the text which is given as input. In linguistics, prosody includes the intonation, rhythm and lexical stress in speech. The prosodic features of a unit of speech can be grouped into syllable, word, phrase or clause level features. These features are manifested as duration, F0 and intensity.

Prosodic units do not need to necessarily correspond to grammatical units. The perceived quality of synthetic speech is largely determined by the naturalness of the prosody generated during synthesis. The correct prosody also has an important role in the intelligibility of synthetic speech. Prosody also conveys paralinguistic information to the user such as joy or anger. In speech synthesis system, intonation and other prosodic aspects must be generated from the plain textual input.

Paper by Ramani Boothalingam (2013) presented the comparison of the performance of Unit Selection based Synthesis (USS) and HMM based speech synthesizer. Difference between the two major speech synthesis techniques namely unit selection-based synthesis and HMM based speech synthesis is explained clearly in this paper.

Unit selection systems usually select from a finite set of units in the speech database and try to find the best path through the given set of units. When there are no examples of units that would be relatively close to the target units, the situation can be viewed either as lacking in the database coverage or that desired sentence to be synthesized is not in the domain of the TTS system. To achieve good quality synthesis, the speech unit database should have good unit coverage.

To obtain various voice characteristics in TTS systems based on the selection and concatenation of acoustical units, a large amount of speech data is needed. It is very difficult to collect and segment large amount of speech data for different languages.

Storing big database in devices having only a small amount of memory is not possible. So, in order to construct a speech synthesis system that can generate various voice characteristics, without big speech database, HMM based speech synthesis was proposed.

In HMM based speech synthesis system, the parameters of speech are modeled simultaneously by HMMs. During the actual synthesis, speech waveforms are generated from HMMs themselves based on maximum likelihood criteria.
The advantage of using HMM based speech synthesis technique for developing TTS system is discussed briefly in “An Overview of Nitech HMM-based Speech Synthesis System for Blizzard Challenge 2005” by Heiga Zen and Tomoki Toda. The concept of accuracy measurement is outlined. The parameters on which the accuracy of a TTS system depends are discussed.

The quality of a TTS system is often determined by using four measures namely intelligibility, naturalness, accuracy and listening ability. These four measures are not independent from one another. For example, serious errors in accuracy will lead to less intelligibility speech, and this will be perceived as less natural and the listening ability of the synthesized speech becomes worse.

CHAPTER 3
METHODOLOGY

3.1 TERMINOLOGY BASE

The Project is completely based on Google's Text-To-Speech API package on the Python programming language. The gTTS uses the following major componential features to convert text to speech precisely and accurately.

3.1.1 PRE-PROCESSOR

Function that takes text and returns text. Its goal is to modify text (for example correcting pronunciation), and/or to prepare text for proper tokenization (for example enduring spacing after certain characters)

3.1.2 TOKENIZER

Function that takes text and returns it split into a list of tokens (strings). In the gTTS context, its goal is to cut the text into smaller segments that do not exceed the maximum character size allowed for each TTS API request, while making the speech sound natural and continuous. It does so by splitting text where speech would naturally pause (for example on ".") while handling where it should not (for example on "10.5" or "U.S.A."). Such rules are called tokenizer cases, which it takes a list of.

3.1.3 TOKENIZER CASE

Function that defines one of the specific cases used by gtts.tokenizer.core.Tokenizer. More specifically, it returns a regex object that describes what to look for a particular case. gtts.tokenizer.core.Tokenizer then creates its main regex pattern by joining all tokenizer cases with "|".

3.2 DESIGN OF RTTS – THE REGIONAL TEXT-TO-SPEECH

3.2.1 FLOW CHART

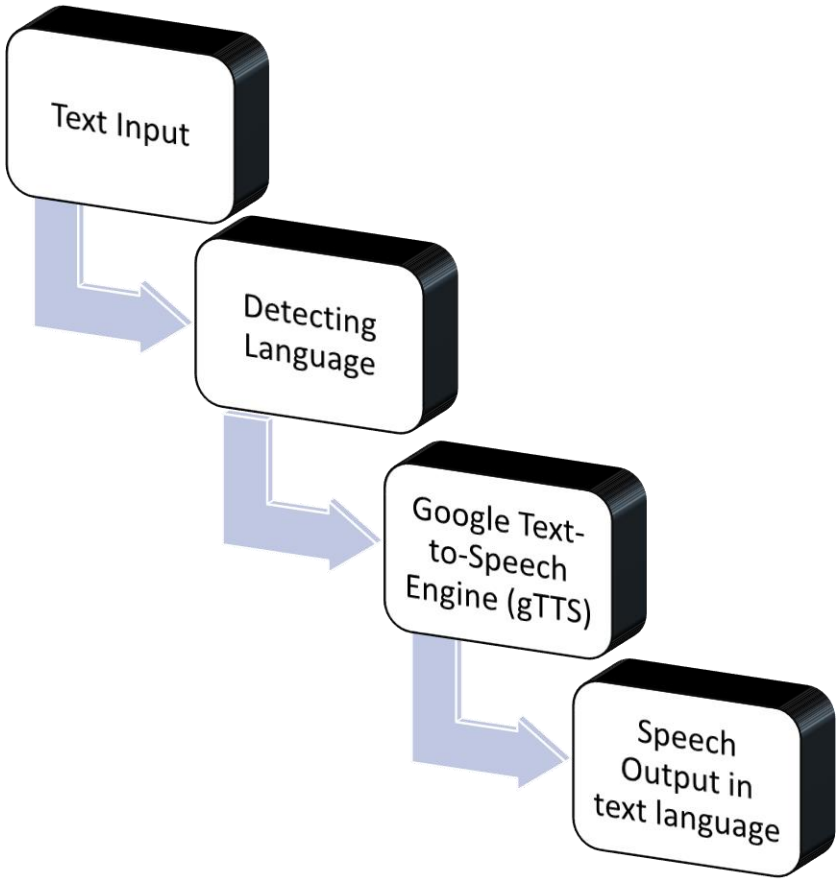

Fig 3.1 Flow Chart for the RTTS Application

The basic flow of the RTTS software is shown in the above flow chart.

a. **Text Input:** The User enters the text in any of the supported regional languages.

b. **Detecting Language:** The detection of language is a key role in the project. It is responsible for language detection of input text and output speech. The 'lang' package of python is used for this feature.

c. **Google Text-to-Speech Engine (gTTS):** The gTTS is the base of the project. Models were trained by google on various languages to redefine the meaning of precise text-to-speech conversion. Refer Chapter 1 for further details.

3.2.2 SUPPORTED LANGUAGES

Table 3.1 Supported Languages

Afrikaans	German	Norwegian	Vietnamese	
Albanian	Greek	Polish	Welsh	English (South Africa)
Arabic	Hindi	Portuguese	Chinese (Mandarin/China)	English (Tanzania)
Armenian	Hungarian	Romanian	Chinese (Mandarin/Taiwan)	French (Canada)
Bengali	Icelandic	Russian	English (US)	French (France)
Bosnian	Indonesian	Serbian	English (Canada)	Portuguese (Brazil)
Catalan	Italian	Sinhala	English (UK)	Portuguese (Portugal)
Croatian	Japanese	Slovak	English (UK)	Spanish (Spain)
Czech	Javanese	Spanish	English (Australia)	Spanish (United States)
Danish	Khmer	Sundanese	English (Ghana)	
Dutch	Korean	Swahili	English (India)	
English	Latin	Swedish	English (Ireland)	
Esperanto	Latvian	Tamil	English (New Zealand)	
Estonian	Macedonian	Telugu	English (Nigeria)	
Filipino	Malayalam	Thai	English (Philippines)	
Finnish	Marathi	Turkish		
French	Myanmar (Burmese)	Ukrainian		

Google Text-to-Speech Engine is a large framework that supports the above 74 languages of the world. This makes gTTS framework the only one to be capable of recognizing and tokenizing large amount of language data while maintaining efficiency, accuracy and precision simultaneously.

3.2.3 UML DIAGRAMS

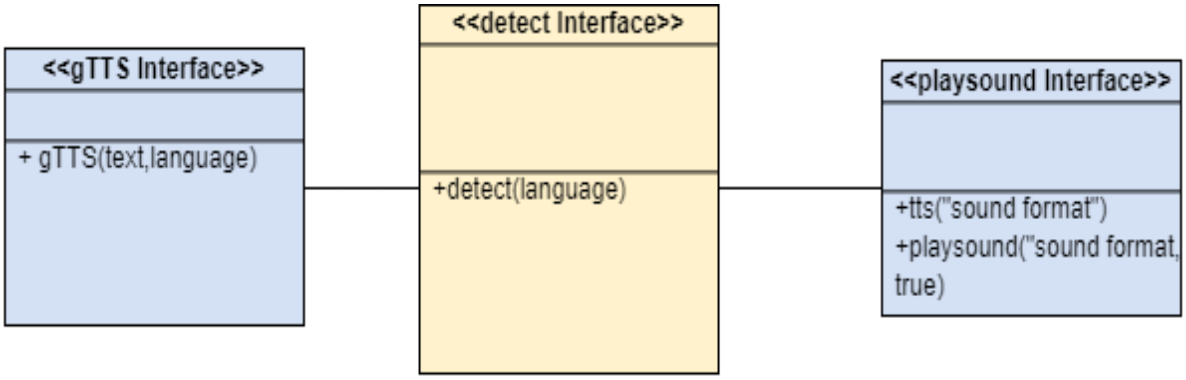

Fig 3.2 Class Diagram

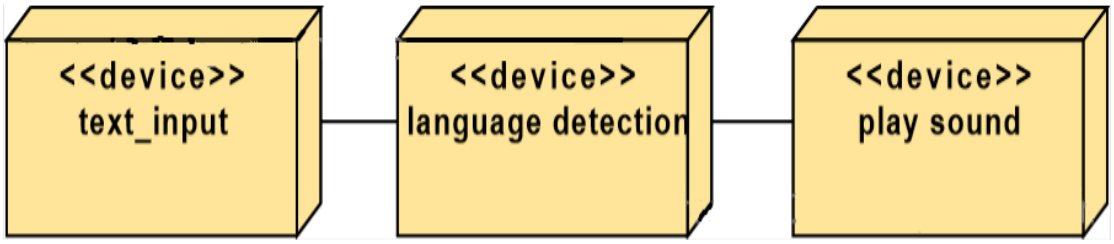

Fig 3.3 Deployment Diagram

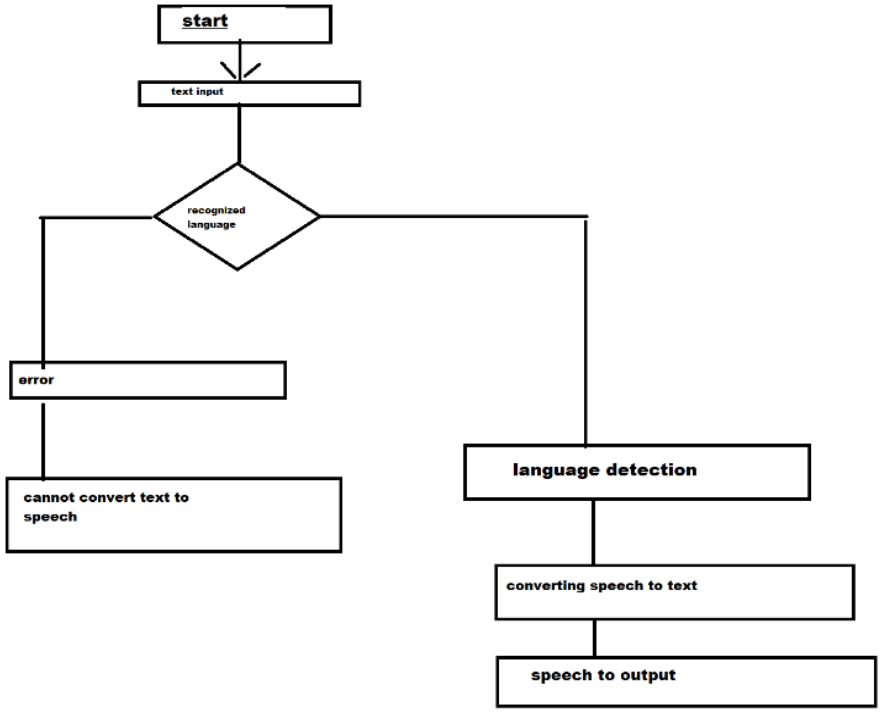

Fig 3.4 Collaboration Diagram

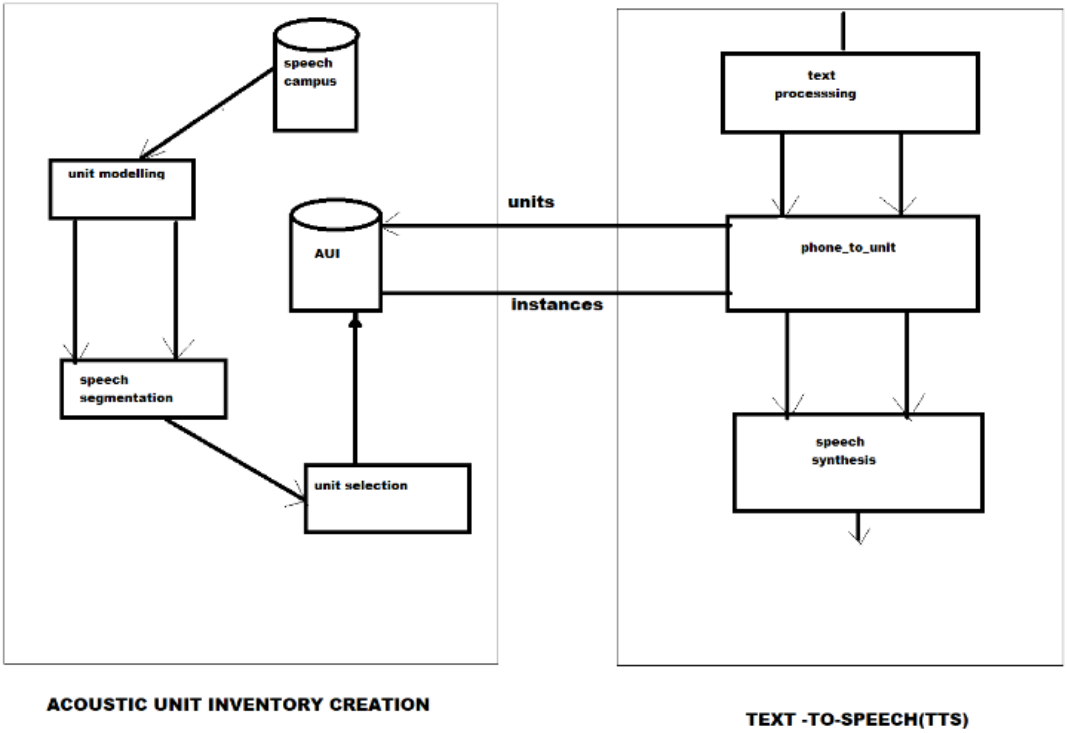

Fig 3.5 Component Diagram

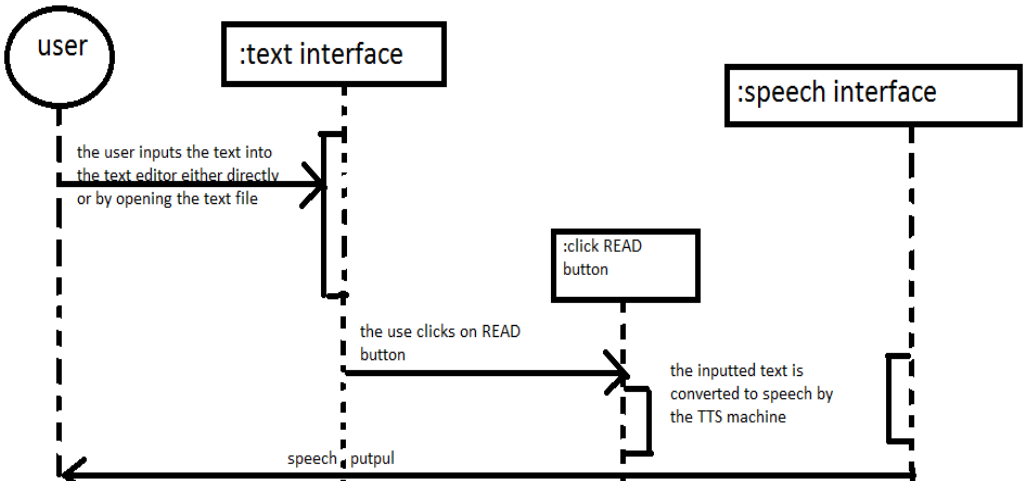

Fig 3.6 Sequence Diagram

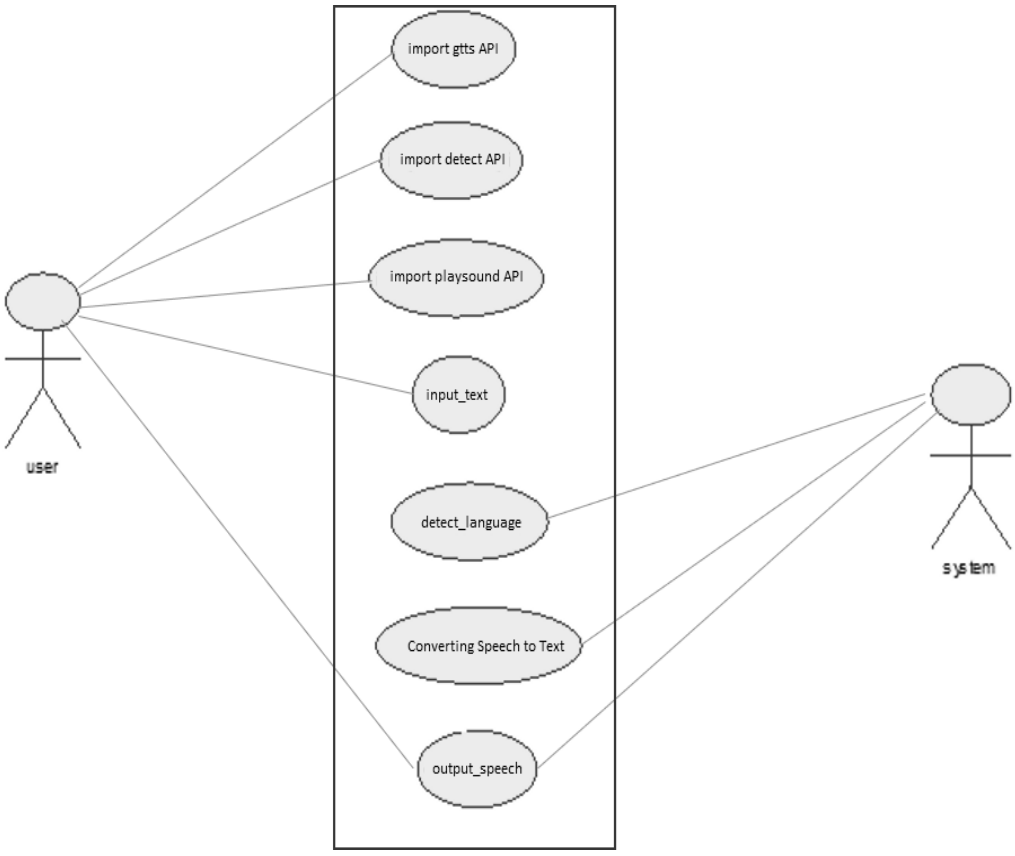

Fig 3.7 Use Case Diagram

3.3 IMPLEMENTATION AND ALGORITHM

We proposed the method of using Google Text-to-Speech (gTTS) python interface. It is a powerful speech engine for conversion of text to speech. It can convert almost all the Indian regional language-text into speech in seconds. Its application is also extended so that it can support not only India regional languages, but all the languages around the world. An App can be made for different platforms like Windows, Linux, Mac, Raspbian, etc. This application can help lots of visually impaired people to read text around them without human assistance. An App can be made for different platforms like Windows, Linux, Mac, Raspbian, etc. This application can help lots of visually impaired people to read text around them without human assistance.

➔ **ALGORITHM**

Step 1: Importing the following python packages

a. gTTS – Google Text-to-Speech Engine
b. lang – Language detection API
c. playsound – Sound Engine for mp3 filesystem

Step 2: Text input is taken from user

text_input = input(“Please enter text here”)

Step 3: Detecting language of text input

detected_language = detect(text_input)

Step 4: Sending text_input to gTTS Engine with detected language as parameter to obtain speech in the same language.

tts = gTTS(text = text_input, lang = detected_language)

Step 5: Saving output sound file and playing the regional language speech

tts.save(<file_name.mp3>)

playsound(<filename.mp3>, True)

3.4 GRAPHICAL USER INTERFACE (GUI) of RTTS

The RTTS Software is a ready to use software with a simple User Interface. It has a text box in which the regional language text should be entered by the user. The user is supposed to click on the button that says 'Speech' to generate the voice/speech of the entered text. A text label displays the detected language code of the entered text. RTTS used Python's TKinter framework to build this simple UI. A Windows executable of this software can be generated using PyInstaller, another python's framework.

PyInstaller collects all the dependencies of a python program (RTTS in this case). These dependencies are packed together and converted into an windows executable file (.exe) to use it as a single file application.

NOTE: **Internet Connection is Mandatory!!!**

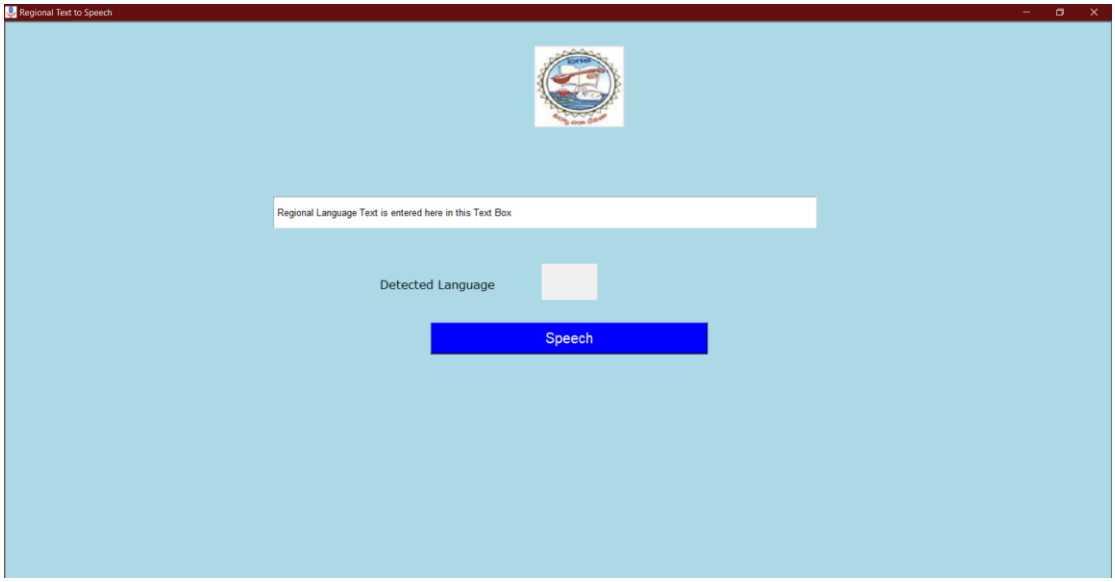

Figure 3.8 UI Look and Feel

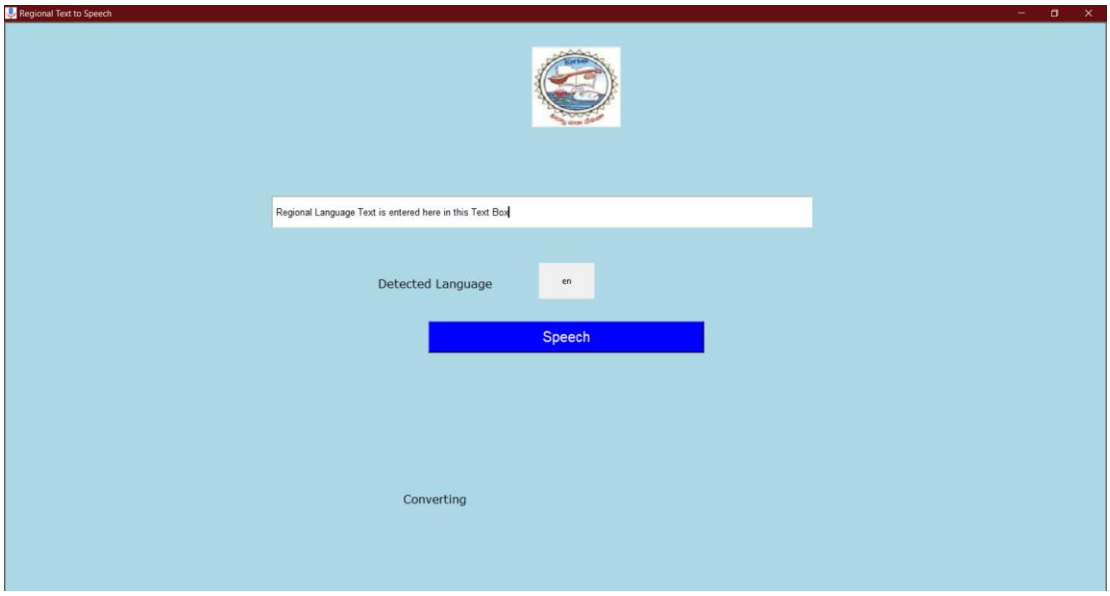

Figure 3.9 Converting Text to Speech

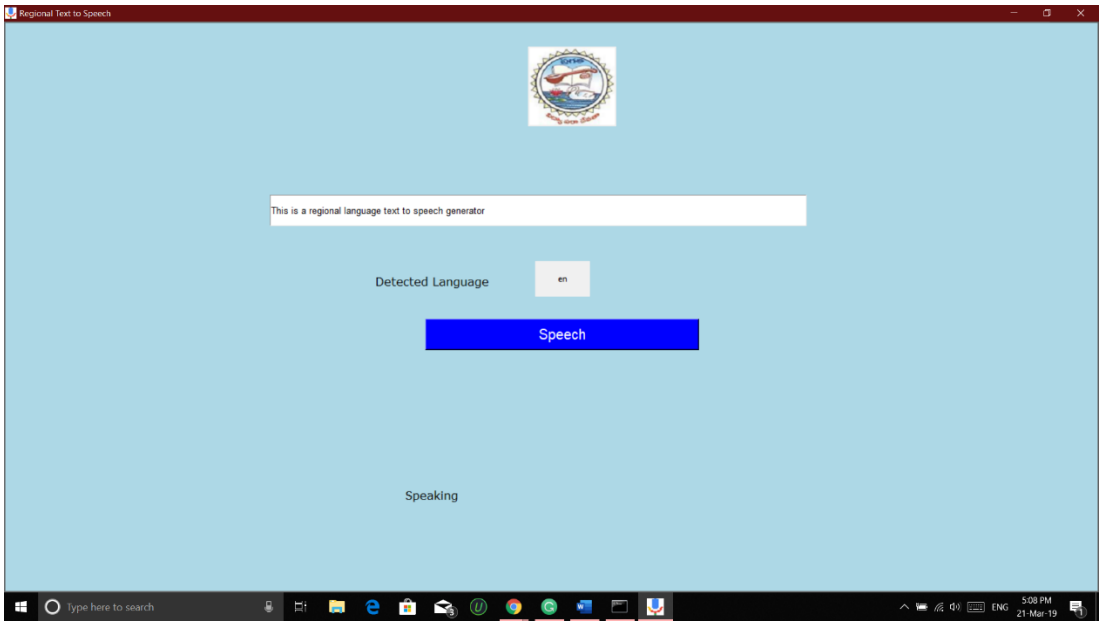

Figure 3.10 Speaking the generated speech

CHAPTER 4

RESULTS AND DISCUSSIONS

The major outcome/result of this Regional Text-to-Speech generation software is a speech. This process requires internet connection.

4.1 WHY IS INTERNET CONNECTION A REQUIREMENT

Conversion of Text-to-Speech in English is an easy task and accessible to everyone around the world. But in case of regional languages, there are a large number of common regional languages around the world and as a result it is impossible to maintain the database, train regional language models using Natural Language Processing (NLP). Practically, to train and collect regional language data, large amounts of resources are required and cannot be collected and performed by an individual developer or a small organization.

4.2 HOW DID WE MANAGE TO SOLVE THIS ISSUE?

As me already know, Google is the largest company in the world and has the capability of computing trillions of Gigabytes of data that is generated all over the world by each and every user. Google is also known to have the best Text and Speech related analysis systems and technologies. It also has a Google Text-to-Speech API, also known as the gTTS, is available as open source for developers to implement further of their projects, ideas and technologies using gTTS as their base.

This is the reason why RTTS uses internet connection in order to send and receive all the 74 language's data to ensure accuracy and precision and this makes RTTS a ready-to-use software.

CHAPTER 5
CONCLUSION AND FUTURE SCOPE

Concluding out project, The RTTS (Regional Text-to-Speech) can be used in many applications.

Real-time Text can be used:

- In conjunction with voice and/or video in a multimedia communication or on its own, on fixed or mobile accesses,
- By people who want a fast and interactive means of conversing,
- In noisy environments where it may be hard to hear,
- In environments where other people are nearby but where communications privacy is required,
- To transfer information e.g. numbers, addresses etc., where exactness is necessary,
- By people who are Deaf or Hard of Hearing or with a speech impairment to communicate with other people, including people who can hear and speak.
- For relay services to offer real-time conversion between different modes of communication as a service to people who are Deaf or Hard of Hearing or with a speech impairment. E.g. to provide real-time captioning of a voice conversation for people who are Hard-of-Hearing.
- To provide all voice callers with a convenient means to accurately pass numbers, addresses and other detailed information in text.
- To allow people who are Deaf or Hard of Hearing or have a speech impairment to use the emergency services without limitations.
- To offer remote interpreter and transcribing (note taking) services for every user who needs it.

BIBLIOGRAPHY

1. Santiago Dimaren, A UX/UI Case Study: Designing a Text-To-Speech App From The Ground Up, Medium. https://blog.prototypr.io/a-ux-ui-case-study-designing-a-text-to-speech-app-from-the-ground-up-1fc95bd04a2b

2. Alfian Losari, Mobile App That Speaks with DeepMind WaveNet Google Cloud Text To Speech ML API and Flutter, Medium https://medium.com/flutter-community/mobile-app-that-speaks-with-deepmind-wavenet-google-cloud-text-to-speech-ml-api-and-flutter-48abf0992cb9

3. Aki Ariga, Text-to-speech based on deep learning for Web site using Amazon Polly and Ruby, Medium https://blog.chezo.uno/text-to-speech-for-web-site-using-amazon-polly-and-ruby-adc1923212cb

4. gTTS Docs, https://gtts.readthedocs.io/en/latest/

5. gTTS Docs, https://gtts.readthedocs.io/en/latest/tokenizer.html

APPENDIX-A
SAMPLE CODE

Programming Language: Python 3.6.5

Project Requirements: Internet Connection

Platform Tested: Windows 10 x64 bit, will work on Linux & Mac too.

```
from gtts import gTTS
from langdetect import detect
from playsound import playsound
text_input = input()
lang = detect(text_input)
print("Text Language: ", lang)
tts = gTTS(text=text_input, lang=lang)
tts.save('speech.mp3')
playsound('speech.mp3', True)
```