TEXT TO SPEECH GENERATING SYSTEM USING NATURAL LANGUAGE PROCESSING

A PROJECT REPORT

Submitted to

SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING

In partial fulfilment of requirements for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

by

A. VENKATESWARA RAO (11606048)
G.V. NIKHILA (11606055)
SK. MOBEENA BEGAM (11606063)

Under the guidance of

Dr. A. RAMA MOHAN REDDY
Professor Department of Computer Science & Engineering

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING,
SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING, TIRUPATI,
2019-20.

SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING, TIRUPATI

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"TEXT TO SPEECH GENERATING SYSTEM USING NATURAL LANGUAGE PROCESSING (NLP)"** genuine and has been carried out under my supervision in the **Department of Computer Science and Engineering**, **Sri Venkateswara University College of Engineering.**

The work is comprehensive, complete and fit for evaluation carried out in partial fulfilment of the requirements for the award of **Bachelor of Technology** in **Computer Science and Engineering** during the academic year 2019-20.

To the best of our knowledge matter embodied in the project has not been submitted to any other University/Institution for the award of any Degree or Diploma.

GUIDE:
Dr. A. RAMA MOHAN REDDY
Professor,
Department of CSE,
SVUCE, Tirupati.

HEAD OF THE DEPT.:
Dr. P. VENKATA SUBBA REDDY
Professor,
Department of CSE,
SVUCE, Tirupati.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SRI VENKATESWARA UNIVERSITY COLLEGE OF ENGINEERING, TIRUPATI

Declaration

The project entitled "**TEXT TO SPEECH GENERATING SYSTEM USING NATURAL LANGUAGE PROCESSING"** is a bona fide work performed by us, for the partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** from **Sri Venkateswara University, Tirupati.**

To the best of our knowledge matter embodied in the project has not been submitted to any other University/Institution for the award of any Degree or Diploma.

(A VENKATESWARA RAO	**11606048)**
(G.V. NIKHILA	**11606055)**
(SK. MOBEENA BEGAM	**11606063)**

ABSTRACT

A Text-to-speech synthesizer is an application that converts text into spoken word, by analyzing and processing the text using **Natural Language Processing** (NLP) and then using **Digital Signal Processing** (DSP) technology to convert this processed text into synthesized speech representation of the text. Text in various Languages can be converted to Speech, Text in the Images can be converted to Speech and we can also use **Optical Character Recognition**(OCR) to dynamically capture images that contain text so that the system can latter convert the text into speech.

The various methods used are **Preprocessing**, **Unicode Conversion**, **Segmentation**, **Concatenation**, **Prosody** and **Smoothing**, to be then combined in an application for easy access and usability. Here, the developed simple application that converts inputted text into synthesized speech and reads out to the user which can then be saved as an mp3. file for future reference. The converted speech can be available in different languages
.so that, User can be able to choose whatever the language they can understand by changing the available options and can also adjust volume, modulation, play back speed and many more options etc.

CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Chapter-1: INTRODUCTION

Overview:

Over the past few years, Cell Phones have become an indispensable source of communication for the modern society. We can make calls and text messages from a source to a destination easily. It is known that verbal communication is the most appropriate modem of passing on and conceiving the correct information, avoiding misquotations.

Quite a few applications used to assist the disabled make use of TTS, STT, and translation. They can also be used for other applications, taking an example: Siri an intelligent automated assistant implemented on an electronic device, to facilitate user interaction with a device, and to help the user more effectively engage with local and/or remote services makes use of voice recognition and text-to-speech (TTS) technology.

Text-to-Speech technology aims to provide a user- friendly application to general users. The main modules used in this application are: Text-to-Speech convertor and Image-to-Text convertor. The application provides a multi-functionality platform for users to communicate, listen or narrate conveniently. The users can choose to convert readable images to text files or read text as such.

The text-to-speech mode converts a text file or inputted text to speech which then is narrated/read using the voice database used by Microsoft SAPI [5] (or) Google Text To Speech (GTTS) engine along with its own Google translator. This processing is done using phonemes and concatenating syllables.

Image-to-text mode converts readable images to a text file which can further be used for speech conversion. Readable images are images that have less complexity in the foreground making it possible to extract the letters in a grammatical fashion. The user is provided with multiple options in this software where he/she can select the mode of operation. When Text-to-Speech mode is selected, the user has to input text using the text input box or a text file. The text is processed and the resulting speech is produced.

Purpose and Motivation:

The primary objective behind developing this system is to combine various modules using modular approach in order to get a simple yet effective way for differentially abled people to interact with others by capturing images and converting the text to speech and thereby making the society better.

An avid reader or a book lover can also use this system to read and understand books written in different languages by simply scanning it and convert it to his/her understandable language(speech). Text To Speech (TTS) system can also help kids who struggle with reading.

The motivation to create this project has many sources: -

- **Saved time and money**

  With TTS technology that is web- or cloud-based on a SaaS (Software as a Service) platform, online content can quickly and easily be speech enabled, and maintenance is minimal

- **Populations are evolving**

  Language proficiency and schooling in the host country's language is a very real problem for migrants and their families.

- **Language Learning**

  Practice makes perfect and adding our highly accurate Text to Speech voices to your language solutions is the best way to enable learners to develop fluency.

  They can test and consolidate skills at any time, without the pressures of the classroom. Moreover, learning styles and study materials can be customized according to each learner's interests.

- **People are increasingly mobile and looking for convenience**

  Text to speech can turn any digital content into a multimedia experience and people can listen to a news or blog article, a text file, or an e-book on the go.

.

- **Word-of-mouth marketing**

  Adding an alternative way to consume content online enhances the user experience. Visitors are far more likely to return to and recommend those where they have had positive experiences.

Objectives:

- The motto of this project is to develop a system capable of taking text, Image, text file as input and the system will automatically detect the source language and convert it into the speech of desired language requested by the user.
- To support translation in all languages that are supported by Google Translator.
- To allow users in order to dynamically capture and load images containing text either in written or printed format and convert it to speech of their native language.
- To help the disabled, children and new language learners to listen and understand the text, pronunciation of the unknown languages.

CHAPTER II

TEXT TO SPEECH

SYNTHESIS

Chapter-2: TEXT TO SPEECH SYNTHESIS

OVERVIEW OF SPEECH SYNTHESIS:

Speech synthesis is the transformation of text to speech. This transformation converts the text to synthetic speech that is as close to real speech as possible in the compliance with the communication norms of special languages. The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood, which can be simplified as two parameters, naturalness of sounding and intelligibility of speech.

A Text-to-Speech system has to model both the generic, phonetic features that make speech intelligible, and the idiosyncratic, acoustic characteristics that make it human. This Text-to-Speech engine or system is mainly composed of two parts:

1.	Natural Language Processing (NLP)	: Converts Texts-to-Phoneme
2.	Digital Signal Processing(DSP)	: Converts Phoneme-to-Speech

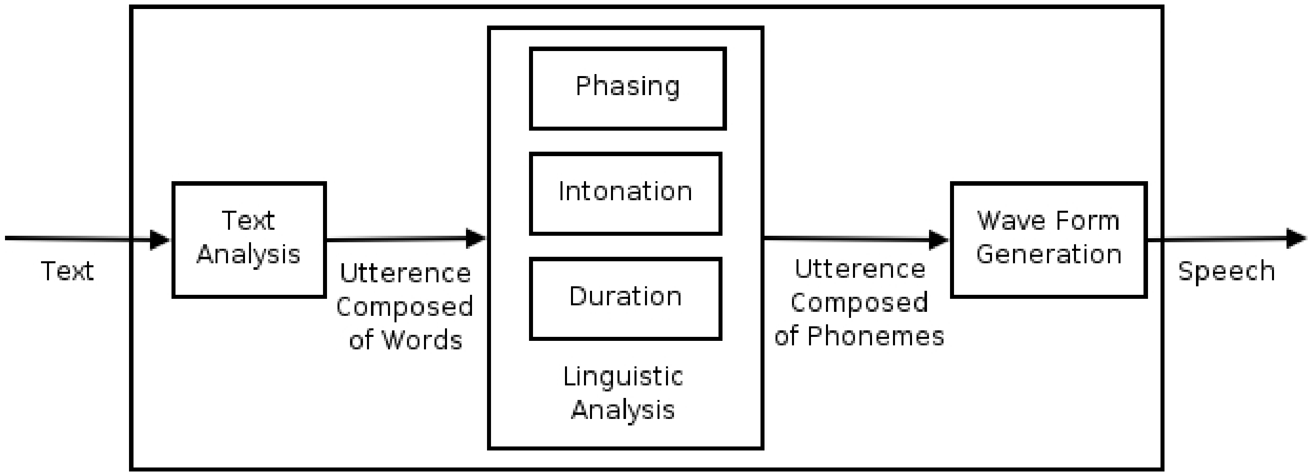


Figure 2.1: An Overview of a typical TTS system

Texts–to–Phoneme:

Also called a Grapheme–to–Phoneme conversion, the process of assigning phonetic transcription to words. The text must be converted into a linguistic representation that includes the phonemes to be produced, their duration, the location of phrase boundaries, and the pitch / frequency contours for each phrase.

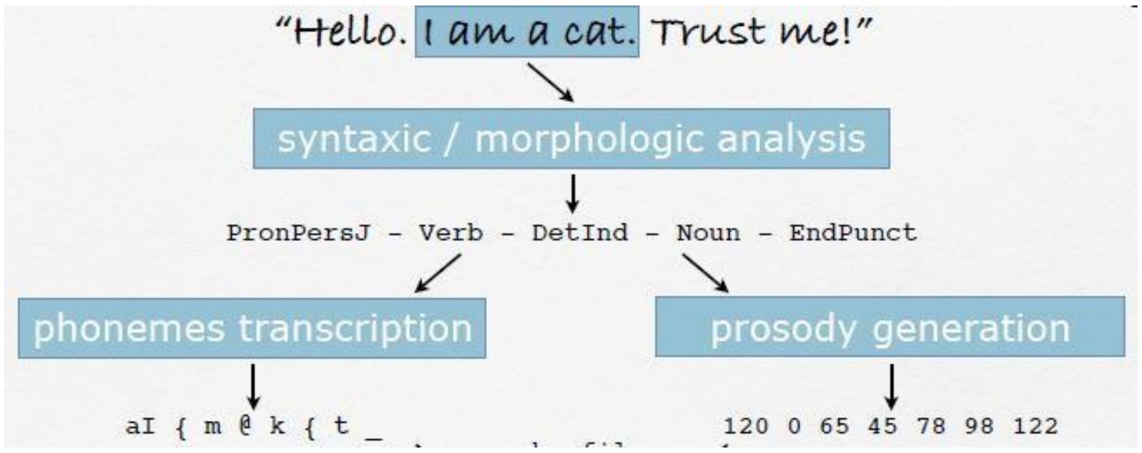

"Hello. I am a cat. Trust me!"
syntaxic / morphologic analysis
PronPersJ – Verb – DetInd – Noun – EndPunct
phonemes transcription
prosody generation
aI { m @ k { t _
120 0 65 45 78 98 122

Figure 2.2: Texts-to-Phoneme

. **Phoneme–to–Speech**:

The Phonetic transcription and prosody information obtained in the linguistic analysis stage are converted into an acoustic waveform.

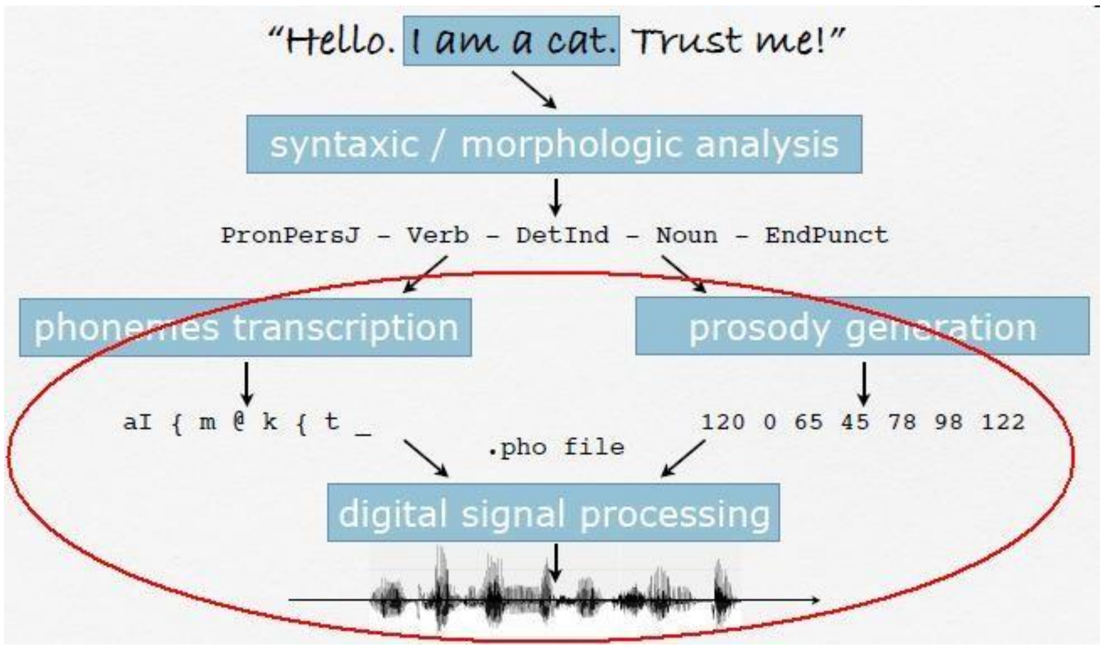

"Hello. I am a cat. Trust me!"
syntaxic / morphologic analysis
PronPersJ - Verb - DetInd - Noun - EndPunct
phonemes transcription
prosody generation
aI { m @ k { t _
.pho file
120 0 65 45 78 98 122
digital signal processing

Figure 2.3: Phoneme – to – Speech

While text is rich in phonetic information, it contains little or nothing about the vocal qualities that denote emotional states, moods, and variegations in emphasis or attitude. The elements of prosody such as register, accentuation, intonation, and speed of delivery are barely represented in the orthography of a text. Yet without them, a synthesized voice sound monotonous and unnatural.

STRUCTURE OF A TTS SYNTHESIZER:

Text-to-speech synthesis takes place in several steps. The TTS systems get a text as input, which it first must analyze and then transform into a phonetic description. Then in a further step it generates the prosody. From the information now available, it can produce a speech signal.

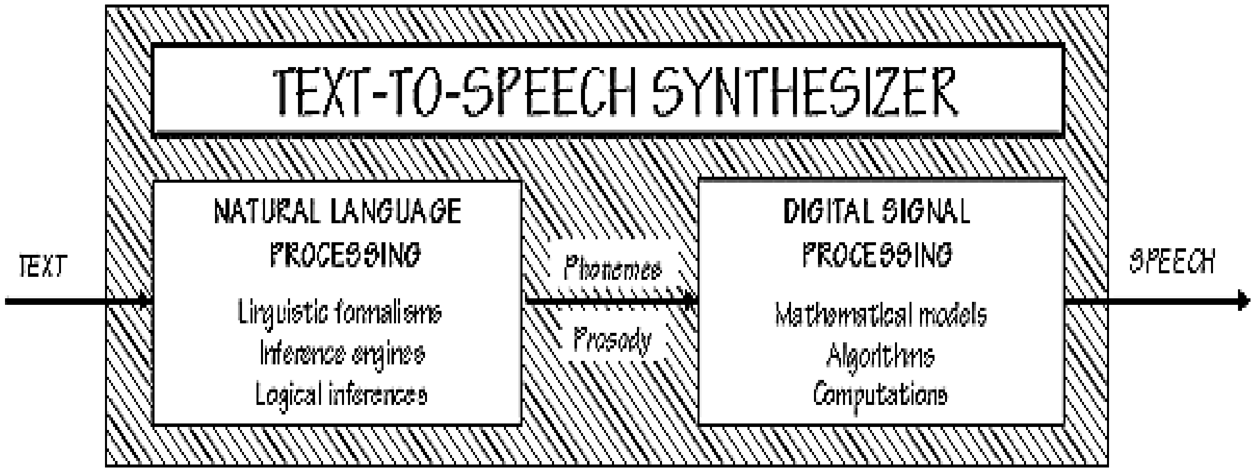


Figure 2.4: A simple but general functional diagram of a TTS system

The structure of the text-to-speech synthesizer can be broken down into major modules:

1. **Natural Language Processing (NLP) module**
2. **Digital Signal Processing (DSP) module**

Natural Language Processing (NLP) module:

It produces a phonetic transcription of the text read, together with prosody. Natural Language processing (NLP) is a widely used technique by which systems can understand the instructions for manipulating text or speech.

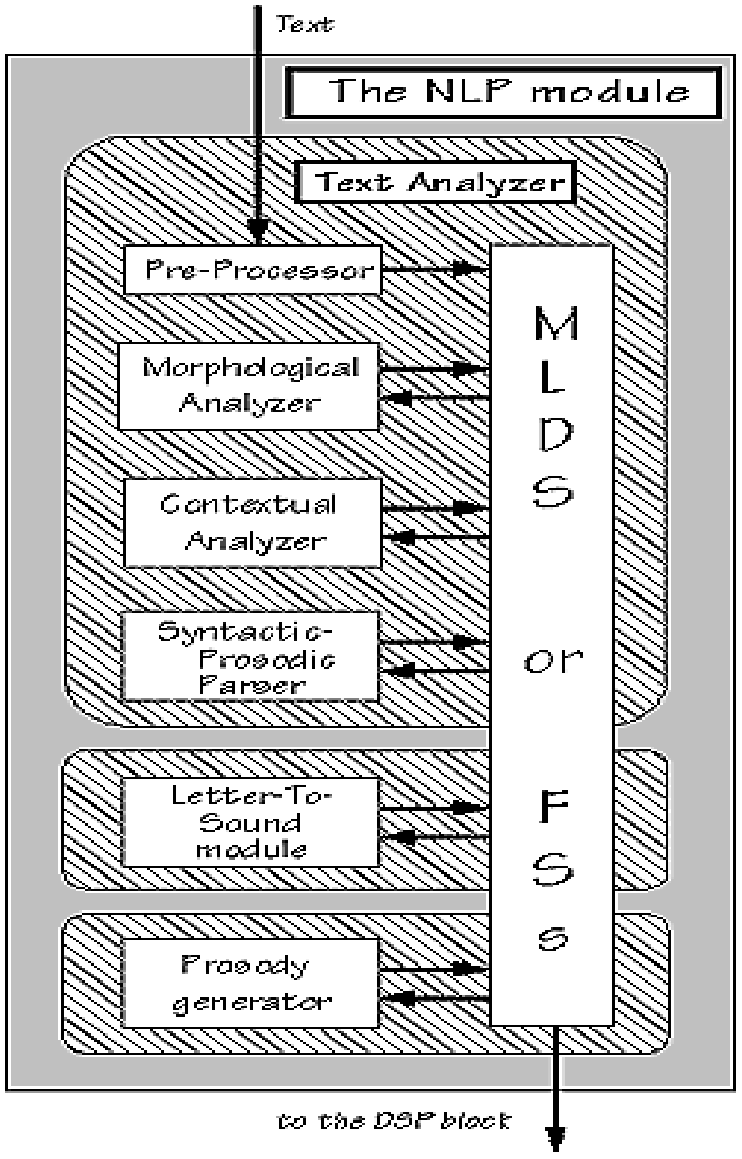


Figure 2.5: Operations of the natural Language processing module of a TTS synthesizer.

The major operations of the NLP module are as follows:

- **Text Analysis:** First the text is segmented into tokens. The token-to-word conversion creates the orthographic form of the token. For the token "Mr." the orthographic form "Mister" is formed by expansion, the token "12" gets the orthographic form "twelve" and "1997" is transformed to "nineteen ninety-seven".

- **Application of Pronunciation Rules:** After the text analysis has been completed, pronunciation rules can be applied. Letters cannot be transformed 1:1 into phonemes because correspondence is not always parallel. In certain environments, a single letter can correspond to either no phoneme (for example, "h" in "caught") or several phoneme ("m" in "Maximum"). In addition, several letters can correspond to a single phoneme ("ch" in "rich"). There are two strategies to determine pronunciation:

    - In **dictionary-based solution** with morphological components, as many morphemes (words) as possible are stored in a dictionary. Full forms are generated by means of inflection, derivation and composition rules. Alternatively, a full form dictionary is used in which all possible word forms are stored. Pronunciation rules determine the pronunciation of words not found in the dictionary.

- In a **rule based solution**, pronunciation rules are generated from the phonological knowledge of dictionaries. Only words whose pronunciation is a complete exception are included in the dictionary.

These two applications differ significantly in the size of their dictionaries. The dictionary-based solution is many times larger than the rules-based solution’s dictionary of exception. However, dictionary-based solutions can be more exact than rule-based solution if they have a large enough phonetic dictionary available.

- **Prosody Generation:** after the pronunciation has been determined, the prosody is generated. The degree of naturalness of a TTS system is dependent on prosodic factors like intonation modelling, phrasing and accentuation, amplitude modelling and duration modelling including the duration of sound and the duration of pauses, which determines the length of the syllable and the tempos of the speech.

Then the output of the NLP module is passed to the DSP module and this is where the actual synthesis of the speech signal happens.

Digital Signal Processing (DSP) module:

The Digital Signal Processor translates the phonetic language specification of the text produced by the NLP into spoken speech. The main challenge of the DSP is to produce a voice that is both intelligible and natural. This is where the actual synthesis of the speech signal happens.

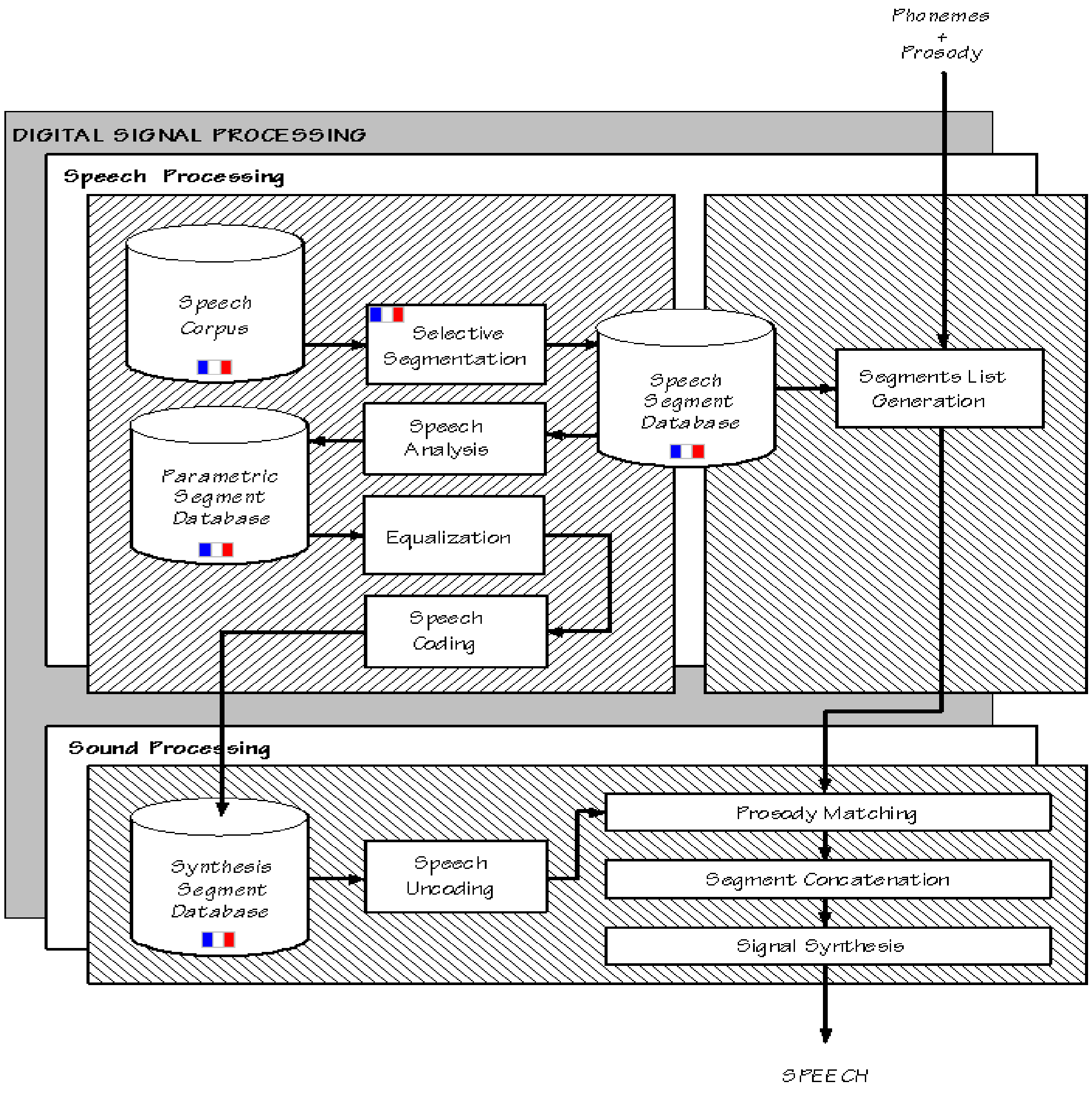


Figure 2.6: DSP component of a general concatenation based synthesizer

Concatenative synthesis and format synthesis are the two primary technologies to generate synthetic speech waveforms.

Concatenative synthesis – Concatenative synthesis is based on the concatenation of segments of recorded speech. Generally, concatenative synthesis produces the most natural-sounding synthesized speech. However, differences between natural variations in speech and the nature of the automated techniques for segmenting the waveforms sometimes result in audible glitches in the output.

Formant synthesis – Formant synthesis does not use human speech samples at runtime. Instead, the synthesized speech output is created using additive synthesis and an acoustic model. Parameters such as fundamental frequency, voicing, and noise levels are varied over time to create a waveform of artificial speech. This method is sometimes called rules-based synthesis; however, many concatenative systems also have rules-based components. Many systems based on formant synthesis technology generate artificial, robotic-sounding speech that would never be mistaken for human speech.

Articulator Synthesis – Uses mechanical & acoustic model for speech generation. It produces intelligible synthetic speech but it is far from natural sound and hence not widely used.

RESEARCH METHODOLOGY:

The survey results were presented in two groups: voice features and non-voice features including all the functional features, operational features and supporting resources. In terms of the voice features, read aloud in long text, technical terms, calculation, product names/unit and functions/formulas were the most demanded in the TTS software tool. Besides the requirements in reading of text, the availability of multiple languages was one of the most significant requirements related to voice quality. Leaving English aside, the usage of Spanish was 73%, and Russian, Chinese, French, German, and Italian were used in a wide range of e-leaning courses as well. Obviously, the demand for multiple languages was one of the most important criteria when evaluating text–to–speech software tools.

On the other hand, the requirements for non-voice features were summarized based on the order of their importance. Generally, the features not viewed as “Not important” would be classified as first priority requirements for voice quality features. Such as

- 90% responses received for requirement of technical support.
- 86% responses received for Speed of Program running.
- Integration of other software requirement also receives more than 50% responses.
- Support of Multiple languages also received more responses.

Based on these things the voice quality, user interface and some functional features should be treated as essential ones and the other ones were also mandatory, nut if not fulfilled, they must be compensated equally useful features.

CHAPTER III

PROPOSED SYSTEM

CHAPTER-3: PROPOSED SYSTEM

OVERVIEW OF SYSTEM

Currently, there are a number of applications; plug-ins and gadgets widely used as speech-synthesis technology tools. A great many Text–to–Speech systems in multiple languages are commonly used for desktop, server, telephone, and internet applications. Here we propose a System that takes text, image and text file as input and convert them into speech of desired language (as the developed TTS system uses google translator, hence it will support many languages).

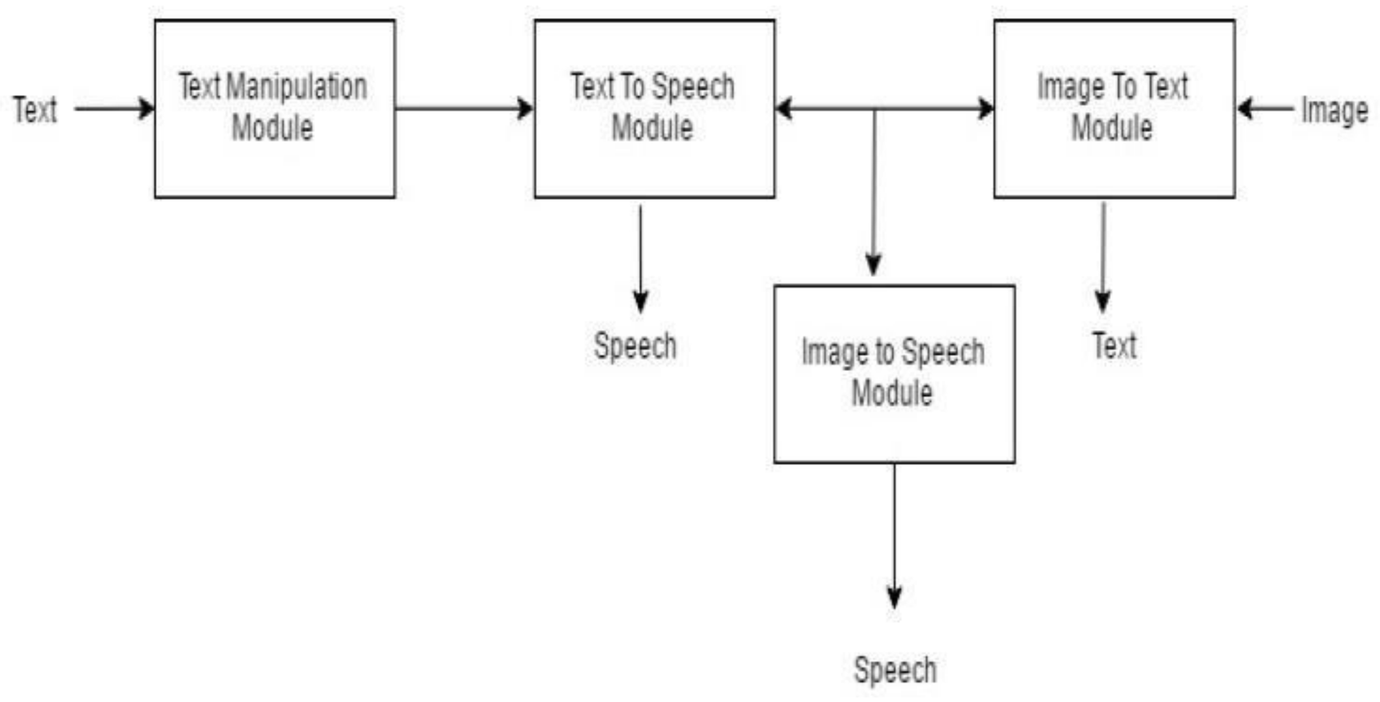


Fig 3.0.1: Overview of the model

PREREQUISITES

It is a combination of both the system requirements and the necessary software libraries required to install.

Hardware Requirements:

- **Central Processing Unit (CPU) —** Intel core i5
- **RAM —** 4 GB minimum
- **Operating System —** Microsoft Windows 8 or higher
- **A HD webcam to take high resolution pictures**

Software Requirements:

- **Python:**

Python is an interpreter, object-oriented, high-level programming language with dynamic semantics. It's high-level built in data structures, combined with the dynamic typing and binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

➢ Kivy:

Kivy is an open source Python library for rapid development of applications that make use of innovative user interfaces, such as multi-touch apps. Using Kivy on your computer, you can create apps that run on Desktop computers (OS X, Linux, and Windows), IOS devices (iPad, iPhone), Android devices (tablets, phones) and any other touch-enabled professional/homebrew devices supporting TUIO (Tangible User Interface Objects). Kivy empowers you with the freedom to write your code once and have it run as-is on different platforms.

➢ gTTS API:

There are several APIs available to convert text to speech in python. One of such APIs is the Google Text to Speech API commonly known as the gTTS API. gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as a mp3 file.

The gTTS API supports several languages including English, Hindi, Tamil, French, German and many more. The speech can be delivered in any one of the two available audio speeds, fast or slow. However, as of the latest update, it is not possible to change the voice of the generated audio.

➢ Google Translator:

Python **googletrans** is a module to translate text. It uses the **Google Translate Ajax API** to detect langauges and translate text. Text **translation** from one language to another is increasingly becoming common for various websites as they cater to an international audience. The python package which allows us to do this is called **googletrans.**

➢ PyTesseract:

PyTesseract is an open source text recognition (OCR) Engine, available under the Apache 2.0 license. It can be used directly, or (for programmers) using an API to extract printed text from images. It supports a wide variety of languages but doesn't have a built-in GUI and there are several available from the 3rdParty page.

Tesseract is compatible with many programming languages and frameworks through wrappers that can be found here. It can be used with the existing layout analysis to recognize text within a large document, or it can be used in conjunction with an external text detector to recognize text from an image.

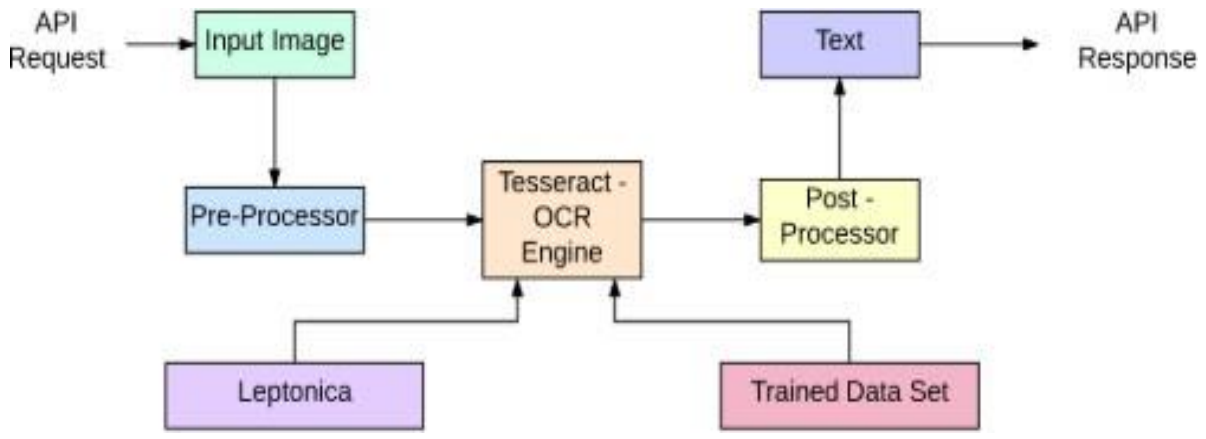


Figure 3.2.1: OCR Process Flow

➢ Python Imaging Library (PIL):

Python Imaging Library abbreviated as PIL in newer versions known as Pillow is a free and open-source additional library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X and Linux.

ENVIRONMENT SETUP

- **Windows Environment**

Installing required software and libraries one by one

- Python for windows it is also can be download directly from the website after the download then install on the PC
- An environment variable path is created for python by adding following paths

  C:\Program Files\Python37 this path to Environment variable
- Pip is python package installer and it is be installed by using the command below

  Download get-pip.py folder, open command prompt and type below command

  python get-pip.py to install
- Similarly install all the other necessary packages by saving all the package names in a text file and by using the following command in the Command Prompt

  File path :\> Pip install –r filename.txt

SUMMARY

It is important to set up the system accurately to run this TTS system. Every command step should follow correctly to complete the procedures.

SOURCE CODE:

➢ Text to speech conversion

```
import kivy

import os

import time

import pyttsx3

engine= pyttsx3.init("sapi5")

from gtts import gTTS

from kivy.app import App

from googletrans import Translator

from kivy.lang import Builder

from kivy.uix.label import Label

from kivy.uix.gridlayout import GridLayout

from kivy.uix.boxlayout import BoxLayout

from kivy.uix.textinput import TextInput

from kivy.uix.dropdown import DropDown

from kivy.uix.button import Button


class MyGrid(GridLayout):

    def__init__(self,**kwargs):
```

```
super(MyGrid,self).__init__(**kwargs)

self.cols=1


self.inside=GridLayout()

self.inside.cols=1


self.add_widget(self.inside)

self.inside.add_widget(Label(text="Enter your text",size_hint=(0.2,0.2)))

self.name=TextInput(multiline=True)

self.inside.add_widget(self.name)


self.LastName=TextInput(hint_text= 'enter your language', multiline=False,
size_hint=(0.1,0.1))

self.inside.add_widget(self.LastName)


self.submit=Button(text="   Download   ",font_size=22,size_hint=(0.1,0.1))

self.submit.bind(on_press=self.pressed)

self.add_widget(self.submit)


self.submit=Button(text="   Synthesize   ",font_size=22,size_hint=(0.1,0.1))

self.submit.bind(on_press=self.clicked)

self.add_widget(self.submit)
```

```
    self.submit=Button(text="   Clear text   ",font_size=22,size_hint=(0.1,0.1))
    self.submit.bind(on_press=self.clear_txt)
    self.add_widget(self.submit)


def clear_txt(self,instance):
    self.name.text= ''
    self.LastName.text= ''


def pressed(self,instance):
    soun =self.name.text
    langu=self.LastName.text
    p= Translator()
    detect= p.detect(soun)
    print("source language: ",detect.lang)
    k = p.translate(soun,dest=langu)
    sem=k.text
    speech = gTTS(text = sem, lang = langu , slow = True)
    print("Translated text: ",sem)
    firs=soun.split()[0]
    timestr = time.strftime("%Y%m%d_%H%M")
    speech.save("TtoAudio{}.mp3".format(firs))
def clicked(self,instance):
        soun =self.name.text
        langu=self.LastName.text
        p= Translator()
        detect= p.detect(soun)
        print("source language: ",detect.lang)
        k = p.translate(soun,dest=langu)
        sem=k.text
        print(sem)
        speech = gTTS(text = sem, lang = langu , slow = True)
        print("Translated text: ",sem)
        speech.save("Audio{}.mp3")
        os.system("start Audio{}.mp3")


class MyApp(App):
    def build(self):
        return MyGrid()


if__name__== "__main__":
    MyApp().run()
```

➢ Text File to Speech Conversion

```
import kivy

import time

from gtts import gTTS

from kivy.lang import Builder

from googletrans import Translator

from kivy.properties import ObjectProperty

import pyttsx3

engine=pyttsx3.init()

from kivy.app import App

kivy.require('1.9.0')

import os

from kivy.uix.floatlayout import FloatLayout

Builder.load_string('''

<Filechooser>:

        orientation: 'vertical'

        BoxLayout:

                FileChooserListView:

                        on_selection: root.click(*args)

                        TextInput:
```

```
                    id: input
                    hint_text: 'enter language'
                    multiLine: False
                    size_hint_y: '0.2dp'
                    height: '10dp'
                Button:
                    text: 'Convert'
                    font: 20
                    size_hint_y: '0.13dp'
                    width: '10dp'
                    on_press: root.crook()
                Button:
                    text: 'Download'
                    size_hint_y: '0.05dp'
                    width: '10dp'
                    on_press: root.pressed()            ''')
class Filechooser(FloatLayout):
    def click(self, *args):
        try: self.text= args[1][0]
        except: passs
```

```
      print(self.text)
def crook(self, *args):
      try: self.text = args[1][0]
      except: pass
      langu=self.ids.input.text
      print(self.text)
      hello=open(self.text)
      result=hello.read()
      print(result)
      p = Translator()
      detect= p.detect(result)
      print("source lang: ",detect.lang)
      k=p.translate(result,dest=langu)
      sem=k.text
      print(sem)
      speech = gTTS(text = sem, lang = langu , slow = False)
      print("Entered text: ",sem)
      speech.save("filetospeech.mp3")
      os.system("start filetospeech.mp3")
 def pressed(self, *args):
```

```
            try: self.text = args[1][0]
            except: pass
            langu=self.ids.input.text
            hello=open(self.text)
            result=hello.read()
            print(result)
            p = Translator()
            detect= p.detect(result)
            print("source lang: ",detect.lang)
            k=p.translate(result,dest=langu)
            sem=k.text
            speech = gTTS(text = sem, lang = langu , slow = False)
            print("Entered text: ",sem)
            tim=time.strftime("%Y%m%d_%H%M%S")
            speech.save("filetospeech{}.mp3".format(tim))
class FileApp(App):
    def build(self):
        return Filechooser()
if__name__== '__main__':
    FileApp().run()
```

➢ Image to Speech Conversion

```
from kivy.app import App

from kivy.lang import Builder

from kivy.uix.boxlayout import BoxLayout

import time

import pytesseract

import pyttsx3

engine = pyttsx3.init()

from gtts import gTTS

from googletrans import Translator

from PIL import Image

Builder.load_string('''

<CameraClick>:

    orientation: 'vertical'

    Camera:

        id: camera

        resolution: (1080, 1920)

        play: False

    ToggleButton:

        text: 'Play'
    on_press: camera.play = not camera.play
    size_hint_y: None
    height: '30dp'
Button:
    text: 'Capture'
    size_hint_y: None
    height: '30dp'
    on_press: root.capture()
TextInput:
    id: input
    hint_text: 'enter your language eg: en-english, hi-hindi etc.'
    multiLine: False
    size_hint_y: '0.1dp'
    height: '10dp'
Button:
    text: 'convert'
    size_hint_y: None
    height: '30dp'
    on_press: root.convert()
Button:
        text: 'Download Audio'

        size_hint_y: None

        on_press: root.pressed()

        height: '30dp'     "')

class CameraClick(BoxLayout):

    def pressed(self):

        langu=self.ids.input.text

        timestr = time.strftime("%Y%m%d_%H%M")

        text=("IMG_{}.png".format(timestr))

        img = Image.open(text)

        print(img)

        pytesseract.pytesseract.tesseract_cmd      ='C:/Program      Files      (x86)/Tesseract-
OCR/tesseract.exe'

        result = pytesseract.image_to_string(img)

        with open('abc.txt',mode ='w') as file:

                                file.write(result)

                                print(result)

        p = Translator()

        k = p.translate(result,dest=langu)

        print(k)
    speech = gTTS(text = k, lang = langu , slow = True)
    print("Name: ",k)
    tim=time.strftime("%Y%m%d_%H%M%S")
    speech.save("CamImgtoAudio{}.mp3".format(tim))

def capture(self):
    camera = self.ids['camera']
    timestr = time.strftime("%Y%m%d_%H%M")
    camera.export_to_png("IMG_{}.png".format(timestr))
    text=("IMG_{}.png".format(timestr))
    print(text)
    print("Captured")

def convert(self):
    langu=self.ids.input.text
    timestr = time.strftime("%Y%m%d_%H%M")
    text=("IMG_{}.png".format(timestr))
    img = Image.open(text)
    print(img)
    pytesseract.pytesseract.tesseract_cmd ='C:/Program Files (x86)/Tesseract-
```

```
OCR/tesseract.exe'
        result = pytesseract.image_to_string(img)
        with open('abc.txt',mode ='w') as file:
                                file.write(result)
                                print(result)
        p = Translator()
        detect= p.detect(result)
        print("source language: ",detect.lang)
        k = p.translate(result,dest=langu)
        sem=k.text
        speech = gTTS(text = sem, lang = langu , slow = True)
        print("Translated text: ",sem)
        speech.save("CamImgtoAudio.mp3")
        os.system("start CamImgtoAudio.mp3")


class TestCamera(App):
    def build(self):
        return CameraClick()


TestCamera().run()
```

CHAPTER IV

SYSTEM DESIGN

Chapter 4: SYSTEM DESIGN

MODULE DESCRIPTION:

The Proposed System consists of the following: Text to speech generating module, Image to Speech generating module and Text file to speech generating module.

Text to speech Data flow diagram:

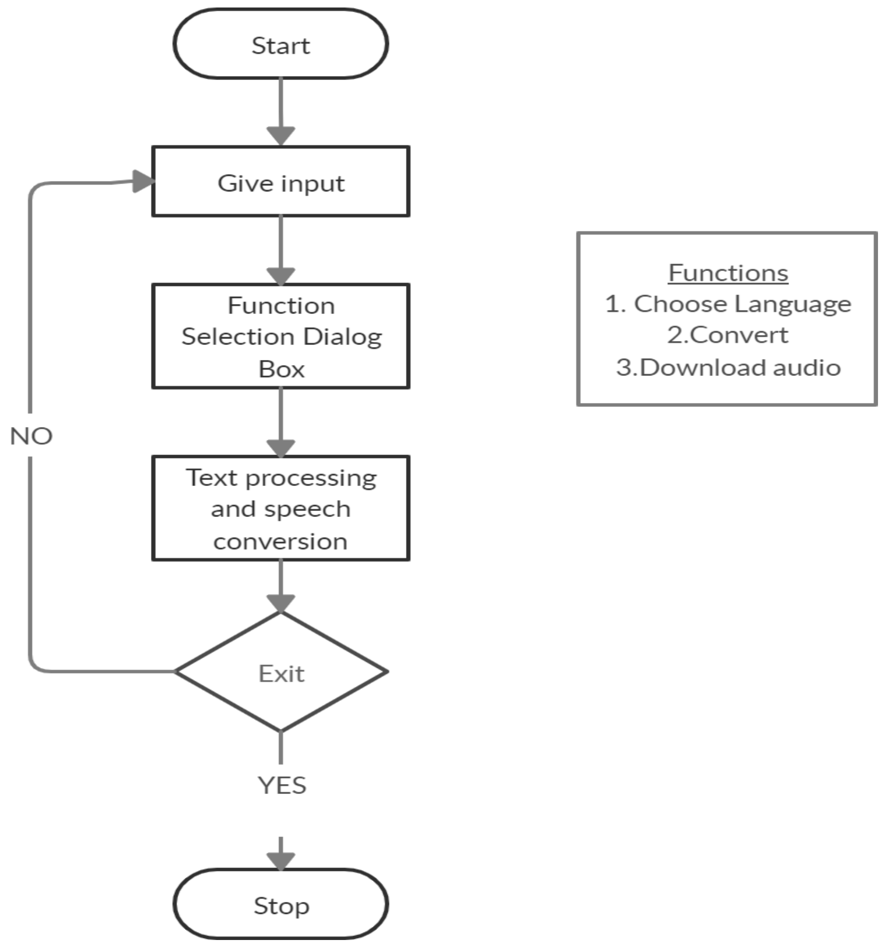
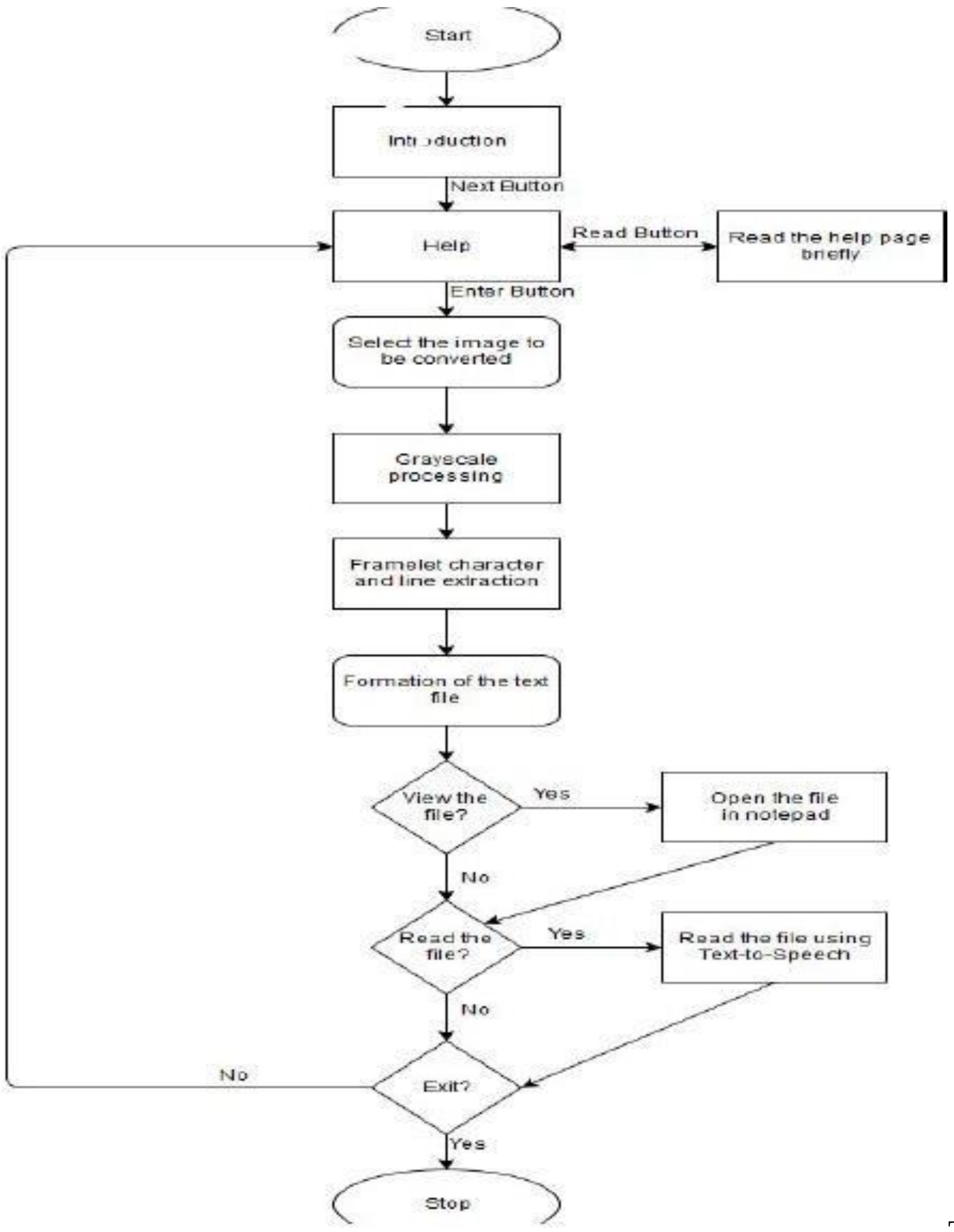


Fig 4.1.0: Text to Speech data flow diagram

The above figure depicts the flow of the text to speech module which is explained in detail as follows: Text to speech conversion can be accomplished by starting with the method of pre-processing of the input text. Here the text abbreviations, acronyms and numbers are expanded. The pre-processed text will then be converted to Unicode. Unicode has the explicit aim of transcending the limitations of traditional character encodings. Here the pre-processed text is used to identify the fonts of input text and is converted to Unicode. Now, the encoded text is segmented into syllables and the duplicates are removed. The syllabled text is then mapped with the syllable sound files in the database. These syllables will be then concatenated and smoothened for resultant outputs. This is done by optimal coupling algorithm and this will gives a smooth human speech output.

$$\boxed{\mathbf{W\ (n) = .054 - 0.46\ cos\ (6.28n/N\text{-}1)}}$$

Image to speech Data flow diagram:



Start
Introduction
Next Button
Help
Read Button
Read the help page briefly
Enter Button
Select the image to be converted
Grayscale processing
Framelet character and line extraction
Formation of the text file
View the file?
Yes
Open the file in notepad
No
Read the file?
Yes
Read the file using Text-to-Speech
No
Exit?
No
Yes
Stop

T

Figure 4.1.2: Image to Speech data flow diagram

Here the contents of the image will be converted to text and thereafter read out as speech. Images containing handwritten text, printed text form is converted to Unicode text, the image can be a scanned file, a photograph of a document or a photograph taken in real-time. The image is converted into an inverted grayscale image in order to remove noise by eliminating hue and saturation. The noise, if any, is removed using the threshold.

$$g = t - f \quad \dots (2)$$

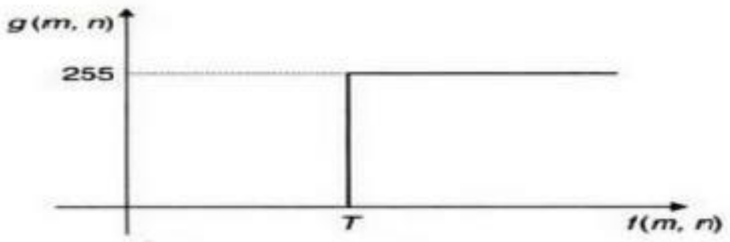
g = Converted Image

t = Threshold calculated from the image

f = Input Image

function line: g = ~im2bw(f, t)

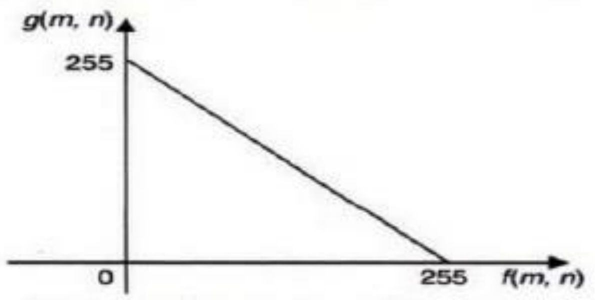
The foreground is made white and the background black due to inverse gray-scaling.



$$g(m,n) = \begin{cases} 0 & \text{for } f(m,n) < t \\ 255 & \text{otherwise} \end{cases}$$

Fig 4.1.3: Graph and Eqn. depicting threshold

This is further segmented using frames to extract characters from the image which is mapped to a matrix which allows the image to be read line by line and character by character.



$$g(m, n) = 255 - f(m, n)$$

Fig 4.1.4: Graph and Eqn. depicting Inverse Transformation

The extracted text is stored into the matrix as it is read, allowing the extracted text to be appropriately related to the image and hence the text saved isn't randomly stored. Further, the text will be entered to a text file and the contents of the text file can be read aloud using the text-to-speech convertor of the application.

SIMULATION

The simulation of Text to Speech conversion module is done by using kivy, python which gives the following results when deployed.

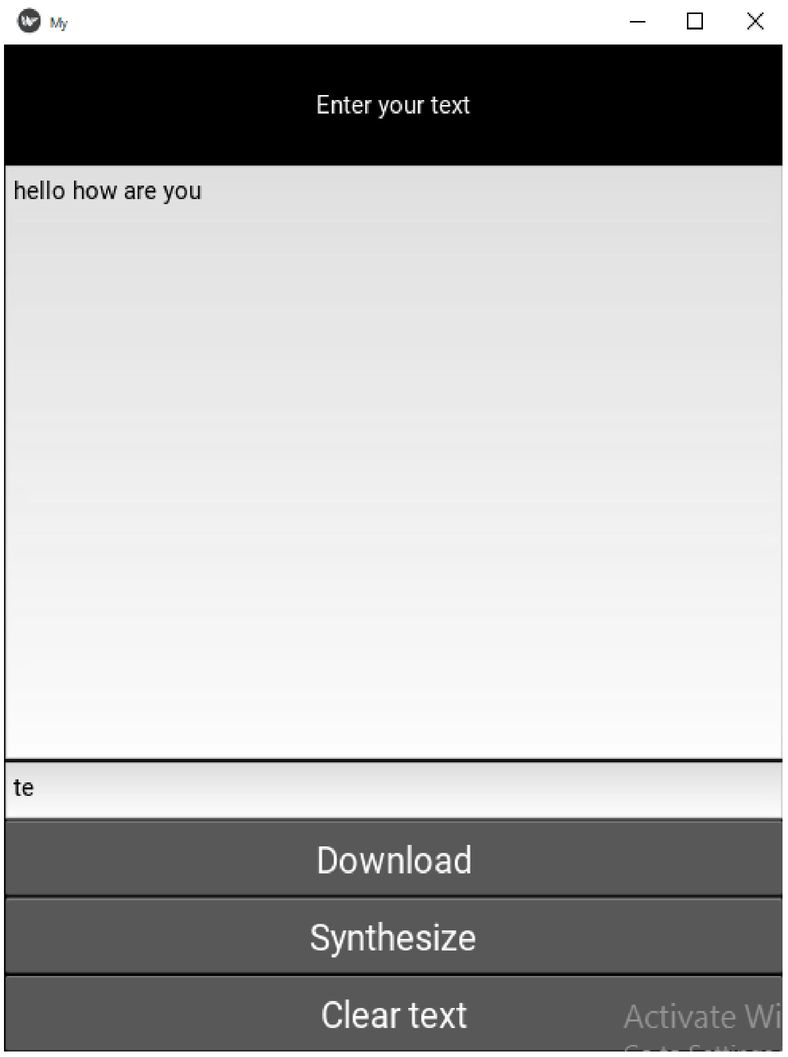


Fig 4.2.1: Text Input page

In this module user can give text as input and choose the destination language to which the given text needs to be converted into speech and then click on synthesize if you want to listen or simply press download button to download the converted audio file.

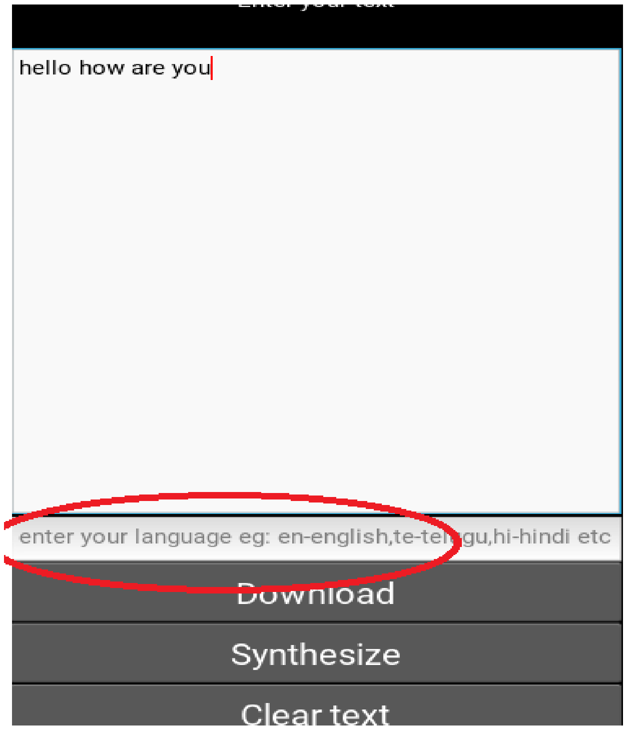
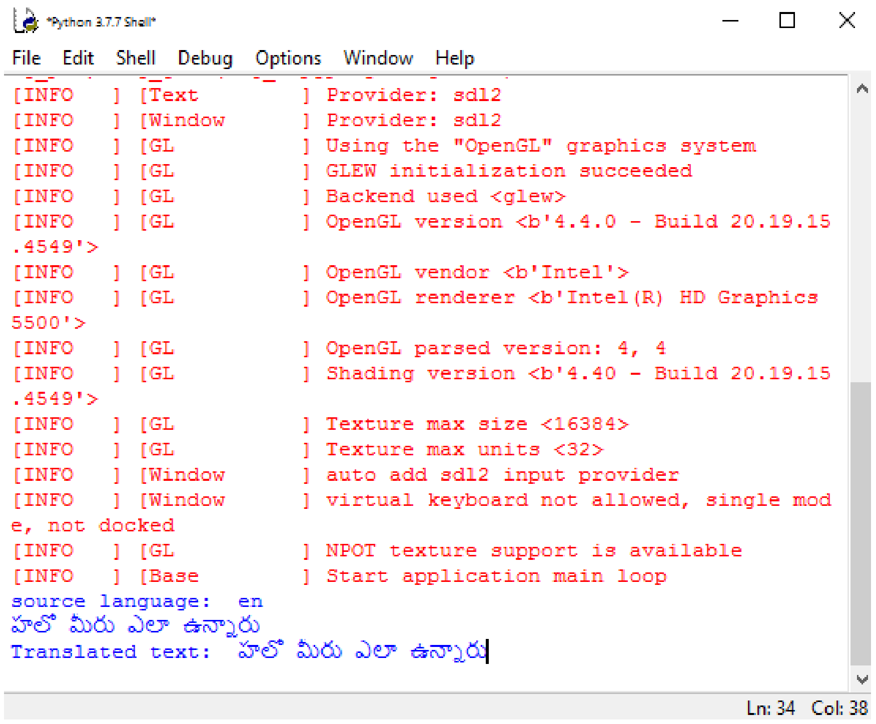


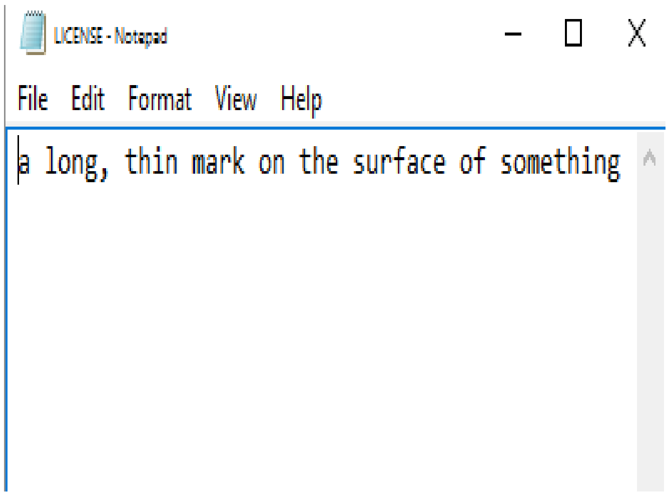
Fig 4.2.2: Choosing destination language

In this module we can saw the converted text of the given input in the chosen language i.e.Telugu

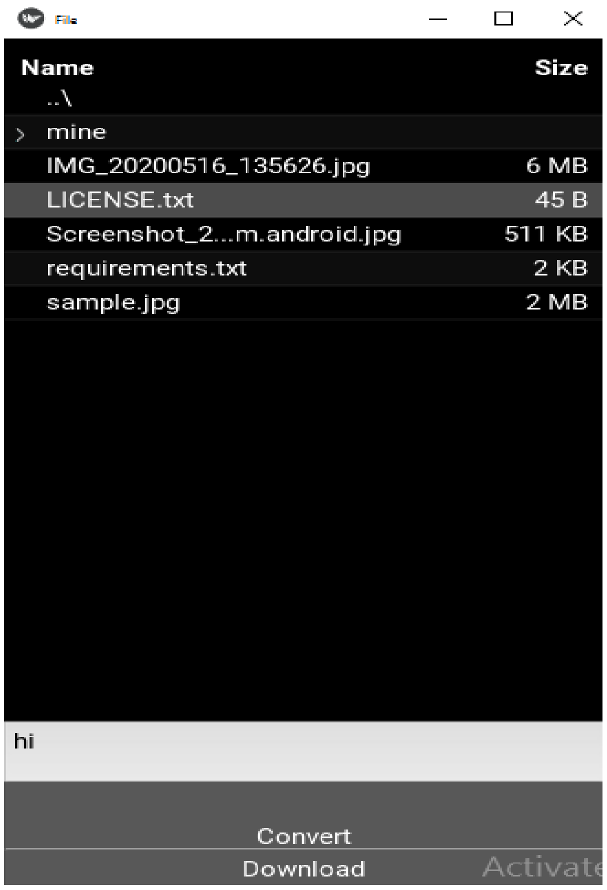


```
[INFO   ] [Text        ] Provider: sdl2
[INFO   ] [Window      ] Provider: sdl2
[INFO   ] [GL          ] Using the "OpenGL" graphics system
[INFO   ] [GL          ] GLEW initialization succeeded
[INFO   ] [GL          ] Backend used <glew>
[INFO   ] [GL          ] OpenGL version <b'4.4.0 - Build 20.19.15
.4549'>
[INFO   ] [GL          ] OpenGL vendor <b'Intel'>
[INFO   ] [GL          ] OpenGL renderer <b'Intel(R) HD Graphics
5500'>
[INFO   ] [GL          ] OpenGL parsed version: 4, 4
[INFO   ] [GL          ] Shading version <b'4.40 - Build 20.19.15
.4549'>
[INFO   ] [GL          ] Texture max size <16384>
[INFO   ] [GL          ] Texture max units <32>
[INFO   ] [Window      ] auto add sdl2 input provider
[INFO   ] [Window      ] virtual keyboard not allowed, single mod
e, not docked
[INFO   ] [GL          ] NPOT texture support is available
[INFO   ] [Base        ] Start application main loop
source language:  en
హలో మీరు ఎలా ఉన్నారు
Translated text:  హలో మీరు ఎలా ఉన్నారు
```

Fig 4.2.3: Text Conversion into given language



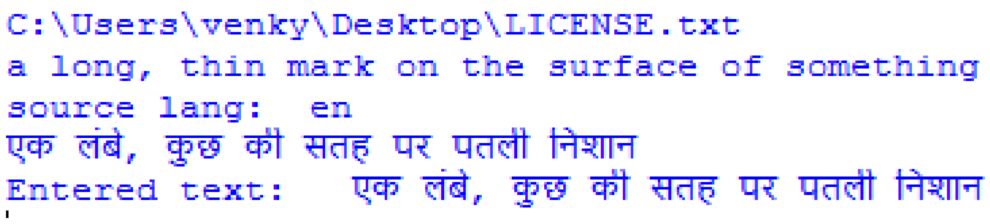
```
a long, thin mark on the surface of something
```



```
Name                          Size
..\
mine
IMG_20200516_135626.jpg       6 MB
LICENSE.txt                   45 B
Screenshot_2...m.android.jpg  511 KB
requirements.txt              2 KB
sample.jpg                    2 MB
hi
```

Fig 4.2.4: LICENSE.txt **Fig 4.2.5: Choosing Text file**

In this module user can select the text file as input and choose the destination language to which the text needs to be converted into speech and then click on convert if you want to listen or press download button in order to download the converted audio file.



```
C:\Users\venky\Desktop\LICENSE.txt
a long, thin mark on the surface of something
source lang:  en
एक लंबे, कुछ की सतह पर पतली निशान
Entered text:   एक लंबे, कुछ की सतह पर पतली निशान
```

Fig 4.2.6: Text in the file converted into desired language

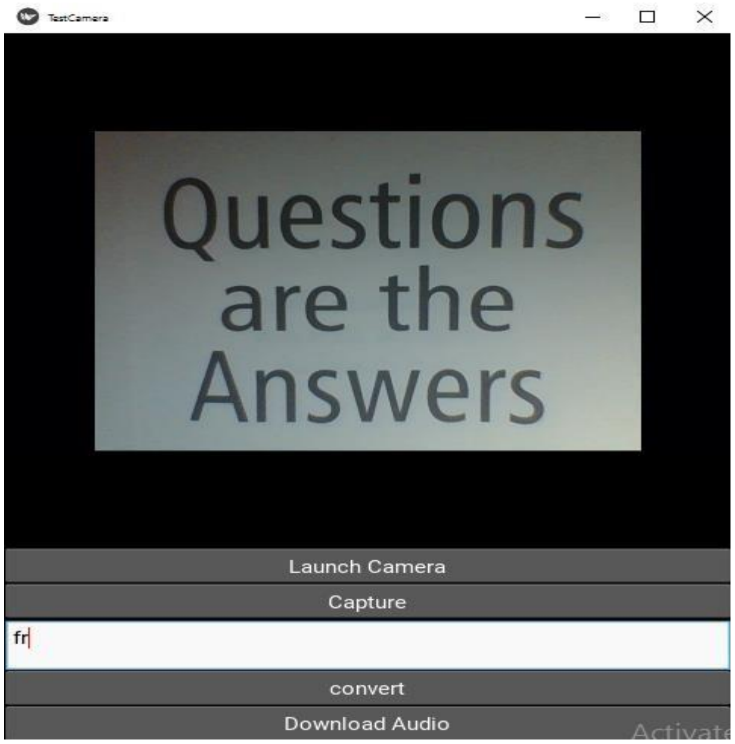
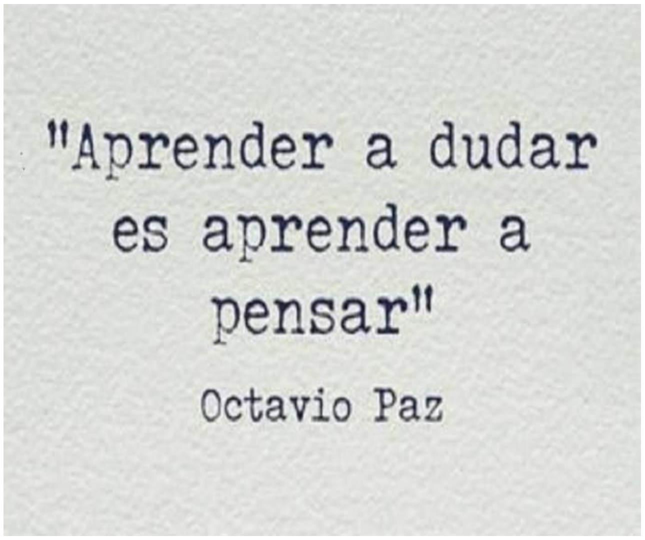


Fig 4.2.7: Captured Image

In this module, we can dynamically capture Images containing text and then convert it to speech of desired language or download the converted audio.

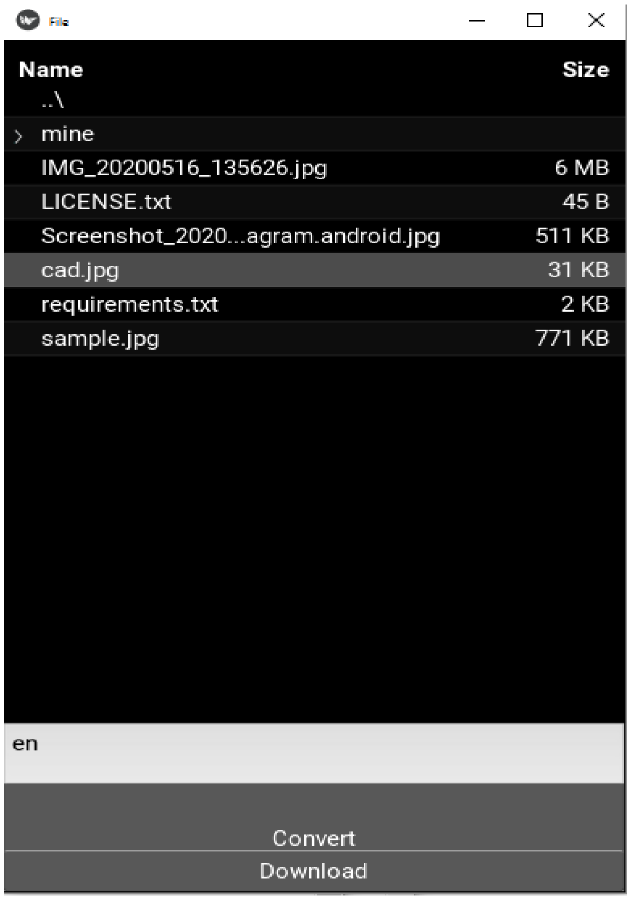
```
<PIL.JpegImagePlugin.JpegImageFile image mode=RGB size=2685x2173 at 0x1C51EC1034
8>
Questions
are the
Answers
source lang:  en
Entered text:  Des questions
sont les
Réponses
```

Fig 4.2.8: Text in the Image converted into desired language



"Aprender a dudar
es aprender a
pensar"

Octavio Paz



```
Name                                    Size
..\
> mine
IMG_20200516_135626.jpg                 6 MB
LICENSE.txt                             45 B
Screenshot_2020...agram.android.jpg   511 KB
cad.jpg                                31 KB
requirements.txt                        2 KB
sample.jpg                            771 KB
```

en

Fig 4.2.9: Cad.jpg **Fig 4.3.0 Selecting Image to convert**

In this module the selected image contains Spanish text and here the chosen language is English to which it needs to be converted.

```
C:\Users\venky\Desktop\cad.jpg
<PIL.JpegImagePlugin.JpegImageFile image mode=RGB size=700x466 at 0x1F8B9DFED88>
"Aprender a dudar
es aprender a
pensar"

Octavio Paz
source lang:  es
Entered text:  "Learning to doubt
is learning to
think"

Octavio Paz
```

Fig 4.3.1: Text in the image is converted to English

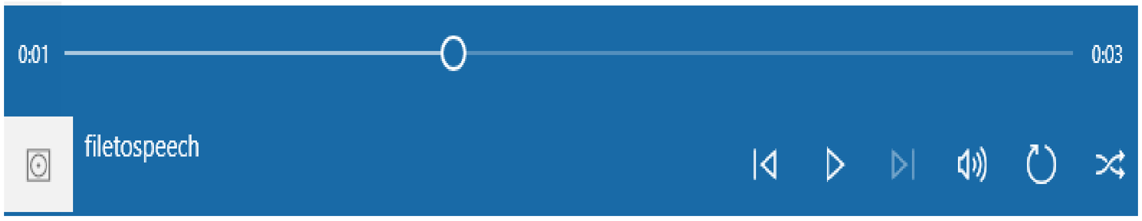


Fig 4.3.2: Text to speech converted audio

The default audio player will be deployed when we click on the convert button and the converted audio will start playing

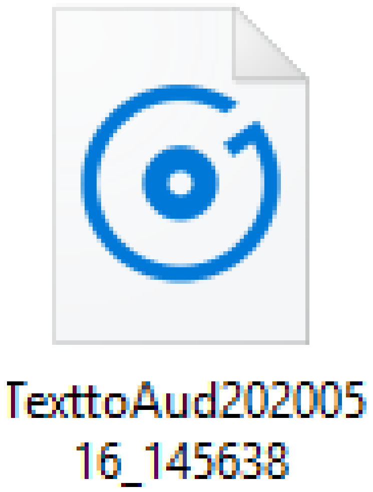


Fig 4.3.3: Downloaded Audio file

When the user Click on the Download option then the converted audio file will be downloaded and saved in the local memory along with the time stamp.

CHAPTER V

EXPERIMENTAL RESULTS

Chapter 5: Experimental Results

CONCLUSION

This system in general is considered an assistive technology tool that can be used in many ways like to help people who are visually impaired, who have trouble reading and also to help children with disabilities, people who have literacy issues and those trying to learn another language etc.

Speech synthesis has been developed steadily over the last decades and it has been incorporated into several new applications. For most applications, the intelligibility and comprehensibility of synthetic speech have reached the acceptable level. However, in prosodic, text preprocessing, and pronunciation fields there is still much work and improvements to be done to achieve more natural sounding speech. Natural speech has so many dynamic changes that perfect naturalness may be impossible to achieve. However, since the markets of speech synthesis related applications are increasing steadily, the interest for giving more efforts and funds into this research area is also increasing.

FUTURE SCOPE

There is a huge scope for future work for improving the application base. Several normal speech processing techniques may be used also with synthesized speech. For example, adding some reverberation it may be possible to increase the pleasantness of synthetic speech afterwards. Other effects, such as digital filtering, chorus, etc., can be also be used to generate different voices.

Some other techniques have been applied to speech synthesis, such as Artificial Neural Networks and Hidden Markov Models. These methods have been found promising for controlling the synthesizer parameters, such as gain, duration, and fundamental frequency. As mentioned earlier, the high-level synthesis is perhaps the least developed part of present synthesizers and needs special attention in the future. Especially controlling prosodic features has been found very difficult and the synthesized speech still sounds usually synthetic or monotonic. As long as speech synthesis needs to be developed, the evaluation and assessment play one of the most important roles.

It is quite clear that there is still very long way to go before text-to-speech synthesis, especially high-level synthesis, is fully acceptable. However, the development is going forward steadily and in the long run the technology seems to make progress faster than we can imagine.

CHAPTER VI

REFERENCES

Chapter 6: REFERENCES

1. https://medium.com/better-programming/an-introduction-to-pyttsx3-a-text-to-speech-converter-for-python-4a7e1ce825c3
2. https://kivy.org/doc/stable/gettingstarted/installation.html
3. https://pypi.org/project/googletrans/
4. https://github.com/pndurette/gTTS
5. https://stackabuse.com/pytesseract-simple-python-optical-character-recognition/
6. Chucai Yi, Yingi Tian, K.Anuradha, Text to Speech Conversion, IEEE Transaction on vol.19,pp.269-278, 2013
7. Digital Image Processing, Tata McGraw Hill Education Private Limited by S Jayaraman, S Esakkirajan, T Veerakumar
8. Text to Speech Synthesis System in Indian English 2016 IEEE Region 10 Conference (TENCON) Conversion of English Text-to-Speech (TTS)
9. Using Indian Speech Signal-ShanthaSelvaKumari, R.Sangeetha,International Journal of Scientific Engineering and Technology-Volume No.4 Issue No.8,01August2015.
10. Implementation of Text to Speech Conversion- Chaw Su Thu, Theingi Zin-International Journal of Engineering Research & Technology (IJERT)- Vol. 3Issue 3,March – 2014
11. S.M.K.Chaitanya, et,al, “Text to Speech Conversion on Intel Atom Processor, International Journal of Engineering Research-Online, Vol.3, Issue.3, 2015