Abstract

- In this project, Machine learning techniques are applied in order to help blind people by detecting the text in front of the person.

- In this project images are detected with the help of a device and result is passed to the user in voice mode. In this Application user can also perform certain actions like send text message by converting the text to speech.

- User will be opening the application using the "OK GOOGLE" command, which opens the application, the camera gets open and starts detecting the Text.

- In between User can also perform different operations like sending a text or voice message just like the mobiles message app.

Introduction

- There is always a challenge when developing a guiding system for blind people in the society and need physical and mental support.

- There are many people who are suffering from visually impaired or blindness, these people face a lot of difficulties in their day-to-day activities.

- The most difficult task for them is moving in an unfamiliar world of which they have no idea what is around them. Generally, people rely on their visions to familiarize with nearby surroundings around them.

- With this project people who are visually impaired or blind can learn and study the text freely around the real time environment without any support as they will be get all the information about the content around him through voice mode.

- This will help users to get a personal support to read, learn and send messages to others without anyone's help.

- With the help of this project, they can recognise different text just by pointing the camera around themselves to know the text and also translate the captured text in which ever languages they want.

Methodology

- pyttsx3 Module

  - An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. During construction, the engine initializes a pyttsx3.driver.

  - Driver Proxy object responsible for loading a speech engine driver implementation from the pyttsx3.driver's module.

  - After construction, an application uses the engine object to register and unregister event call backs; produce and stop speech; get and set speech engine properties; and start and stop event loops.

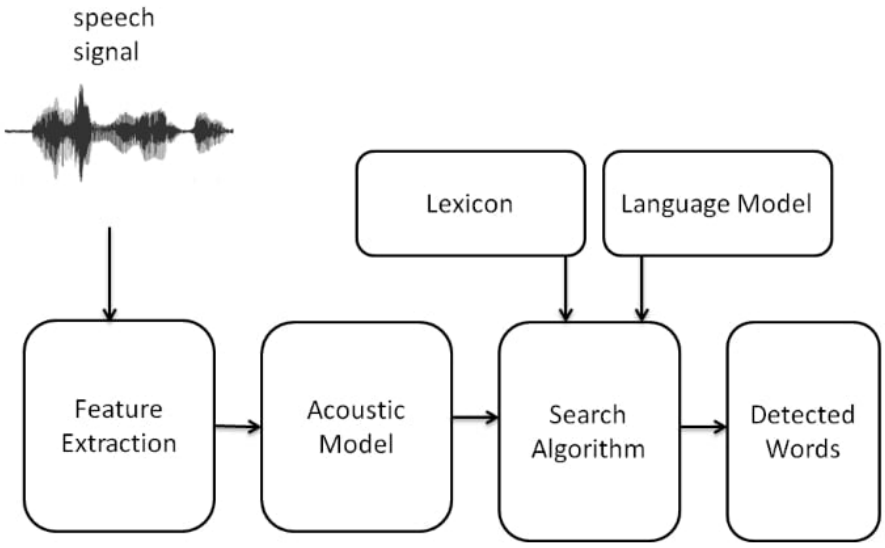

Fig-1 Flow diagram of pyttsx3

The Engine factory

pyttsx3.init([driverName : string, debug : bool]) → pyttsx3.Engine

Gets a reference to an engine instance that will use the given driver. If the requested driver is already in use by another engine instance, that engine is returned. Otherwise, a new engine is created.

➢ Speech Recognition

- ❖ This is a service which basically recognizes voice and converts it into text. It basically has many functions with it like post speaking, Buffer Speaking etc.
- ❖ For performing Speech Recognition, there is speech Recognition library which is open source and the best thing is that several engines and APIs provide in both modes online and offline mode. For Speech Recognition is in python.
- ❖ speech Recognition is used highest among the audience and it has many examples. Audio files provided in speech Recognition can be in AfIFF-C, AIFF, WAV format, commonly wav format is widely used.

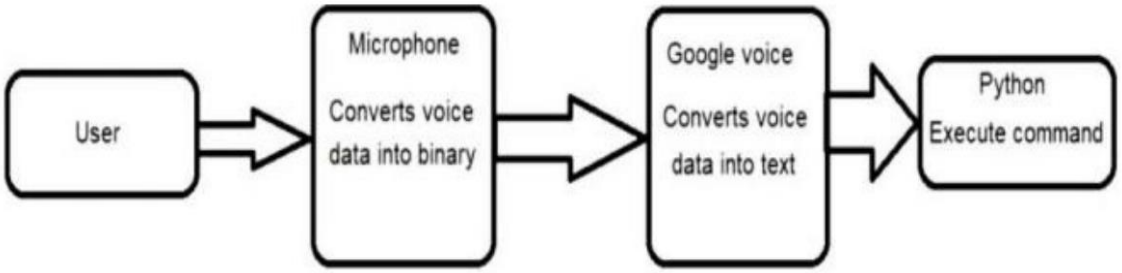

Fig-2 working of google's speech recognition

- ❖ As a result, we do not need to build any machine learning model from scratch, this library provides us with convenient wrappers for various well-known public speech recognition APIs (such as Google Cloud Speech API, IBM Speech To Text, etc.).

➢ pytesseract

Pytesseract or Python-tesseract is an OCR tool for python that also serves as a wrapper for the Tesseract-OCR Engine. It can read and recognize text in images and is commonly used in python ocr image to text use cases.

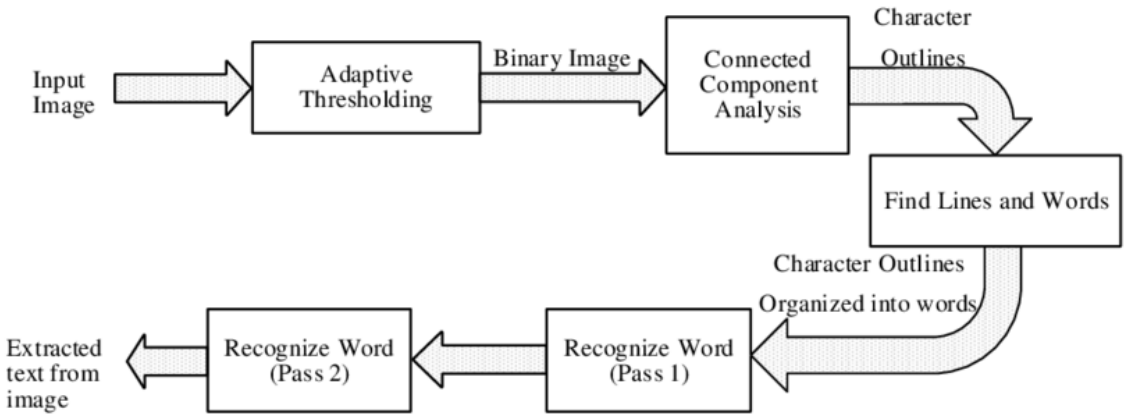

Fig-3 Flow diagram of recognizing text from image

It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, bmp, tiff, and others.

- gTTS

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 ensuring spacing after certain characters).

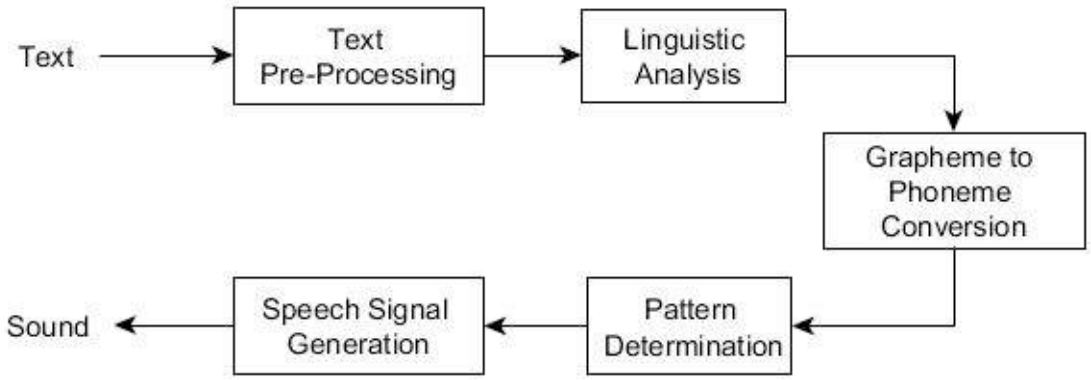

Fig-4 background working of converting text to speech

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.

➢ Googletrans

translate.google.com uses a token to authorize the requests. If you are not Google, you do have this token and will have to pay for use.

This class is the result of reverse engineering on the obfuscated and minified code used by Google to generate such token.

To detect the language of the given text we use detect function from the translator class present in the module.

The total languages that Googletrans can use are as follows:

```
import googletrans

print(googletrans.LANGUAGES)
```

```
{'af': 'afrikaans', 'sq': 'albanian', 'am': 'amharic', 'ar': 'arabic', 'hy': 'armenian', 'az': 'azerbaijani', 'eu': 'basque',
'be': 'belarusian', 'bn': 'bengali', 'bs': 'bosnian', 'bg': 'bulgarian', 'ca': 'catalan', 'ceb': 'cebuano', 'ny': 'chichewa',
'zh-cn': 'chinese (simplified)', 'zh-tw': 'chinese (traditional)', 'co': 'corsican', 'hr': 'croatian', 'cs': 'czech', 'da': 'da
nish', 'nl': 'dutch', 'en': 'english', 'eo': 'esperanto', 'et': 'estonian', 'tl': 'filipino', 'fi': 'finnish', 'fr': 'french',
'fy': 'frisian', 'gl': 'galician', 'ka': 'georgian', 'de': 'german', 'el': 'greek', 'gu': 'gujarati', 'ht': 'haitian creole',
'ha': 'hausa', 'haw': 'hawaiian', 'iw': 'hebrew', 'he': 'hebrew', 'hi': 'hindi', 'hmn': 'hmong', 'hu': 'hungarian', 'is': 'icel
andic', 'ig': 'igbo', 'id': 'indonesian', 'ga': 'irish', 'it': 'italian', 'ja': 'japanese', 'jw': 'javanese', 'kn': 'kannada',
'kk': 'kazakh', 'km': 'khmer', 'ko': 'korean', 'ku': 'kurdish (kurmanji)', 'ky': 'kyrgyz', 'lo': 'lao', 'la': 'latin', 'lv': 'l
atvian', 'lt': 'lithuanian', 'lb': 'luxembourgish', 'mk': 'macedonian', 'mg': 'malagasy', 'ms': 'malay', 'ml': 'malayalam', 'm
t': 'maltese', 'mi': 'maori', 'mr': 'marathi', 'mn': 'mongolian', 'my': 'myanmar (burmese)', 'ne': 'nepali', 'no': 'norwegian',
'or': 'odia', 'ps': 'pashto', 'fa': 'persian', 'pl': 'polish', 'pt': 'portuguese', 'pa': 'punjabi', 'ro': 'romanian', 'ru': 'ru
ssian', 'sm': 'samoan', 'gd': 'scots gaelic', 'sr': 'serbian', 'st': 'sesotho', 'sn': 'shona', 'sd': 'sindhi', 'si': 'sinhala',
'sk': 'slovak', 'sl': 'slovenian', 'so': 'somali', 'es': 'spanish', 'su': 'sundanese', 'sw': 'swahili', 'sv': 'swedish', 'tg':
'tajik', 'ta': 'tamil', 'te': 'telugu', 'th': 'thai', 'tr': 'turkish', 'uk': 'ukrainian', 'ur': 'urdu', 'ug': 'uyghur', 'uz':
'uzbek', 'vi': 'vietnamese', 'cy': 'welsh', 'xh': 'xhosa', 'yi': 'yiddish', 'yo': 'yoruba', 'zu': 'zulu'}
```

Fig-5 the available languages in google translator

We determine the language of the text with detect and print the result to the console. The method prints the language and the confidence value, which is the probability of the correct guess of the language.

Implementation

```
from gtts import gTTS
from googletrans import Translator,LANGUAGES
import speech_recognition as sr
import pytesseract as pytesseract
import pytesseract as tess
from PIL import Image
import os
import pyttsx3

engine = pyttsx3.init()
engine.runAndWait()
engine.setProperty('rate', 150)
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[0].id)

translator = Translator()
pytesseract.pytesseract.tesseract_cmd =
"C:\\Users\\Ravi\\AppData\\Local\\Tesseract-OCR\\tesseract.exe"


def I2T():
    img = Image.open("text.png")
    text = tess.image_to_string(img)
    print("The text was : ")
    print(text)
    ex = translator.detect(text)
    engine.say("The text was in :"+LANGUAGES[ex.lang])
    engine.runAndWait()

    engine.say("Do you Want to translate")
    engine.runAndWait()
    ch = input("Do you Want to translate : ")
    if ch == 'y':
        text, lan = T2T(text)
        T2S(text, lan)
```

```
    elif ch == 'n':
        T2S(text, ex.lang)


def T2S(text, lan):
    audio = gTTS(text=text, lang=lan, slow=False)
    audio.save("voice.mp3")
    os.system("start voice.mp3")


def S2T():
    r = sr.Recognizer()
    mic = sr.Microphone()

    with mic as source:
        engine.say("Say Something...!")
        print('Say Something...!')
        engine.runAndWait()
        audio = r.listen(source)
    print('end')
    try:
        print("\nYour message was : " + r.recognize_google(audio))
    except sr.UnknownValueError:
        engine.say("could not understand, can you speak again...!")
        print('could not understand, can you speak again...!')
        engine.runAndWait()
        S2T()
    except sr.RequestError as e:
        print('could not recognize the audio...!', e)


def T2T(message):
    trans = Translator()
    print("\n0 -> Hindi.\n1 -> Telugu.")
    print("2 -> Tamil.\n")
    engine.say("In with language do you want ")
    engine.runAndWait()
    ip = int(input("In with language do you want : "))
    match ip:
        case 0:
            l = "hi"
        case 1:
```

```
            l = "te"
        case 2:
            l = "ta"
        case default:
            l = "en"
    t = trans.translate(message, dest=l, src="en")
    return t.text, l


def option(a):
    match a:
        case 'l' | 'L':
            I2T()
        case 's' | 'S':
            S2T()
        case 'e' | 'E':
            exit(0)
        case default:
            print('Please enter correct option...!')


while 1:
    print("\n-----------------------------------------\n")
    print("L -> Listen to the image\nS -> Send a message.\nE -> Escape\n")
    opt = input('Enter your option : ')
    option(opt)

engine.runAndWait()
engine.stop()
```

Inputs

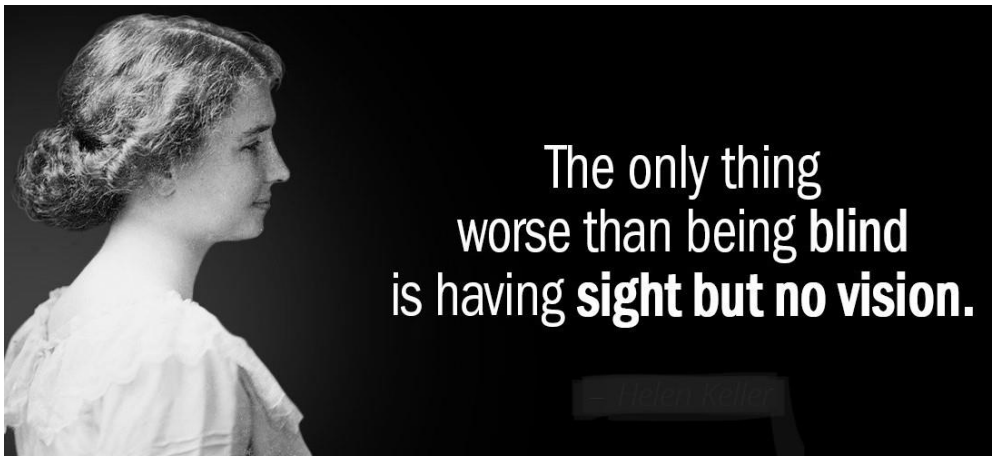

Fig-6 Input for capturing text from image

Outputs

The program continuously asks whether to get the text from the captured image or to give a text message from our voice.

Fig-7 The options available for the blind

Again, if the user selects to get the text, the program will detect and give the response about the language of the text.

```
Enter your option : 1
The text was :
The only thing
worse than being blind
is having sight but no vision.
```

If the user wants the text to be translated in a particular language, the program gives the basic 3 languages to translate.

```
Do you Want to translate : y

0 -> Hindi.
1 -> Telugu.
2 -> Tamil.

In which language do you want : 1
```

In order to send a message, the program captures the users voice and converts it into a text message.

```
L -> Listen to the image
S -> Send a message.
E -> Escape

Enter your option : s
Say Something...!
end

Your message was : how are you
```

If the voice was not recognized properly, it will again reply's to speak again.

```
L -> Listen to the image
S -> Send a message.
E -> Escape

Enter your option : s
Say Something...!
end
could not understand, can you speak again...!
Say Something...!
end

Your message was : this is a sample voice to test
```

Also if we didn't speak also the program will reply that we didn't speak

Limitations

- This system is working fine and it is covering all the modules but it has some limitations too. Also, the camera should be a better quality one.
- There comes little latency because the voice command is passed after all those detections, classification, distance calculation and finding the object which is closer to the user. so, this shows little latency.
- To use the google assistant we need 24/7 internet connectivity and minimal battery supply to run in any conditions.
- This application does not work at night because of visibility issues of the object. These all issues will work out as the technology is evolving at a very fast rate. so, a system with a good processor will cover all these limitations.

Conclusion

- There are many systems which are already developed which can give better results but they are very hard to carry out in normal day to day life. Even they are not feasible and more expensive.

- The proposed system discussed here will be easy for visually impaired or bind people to carry around. This system will assist the user to learn and read without anybody's help, also reply textually, or send a voice reply.

- This system will help them to communicate and read from surroundings. Using this application blind people interact with others and express their feelings.

Project Outcome

- From this project, we have learnt that how can we identify the problem from our surroundings and how can we help in tackling and giving a solution to this day-to-day problem using technology.

- From this project, we came to know various domains and what problems each domain has and how to select a particular problem and derive a particular solution path to the problem we choose. There might be many solutions to a problem but choosing the optimum solution made it easier for our problem.

- From this project, we have learnt how to improve our presentation skills, how to make an effective presentation and even learnt how to present our thoughts and implementation through this.

- From this project, team management skills improvised a lot, each one of us enthusiastically took each part of theirs and worked on it. We learnt how coordination can help make a bigger problem into a smaller problem and each person was the leader of the team in their way.

- From this project, communication skills have got better and how to represent ourselves in front of a crowd improved, communicating among ourselves also helped in making our work easier.

- From this project, we have learnt a new module in python like pytesseract, pyttsx3, gTTS, speechrecognition, pyaudio and googletrans frameworks which is used to implement complicated machine learning algorithm's using python.

- Finally, this project helped us to pave the path learning new skills, helped us to think out of the box, improve our thoughts towards society and how to take up a problem from our daily life and put some work on it to give a minimal solution which would positively benefit the society.

References

- "Using pyttsx3 — pyttsx3 2.6 documentation" https://pyttsx3.readthedocs.io/en/latest/engine.html
- Convert Text to Speech in Python - Java point https://www.javatpoint.com/how-to-convert-text-to-speech-in-python
- "Python: Convert Text to Speech [Beginner's Guide] - codingem.com" https://www.codingem.com/python-text-to-speech/
- Convert speech to text: https://www.thepythoncode.com/article/using-speech-recognition-to-convert-speech-to-text-python
- Implementation of speech recognition: https://www.simplilearn.com/tutorials/python-tutorial/speech-recognition-in-python