International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2018 IJSRCSEIT | Volume 3 | Issue 3 | ISSN : 2456-3307

Easy Reader: Audio Book listener

Anjali A. Thakre, Pooja J. Shivhare, Pragati V. Palsapure, Ashwini V. Bhoyar, Shruti P. Bhatkulkar
Information Technology, RTMNU/ K.D.K.C.E/ Nagpur, Maharashtra, India

ABSTRACT

In this paper, we represent our study about the text to speech, mainly it work on text which is converted into the speech or called audio. In this android application we can convert the pdf file into the audio using Google API's, usually the text get converted into the audio but not any file, specially its work for the pdf file. The people can easily use this application and it can take less time and memory. The development of a text to speech synthesizer will be of great help to people with visual impairment.

Keywords: speech synthesis, speech recognition, speech analysis and processing, pdf converter, text-to-phoneme conversion.

I. INTRODUCTION

An audio text is a recording of a text being read. A reading text is noted as unabridged while reading of reduce version or bridgement of the text are label as bridge. Audio text readings of text down by usually one reader. Human beings have long had an oral tradition .Audio text are in that oral tradition. Audio text can give you a perspective. Audio text and a new generation of listening devices are helpful for readers with limited vision and are in increasingly popular reading option for sighted audiences. This audio text is digital files that contain the same content as convension audio books. The text portion is display on a screen.

Human being long had an oral tradition. We use to sit around the campfire listening to stories. Children love having stories read to them. Audio books are in that oral tradition. More than that you can listen to audio books when you are doing something else. Many people get terribly car sick (well car sick, bus sick, plane sick…any kind of motion sick), and for them there is no way to read a book while in a car or a bus. Audio books are a fantastic on along car/bus/plane/boat journey. The just time files as you are taken away on an imaginary journey that is just like reading.

Visually impaired people are dependent solely on Braille books and audio recordings provided by NGO's. Owing to many constraints in above two approaches blind people cannot have book of their choice. The presented work will provide them an opportunity to have an audio book of their choice in English or Marathi language of any printed book having English, Marathi or Braille script. Printed text from text book having English, Marathi or Braille script will be taken as input in the form of an image which will be converted into plain editable text with the help of optical character recognition (OCR). This plain text will be then fed to text to speech (TTS) converter which will generate the audio output file in English or Marathi language corresponding to the input text image script. Printed book to audio book converter has been successfully implemented and satisfactory results were obtained [1].

II. PRODUCT DESCRIPTION

In order to reproduce the natural sound of each language, a narrator records a series of texts (poetry,

political news, sports results, stock exchange updates, etc.) which contain every possible sound in the chosen language. These recordings are then sliced and organized into an acoustic database. During database creation, all recorded speech is segmented into some or all of the following: diphones, syllables, morphemes, words, phrases, and sentences. To reproduce words from a text, the TTS system begins by carrying out a sophisticated linguistic analysis that transposes written text into phonetic text. A grammatical and syntactic analysis then enables the system to define how to pronounce each word in order to reconstruct the sense. We call this the prosody: it gives the rhythm and intonation of a sentence. Finally, the system produces information associating the phonetic writing with the tone and required length of the pronunciation. The chain of analysis ends here and sound is generated by selecting the best units stocked in the acoustic database.

speech synthesis:

Speech synthesis is a artificial production of a human speech. A computer system used for this purpose is called a speech computer or speech synthesizer, and can be implemented in a software or hardware product. A text-to-speech system converts normal language text into speech; other system renders symbolic linguistic representation like phonetic transcription into speech [2].

Synthesized speech can be created by concatenating Places of recorded speech that are stored in database. Systems differ in a size of the stored speech units; a system that stores phones or diphones provides the largest output range, but may lack clarity. For specific usages domains, the storage of entire words or sentences allows for high quality output. Alternatively, a synthesizer can incorporate a model of the vocal tract and other human voice characteristics to create a completely "synthetic" voice output. [3] The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood clearly. An intelligible text – to- speech program allows people with visual impairments or reading disabilities to listen to written words on a home computer.

SPEECH RECOGNITION:

Speech recognition is the inter-disciplinary sub-field of computational linguistics that develop methodologies and technologies that enable the recognition and translation of spoken language into text by computers. It is also known as "automatic speech recognition" (ASR), "computer speech recognition", or just "speech to text" (STT). It incorporates knowledge and research in the linguistics, computer science, and electrical engineering fields.

Some speech recognition systems require "training" (also called "enrollment") where an individual speaker read text or isolated vocabulary into the system. The system analyzes the person's specific voice and uses it to fine-tune the recognition of that person's speech, resulting in increased accuracy. Systems that do not use training are called "speaker independent" [4] systems. Systems that use training are called "speaker dependent".

Speech recognition applications include voice user interfaces such as voice dialing (e.g. "call home"), call routing (e.g. " I would like to make a collect call"), domotic appliance control, search (e.g. find a podcast where particular words were spoken), simple data entry (e.g., entering a credit card number), preparation of structured documents (e.g. a radiology report), speech-to-text processing (e.g., word processors or emails), And aircraft (usually termed direct voice input).

III. METHODOLOGY

As an emerging technology, not all developers are familiar with speech recognition technology. While

the basic functions of both speech synthesis and speech recognition takes only few minutes to understand (after all, most people learn to speak and listen by age two), there are subtle and powerful capabilities provided by computerized speech that developers will want to understand and utilize.

Despite very substantial investment in speech technology research over the last 40years, speech synthesis and speech recognition technologies still have significant limitations. Most importantly, speech technology does not always meet the high expectations of users familiar with natural human-to-human speech communication. Understanding the limitations - as well as the strengths - is important for effective use of speech input and output in a user interface and for understanding some of the advanced features of the Java Speech API.

An understanding of the capabilities and limitations of speech technology is also important for developers in making decisions about whether a particular application will benefit from the use of speech input and output. A speech synthesizer converts written text into spoken language. Speech synthesis is also referred to as
Text-to-speech (TTS) conversion. The major steps in producing speech from text are as follows:

Structure analysis:

Process the input text to determine where paragraphs, sentences and other structures start and end. For most languages, punctuation and formatting data are used in this stage.

Text pre-processing :

Analyze the input text for special constructs of the language. In English, special treatment is required for abbreviations, acronyms, dates, times, numbers, currency amounts, email addresses and many other forms. Other languages need special processing for these forms and most languages have other specialized requirements. The remaining steps convert the spoken text to speech.

Text-to-phoneme conversion:

Convert each word to *Phonemes*
A phoneme is a basic unit of sound in a language. US English has around 45 phonemes including the consonant and vowel sounds. For example, "times" is spoken as four phonemes "t ay m s". Different languages have different sets of sounds (different phonemes). For example, Japanese has fewer phonemes including sounds not found in English, such as "ts" in "tsunami".

Prosody analysis :

Process the sentence structure, words and phonemes to determine appropriate *prosody*
For the sentence. Prosody includes many of the features of speech other than the sounds of the words being spoken. This includes the pitch (or melody), the timing (or rhythm), the pausing, the speaking rate, the emphasis on words and many other features. Correct prosody is important for making speech sound right and for correctly conveying the meaning of a sentence.

Waveform production :

Finally, the phonemes and prosody information are used to produce the audio waveform for each sentence. There are many ways in which the speech can be produced from the phoneme and prosody information. Most current systems do it in one of two ways:

Concatenation Of chunks of recorded human speech, or *formant synthesis*
Using signal processing techniques based on knowledge of how phonemes sound and how prosody affects those phonemes. The details of

waveform generation are not typically important to application developers.

Speech Synthesis : javax.speech. synthesis

A speech synthesizer is a speech engine that converts text to speech. The javax.speech.synthesis package defines the Synthesizer interface to support speech synthesis plus a set of supporting classes and interfaces. As a type of speech engine, much of the functionality of a Synthesizer is inherited from the Engine interface in the javax.speech package and from other classes and interfaces in that package.

Create:

The Central class of javax.speech package is used to obtain a speech synthesizer by calling the create Synthesizer method. The Synthesizer Mode Descargument provides the information needed to locate an appropriate synthesizer. In this example a synthesizer that speaks English is requested.

Allocate and Resume:

Allocate and resume methods prepare the Synthesizer to produce speech by allocating all required resources and putting it in the RESUMED state.

Generate:

The speak Plain Text method requests the generation of synthesized speech from a string.

Deallocate:

The wait Engine State method blocks the caller until the Synthesizer is in the QUEUE_EMPTY state - until it has finished speaking the text. The deallocate method frees the synthesizer's resources.

Synthesizer as an Engine

The basic functionality provided by a Synthesizer is speaking text, management of a queue of text to be spoken and producing events as these functions proceed. The Synthesizer interface extends the Engine interface to provide this functionality. The following is a list of the functionality that the javax.speech. Synthesis package inherits from the javax.speech package and outlines some of the ways in which that functionality is specialized.

The properties of a speech engine defined by the Engine Mode Desc class apply to synthesizers. The Synthesizer Mode Desc class adds information about synthesizer voices

Synthesizers are searched, selected and created through the Central class in the javax.speech package.

Synthesizers inherit the basic state system of an engine from the Engine interface. The basic engine states are ALLOCATED, DEALLOCATED, ALLOCATING_RESOURCES and DEALLOCATING_RESOURCES for allocation state, and PAUSED and RESUMED for audio output state. The getEngineState method and other methods are inherited for monitoring engine state. An EngineEvent indicates state changes.

Synthesizers produce all the standard engine events The javax.speech.synthesis package also extends the EngineListener interface as SynthesizerListener to provide events that are specific to synthesizers.

Selecting Voices

Most speech synthesizers are able to produce a number of voices. In most cases voices attempt to sound natural and human, but some voices may be deliberately mechanical or robotic. The Voice class is used to encapsulate the four features that describe each voice: voice name, gender, age and speaking style. The voice name and speaking style are both String objects and the contents of those strings are determined by the synthesizer. Typical voice names might be "Victor", "Monica", "Ahmed", "Jose", "My Robot" or something completely different. Speaking styles might include "casual", "business", "robotic" or "happy" (or similar words in other languages) but the API does not impose any restrictions upon the speaking styles. For both voice name and speaking style, synthesizers are encouraged to use strings that

are meaningful to users so that they can make sensible judgements when selecting voices. By contrast the gender and age are both defined by the API so that programmatic selection is possible. The gender of a voice can be GENDER_FEMALE, GENDER_MALE, GENDER_NEUTRAL or GENDER_DONT_CARE. Male and female are hopefully self-explanatory. Gender neutral is intended for voices that are not clearly male or female such as some robotic or artificial voices. The "don't care" values are used when selecting a voice and the feature is not relevant.

20The age of a voice can be AGE_CHILD (up to 12 years), AGE_TEENAGER (13-19), AGE_YOUNGER_ADULT (20-40), AGE_MIDDLE_ADULT (40-60), AGE_OLDER_ADULT (60+), AGE_NEUTRAL, and AGE_DONT_CARE. Both gender and age are OR able values for both applications and engines. For example, an engine could specify a voice as: Voice("name", GENDER_MALE, AGE_CHILD | AGE_TEENAGER, "style");In the same way that mode descriptors are used by engines to describe themselves and by applications to select from amongst available engines, the Voice class is used both for description and selection. The match method of Voice allows an application to test whether an engine-provided voice has suitable properties.

IV. REQUIREMENTS

An Android is mobile operating system developed by Google, based on modified version of linux kernel and the open source software and design primarily for touch screen mobile devices such as smartphones and tablets. In addition Google has further developed Android TV for televisions, Android Auto for cars and Android Wear for Wrist Watches, each with a specialized user interface. Variants are Android also used on a game consoles, digital cameras, PCs and other electronics.

Initially developed by Android Inc., which Google bought in 2005, Android was unveiled in 2007, with the first commercial Android device launched in September 2008. The operating system has since gone through multiple major releases, with the current version 8.1 “Oreo”, released in December 2017. Android has growing selection of third party applications, which can be acquired by user by downloading and installing applications APK file, or by downloading them using an application store program that allows user to install, update and remove applications from their device. The mobile operating system began with the public release of the Android beta in November 5, 2017.

Java:

Java is a general purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers “write once run anywhere” (WORA), meaning that compiled java code can run on all platforms that support java without need for recompilation. Java applications are typically compile to byte code that can run on any Java virtual Machine (JVM) regardless of computer architecture. Java was originally developed by James Gosling at Sun Microsystem and released in 1995 as a core component of Sun Microsystem’s java platform. The latest version is Java 9, released on September 21, 2017, and is one of the two versions currently supported for free by Oracle.

The Java Development Kit (JDK) is an implementation of either one of the Java Platform, Standard Edition, Java Platform, Micro Edition platforms[5] released by Oracle Corporation in the form of a binary product aimed at Java developers on Solaris, Linux, macOS or Windows. The JDK includes a private JVM and a few other resources to finish the development of a Java Application[6]. Since the introduction of the java platform, it has been by

far the most widely used Software Development Kit (SDK).

V. DESIGN ENVIRONMENT

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Instant Run to push changes to your running app without building a new APK
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine
- This page provides an introduction to basic Android Studio features. For a summary of the latest changes, see Android Studio Release Notes.

VI. DESIGN AND IMPLEMENTATION

Our software is called the Android application, a simple and one page application with the text to speech functionality. The application is developed using java programming language. Java is used because it's robust and independent platform. The application is divided into two main modules- the main application module which include the basic Graphical user Interface component which handle the basic operations of the app such as input of parameter and files for conversation either via the file browser or direct keyboard input or the browser. This would make use of the open source API's provided by Google. The second main module, the main conversion engine which integrated into the main module is for acceptance of date and PDF files hence the conversation.

VII. CONCLUSION

Text to speech synthesis is a rapidly growing aspect of computer technology and is increasingly playing a more important role in the way we interact with the system. We have also developed a very simple and attractive graphical user interface application which allows the user to choose the any file for converting it in audio. This already exists in some tools not in app and not for the pdf file. Another area of further work is the implementation of a text to speech system and any other platforms where text to speech technology would be an added advantage and increase functionality.

VIII. FUTURE SCOPE

There is a future scope of this facility that many more features such as there is only three languages are used and in future will use many more languages. Another scope is to the audio files are convert into the files or text, it is also beneficial for everyone.

IX. REFERENCES

[1]. India Educators' Conference (TIIEC), 2013 Texas instruments.

[2]. Allen, Jonathan; Hunnicutt, M. Sharon; Klatt, Dennis (1987). From Text to Speech: The MITalk system. Cambridge University Press. ISBN 0-521-30641-8.

[3]. Rubin, P.; Baer, T.; Mermelstein, P. (1981). "An articulatory synthesizer for perceptual research".

Journal of the Acoustical Society of America. 70 (20): 321-328.

[4]. "Speaker Independent Connected Speech Recognition-Fifth Generation Computer Corporation". Fifthgen.com. Retrieved 15 June 2013.

[5]. "Java SE 7 Features and Enhancement". Oracle Corporation. Retrieved 1 January 2013.

[6]. "OpenJDK homepge". Oracle Corporation and/or its affiliates. Retrieved 1 January 2013.