Project Report
On

“ANDROID APPLICATION TO ASSIST BLIND PEOPLE IN READING TEXT IN NATIVE LANGUAGE”

Submitted in partial fulfillment for the award of degree

Of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

Submitted by

Suraj Gurav (16141238)

Madhuri Ghadge (16141139)

Aishwarya Swami (16141168)

Under the Guidance of

Prof. Y.D. Chavhan

Government College of Engineering, Karad

(An Autonomous Institute of Government of Maharashtra)

Academic Year 2019-2020

Government College of Engineering, Karad

(An Autonomous Institute of Government of Maharashtra)

Department of Information Technology

CERTIFICATE

This is to certify that the project entitled **"Android Application to assist blind people in reading text in native language"** has been carried out by team:

1.Suraj Gurav (16141238)

2. Madhuri Ghadge (16141139)

3.Aishwarya Swami (16141168)

of B. Tech IT class under the guidance of Prof. Y. D. Chavhan during the academic year 2019 -20 (Sem-VIII).

Prof. Y.D.Chavhan

Project Guide

Dr. S. J. Wagh
Head

Information Technology
Department

External Examiner

ACKNOWLEDGEMENT

Apart from individual efforts, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental throughout the project work.

It is our privilege to express our gratitude towards our project guide, Prof. Y.D. Chavhan, for their valuable guidance, encouragement, inspiration and whole-hearted cooperation throughout the project work. We thank him for being a motivation through all our highs and importantly, our lows.

We deeply express our sincere thanks to our Head of Department **Dr. S. J. Wagh** for encouraging and allowing us to present the project on the topic " **ANDROID APPLICATION TO ASSIST BLIND PEOPLE IN READING TEXT IN NATIVE LANGUAGE** " and providing us with the necessary facilities to enable us to fulfill our project requirements as best as possible. We take this opportunity to thank all faculty members and staff of Department of Information Technology, who have directly or indirectly helped our project.

We pay our respects to honorable Principal **Dr. A. T. Pise** for their encouragement. Our thanks and appreciations also go to our family and friends, who have been a source of encouragement and inspiration throughout the duration of the project.

TABLE OF CONTENTS

ABSTRACT

Blind people are unable to perform visual tasks. For instance, text reading requires the use of Braille reading system or a digital speech synthesizer (if the text is available in digital format). The majority of published printed works does not include Braille or audio versions, and digital versions are still a minority. On the other hand, blind people are not able to read the simple warnings on walls or signals that surround us. Thus, the development of a mobile application that can perform the image to speech conversion, whether it's a text written of a wall, a sheet of writing paper or in another support, has a great potential and utility.

The technology of optical character recognition (OCR) enables the recognition of texts from image data. This technology has been widely used in scanned or photographed documents, converting them into electronic copies, which one can edit, search, play its content and easily carry. The technology of speech synthesis (TTS) enables a text in digital format to be synthesized into human voice and played through an audio system. The objective of the TTS is the automatic conversion of sentences, without restrictions, into spoken discourse in a natural language, resembling the spoken form of the same text, by a native speaker of the language. This technology has had significant progress over the last decade, with many systems being able to generate a synthetic speech very close to the natural voice. Research in the area of speech synthesis has grown as a result of its increasing importance in many new applications.

LIST OF FIGURES

ABBREVATIONS

Acronym	Definition
OCR	Optical Character Recognition
XML	Extensible Markup Language
API	Application Programming Interface
GUI	Graphical User Interface
TTS	Text to Speech
CLI	Command Line Interface

Chapter 1
INTRODUCTION

Background:

The visually impaired people face many problems in their day to day life like reading texts on walls, streets, messages on the boards, signals for the vehicles, etc. Many Braille systems for any language and digital synthesizer for digital text are available for those people to read the text. But the published work is mostly in the form of regional languages or in the English. Those formats are not understandable by blind people which may restrict their capabilities to gain the knowledge from all the fields. To support them there is need of providing the applications which may give the result understandable by those people. Such applications scan and read the text documents, text images and translate it and convert them in the electronic form. And will also help in Translating to there native language.

Importance of the project:

Congenital position in the eye, eye injuries, eye issues, brain trauma and many other diseases causes visual impairments of the people. Almost 50% blind population of the world is only from the India. As those people are the part of our society, they have a right to learn and develop themselves in any field, so that the blind students are provided with Braille language books and plates, Audio books or the personal assistant for education. But the availability of the study material is too less to gain knowledge of each and every field which student has an interest and which he wants to gain. Thus, availability of material is not much effective as it limits their capabilities.

The objective of the project is to assist the blind people and provide them the technical solution which overcomes the above problem. The application provides access to any text resource maybe text documents, text images, street boards, traffic signals, text plates, etc. and provides the output in the audio format effectively and translates the text. The android application is supporting all the android mobile devices which are easy to carry anywhere with the user. The earlier developing applications need to capture and load the images to further process to recognize characters. This application overcomes this issue by capturing and automatically processing all the activities to load, to extract, to clean, to recognize, to translate and converting into audio format.

MOTIVATION:

Human communication in the society enhances via watching, reading, hearing, writing and interacting. Blind people are the integral part of human society but they can only listen the things and are unable to do other activities which needs normal vision. They are dependent on others for the basis activities like travelling, reading, shopping, etc. Blind people are unable to read simple text on boards, walls, etc. The most printed documents are in regional, national language or in English and doesn't support the Braille or digital systems. Thus, the need for an android application is increasing.

SCOPE:

The scope of our project is to provide an efficient and enhanced software tool for the users to perform document image analysis, document processing, by reading and recognizing the characters that are having large pool of documented, scanned images. Irrespective of the size of documents and the type of characters in documents, the product is recognizing them, searching them and processing them faster according to the needs of the environment.

EXPECTED OUTCOME:

- The project provides a clear concept for the text reading in English as an input in regional language (Hindi or Marathi) for blind people and also it translates the text in any other language.
- The main advantages of this project are it requires less consumption of time in recognizing and reading text with lower operational costs and also text of different fonts can be recognized.

Chapter 2

LITERATURE SURVEY

1. A Detailed Study and Analysis of OCR Research in South Indian Scripts:[1]
   Author: M. Abdul Rahiman and M. S. Rajasree
   Abstract: This paper provides an overview of the OCR research in south Indian languages. OCR reading technology is benefited by the evolution of high-powered desktop computing allowing for the development of more powerful recognition software that can read a variety of common printed fonts and handwritten texts.

2. Implementation of an Optical Character Reader (OCR) for Bengali Language:[2]
   Author: Muhammed Tawfiq Chowdhury, Md. Saiful Islam, Baijed Hossain Bipul and Md. Khalipur Rhaman
   Abstract: Optical Character Recognition (OCR) is the process of extracting text from an image. The main purpose of an OCR is to make editable documents or image files. Significant number of algorithms is required to develop an OCR and basically it works in two phases such as character and word detection.

3. Review of the Character Recognition System Process and Optical Character Recognition Approach:[3]
   Author: Jaswinder Kaur and Mrs. Rupinder Kaur
   Abstract: Character recognition in pictures could be an analysis space that makes an attempt to develop a computing system with the power to mechanically scan the text from pictures. Currently there’s a large demand in storing data the knowledge the data out there in paper document format into a hardware disk and so later reusing this information by looking method.

Chapter 3

RELATED THEORY AND PROBLEM DEFINITION

Problem Definition:

Among 37 billion blinds across world 15 billion are from India. There are various solutions available for the blind people using different technologies like OCR technology. In early days OCR technology was only used with computers but now a days we can use this technology with android mobile devices also. There are several applications in market to read books, text images etc., in audio format but that are less effective as they need to capture image by users and to load it in application. [4] We are providing the OCR for android application which once started continuously capture and loads images and recognize text which gives speech as an output until user stops the application. Application is available in real time and in regional language to effectively convert text in any language and then speak for visually impaired persons. Simple OCR algorithm and libraries extract text from images and android application provides front end interface to the users supporting to all android mobile devices,

Related Theory:

- **OCR Technology:**

OCR is a widely used technology to recognize the printed, hidden, handwritten, typed text or the text on nameplates, boards of buses, trains, cabins, shops, advertises, text superimposed on the images etc. and to convert that text to machine readable language. After the document or text scanning it is stored as a bitmap file in the form of TIF file format. For the computers the text Images are just like dots. OCR [5] technology is developing towards giving the best solutions for text recognition from the 1990s till now.

Firstly, the OCR is made to help the sight-impaired people and till date it is developed large scale to search the text or digits in large dataset, to sort the mails, magazines, letters etc. Zonal OCR is automating the complex text documents. There are variety of OCR techniques available in the market for the various application purpose. Few of them are as follows:

- **Pre-processing:**

OCR pre-processes the input images to make further implementation of algorithm effectively towards the success.[6]

- De-skew: The non-aligned document has to rotate few degrees in clockwise direction or in anti-clockwise direction in order to make it perfectly vertical or horizontal.
- Despeckle: Removal of positive and negative dots, smoothening of an edge of image.
- Binarisation: Image conversion from colored and greyscale image to black-white type in age. The task is to separate the text from the background.
- Removal of Lines: Cleaning of boxes and lines in the image that are not required.
- Layout analysis: Identification of the columns, rows/lines, paragraphs etc. as identifiable blocks. It is mainly important in case of multiple column layout with tables.
- Word or Line detection: Develops the baseline for word or character size and shapes and makes division in word if it is necessary.
- Recognition of Script: In the documents containing multiple languages, the format of the script file can be changed with respect to words so identifying the script file is necessary for that the expected OCR may be used to control the particular script file.
- Segmentation or Isolation of Characters: Separation of multiple characters connected with the image artifacts is required. Single characters broken into multiple pieces due to artifacts must be connected. The broken single character in many parts because of artifacts should be connected for identification.
- Normalizing the scale and aspect ratio

- **Character recognition:**

Two types of OCR algorithms are available to make candidate characters' ordered list:

- The input image is compared with the image stored for each pixel is called as matrix/pattern matching, pattern recognition or image correlation[1]. Correct isolation of image characters depends on the input images containing the curved or curly characters rather than other chars and also it is taken in consideration that in which font and scale the glyph is stored. This method[7] works better with typewritten words/text and doesn't work well with new fonts added in the file.

- The features like lines, its closed loops, directions and intersections are extracted from the input images[2]. Due to this the representation dimensionality is reduced for better efficiency of recognition activity. The extracted features are tried to match with the vector of character representation. To compare input image features with stored image features, classifiers like k-nearest neighbors' algorithm are needed and the nearest path that is available is chosen.
- Data records on the printed papers is gone through the information entry in a large scale for such entry forms like documents of passport, cheques, statement of bank, electronic receipts, mail, prints or other documents are required the pre-processing before recognition process. Printed text is converted in digital form to enable the editing, searching, storing, displaying online, used in cognitive computing, machine language translation, TTS.

➢ **Post-processing:**

- By listing the words allowed to occur in the output document, the accuracy of an OCR is increased. In all the words in the English language. In case when the words out of the listed lexicon occurs in the document then accuracy of the recognition reduces very deliberately. The dictionary of Tesseract is used to enhance the segmentation of character in order to improve accuracy of OCR.
- In many cases the output is in the form of text or char files but in more advance OCR systems, the output matches the layout of an input format. E.g. Pdf containing root image and its textual representation which is searchable.
- The frequencies of the characters occurring together are used for the analysis of nearer neighbor for correcting the occurred errors by analyzing that certain words are often seen together. E.g. "ing", " ed" etc.
- Information about the grammar of language used in input document to be scanned helps in guessing nouns and verbs for increased performance. For optimizing the outputs of OCR API, the algorithm of Levenshtein Distance is used in OCR post processing.

- **Tesseract Application:**

- Tesseract is an open source OCR engine having greater accuracy and is a free software with Apache License, Version 2.0 that can be run on different platforms. In 2006, Tesseract was one of the engines from top three with respect to character accuracy.
- Tesseract 3.9 supports formatting of an output text, information position and analysis of page-layout. To support the formats of new images the library called Leptonica is added Tesseract can identify the monospaced or proportionally spaced text Tesseract is trained to work in many languages like English, Malayalam, and Afrikaans etc. [8] Tesseract processes right-to-left text like Arabic or Hebrew language.
- In case of the unprocessed input images the tesseract gives very poor-quality output, Images must be scaled up to 20 pixels, rotating an image or an orientation should be corrected otherwise text detection will become hard. Binarization of Tesseract stage will delete much more of the page, and the dark borders should be removed to avoid misinterpretation of characters.
- Tesseract is run from the CLI (command-line interface) as it is not having GUI, but various projects supplies GUI for it.

- **Text to Speech:**

Google Cloud TTS enables programmers to analyze the natural voice with 30 different spices in multiple languages. It uses the research of DeepMind for WaveNet and the neural networks of Google for delivering the audio of high fidelity. Using those APIs, you can make inter actions with users all over the world across many applications and the devices.

- **jTessBox Editor:**

In the Tesseract OCR the training is done via Box Editor with full automation. It is used for both the versions of Tesseract 2.0 and 3.0 too. It can read basic formats of images with the TIFF many pages, Program requires JAVA Runtime Environment-7 or later. It is rewritten in Java gauge to render the issue of the complex JavaScript.

The box or TIFF files are provided as an input to the box editor. The type of input images should be of 300 DPI and 1 ppb black-white or 8bpp grey scale uncompressed images. The Box files are generated in UTF-8 format by Tesseract[8] executables or by the box generator

- **Gradle NDK:**

To make the library project as the Gradle build dependency, the Gradle is provided by allocating the path of CMake or ndk-build script file. As an application is executed the gradle run the CMake or ndk-build script file to share the libraries with the APK and the files to include in the android project are guessed by gradle with build script files. For any project, it is necessary to create build-script file initially if not available before further procedures

Every module in the project is linked with the only one ndk-build script file. For the number of modules in project there is need to create number of build script files. The CMakeLists.Txt is then used as a top-level build script for the project and other script files as the dependencies of that top script file. For ndk-build the Android-mk script file can be used to include other Makefiles in project.

- **Tess-two with Gradle:**

To integrate tesst-two with gradle android project there are following steps:

1. Create a libraries folder in the main directory of the project. For example, if project is First Project, create a First Project/libraries folder in main directory.
2. Copy tess-two directory in the libraries folder just created.
3. Delete the libs folder in the tess-two directory, project.properties, build.xml, .classpath and .project files as.
4. Create a build.gradle file in the tess-two directory:

Cuneiform and Tesseract software uses two pass method for recognizing characters. This pass is also called as adaptive type recognition and it uses the shapes of letters that are recognized. To recognize the letters in second pass the letters recognized in first pass are taken in account for better recognition efficiency. This is advantageous in case of unusual fonts or low-quality scanned docs or images where the font is blurred.

Use:

- For using tess two in the application, modify the build gradle build file of app module in order to insert tess two dependency externally in the project application. Dependency is:

  ```
  Dependencies
  {
     implementation 'com.rmtheis:tess-two:9,0.0
  }
  ```

- Open build.gradle and add Mobile Vision dependencies in it as follows:

  ```
  Dependencies
  {
  implementation file Tree(dir:'libs’, include:[‘*jar’])
  implementation ‘com.android.support:support-v4:26.1.0’
   implementation ‘com.android.support:design:26.1.0’
   implementation ‘ com.google.android.gms:play-service-vision:15.0.0’
  }
  ```

Chapter 4

DESIGN METHODOLOGY

Proposed system Architecture:

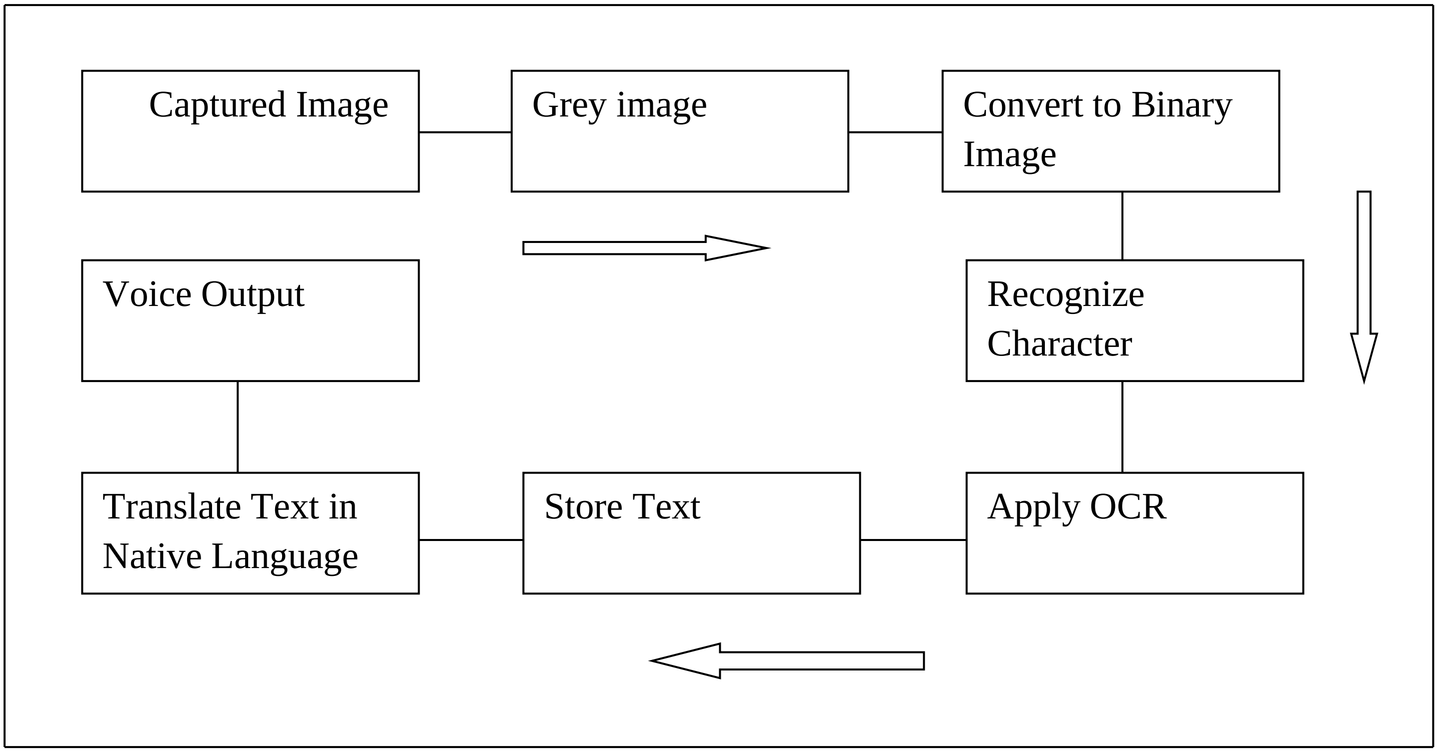


Fig.4.1: System architecture

Internal Logic of system:

The project consists of OCR technology, JBox Text Editor, VIET OCR, Android Studio, NDK, ANT Java Package Manager, text to speech and Tesseract Command Line Interface. The OCR technology consists preprocessing, character recognition and post-processing. The output text is converted into voice by text to speech conversion method.

OCR Technique: The optical character recognition is used to recognize the characters from the input images provided. The OCR uses character recognition algorithm. The logic of algorithm is as follows:

- Loading the image of any type in any format from the source as given.
- Converting input text image into the gray-scale image and then do binarization of it using Otsu algorithm with the threshold value
- Detecting features of image as resolution or inversion to convert it in a simple image to next processing.
- Detect the lines and remove it to improve the analysis of page layout to gain better quality of recognition for the character or text.
- Analysis of Page Layout: identifying the text set available in image to use that portion for recognition.
- Detecting text lines or words using variety of fonts and the spaces in words.
- Recognition of text: Each character image should be changed into right code of character. At times the algorithm generates number of character codes for one image. E.g. Recognition of ‘I’ may generate ‘I’, ‘|’, ‘l’ and ‘l’ codes. The resultant character code is chosen afterwards.
- Results are saved in chosen output format as pdf, doc, rtf and txt. Previous layout of page is saved.

Image Processing: The image is captured the camera of an Android Mobile device and then is used by an application. Image pre-processing is done to avoid the blurriness, extra brightness and the noise in image to clean it for the best performance.

Character Recognition: The pre-processed image is further uses algorithm for character recognition. The recognized characters are compared with patterns for semantic analysis. The matched words are then treated as recognized words.

Text to Speech Conversion: The extracted words from the text image or document are then conned into voice output for the user of an application

Data Flow and UML Diagram

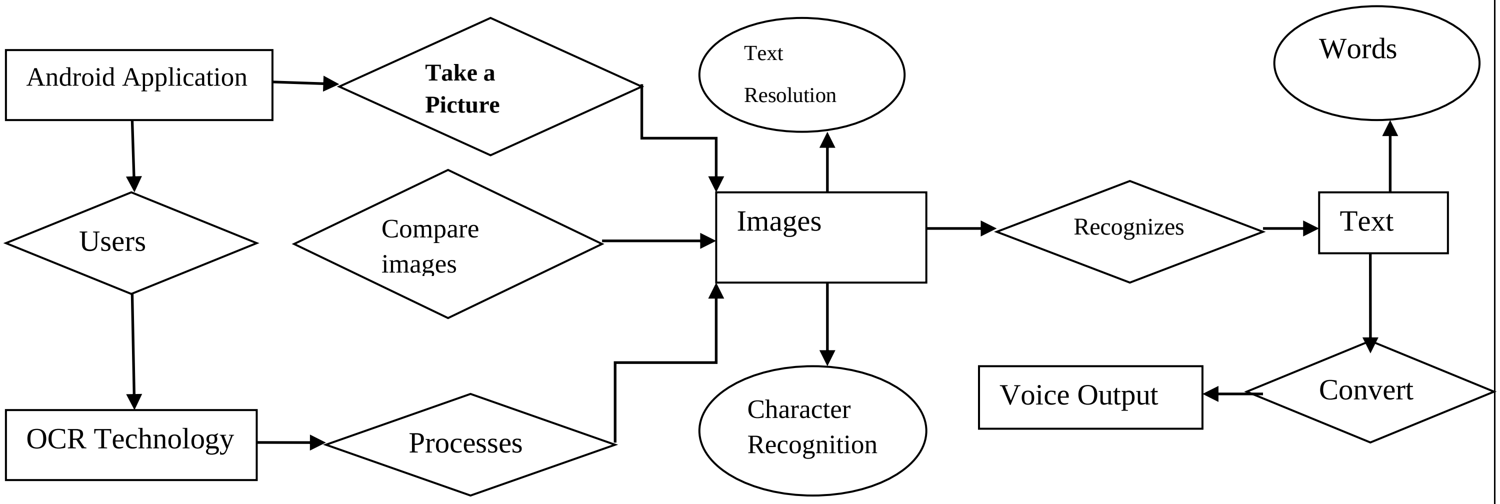


Fig.4.2: Overall Flow of System

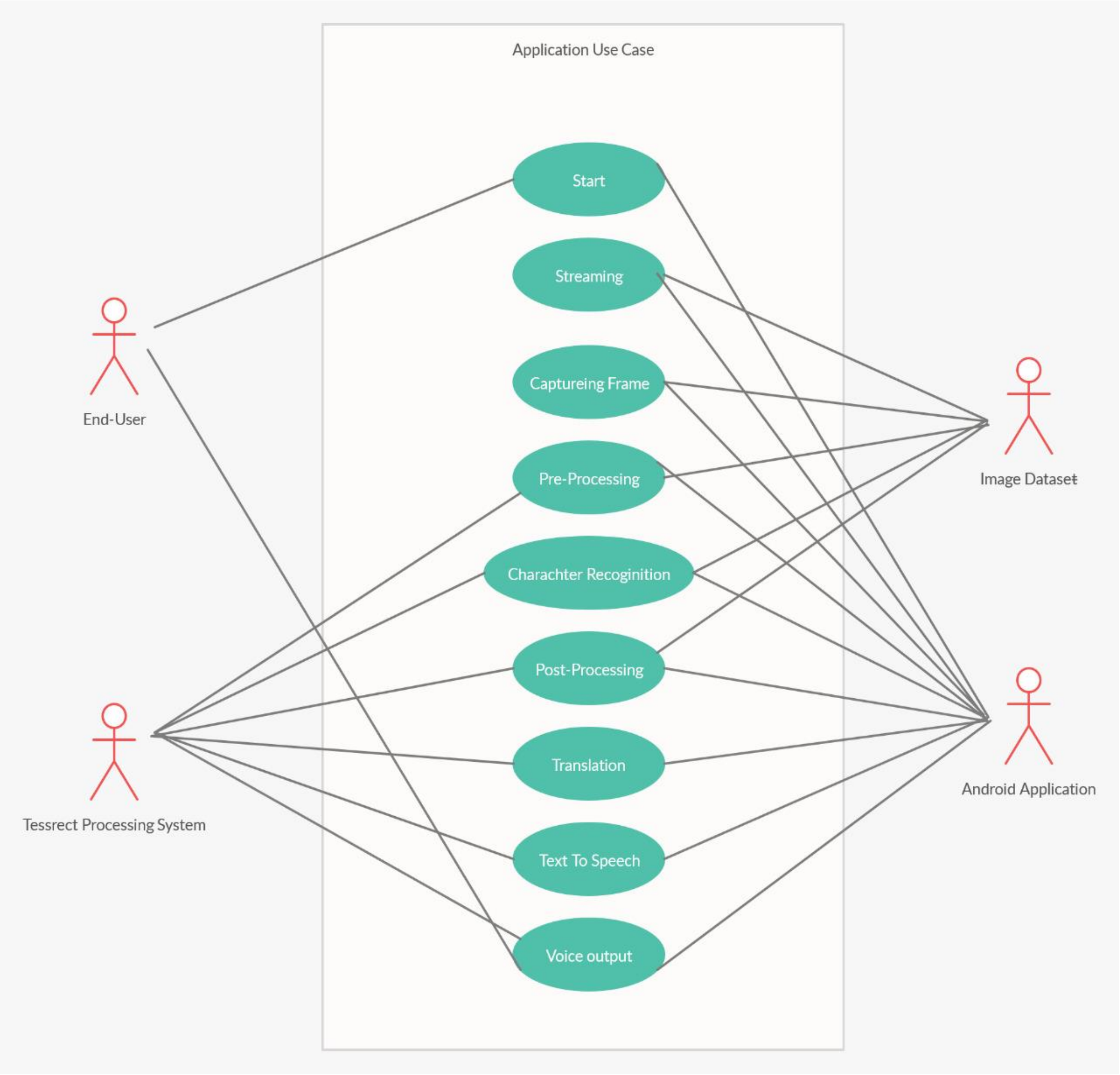


Fig.4.3: Use Case Diagram

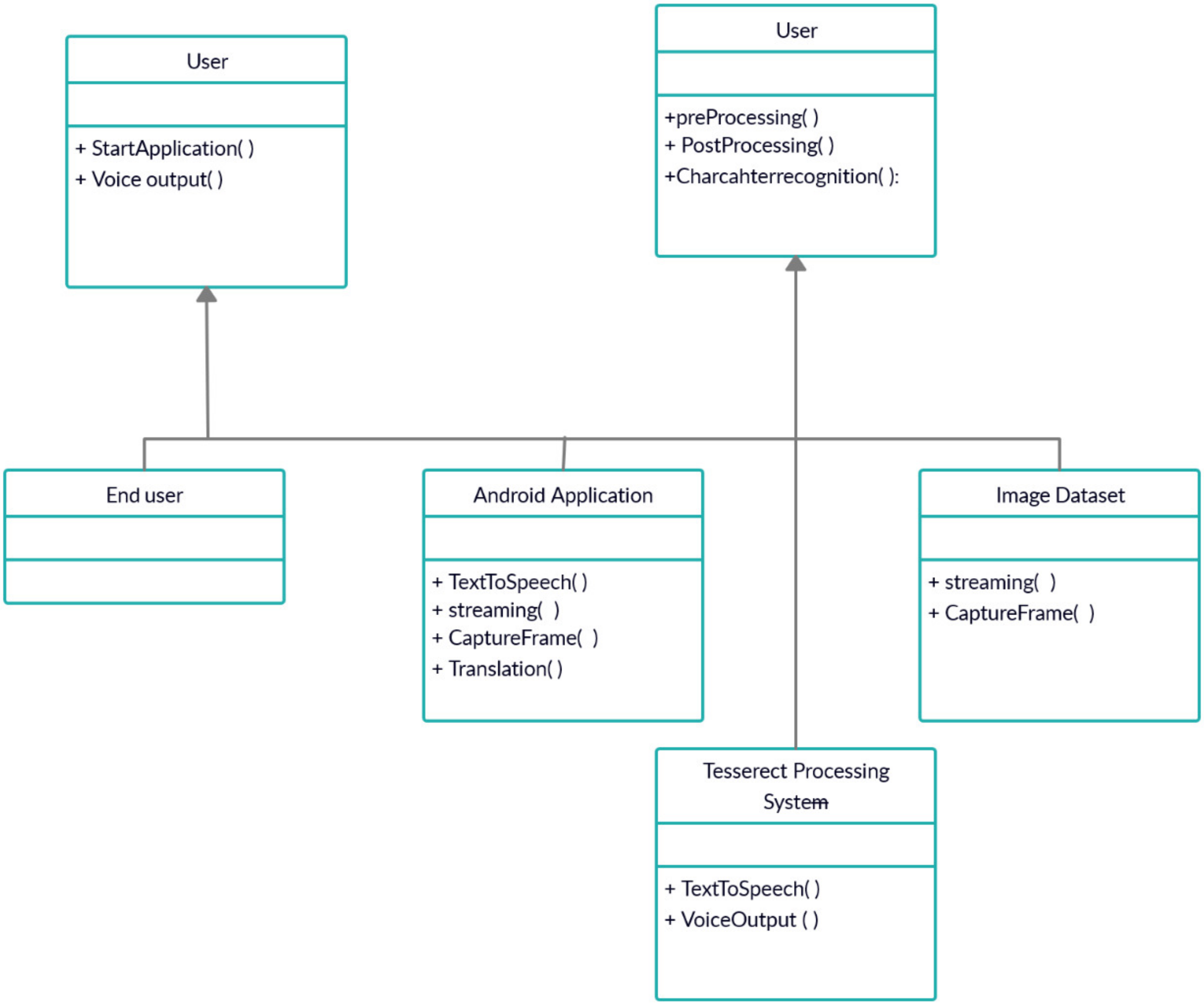


Fig.4.4: Class Diagram

Technical Specifications:

Software Specification:

- Tesseract Engine(version 3.4)
- jBox Tess Editor
- VIET OCR
- NDK Build
- Android Studio(version 3.1)
- ANT (Java Package Manager)

Hardware Specification:

- Android Mobile Device with High Resolution Camera
- Connector Cable
- High Speech and Audibility Ear-phones

Chapter 5

IMPLEMENTATION

Implementation of Proposed System:

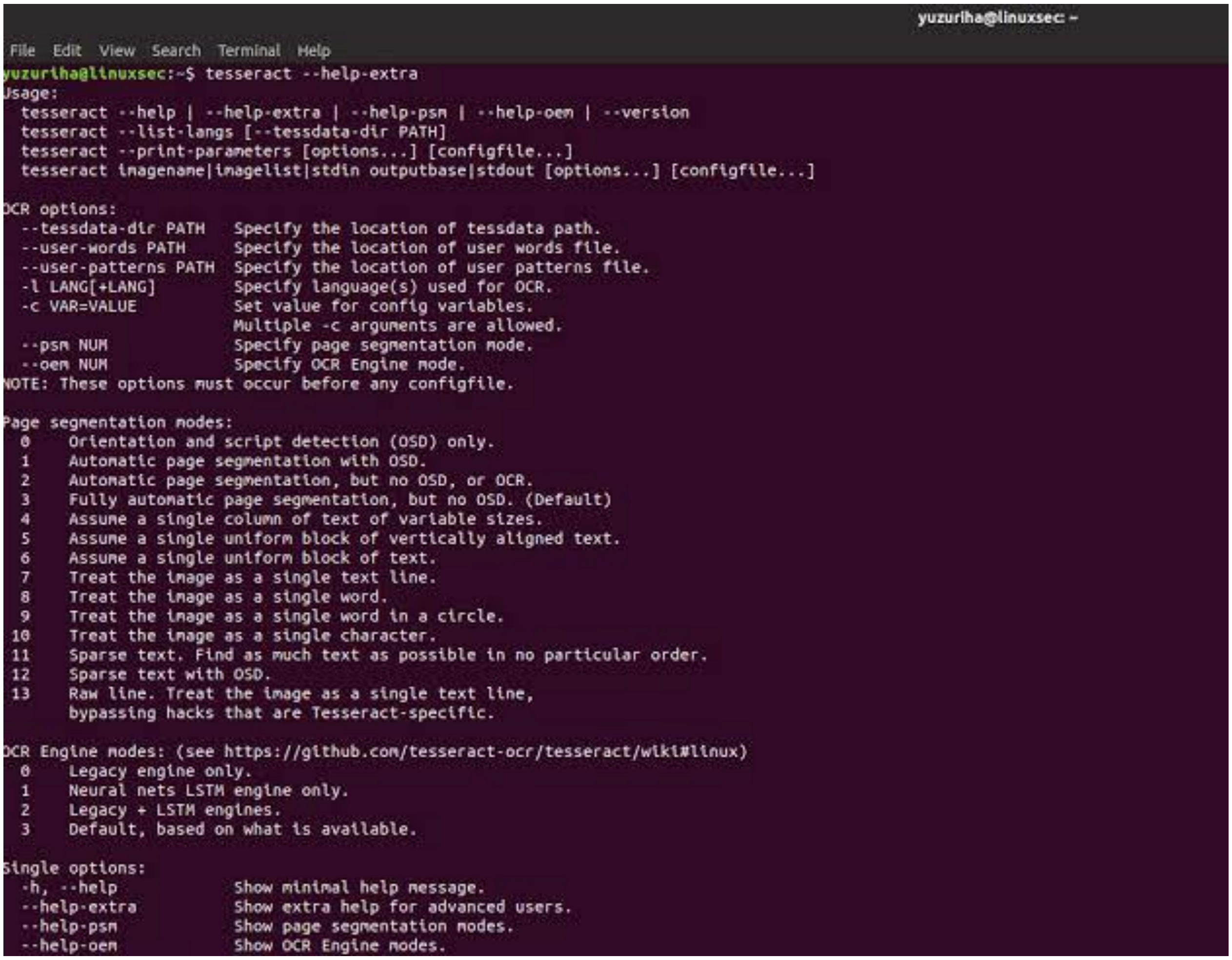
Code for Tesseract to implement OCR:

```
Public TessOCR()

{

    //TODO Auto-generated constructor stub

    mTess = new TessBaseAPI();

    String datapath = Environment.getExternalStorageDirectory()+ “/tesseract/”;

    String Language = “eng”;

     File dir= new File(datapath + “tessdata/”);

     If(!dir.exists())

            Dir.mkdirs();

            mTess.init (datapath,  Language);

}

Public String getOCRResult(Bitmap bitmap)

{

     mTess.setImage(bitmap);

     String result = mTess.getUTF8Text();

     return result;

}
```

Implementation:

Tesseract Activity to Implement OCR:

The installed Tesseract application is executing for an OCR and asking for the language specifications used for OCR conversion. It is necessary here to specify it while configuring it. The tesseract is available for variety of languages like English, Malayam, Afrikaans and many Albanian languages.

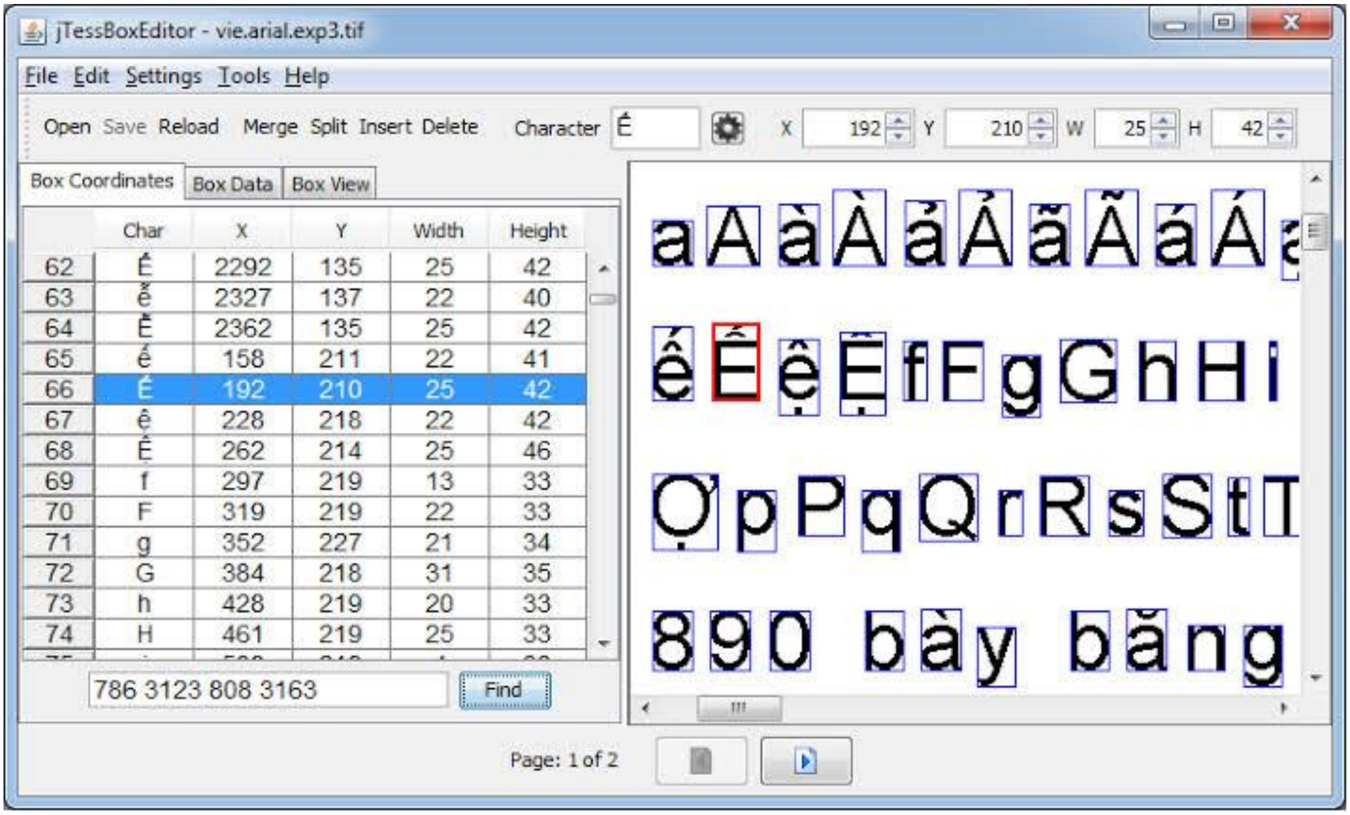


```
yuzuriha@linuxsec:~$ tesseract --help-extra
Usage:
  tesseract --help | --help-extra | --help-psm | --help-oem | --version
  tesseract --list-langs [--tessdata-dir PATH]
  tesseract --print-parameters [options...] [configfile...]
  tesseract imagename|imagelist|stdin outputbase|stdout [options...] [configfile...]

OCR options:
  --tessdata-dir PATH   Specify the location of tessdata path.
  --user-words PATH     Specify the location of user words file.
  --user-patterns PATH  Specify the location of user patterns file.
  -l LANG[+LANG]        Specify language(s) used for OCR.
  -c VAR=VALUE          Set value for config variables.
                        Multiple -c arguments are allowed.
  --psm NUM             Specify page segmentation mode.
  --oem NUM             Specify OCR Engine mode.
NOTE: These options must occur before any configfile.

Page segmentation modes:
  0    Orientation and script detection (OSD) only.
  1    Automatic page segmentation with OSD.
  2    Automatic page segmentation, but no OSD, or OCR.
  3    Fully automatic page segmentation, but no OSD. (Default)
  4    Assume a single column of text of variable sizes.
  5    Assume a single uniform block of vertically aligned text.
  6    Assume a single uniform block of text.
  7    Treat the image as a single text line.
  8    Treat the image as a single word.
  9    Treat the image as a single word in a circle.
 10    Treat the image as a single character.
 11    Sparse text. Find as much text as possible in no particular order.
 12    Sparse text with OSD.
 13    Raw line. Treat the image as a single text line,
       bypassing hacks that are Tesseract-specific.

OCR Engine modes: (see https://github.com/tesseract-ocr/tesseract/wiki#linux)
  0    Legacy engine only.
  1    Neural nets LSTM engine only.
  2    Legacy + LSTM engines.
  3    Default, based on what is available.

Single options:
  -h, --help            Show minimal help message.
  --help-extra          Show extra help for advanced users.
  --help-psm            Show page segmentation modes.
  --help-oem            Show OCR Engine modes.
```

Fig.5.1:Teserract Implementation

Box Editor in OCR:

The below shown box editor executes the input files that are in box or TIFF format, 1bpp black-and-white or 8bpp grey scale images. The output is the code for each character.


jTessBoxEditor - vie.arial.exp3.tif
File Edit Settings Tools Help
Open Save Reload Merge Split Insert Delete
Box Coordinates
Box Data
Box View
Find
aAàÀảẢãÃáÁ
ếẾệỆfFgGhHi
ỢpPqQrRsStT
890 bày băng

Character Ế X 192 Y 210 W 25 H 42

	Char	X	Y	Width	Height
62	Ế	2292	135	25	42
63	ễ	2327	137	22	40
64	Ễ	2362	135	25	42
65	ế	158	211	22	41
66	Ế	192	210	25	42
67	ệ	228	218	22	42
68	Ệ	262	214	25	46
69	f	297	219	13	33
70	F	319	219	22	33
71	g	352	227	21	34
72	G	384	218	31	35
73	h	428	219	20	33
74	H	461	219	25	33

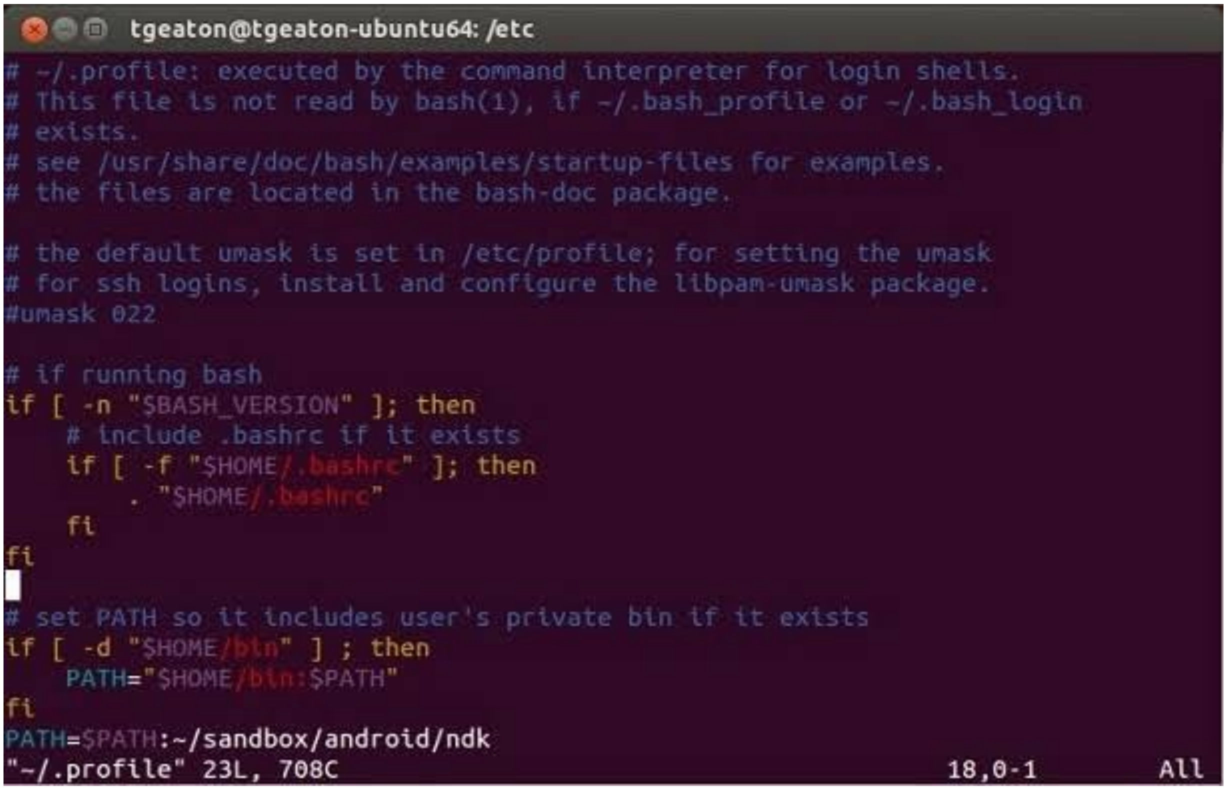
786 3123 808 3163

Page: 1 of 2

Fig.5.2:Box Editor in OCR

NDK-Build:

NDK-build script files are initially provided to any android project. The CMake or NDK-build script files are executed by the gradle for any application to share the files with the APK. Every module of project requires one separate NDK-build script file so that the number of modules is provided with the same number of NDK-build script files. One CMakeLists.txt file is made on the top of directory and other files are given dependencies with this file in order to work collaboratively.

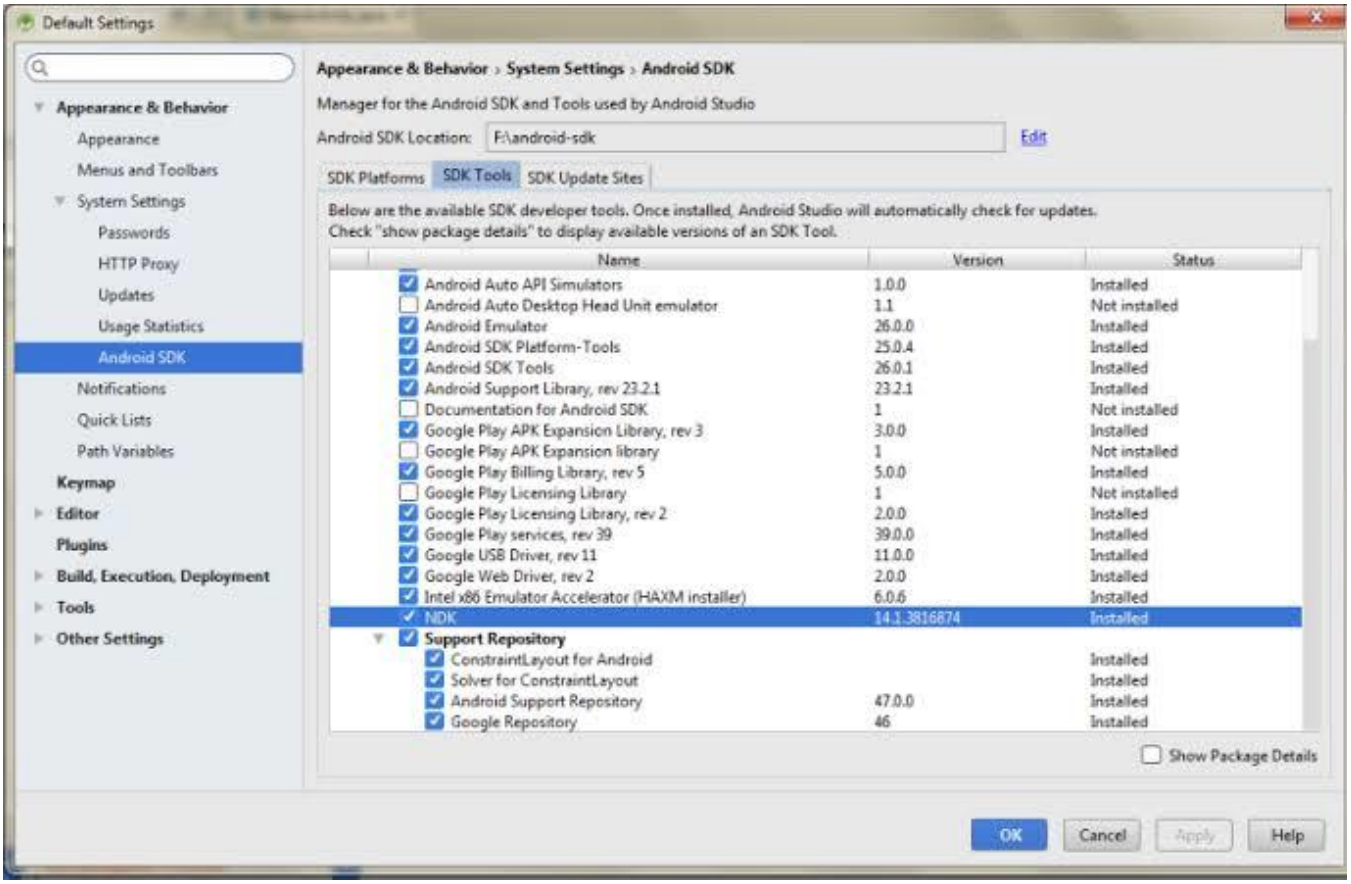


```
tgeaton@tgeaton-ubuntu64: /etc
# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
#umask 022

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi
PATH=$PATH:~/sandbox/android/ndk
"~/.profile" 23L, 708C                                        18,0-1        All
```

Fig.5.3NDK-Build

NDK-Path:

The below figure is indicating the path of NDK-build script file or the CMake file which is then used by the Gradle to execute it to share the resources with the APK. Every project is initially provided the NDK-Path, if not then it is necessary to mention at first to avoid further errors in the APK process.



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: F:\android-sdk

Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates.
Check "show package details" to display available versions of an SDK Tool.

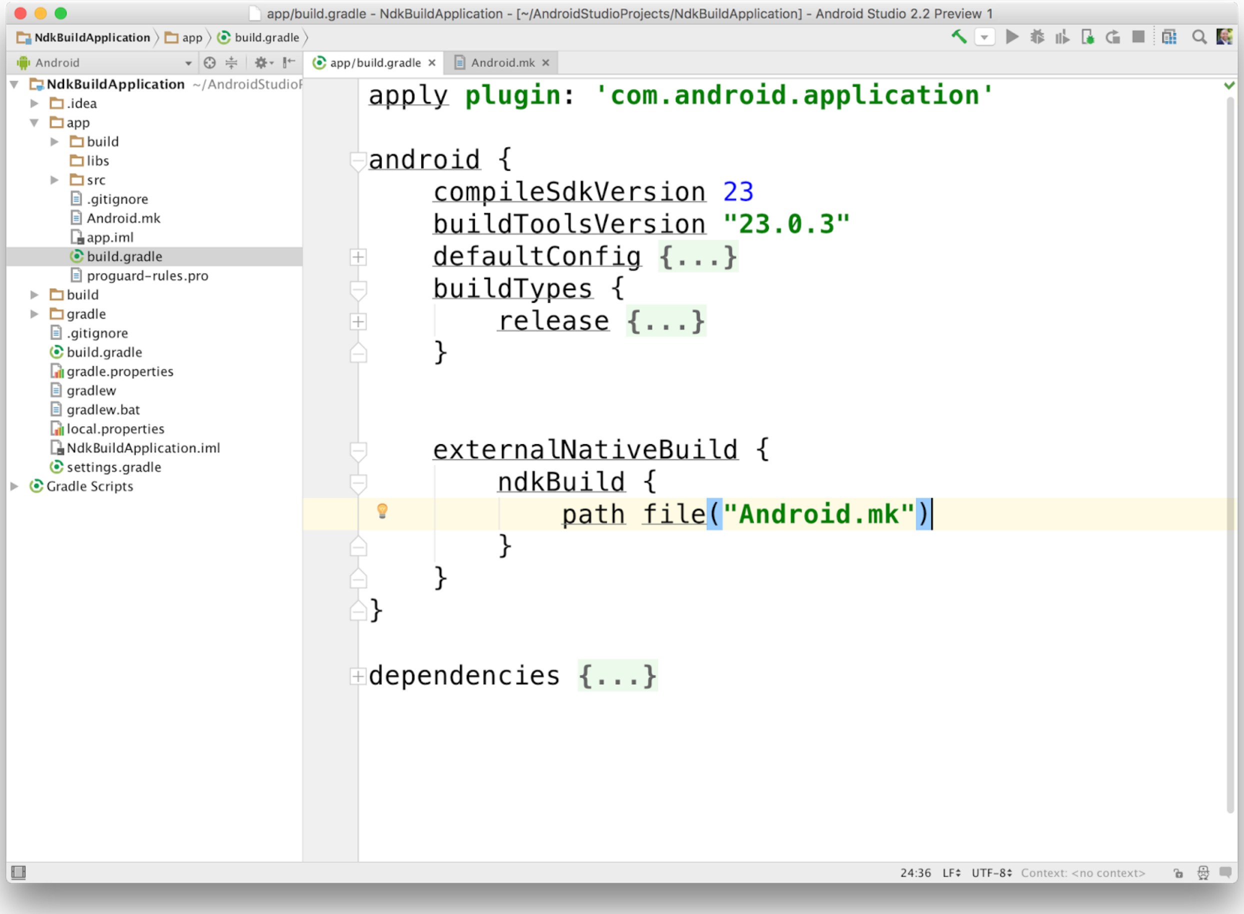
Name	Version	Status
☑ Android Auto API Simulators	1.0.0	Installed
□ Android Auto Desktop Head Unit emulator	1.1	Not installed
☑ Android Emulator	26.0.0	Installed
☑ Android SDK Platform-Tools	25.0.4	Installed
☑ Android SDK Tools	26.0.1	Installed
☑ Android Support Library, rev 23.2.1	23.2.1	Installed
□ Documentation for Android SDK	1	Not installed
☑ Google Play APK Expansion Library, rev 3	3.0.0	Installed
□ Google Play APK Expansion library	1	Not installed
☑ Google Play Billing Library, rev 5	5.0.0	Installed
□ Google Play Licensing Library	1	Not installed
☑ Google Play Licensing Library, rev 2	2.0.0	Installed
☑ Google Play services, rev 39	39.0.0	Installed
☑ Google USB Driver, rev 11	11.0.0	Installed
☑ Google Web Driver, rev 2	2.0.0	Installed
☑ Intel x86 Emulator Accelerator (HAXM installer)	6.0.6	Installed
☑ NDK	14.1.3816874	Installed
☑ Support Repository		
☑ ConstraintLayout for Android		Installed
☑ Solver for ConstraintLayout		Installed
☑ Android Support Repository	47.0.0	Installed
☑ Google Repository	46	Installed

□ Show Package Details

Fig.5.4: NDK Path

GradleNDK:

To make the library project as the Gradle build dependency, the Gradle is provided by the allocating the path of CMake or NDK-build script file. As an application is executed the gradle run the Cmake or the NDK-build script file to share the libraries with the APK and the files to include in the android project are guessed by gradle with build script files for any project, it is necessary to create build-script file initially if not available before further procedures.

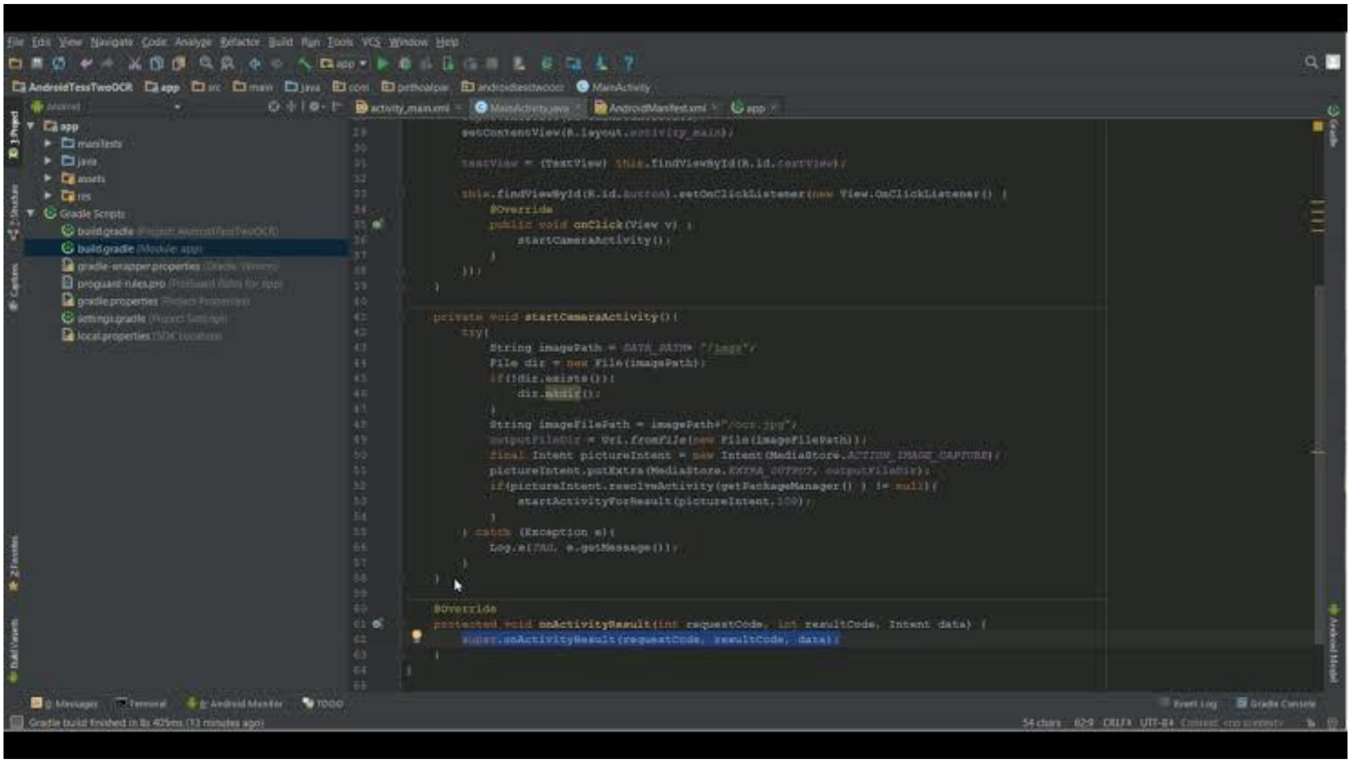


```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 23
    buildToolsVersion "23.0.3"
    defaultConfig {...}
    buildTypes {
        release {...}
    }


    externalNativeBuild {
        ndkBuild {
            path file("Android.mk")
        }
    }
}

dependencies {...}
```

Fig.5.5: Gradle NDK

Tess Two with Gradle:

Tesseract uses two pass method to recognize text. This is also called as adaptive type of recognition which uses the shapes of recognized letters. To recognize letters in second pass, letters recognized in first pass are taken in account for better recognition efficiency. This is advantageous in case of unusual fonts or low-quality scanned docs or images where the font is distorted.



```
        setContentView(R.layout.activity_main);

        textView = (TextView) this.findViewById(R.id.textView);

        this.findViewById(R.id.button).setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startCameraActivity();
            }
        });
    }

    private void startCameraActivity(){
        try{
            String imagePath = DATA_PATH+ "/imgs";
            File dir = new File(imagePath);
            if(!dir.exists()){
                dir.mkdir();
            }
            String imageFilePath = imagePath+"/ocr.jpg";
            outputFileDir = Uri.fromFile(new File(imageFilePath));
            final Intent pictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
            pictureIntent.putExtra(MediaStore.EXTRA_OUTPUT, outputFileDir);
            if(pictureIntent.resolveActivity(getPackageManager() ) != null){
                startActivityForResult(pictureIntent,100);
            }
        } catch (Exception e){
            Log.e(TAG, e.getMessage());
        }
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
    }
}
```

Fig.5.6: Tess Two with Gradle

On-field Testing:

Testing is an important step that helps to detect errors. Testing is a process of finding faults that might occur during the implementation phase. It is also a way to test if the product fulfills the requirements and to check the components functionalities. There exist many ways of testing where each one of them has a distinct requirement, but the only testing that we made is the acceptance testing.

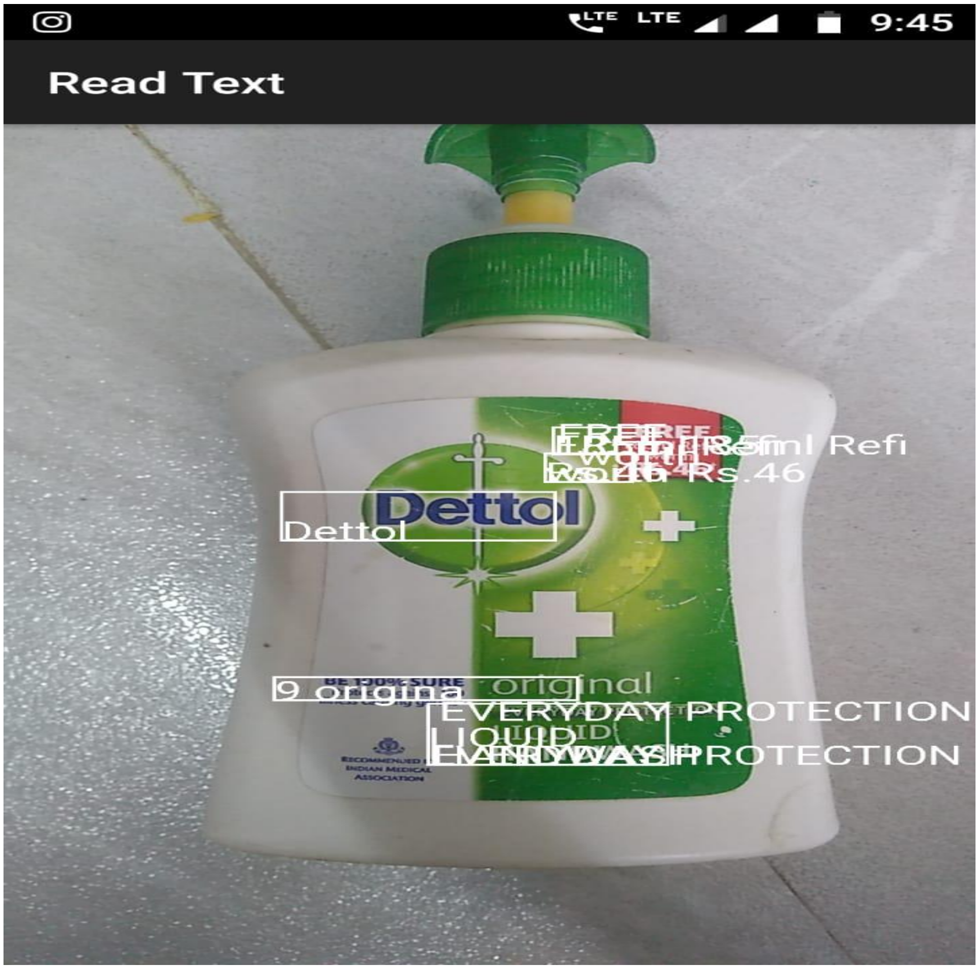
We have tested the application, Android based device, with the supervisor using acceptance testing strategy.

Chapter 6

RESULT AND DISCUSSION

Result:

The simple OCR application uses the input images in different languages and in variety of fonts. The image datasets are provided to recognize the text. The OCR recognizes each and every character in the image in the left hand side and providing the output words at the right which are further used to provide voice output to the user.



Screen o/p

On-field Result Analysis:

- **Economic:** Since people will be using such technology, so instead of wasting time typing on their keyboards or writing on papers, many actions might be performed in few minutes which will contribute on the growth of the economy.
- **Political:** No political trend.
- **Technological:** This application is introducing a new technology that will help its users to save time and perform many actions in few minutes. For instance, students can take a picture of the slides and copy them directly instead of wasting time typing and then copying the desirable data.
- **Legal:** No legal trend.
- **Environmental:** Since the use of papers will reduce, so the number of wastes will decrease as well.
- **Social:** This application provides a social issue which is saving time and performing actions in few minutes, helping foreigners to copy and translate the desirable text just by taking a picture of whatever they need.
- **Ethical:** This application should protect the confidentiality of the user's personal information and any personal data stored on the mobile phone.

Discussion:

The project has been developed in order to achieve the objective of text recognition and voice output to assist the blind people in their regional languages. Initially, we have started the project to build an android application continuously giving a voice output for the captured frames by the camera in real time using the OCR technology. We have faced many problems like image loading in an application, frame drops and image brightness while capturing the images, frame-rate variation, orientation and font of the images etc., To overcome all this problems that we have faced in the application building process we used the Google API for Text To Speech conversion. The speech-rate while using API was then too high with the lesser delay between the frames in the voice processing and it supports the English language and not all the regional languages.

To support the application with regional languages there was a need to provide various databases for various languages in different fonts and angles etc. We have made two datasets with the images in varying angles, brightness, noise and angles etc. The database supports the system and recognizes the text efficiently. According to the cleanliness, noise and skew level the efficiency of an android application varies. The project is built using the Box Tess Editor, tesseract and ndk Gradle to manage resources and packages with the APK. The real time implementation of an application with low cost makes it easy to use in daily life for educational purpose in case of blind people and to use in other applications as a base.

Chapter 7

CONCLUSION AND FUTURE SCOPE

Conclusion:

The project provides a clear concept for text reading in any form as an input in any regional language for the blind people all across the world. TTS synthesis is largely growing phenomenon of computer science and it is vastly executing the major role as an interaction with system and interfaces on the variety of platforms. The planned system provides the simplest technique for ITS transformation. Char inputs like alphabet, sentences, words and numbers etc. are provided to system to recognize Text to voice conversion is obtained and receives the best result which is hearable and accurate. OCR performs text character recognition on the text fields in variety of documents and images and transforms it into audio output for blind users. The main advantages this project is it requires less consumption of time in recognizing and reading text with lower operational costs and also text of different fonts can be recognized. It plays a significant role for visually impaired students in their education.

Future Scope:

- **Direction:** It will be possible to develop an application accepting the source and destination of travelling and providing the proper and efficient destination to the users.
- **Allocation of text:** The visually impaired people can move with the mobile device which can beeps when the camera captures the text on any type of text in captured images so that the user can stop there by directing the camera towards the text image to listen it in the voice format.
- **Availability in all languages:** The text conversion is possible in all languages which on need to build the necessary database of images for real time implementation.

- **Automation:** We can make the application automated by using AI by doing any activity like travelling, searching for new place, guessing roads, statues, shops etc. and to analyze the frontier person gestures or face recognition.

- **Alert System:** Using GPS system we can make the application track able in accidental or in any necessary situation.

REFERENCES

[1]Mamta Kadyan (M.Tech Scholar), Deepti Ahlawat (Assistant Professor) N.C. College of Engineering, Israna, Panipat, India. A Review on Character Recognition Using OCR Algorithm, Volume 7, Issue 5, May 2017, Journal of Network Communications and Emerging Technologies

[2]Fatima Ezzahra Zdadou, "The Optical Capture Recognition Mobile Application". 2015.

[3] A.Subbiah. T.Arivukkarasu. MS Saravanan and V.Balaji. "Camera Based Label Reader for Blind People" Int. J. Chem. Sci.: 14(S3), 2016. 840-844 ISSN 0972768 X, Department of EC, Arupadai Veedu Institute of Technology, Chennai, India.

[4] Noman Islam. Zeeshan Islam. Nazia Noor, "A Survey on Optical Character Recognition Sys tem". Journal of Information & Communication Technology-JICT Vol. 10 Issue. 2. December 2016

[5] https:/develoerandroid.com

[6] https://rmtheis.github.io/tess-two/javadoc/

[7] https://github.com/tesseract-ocr/

[8] https://ocrsdk.com

[9] http://researchgate.net

[10] http://stackoverflow.com/