An android based text detection application for visually impaired

A Bachelors Mini Project (5th Semester)

Of the Undergraduate Program in

INFORMATION TECHNOLOGY

Submitted By:

1. Gaurav Kumar Chandel (IIT2012056)

2. Nithin Srikar Karnala (IIT2012014)

3. Ritesh Kumar Sinha (IIT2012003)

4. Mohit Purbey (IIT2012055)

5. Harivansh Kumar (IIT2012026)

Under the Guidance Of:

**Prof. U.S Tiwary , IIIT Allahabad.**



Indian Institute of Information Technology Allahabad

Devghat, Jhalwa, Allahabad-211012, U. P. INDIA

Candidate’s Declaration

We hereby declare that the work presented in this project entitled “Development Of An Android Based Text Detection Application For Visually Impaired”, submitted in partial fulfilment of the Fifth semester of Bachelor of Technology (B.Tech) program, in Information Technology at Indian Institute Of Information Technology, Allahabad, is an authentic record of our original work carried out under the guidance of Prof. U.S Tiwary, due acknowledgements have been made in the text of the project to all other material used. This work was done in full compliance with the requirements and constraints of the prescribed curriculum.

Place: Allahabad Gaurav Kumar Chandel(IIT2012056)

Date: Nithin Srikar Karnala(IIT2012014)

Ritesh Kumar Sinha(IIT2012003)

Mohit Purbey(IIT2012055)

Harivansh Kumar(IIT2012026)

Acknowledgement

The authors acknowledge the valuable contributions of **Prof. U. S. Tiwary** who guided through

this project. His keen awe-inspiring personality, superb guidance and constant encouragement are the motive forces behind this project work. We would like to thank him sincerely for his help and support.

Gaurav Kumar Chandel (IIT2012056)

Nithin Srikar Karnala (IIT2012014)

Ritesh Kumar Sinha (IIT2012003)

Mohit Purbey (IIT2012055)

Harivansh Kumar (IIT2012026)

**Abstract**

We are going to develop an android based application which can be used by the visually impaired people. In short our application processes the image of the text which user wants to read and reads it aloud so that the user will be able to get the content of that text in the form of audio. We are developing our software on android platform for enhanced mobility by keeping in mind the recent trend of usage on the go. We are considering that the text to be read will be in English (US) and the TTS engine will read it aloud in English.

Key terms: Text detection and extraction, text to speech, OCR, Android.

Table of Contents

|  |  |  |
| --- | --- | --- |
| Sl. No. | Topic | Page No. |
| 1. | Introduction | 1 |
| 2. | Literature Review | 2 – 5 |
| 3. | Methods and Techniques used | 6 – 9 |
| 4. | Input - Output | 10 – 12 |
| 5. | Scope and Future plans | 13 |
| 6. | References | 14 |
| 7. | Conclusion | 15 |
| 8. | Remarks and Suggestions | 16 |

**Introduction**

The key motivation of this project is to help the visually impaired people and make their lives better by providing them a user friendly (precisely easy to use, targeting visually impaired) application which helps them in their day to day reading activities. As we all know it has become very easy these days to acquire a considerable android mobile and keeping that aspect in mind, we are targeting android platform.

Our application will prompt the user to put mobile’s camera facing the text which he/she wants to read and tap the mobile screen, there by capturing a picture of it, processes that image and reads aloud the text present in that image.

The application which we are going to develop includes preprocessing steps, then optical character recognition (OCR), then post processing steps and at last text to speech synthesis (TTS).

The preprocessing steps include

1. Converting the picture taken from camera to grayscale.
2. Canny edge detection
3. Deskewing of the image to align the text perfectly.
4. Binarisation.

Post processing steps include matching the words generated using English dictionary (lexicon).

We decided to first achieve a sub goal of recognizing a character (simple background) and then recognizing a word and in turn a paragraph (complex background).

**Literature Review**

Many different methods have been proposed for text detection and extraction. We will brieﬂy summarize the two main categories of these methods:

1. Connected component based method: Connected component based methods use bottom up approach to group smaller components into larger components until all regions are identified in the image. A geometrical analysis is later needed to identify text components and group them to localize text region.[1]
2. Texture-analysis method: Texture-based methods use the observation that text in images have distinct textural properties that distinguish them from the background. The techniques based on Gabor filters etc. can be used to detect the textural properties of a text region in an image.[2]

Preprocessing steps help to reduce errors and improve the result of the text detection. Various preprocessing steps include:

1. Canny edge detection algorithm:

Canny Edge Detection is a popular edge detection algorithm. It was developed by John F. Canny in 1986. It is a multi-stage algorithm and we will go through each stages.

1. Noise Reduction
2. Hysteresis thresholding:

This stage decides which of all edges are really edges and which are not. For this, we need two threshold values, *minVal* and *maxVal*. Any edges with intensity more than *maxVal* must be edges and those below *minVal* must be non-edges, so they will be discarded. Those edges that lie between these two thresholds are classified into edges or non-edges based on their connectivity. If they are connected to edges with intensity above maxval pixels, they are considered to be part of edges. Otherwise, they are also discarded. See the image below:



Fig 1. Hysteresis thresholding

The edge A is above the *maxVal*, so considered as sure-edge. Although edge C is below *maxVal*, it is connected to edge A, so that also considered as valid edge and we get that full curve. But edge B, although it is above *minVal* and is in same region as that of edge C, it is not connected to any sure-edge, so that is discarded. So it is very important that we have to select *minVal* and *maxVal* accordingly to get the correct result.

This stage also removes small pixels noises on the assumption that edges are long lines.

So what we finally get is strong edges in the image.



Fig 2. Before Canny Edge Detection



Fig 3. After Canny Edge Detection

1. Binarization:

Binarization is performed either globally or locally.

For the global methods, a single calculated threshold value is used to classify image pixels into object or background classes [3]. It works well when there is a significant difference in the intensities of foreground and background pixels. Global thresholding methods are not sufﬁcient since document images usually are degraded and have poor quality including shadows, non-uniform illumination, low contrast, large signal-dependent noise.

For the local method, local area information determines the threshold value for each pixel. According to this analysis, local algorithms work well for a slowly changing background.

Our primary work is to apply various preprocessing methods and feed the result into the OCR engine.

A number of OCR engines are widely available and in current use. Open-source projects include GOCR, OCROpus [7], OpenOCR and Tesseract [8], [9]. Commercial products supporting mobile platforms include ABBYY FineReader, Nuance OmniPage and ExperVision OpenRTK.

We are going to use OCR and TTS which are explained in detail below:

OCR

OCR stands for Optical Character Recognition. This software allows to automatically recognize characters from images. But the performance of OCR is directly dependent on quality of input images (Which we are enhancing using various preprocessing techniques). OCR is designed to process images that consist almost entirely of text, with very little non-text areas obtained from a picture captured by a camera. The application which we are going to use is for the Android mobile operating system that combines Google’s open-source OCR engine, Tesseract. It was firstly developed during 1984 to 1994 at HP. Finally in 2005, Tesseract was released as open source by HP. Tesseract performs activity to recognize words. This recognition activity is mainly consists of two passes. The first pass tries to recognize the words. Then satisfactory word is passed to Adaptive Classifier as training data, which recognizes the text more accurately. During second pass, the words which were not recognized well in first pass are recognized again through run over the page.

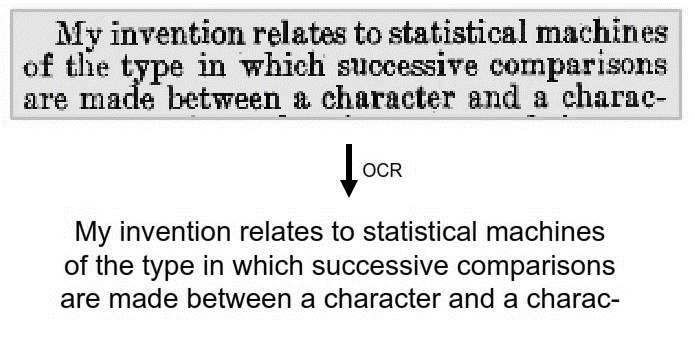


Fig.4 Working Of OCR

TTS

A text to speech (TTS) synthesizer is a system that can read text aloud automatically upon feeding a text (which is extracted from Optical Character Recognition (OCR) and post processed in our case). Speech synthesis is the artificial production of human speech. A computer system used for this purpose is called a speech synthesizer. A text-to-speech (TTS) system converts normal language text into speech. A synthesizer can incorporate a model of the vocal tract and other human voice characteristics to create a completely “synthetic” voice output.



Fig 5. Tesseract Flow Diagram

**Methods and Techniques used**

1. Opening of the Application and initial steps:

We are going to design the application such that it can be opened by pressing some combination of hardware buttons for the ease of user. When the application is opened the user will be prompted to put the mobile in front of text to be read aloud (We are considering a constraint that he will put at a length of around 30cms i.e., arm’s length) and tap the screen when he/she is ready upon which a snap will be taken (We are considering that there will be optimum light for reading and text to be read will be completely within the captured image which can be said to be a constraint).

1. Preprocessing steps:

Various preprocessing steps which we are going to employ are as follows:

1. Convert the obtained image to Grey Scale

We will be using the weighted average of RGB as follows for each pixel

R = G = B = (0.299 \* R + 0.587 \* G + 0.114 \* B)



Fig 6. Conversion to Grey Scale

1. Apply Canny edge detection algorithm

Canny edge detection is a four step process.

1. A Gaussian blur is applied to clear any noise and free the image of noise.
2. A gradient operator is applied for obtaining the gradients' intensity and direction.
3. Non-maximum suppression determines if the pixel is a better candidate for an edge than its neighbors.
4. Hysteresis thresholding finds where edges begin and end.
5. Detect the Skew angle and deskew it:

In order to detect the skew we cast rays from left to right and find how many blackish pixels the rays intersect with. Then, we rotate the angle of the rays by a small amount and cast the rays again, and do the same process again and again.

The angle at which most white space is encountered (in other words, most black pixels are shot down), we consider that to be the skew angle. Most likely, we would start from 0 degree and go down to 90 degrees, and then again, from 0 degree go up to 90 degrees with the angle of the rays. Whenever the white ratio reaches the maximum peak and then starts to come down again, we stop immediately, and we know we have found our skew angle. This is the basic idea of detecting skew for text images.

The deskewing of an image is nothing but rotating the image by the same amount of the skew in the reverse direction.

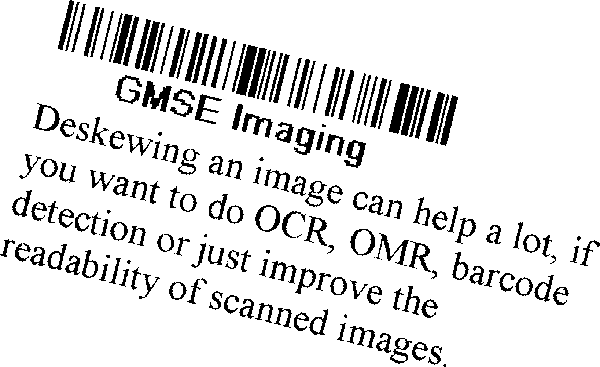


Fig 7. Image before deskewing

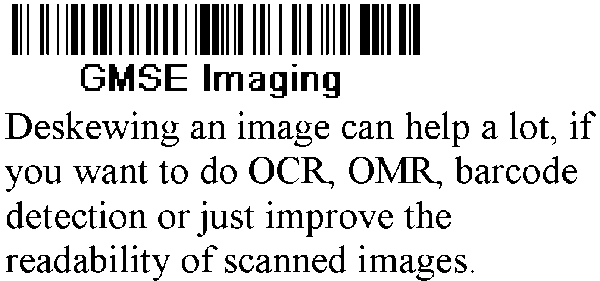


Fig 8. After deskewing

1. Binarization:

**Otsu's method** is used to automatically perform clustering-based image thresholding [11], or the reduction of a gray level image to a binary image. The algorithm assumes that the image contains two classes of pixels foreground pixels and background pixels, it then calculates the optimum threshold separating the two classes so that their combined spread is minimal [12].



Fig 9. Original image



Fig 10. Image thresholded using Otsu’s algorithm

1. The output obtained after applying the above steps is fed into Tesseract OCR engine. We have chosen Tesseract because of widespread approbation, its extensibility and flexibility, its community of active developers.
2. The text obtained from the Tesseract engine will not be 100% error free and lexically correct (as stated in the Tesseract official documentation). Therefore we decided to apply Post Processing Step of checking the correctness of the words lexically by implementing a spell checker and from the suggestions obtained we will choose the best suggestion provided from the spell checker service.
3. The final text obtained after applying all the above steps will be fed into TTS engine which reads aloud the text.

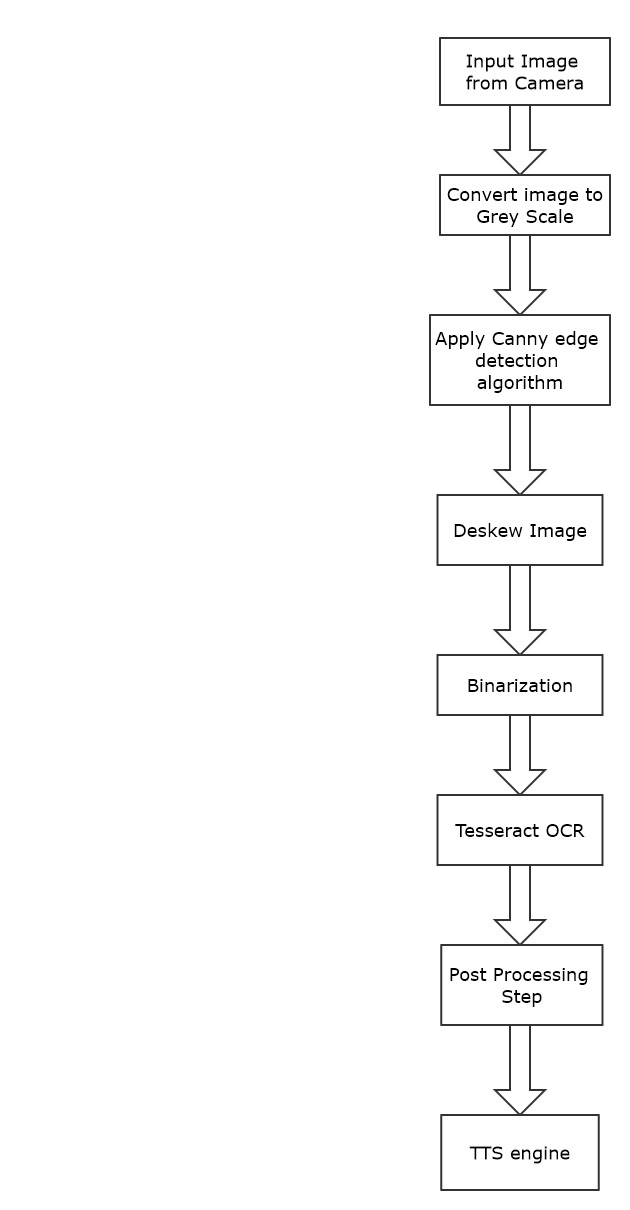


Fig.11 Flowchart of various modules employed in the project

Input - Output

1. 

It doesnft take a genius.

C:\Users\nithin srikar\Desktop\im4.tif

Facebook

C:\Users\Ritesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\t4.tif

SoftwareEngineering

C:\Users\Ritesh\AppData\Local\Microsoft\Windows\INetCache\Content.Word\t2.tif

Motivational and Inspirational Quotes

A stone is broken by the last stroke. This does not mean that ﬁrst

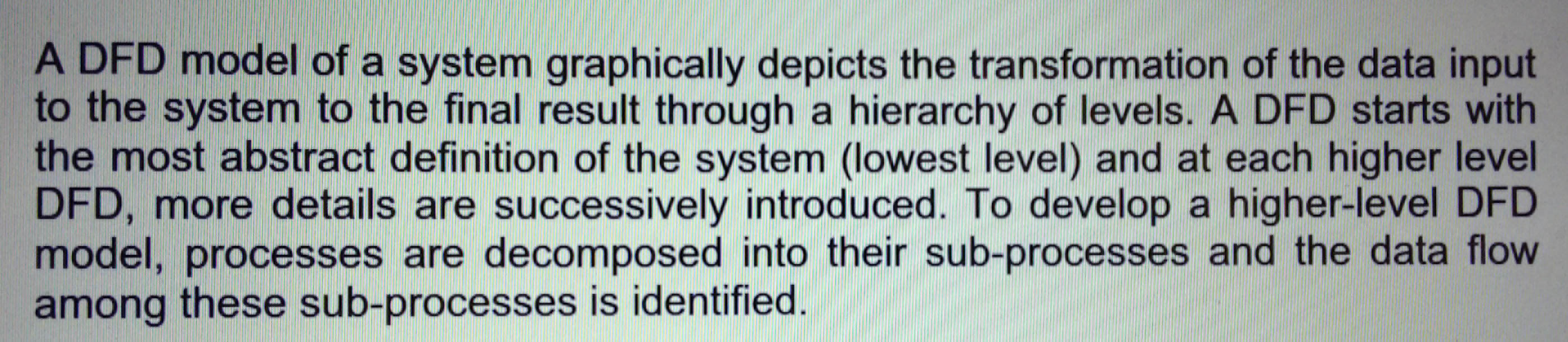
stroke was useless. Success is a result of continuous daily effort.

——Unknown

Stand up, be bold, be strong. Take the whole responsibility on your own

shoulders, and know that you are the creator of your own destiny.

——Swami Vivekananda



DFDﬁnwhlﬂdl‘é system graphically depicts the transformation of the data input

to thesy8térhtdtheﬁ‘nal result through a hierarchy of levels. A DFD starts with

the mostabst‘ract deﬁnition of the system (lowest level) and at each higher level

DFD, mere details are successively introduced. To develop a higher-level DFD

model, processes are decomposed into their subnprocesses and the data ﬂow

among these sub—processes is identified.

E:\Ritz\IIITA\5th Sem\COURSE\Mini Project\test cases\t\t5.tif

The product is intangible The manager of a shipbuilding project or of a civil

engineering project can see the product being developed. If a schedule slips,

the effect on the product is visible—pans of the structure are obviously unﬁn-

ished. Software is intangible. It cannot be seen or touched. Software project

managers cannot see progress. They rely on others to produce the documenta—

tion needed to review progress.

**Scope and Future plans**

The scope of this project can be extended in several facets such as:

1. Real time processing:

Instead of processing image captured, frames can be processed in real time and the text within the frames can be read aloud then and there.

1. Improved Binarization techniques:

Improved binarization techniques can be employed at the preprocessing stage since the main disadvantage of Otsu’s method is that they do not have an ability to exploit information of the characteristics of target images that they threshold. Other techniques like hybrid image thresholding [13] can be applied for improved results.

1. Stoke width transform can be applied at preprocessing stage for better results
2. Translation:

The application can further be developed to translate the read text to other language and convert it into audio in other language which can help tourists for reading sign boards etc.

1. Support for Many languages:

The application at present is confined to only English (US) but it can be further developed to support reading text of any language and then reading it aloud in any language by using translation.

**References**

[1] A. Jain and B. Yu., “Automatic text location in images and video frames”, Aug 1998, pp. 1497–1499.

[2] C. W. Lee, K. Jung, and H. J. Kim, “Automatic text detection and removal in video sequences”, November 2003, pp. 2607–2623.

[3] H. Yan, “Uniﬁed formulation of a class of image thresholding techniques”, 1996, pp. 2025-2032.

[4] J. Yang, Y. Chen, W. Hsu, “Adaptive thresholding algorithm and its hardware implementation”, 1994.

[5] O.D. Trier, A.K. Jain, “Goal-directed evaluation of binarization methods, IEEE Trans. Pattern Anal. Mach. Intell”, 1995.

[6] M. Sezgin, B. Sankur, “Survey over image thresholding techniques and quantitative performance evaluation”, 2004.

[7] T. M. Breuel, “The OCRopus open source OCR system,” in Proc. IS&T/SPIE 20th Annu. Symp., 2008.

[8] R. Smith, “An overview of the Tesseract OCR engine.” in ICDAR ’07 Proc. 9th Int. Conf. Document Analysis and Recognition, Curitiba, Brazil, 2007, pp. 629-633.

[9] R. Smith, D. Antonova, and D. Lee, “MOCR ’09 Adapting the Tesseract open source OCR engine for multilingual OCR,” in Proc. Int. Workshop Multilingual OCR, Barcelona, 2009.

[10] Ravina Mithe, Supriya Indalkar, Nilam Divekar ,“Optical Character Recognition”, International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-2, Issue-1, March 2013.

[11] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation", 2004.

[12] Nobuyuki Otsu, "A threshold selection method from gray-level histograms". IEEE Trans. Sys., Man., Cyber, 1979.

[13] Febriliyan Samopa and Akira Asano , “Hybrid Image Thresholding Method using Edge Detection”, Department of Information Engineering, Graduate School of Engineering, Hiroshima University, Higashi-Hiroshima, Hiroshima, Japan.

Conclusion

Text detection and extraction from images can be very useful in various fields but it faces many challenges like uneven illumination, noise, low resolution etc. that lead to degradation in the performance of the software. Though many techniques have been proposed above to improve the performance and efficiency of OCR such as preprocessing of images like binarization, edge detection and noise removal and post-processing of the results obtained from the OCR but still the maximum accuracy that has been achieved so far from an OCR is 85% which is good but not perfect.  
So we can say that for the time being, it is unrealistic to create a generic recognition system that reaches signiﬁcant results for all kind of text images.

Remarks and Suggestions