CHAPTER 1

INTRODUCTION

Scene text is text that appears in an image captured by a camera in an outdoor environment. Scene-text detection in natural-scene images is an important technique because scene texts contain location information such as names of places and buildings, but many difficulties still remain regarding practical use. Text embedded in natural scene images provide rich semantic information about the scene, which is of great value for content-based image, indexing and retrieval applications. Text characters embedded in images and video sequences represents a variety of rich source of information. Text information in scene image serves as important clue for many computer vision applications. Text component in an image is of particular interest, as it is easy to understand by both humans as well as machines. Very large number of applications can be built by extracting text from images such as scene understanding, content-based image retrieval, automatic robot navigation and license plate detection. The goal of scene text extraction is to identify and isolate text element in a picture. It is designed to operate on raw images and produce binary versions, in which all non-text elements has been removed.

Text-recognition methods have been extensively investigated and researchers have proposed effective ones with the required accuracy. An optical-character-recognition method for recognizing printed text in scanned documents is one of the most successful. However, it still difficult for computers to detect and recognize scene text, which is text in natural-scene images because scene text has various interference factors such as appearance variation (e.g. changes in character size and font), language, orientation, distortion, noise, occlusion, and complex background. Text detection approaches can be divided into two main categories (a) sliding window based approaches (b) connected components based approaches. There are three main methods used in text detection: colour-based method, texture-based method and stroke-based method.

The proposed methodology is an application which is able to extract the text from scene image, translate the same from one language to other and also perform a web-search is developed. Extraction is performed using stroke width transform (SWT) approach and connected component analysis. Texts are extracted accurately using machine learning algorithm namely Support Vector machine(SVM) and this is recognized using Google's open source optical character recognition (OCR) engine ‘Tesseract’ and web-search is performed using Google’s platform.

1.1 Image Processing

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Image Processing is a technique to enhance raw images received from cameras/sensors placed on satellites, space probes and aircrafts or pictures taken in normal day-to-day life for various applications. Image processing is used in various application such as: Remote Sensing, Medical Imaging, Printing Industry, Graphics art, Document Processing etc. It is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal-processing techniques to it. It is among rapidly growing technologies today, with its applications in various aspects of a business. Image Processing forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps.

- Importing the image via image acquisition tools.
- Analysing and manipulating the image which includes data compression and image enhancement and spotting patterns that are not to human eyes like satellite photographs.
- Output in which result can be altered image or report that is based on image analysis.

The purpose of image processing is divided into 5 groups. They are:

- Visualization - Observe the objects that are not visible.
- Image sharpening and restoration - To create a better image.
- Image retrieval - Seek for the image of interest.
- Measurement of pattern – Measures various objects in an image.
- Image Recognition – Distinguish the objects in an image.

The two types of methods used for Image Processing are Analog and Digital Image Processing.

- Analog or visual techniques of image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. The image processing is not just confined to area that has to be studied but on knowledge of analyst. Association is another important tool in image processing through visual techniques. So analysts apply a combination of personal knowledge and collateral data to image processing.
- Digital image processing is the use of computer algorithms to perform image processing on digital images Digital Processing techniques help in manipulation of the digital images by using computers. As raw data from imaging sensors from satellite platform contains deficiencies. To get over such flaws and to get originality of information, it has to undergo various phases of processing. The three general phases that all types of data have to undergo while using digital technique are Pre-processing, enhancement and display, information extraction.

1.2 Stroke Width Transform

The Stroke Width Transform (SWT) is a local image operator which computes per pixel the width of the most likely stroke containing the pixel. The output of the SWT is an image of size equal to the size of the input image where each element contains the width of the stroke associated with the pixel. A stroke is defined to be a contiguous part of an image that forms a band of a nearly constant width. The SWT operator calculates SWT value, the stroke width of each pixel between two opposite gradient of edge pixel, and produces SWT image.

1.3 Connected-Component Analysis

Connected-Component Clustering algorithm has two steps – Labelling and Clustering.

- Connected components labelling is defined as the creation of a labelled image in which the positions associated with the same connected component of the binary input image have a unique label. Labelling is used in computer vision to detect connected regions in binary digital images. Connected-component labelling (alternatively connected-component analysis, blob extraction, region labelling, blob discovery, or region extraction) is an algorithmic application of graph theory, where subsets of connected components are uniquely labelled based on a given heuristic. Connected-component labelling is used in computer vision to detect connected regions in binary digital images, although colour images and data with higher dimensionality can also be processed. When integrated into an image recognition system or human-computer interaction interface, connected component labelling can operate on a variety of information. Blob extraction is generally performed on the resulting binary image from a thresholding step, but it can be applicable to gray-scale and colour images as well. Blobs may be counted, filtered, and tracked.

- Clustering is performed by taking account into proximity of the different blobs and grouping them present in image.

1.4 Support Vector Machine(SVM)

Support vector machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. When data are not labelled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups.

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labelled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples. In two dimensional space this hyperplane is a line dividing a plane in two parts where in each class lay in either side.

A support vector machine constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks like outliers detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

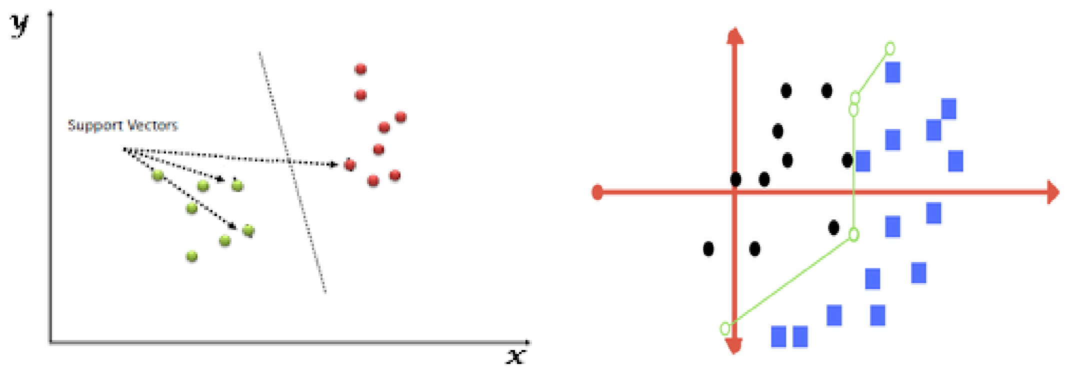


Figure 1.1 SVM Principle

1.5 Google OCR – Tesseract

Optical character recognition (also optical character reader, OCR) is the mechanical or electronic conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example from a television broadcast). Advanced systems capable of producing a high degree of recognition accuracy for most fonts are now common, and with support for a variety of digital image file format inputs. OCR engines have been developed into many kinds of domain-specific OCR applications, such as receipt OCR, invoice OCR, check OCR and legal billing document OCR. They can be used for:

- Data entry for business documents, e.g. check, passport, invoice, bank statement and receipt.
- Automatic number plate recognition, Automatic insurance documents key information extraction.

- Make electronic images of printed documents searchable, e.g. Google Books
- Converting handwriting in real time to control a computer (pen computing).
- Defeating CAPTCHA anti-bot systems, though these are specifically designed to prevent OCR.
- Assistive technology for blind and visually impaired users.

Tesseract is an optical character recognition engine for various operating systems. It is free software, released under the Apache License, Version 2.0 and development has been sponsored by Google since 2006. Tesseract is considered one of the most accurate open-source OCR engines then available. Tesseract works best when there is a (very) clean segmentation of the foreground text from the background. Tesseract is the most advanced text recognition library available. It performs all the functions required to recognize text such as – Paragraph Segmentation, Line Segmentation and Character Segmentation.

1.6 Motivation

- Text can be detected by exploiting the discriminate properties of text characters such as the vertical edge density, the texture or the edge orientation variance. One early approach for localizing text in covers of Journals or CDs assumed that text characters were contained in regions of high horizontal variance satisfying certain spatial properties that could be exploited in a connected component analysis process.
- Since commercial OCR engines achieve high recognition performance when processing black and white images at high resolution, almost all the methods in the literature that addressed the issue of text recognition in complex images and videos employed an OCR system.
- There are two problems in obtaining efficient and robust text detection using machine learning tools. One is how to avoid performing computational intensive classification on the whole image, the other is how to reduce the variance of character size and gray scale in the feature space before training
- Detecting text in a natural scene is an important part of many Computer Vision tasks.

1.7 Objective

- To build a robust system which would be able to detect natural-scenic text accurately.

- Machine Learning should be implemented to train the model such that efficiency of text detection is improved.
- The system shall have additional capabilities which facilitate the detected text to be translated into other languages and search for its related content on the web.
- The scope can be extended by using a larger data-set for improved text detection.
- Using different formulae for SWT method, we can detect different languages more effectively.

CHAPTER 2

LITERATURE SURVEY

A literature survey shows the various analyses and research made in the field of our interest and the results already published, taking into account the various parameters of the project and the extent of the project. It is the most important part of our report as it gives us a direction in the area of your research. It helps us set a goal for our analysis - thus giving us our problem statement. It includes the researches made by various analysts - their methodology and the conclusions they have arrived at. All this research has influenced our thesis.

Scene text extraction is recent research area in the field of Computer Vision. Text component in an image is of particular interest, as it is easy to understand by both humans as well as machines and can be used in variety of applications like content-based image retrieval, aids for visually impaired people, translator for tourists. Although there exists a lot of research activities in this field, it is still remained as a challenging problem, mainly due to two issues: different variety of text patterns like fonts, colors, sizes, orientations; and presence of background outliers similar to text characters, such as windows, bricks. In this paper [1], an android application named as TravelMate is developed, which is a user-friendly application to assist the tourist navigation, while they are roaming in foreign countries. Proposed application is able to extract text from an image, which is captured by an android mobile camera. Extraction is performed using stroke width transform (SWT) approach and connected component analysis. Extracted texts are recognized using Google's open source optical character recognition (OCR) engine `Tesseract' and translated to a target language using Google's translation. The entire result is displayed back onto the screen of the mobile Phone. Application is able to extract text from an image [1], which is captured by an android mobile camera. Extraction is performed using stroke width transform (SWT) approach and connected component analysis. Extracted texts are recognized using Google’s open source optical character recognition (OCR) engine ‘Tesseract’ and translated to a target language using Google’s translation. Text in the horizontal direction is extracted accurately. Speed is favored instead of flexibility in this system. But, it cannot extract text which is oriented arbitrarily. The methodology fails for low resolution images under varying light conditions.

Ezaki, Bulcau and Schomaker [2] proposes a system that reads the text encountered in natural scenes with the aim to provide assistance to the visually impaired persons. This paper describes the system design and evaluates several character extraction methods. Automatic text recognition from natural images receives a growing attention because of potential applications in image retrieval, robotics and intelligent transport system. Camera-based document analysis becomes a real possibility with the increasing resolution and availability of digital cameras. However, in the case of a blind person, finding the text region is the first important problem that must be addressed, because it cannot be assumed that the acquired image contains only characters. At first, this system tries to find in the image areas with small characters. Then it zooms into the found areas to retake higher resolution images necessary for character recognition. In the present paper, the proposed four character-extraction methods based on connected components. The effectiveness of the proposed methods was tested on the ICDAR 2003 Robust Reading Competition data. The performance of the different methods depends on character size. In the data, bigger characters are more prevalent and the most effective extraction method proves to be the sequence: Sobel edge detection, Otsu binarization, connected component extraction and rule-based connected component filtering. It uses Sobel Edge Detection, Otsu Binarization, Connected Component Extraction and rule based Connected- Components Selection. High recall rate is achieved by collecting all the candidate text areas provided by the methods. The drawbacks are that it is not useful for practical use. Small text characters cannot be detected with high accuracy.

In few decades, text detection methods have been proposed with significant results. Yet, most of them are only implemented for certain language. Thus, this work [3] proposes a text detection method for multi language text. To do such a task, a Fast Stroke Width Transform (FSWT) is introduced. FSWT is a robust local image operator for text detection with complex background. It is improved version of Stroke Witdh Transform (SWT) in order to obtain faster computation. The FSWT calculates the stroke width on each pixel. The pixels in the FSWT image connected into components according to stroke width value. These components are then classified as text or nontext. In standard SWT, the components should be generated two times according two types of text, bright text on dark background (BoD) and dark text on bright background (DoB). However, the FSWT could automatically determine the type of text by utilizing edge component labeling and tree of edge component. To compare with standard SWT, most of the images are archived in database with BoD and DoB text. In the experiment, the proposed method achieves almost three times faster than the standard SWT in computation time. In consequence, the proposed method is expected to be implemented with faster process in real-time case. An efficient method for detecting Multi-language text has been successfully implemented. Utilization of edge component labelling and tree of edge component. Three times faster than regular SWT and component clustering. Whereas, the component generating before tree of component process can be performed by other features such as color or Texture. This does not allow the detection image which contains touching characters.

Jain, Peng, Zhuang, Natarajan and Cao [4] proposed an end-to-end system for text detection and recognition in natural scenes and consumer videos. Maximally Stable Extremal Regions which are robust to illumination and viewpoint variations are selected as text candidates. Rich shape descriptors such as Histogram of Oriented Gradients, Gabor filter, corners and geometrical features are used to represent the candidates and classified using a support vector machine. Positively labeled candidates serve as anchor regions for word formation. The method then group candidate regions based on geometric and color properties to form word boundaries. To speed up the system for practical applications, the method uses Partial Least Squares approach for dimensionality reduction. The detected words are binarized, filtered and passed to a hidden Markov model based Optical Character Recognition (OCR) system for recognition. The method shows significant improvement in text detection and recognition tasks over previous approaches on a large consumer video dataset. Furthermore, the event detection system built upon the OCR output of this approach outperformed multiple other OCR-only based submissions in the recently concluded NIST TRECVID 2013 multimedia event detection evaluations. Uses SVM classifier for improved performance. A merging scheme has been proposed which overcomes the mistakes of SVM classification step and preserves word boundaries. Here, the time for training the machine with the dataset tends to be higher. This method produces better results at the expense of speed.

2.1 Existing System

Generally, there are three main approaches used in text detection: color-based method [5], texture-based method [6], and stroke-based method [7]–[9], [10]. However, most of the researchers use stroke width features to implement their method, since the stroke is a robust feature, always appears in text and can be applied in both of color images and black-and white images. In [11], stroke width was calculated by using the horizontal scan line to record the intensity variations around the edge pixels, usually a pair of impulses on the strokes with equal magnitudes and in opposite direction. A character consists of stroke in multiple orientations, while the horizontal scan line can only derive the width of vertical strokes [12].Stroke Width Transform (SWT), proposed by [9] and adopted in [8], [9], [10], [12], [13], is one of methods which deals with stroke feature. Compared with other methods, the SWT is more robust operator to obtain stroke width, since this operator can deal in multi-orientation text stroke.

CHAPTER 3

SYSTEM REQUIREMENT SPECIFICATION

A System Requirements Specification (SRS) is a comprehensive description of the intended purpose and environment for software under development. The SRS fully describes what the software will do and how it will be expected to perform. An SRS minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs and human users in a wide variety of real-world situations. Parameters such as operating speed, response time, availability, portability, maintainability, footprint, security and speed of recovery from adverse events are evaluated.

3.1 Functional Requirements

The functional requirements for a system describe what the system should do. These requirements depend on the type of software being developed, the expected users of the software and the general approach taken by the organization when writing requirements. When expressed as user requirements, the requirements are usually described in a fairly abstract way. However, functional system requirements describe the system function in detail, its inputs and outputs, exceptions, and so on.

Functional requirements are as follows:

- The system should process the input given by the user only if it is an image file.
- System should detect text regions present in image.
- System should suppress non text regions from the initial regions.
- System should recognize individual characters and then form words.
- System should perform a google search with the recognized text as the parameter.

3.2 Non-Functional Requirements

Non-functional requirements, as the name suggests, are requirements that are not directly concerned with the specific functions delivered by the system. They may relate to emergent system properties such as reliability, performance and availability. Alternatively, they may define constraints on the system such as the capabilities of I/O devices and the data representations used in system interfaces. Non-functional requirements may be more critical than the functional requirements.

- **Performance:** The text in the input image is recognized and classified with high accuracy.
- **Availability:** This system will process the image and give proper results only if the text is properly visible in the image.
- **Flexibility:** It recognizes text in both the conditions, that is, dark text on light background and light text on dark background.
- **Ability:** The software is very easy to use.
- **Reliability:** This software works well with all resolution images and not with graphical images.

3.3 Hardware Requirement Specifications

Processor	Dual Core, i7 Core Intel Processor
Architecture	64 Bit Architecture
RAM	8 GB
Hard disk	1 GB (Varies greatly depending on data-set) .
Monitor	SVGA or HDMI
Mouse	Two or three mouse button

Table 3.1 Hardware Requirements.

3.4 Software Requirement Specifications

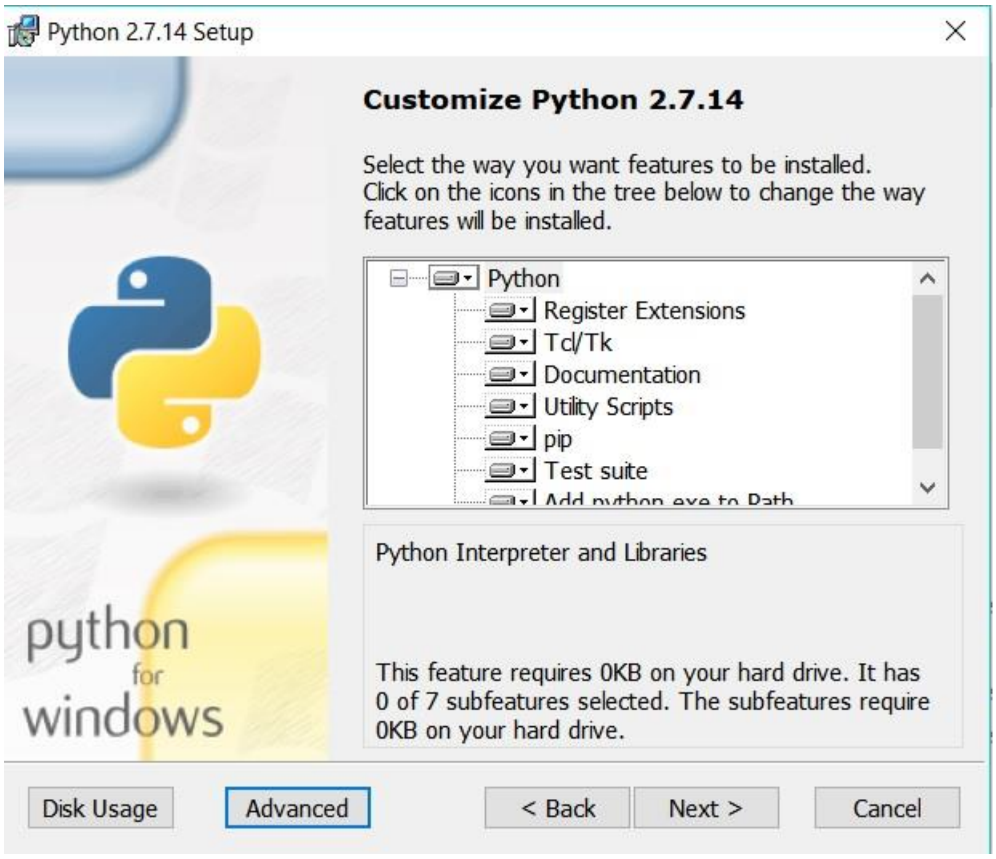
- Windows 10
- Python 2.7
- Enthought Canopy (Python IDE)

- Tesseract OCR Binaries
- Android Studio and Android SDK
- Java SDK
- Libraries – OpenCV, Matplot, Python Image Processing Library (PILLOW), pytesseract, sklearn, pickle, web browser.

3.5 Installing Python2.7

Download the python installation file from www.python.org. Select the appropriate file from the list of given files based on your operating system and system architecture.

Choose the Installation directory and click Next.


Python 2.7.14 Setup

Customize Python 2.7.14

Select the way you want features to be installed.
Click on the icons in the tree below to change the way features will be installed.

- Python
  - Register Extensions
  - Tcl/Tk
  - Documentation
  - Utility Scripts
  - pip
  - Test suite
  - Add python.exe to Path

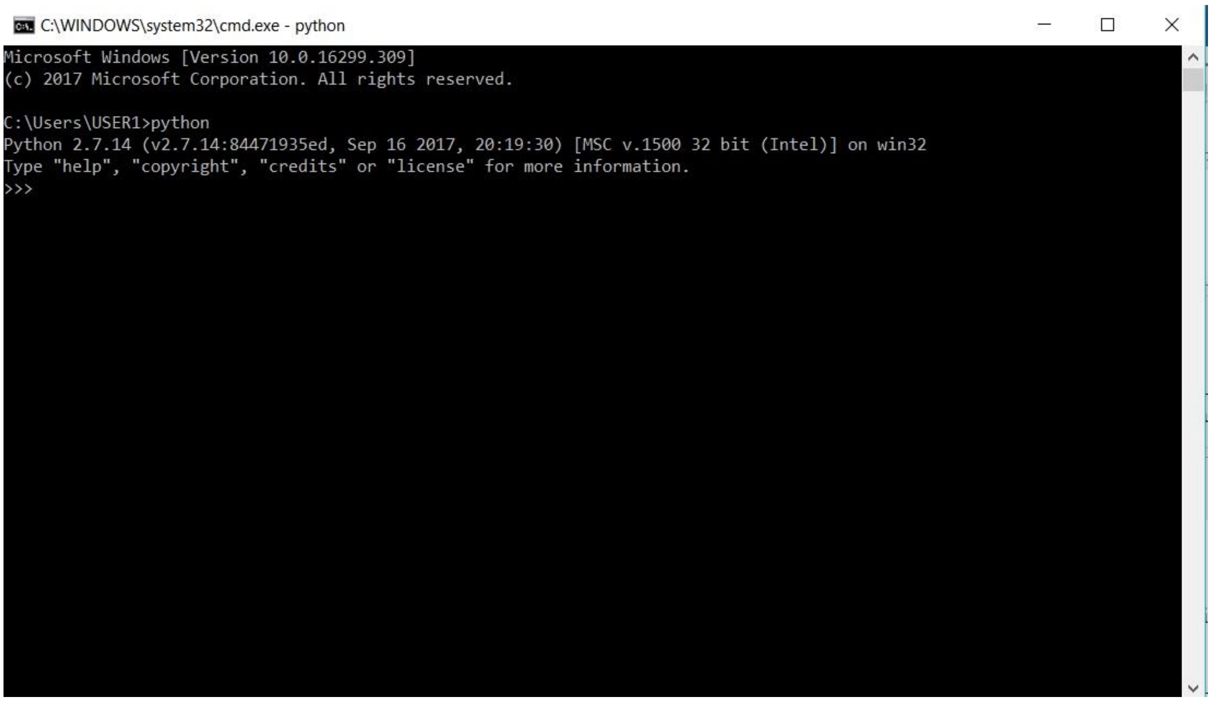
Python Interpreter and Libraries

This feature requires 0KB on your hard drive. It has 0 of 7 subfeatures selected. The subfeatures require 0KB on your hard drive.

Figure 3.1 Python Installation

Select all the features and click on Next. The installation will take some time to complete. Once the installation is complete we can verify it by using the command "python" in command prompt window.

If python is installed on our computer, we get an output similar to this –

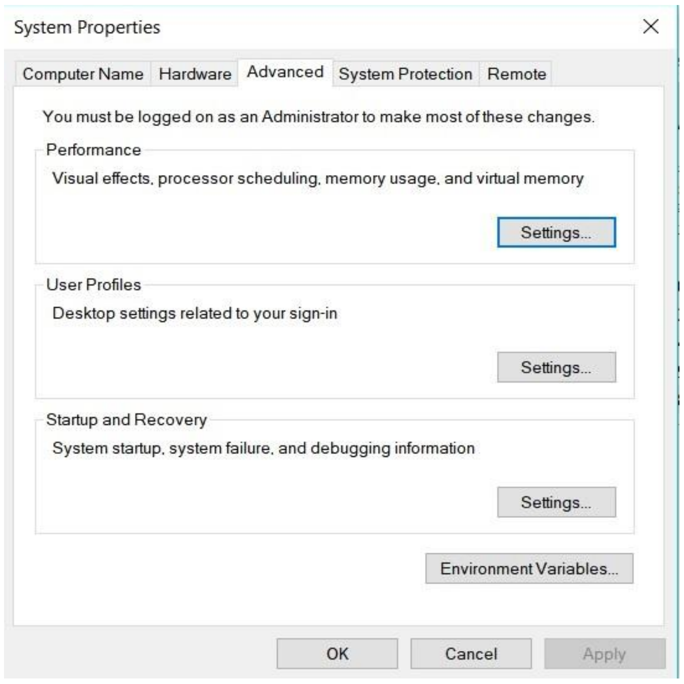


```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.16299.309]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\USER1>python
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 3.2 Verifying Python Installation

If we encounter an error in this stage, we will have to manually update the environmental variable. In Windows, we can accomplish this by searching for "Environmental Variables" in Windows search bar.



System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...

User Profiles
Desktop settings related to your sign-in
Settings...

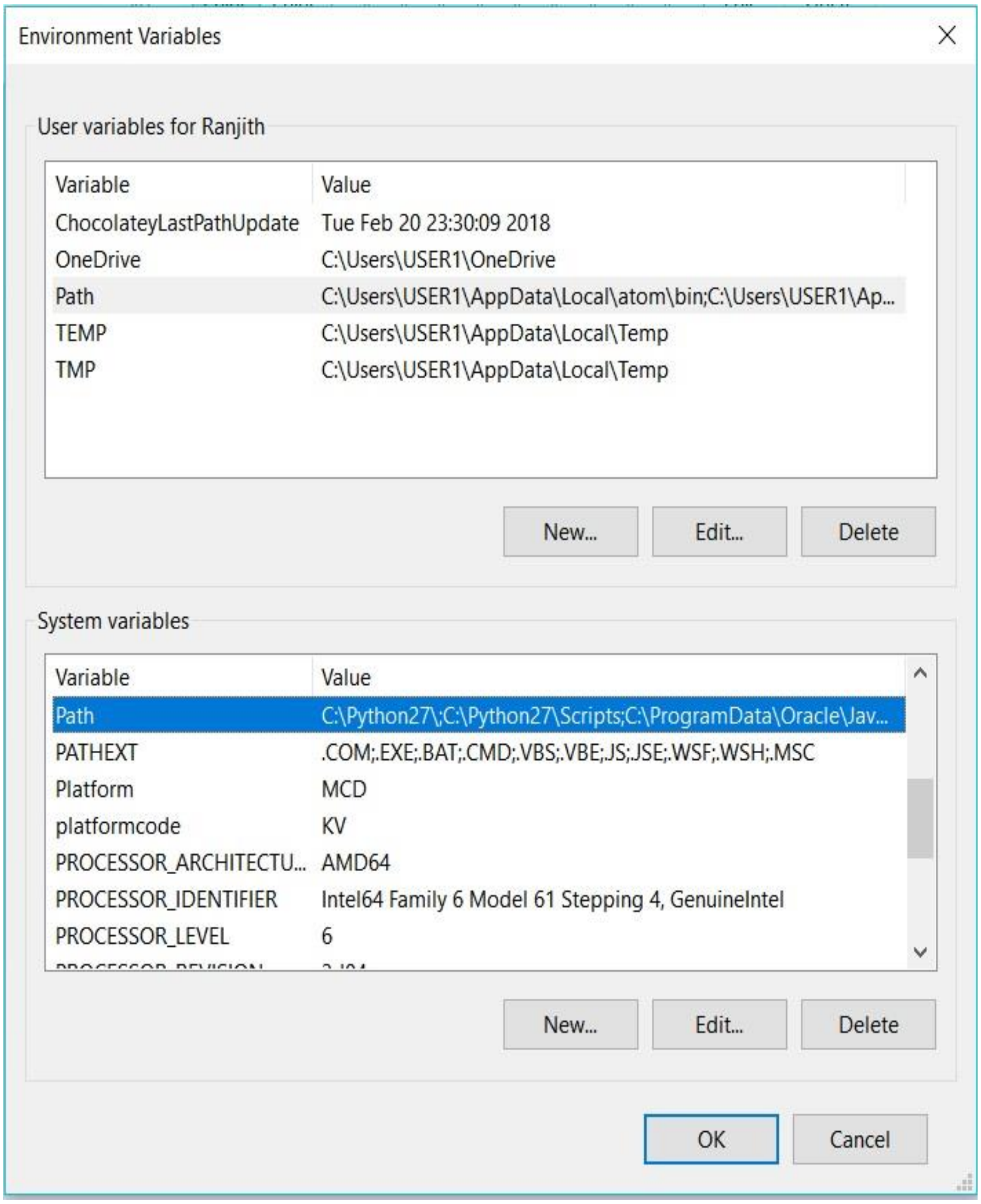
Startup and Recovery
System startup, system failure, and debugging information
Settings...

Environment Variables...

OK Cancel Apply

Figure 3.3 Opening Environmental Variable

Now, we have to click on “Environmental Variables”. Next we will have to select a variable called “Path”. Once we select the variable we need to add the path where python is installed. In our case it was “C:\Python27”.


Environment Variables

User variables for Ranjith

Variable	Value
ChocolateyLastPathUpdate	Tue Feb 20 23:30:09 2018
OneDrive	C:\Users\USER1\OneDrive
Path	C:\Users\USER1\AppData\Local\atom\bin;C:\Users\USER1\Ap...
TEMP	C:\Users\USER1\AppData\Local\Temp
TMP	C:\Users\USER1\AppData\Local\Temp

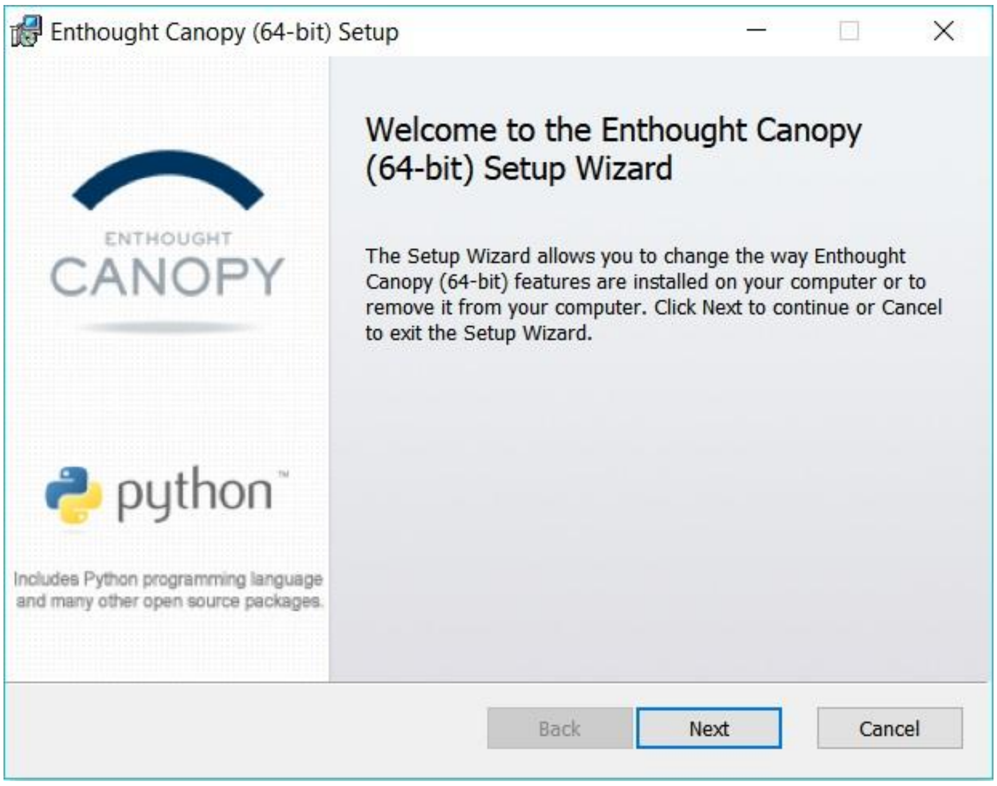
System variables

Variable	Value
Path	C:\Python27\;C:\Python27\Scripts;C:\ProgramData\Oracle\Jav...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
Platform	MCD
platformcode	KV
PROCESSOR_ARCHITECTU...	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 61 Stepping 4, GenuineIntel
PROCESSOR_LEVEL	6

Figure 3.4 Setting the Environmental Path for Python

3.6 Installing Enthought Canopy

We need to first download the setup file from https://www.enthought.com/product/canopy/. Select the appropriate version based on your Operating System and System Architecture. Make sure to download the version that is compatible with Python 2.7. Select the installation directory for Canopy to be installed. Once the installation is done you will have to wait for it to automatically set up the working environment for development.


Enthought Canopy (64-bit) Setup

Welcome to the Enthought Canopy (64-bit) Setup Wizard

The Setup Wizard allows you to change the way Enthought Canopy (64-bit) features are installed on your computer or to remove it from your computer. Click Next to continue or Cancel to exit the Setup Wizard.

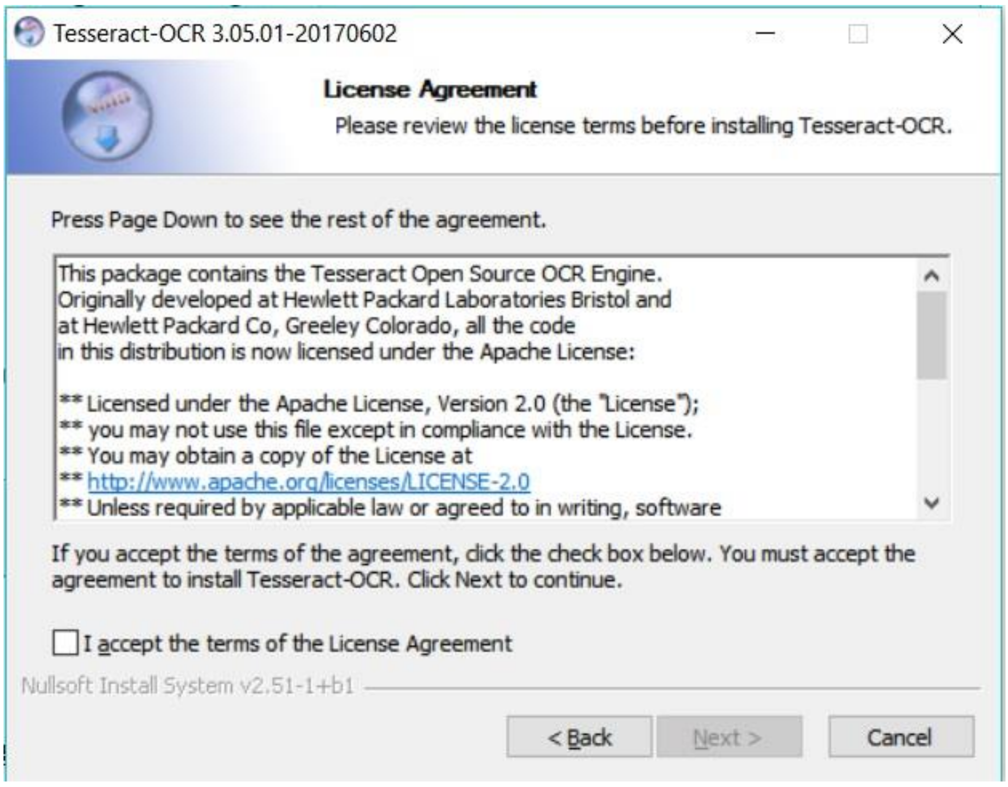
Includes Python programming language and many other open source packages.

Figure 3.5 Installing Enthought Canopy

3.7 Installing Tesseract OCR Binaries

Download the Tesseract setup files from https://github.com/UB-Mannheim/tesseract/wiki.

Make sure to download the correct setup which is named “tesseract-ocr-setup-3.05.01.exe”. Once downloaded, you can begin the installation procedure. The setup will ask you to select language of instruction. Once you select the language you will be prompted to accept the terms and conditions. Once that is done you can select the language packs that you want to download along with English.


Tesseract-OCR 3.05.01-20170602

License Agreement

Please review the license terms before installing Tesseract-OCR.

Press Page Down to see the rest of the agreement.

This package contains the Tesseract Open Source OCR Engine.
Originally developed at Hewlett Packard Laboratories Bristol and
at Hewlett Packard Co, Greeley Colorado, all the code
in this distribution is now licensed under the Apache License:

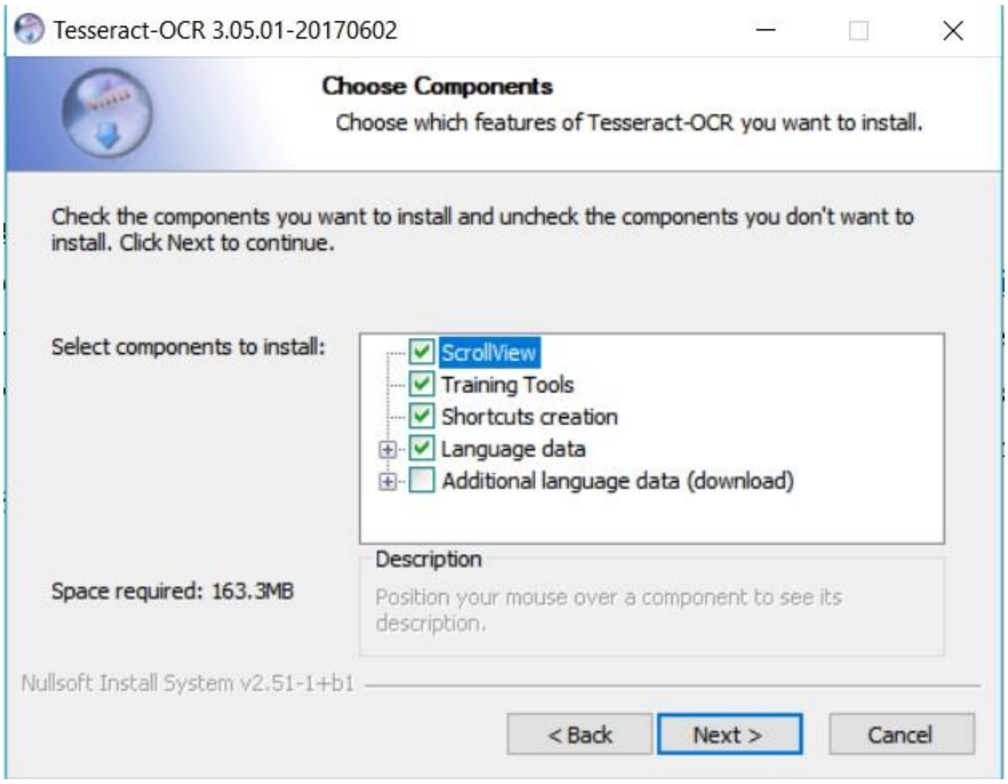
** Licensed under the Apache License, Version 2.0 (the "License");
** you may not use this file except in compliance with the License.
** You may obtain a copy of the License at
** http://www.apache.org/licenses/LICENSE-2.0
** Unless required by applicable law or agreed to in writing, software

If you accept the terms of the agreement, click the check box below. You must accept the agreement to install Tesseract-OCR. Click Next to continue.

☐ I accept the terms of the License Agreement

Nullsoft Install System v2.51-1+b1

Figure 3.6. Installing Tesseract OCR


Tesseract-OCR 3.05.01-20170602

Choose Components

Choose which features of Tesseract-OCR you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

- ✓ ScrollView
- ✓ Training Tools
- ✓ Shortcuts creation
- ✓ Language data
- ☐ Additional language data (download)

Space required: 163.3MB

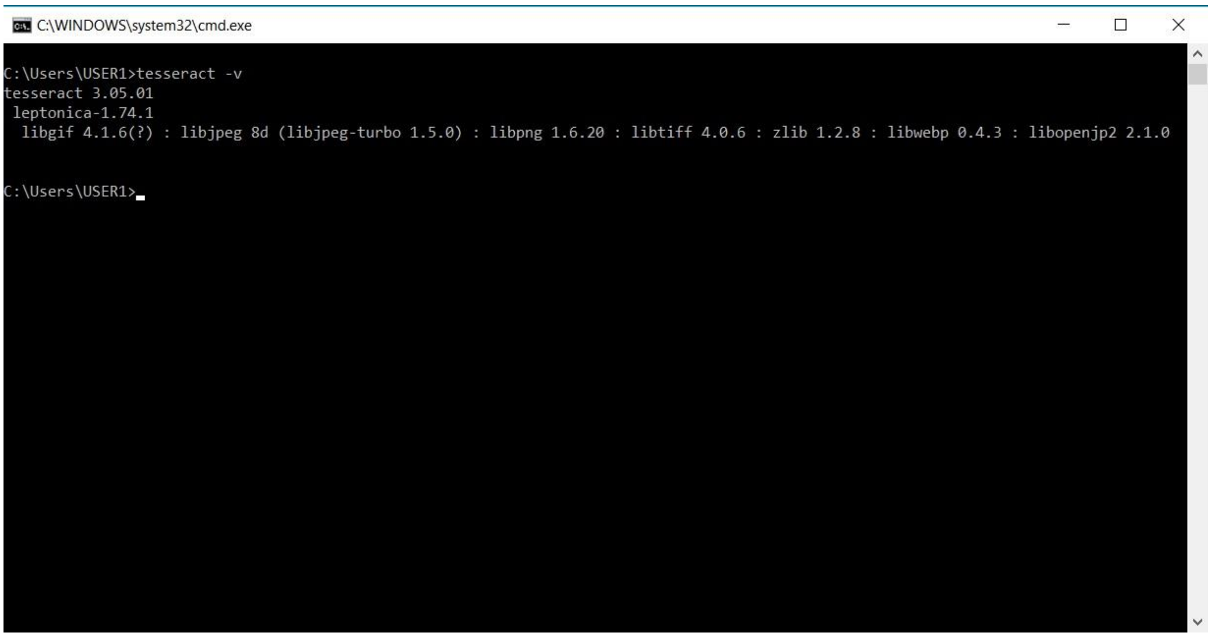
Description

Position your mouse over a component to see its description.

Nullsoft Install System v2.51-1+b1

Figure 3.7 Selecting Language Packs in Tesseract OCR

To check the integrity of installation, we will use the command prompt again. By using the command “tesseract -v” we can get to know the version and if the binaries have been installed properly. If it has been installed properly the output will be something similar to this -



```
C:\Users\USER1>tesseract -v
tesseract 3.05.01
 leptonica-1.74.1
  libgif 4.1.6(?) : libjpeg 8d (libjpeg-turbo 1.5.0) : libpng 1.6.20 : libtiff 4.0.6 : zlib 1.2.8 : libwebp 0.4.3 : libopenjp2 2.1.0


C:\Users\USER1>
```

Figure 3.8 Verifying Tesseract Installation

If the command gives an error, we need to add 2 variables to the Environmental Variable. The procedure is the same as before to open the variable named “Path” and add the directory of the tesseract library in the variable. In our case we need to add “C:\Program Files(x86)\Tesseract-OCR”. Along with this we also need to create an additional variable called “TESS_DATA_PREFIX” and set its value to “C:\Program Files(x86)\Tesseract-OCR\tessdata”.

3.8 Installing Android Studio

We need to download the latest version of Android Studio and install it. The procedure for installation is very simple. We need to first accept to the terms and conditions. Once that is done it will automatically check the java sdk installation and setup the environment for working.

3.9 Installing Java SDK

Java SDK can be downloaded and installed from Oracle website. The installation is straightforward. We need to just accept the terms and conditions. The default path for installation is preferred and we just need to click Next. A restart will be prompted once installation is complete.

3.10 Installing Libraries

We can install all the libraries by using canopy command prompt. We use the "pip" command to install all the libraries.

To Install OpenCV, we use the command – "pip install opencv-python".

To Install Matplot, we use the command – "pip install matplot".

To Install Python Image Processing Library, we use the command – "pip install pillow".

To Install pytesseract, we use the command – "pip install pytesseract".

To Install pickle, we use the command – "pip install cpickle".

To Install sklearn, we use the command – "pip install sklearn".

Every other library used while coding the project are in-built libraries that are installed along with Python 2.7.

CHAPTER 4

PROPOSED SYSTEM

4.1 Problem Definition

The application being developed will ideally provide information to the user based on the text which is present in the image that is being fed as the input. A combination of detection algorithm, machine learning techniques and optical character recognition will ultimately help in deciphering the text present in an image to obtain more valuable information related to it. The accuracy of detection and recognition plays a vital role and we hope to improve this.

The proposed system consists of four major parts:

- Text detection
- Machine Learning procedure (SVM) for non-text rejection
- Text Recognition
- Google Search

4.2 Text Detection

- The Stroke width Transform (SWT) technique is performed to detect the text from the images.
- SWT detects the alphabets in an image based on the gradient direction of each letter.
- Then Connected Component analysis method is done to specify the individual words which are present in the image.
- This gives us an image with boxes around the words.

4.3 Support Vector Machine

- Support Vector Machine (SVM) is used to train the machine for achieving faster results.

- SVM algorithm is used to train the machine for separating the textual information in the image.
- SVM provides more refined and fast results.

4.4 Text Recognition

- Text Recognition is performed by utilizing the power of Optical Character Recognition (OCR)
- OCR is performed by using an open-source library called “Tesseract”.
- Tesseract is the most advanced text recognition library available.
- It performs all the functions required to recognize text such as – Paragraph Segmentation, Line Segmentation and Character Segmentation.

4.5 Google Search

- Once the text is recognized we can use the result to perform a Google Search and retrieve the required result.
- Thus we would obtain the required information of the text in the image which would be very useful.

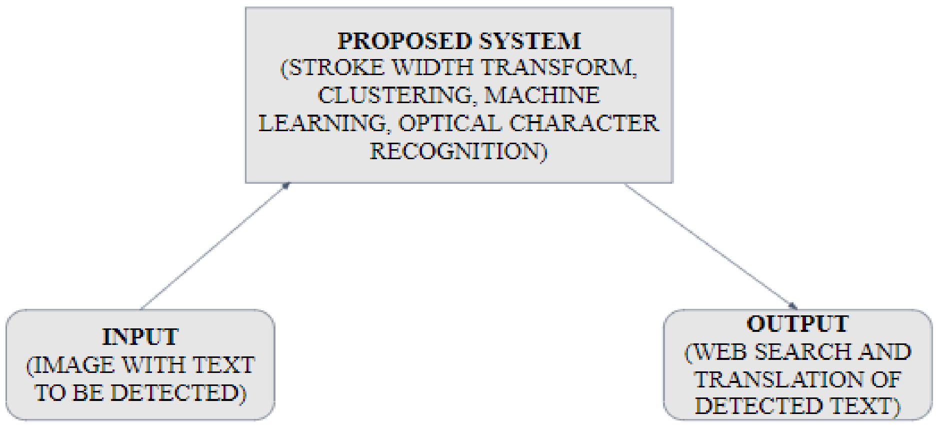

PROPOSED SYSTEM
(STROKE WIDTH TRANSFORM, CLUSTERING, MACHINE LEARNING, OPTICAL CHARACTER RECOGNITION)
INPUT
(IMAGE WITH TEXT TO BE DETECTED)
OUTPUT
(WEB SEARCH AND TRANSLATION OF DETECTED TEXT)

Figure 4.1 Basic Workflow diagram

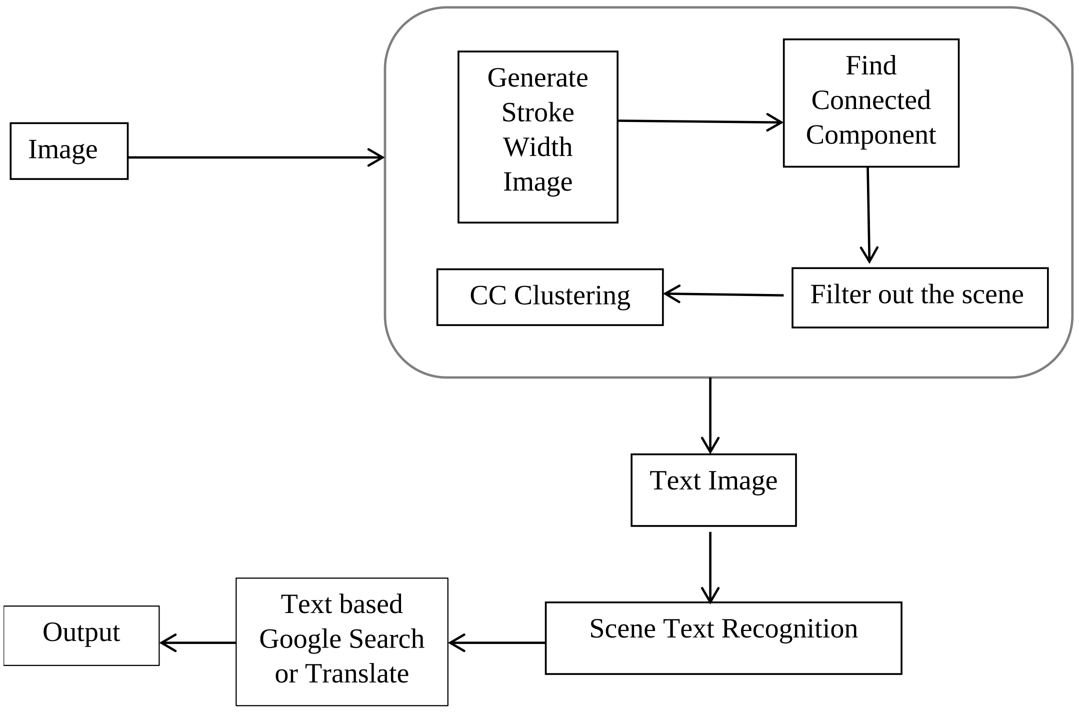


Figure 4.2 Workflow Diagram for the Application

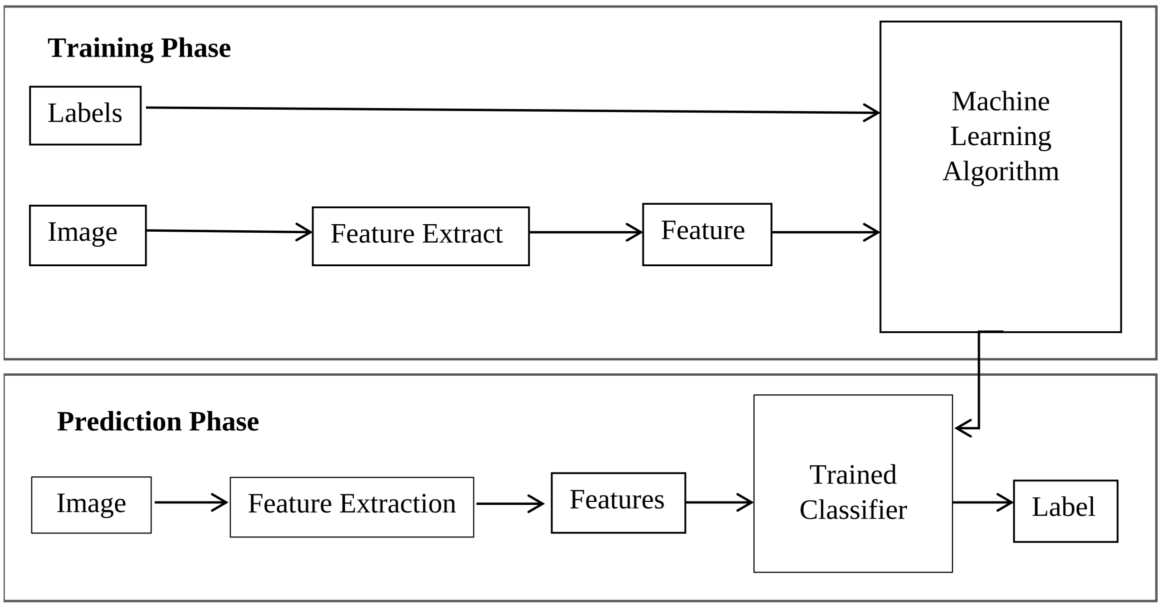


Figure 4.3 Workflow Diagram for Training

CHAPTER 5

SYSTEM DESIGN

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development.

The purpose of the system design is to plan the solution of a problem specified by the requirements document. This phase is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs. The design of the system is perhaps the most critical factor affecting the quality of the software and has a major impact on the later phases, particularly testing and maintenance. System design aims to identify the modules that should be in the system, the specifications of these modules are to interact with each other to produce the desired results.

5.1 Data Flow Diagram for the application

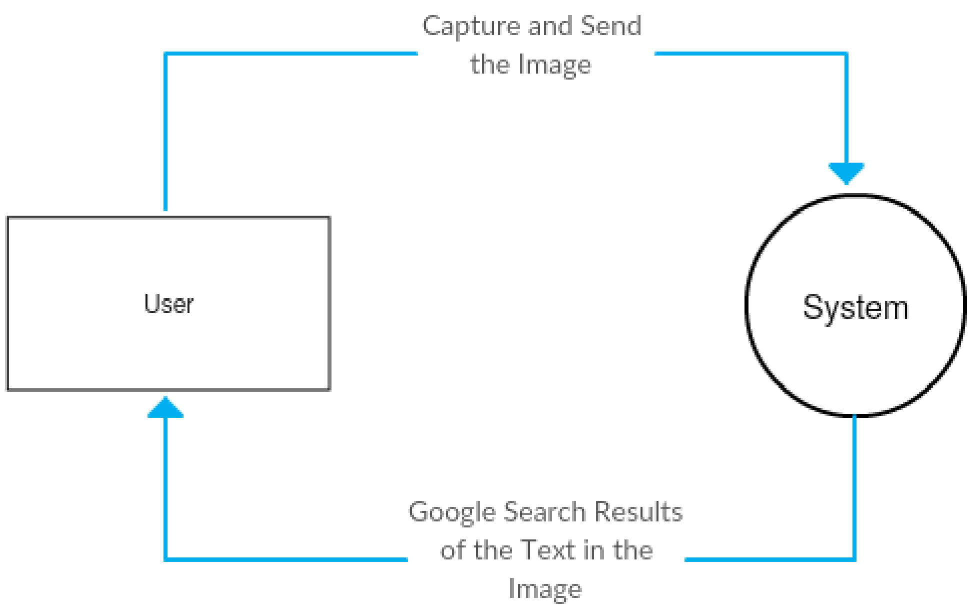

Capture and Send the Image
User
System
Google Search Results of the Text in the Image

Figure 5.1 DFD Level 0 Diagram

Data flow diagram level 0 illustrates abstractly the data flow in the system from input from the user to Google Search results of the text in the image as shown in figure 5.1.1. The user is asked to capture an image and then send it to the program. The image then undergoes some processing to give the required results. The output would contain the Google search results of the text in the image along with the image of detected text with boxes around it and the recognized text.

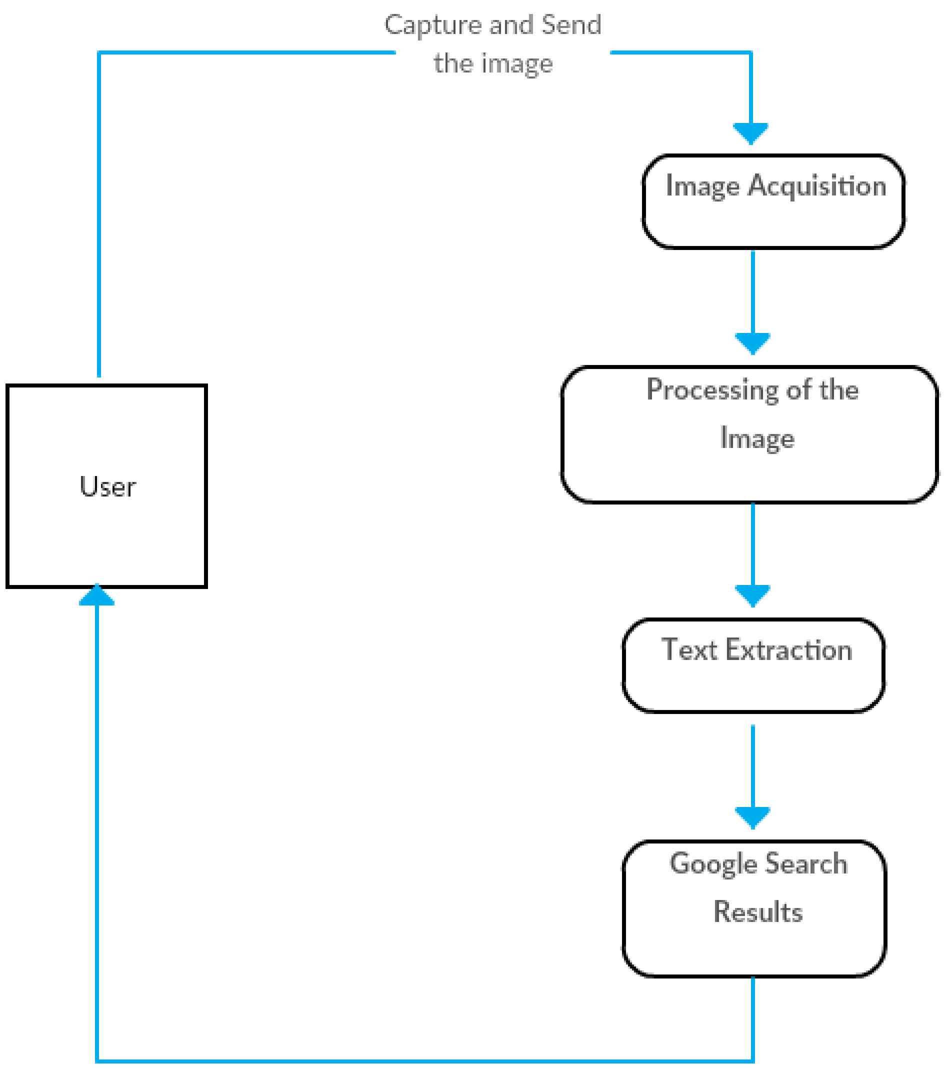


Figure 5.2 DFD Level 1 Diagram

Data flow diagram level 1 illustrates in brief the data flow in the system as shown in figure 5.1.2. The user captures an image and then sends it to the program. The image then undergoes some processing to give the required results. All the steps mentioned in the above diagram takes place accordingly. The output would contain the Google search results of the text in the image along with the image of detected text with boxes around it and the recognized text.

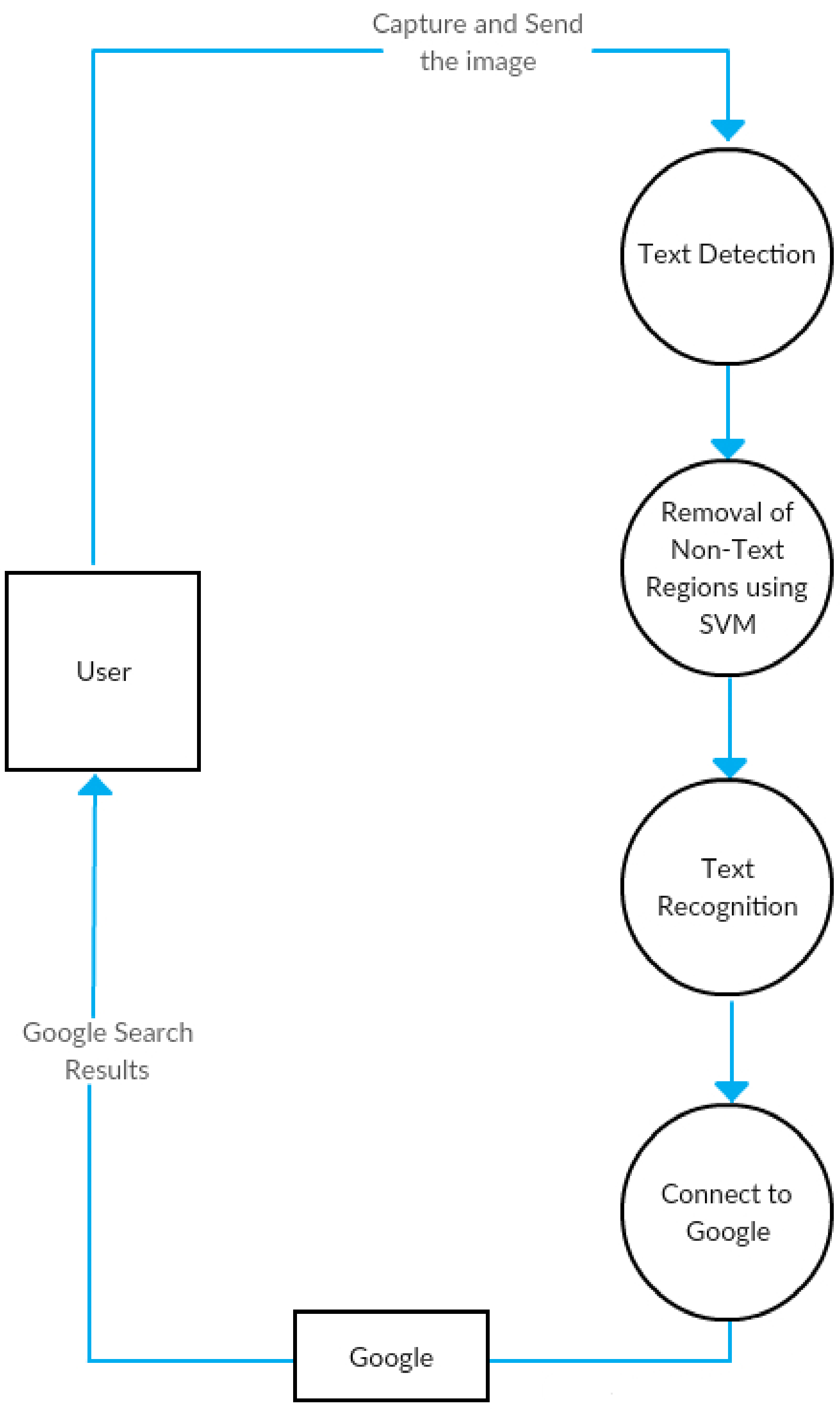


Figure 5.3 DFD Level 2 Diagram

Data flow diagram level 2 illustrates in brief the data flow in the system as shown in figure 5.1.3. The user captures an image and then sends it to the program. The image then undergoes some processing to give the required results. All the steps mentioned in the above diagram takes place accordingly. Text Detection is first done by SWT and Connected Component Analysis followed by rejection of non-text regions using SVM. The text is then recognized and then google search is performed for that tex. The output would contain the Google search results of the text in the image along with the image of detected text with boxes around it and the recognized text.

5.2 Use Case Diagram

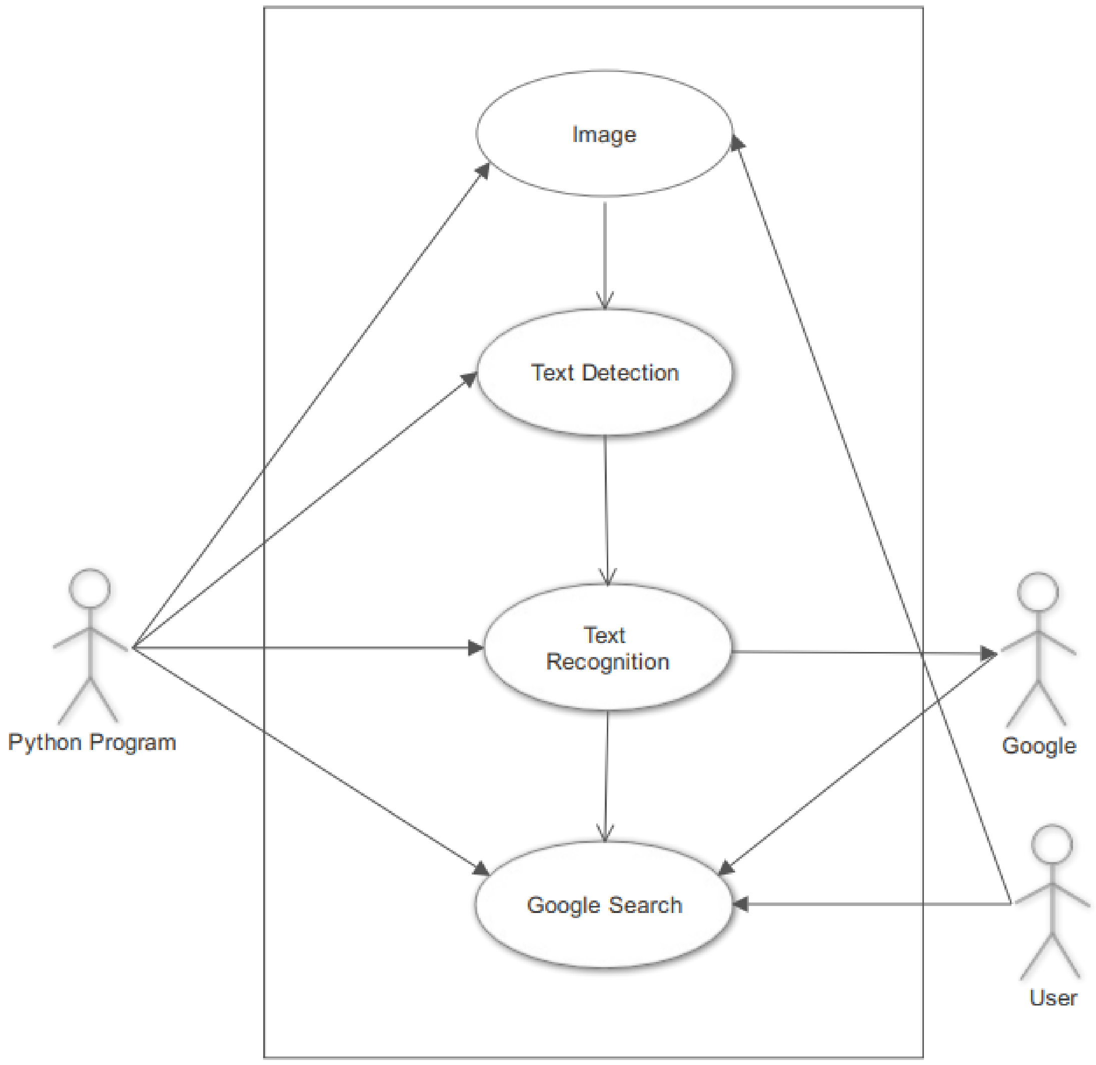


Figure 5.4 Use Case Diagram for the Application

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

In this diagram we show as to what will happen as the user provides the input to the program and the entities the users can access. It also provides the required results at the end to the user.

5.3 Sequence Diagram

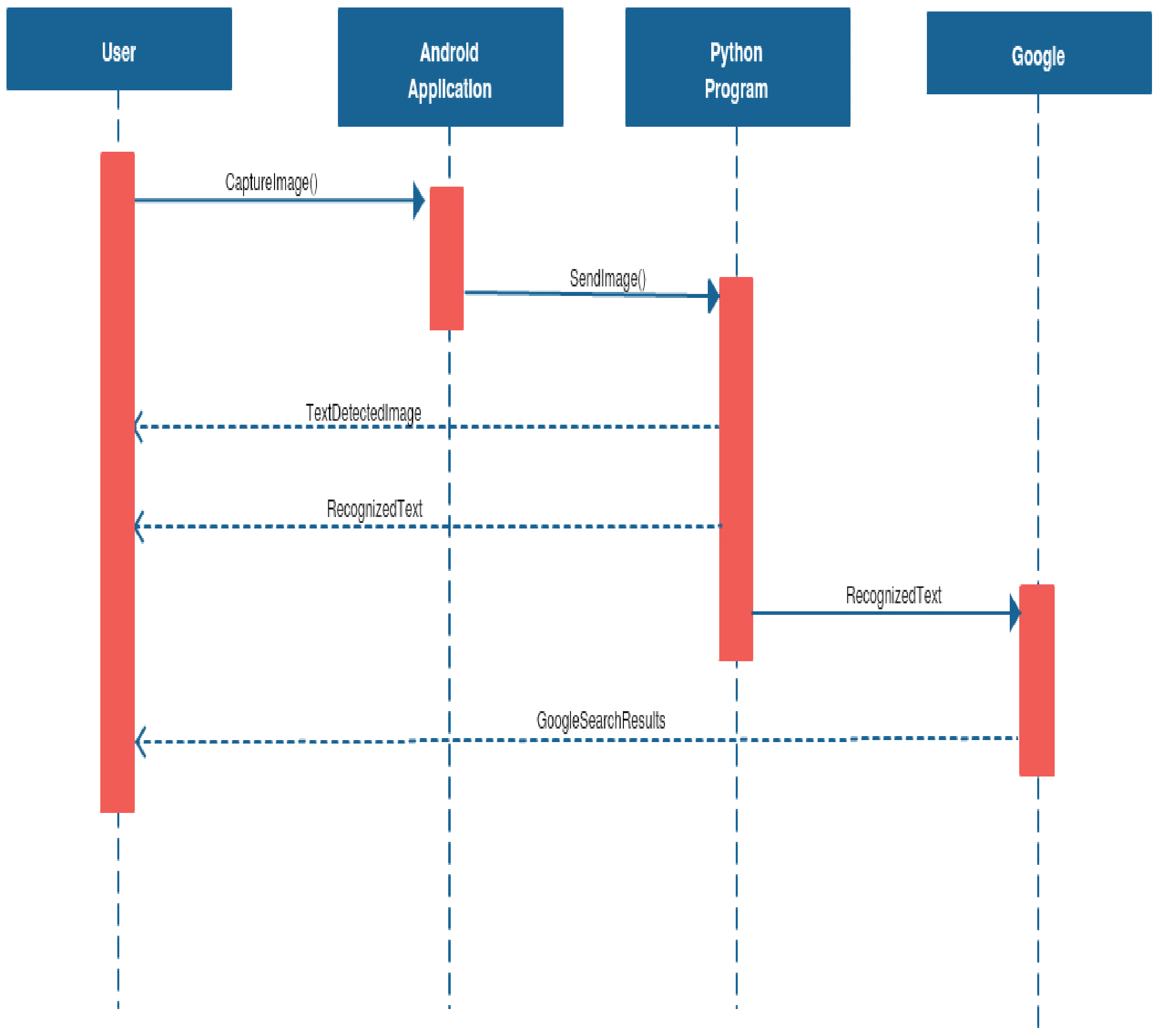


Figure 5.5 Sequence Diagram for the Application

Sequence diagram provides the sequence of activities performed during the execution of a module or component in the system. The Figure 5.3depicts the sequence of activities between the user and the Python program. The user captures the image of his choice and then sends it to the Python program. Python program then executes the required steps, provides the image with the green boxes which depict the detected text in the image, returns the recognized text to the user and also sends the recognized text to the Google web platform. The Google web platform in turn executes the search for the provided text and returns the results of the search results to the user.

5.4 Flow Chart for the Application

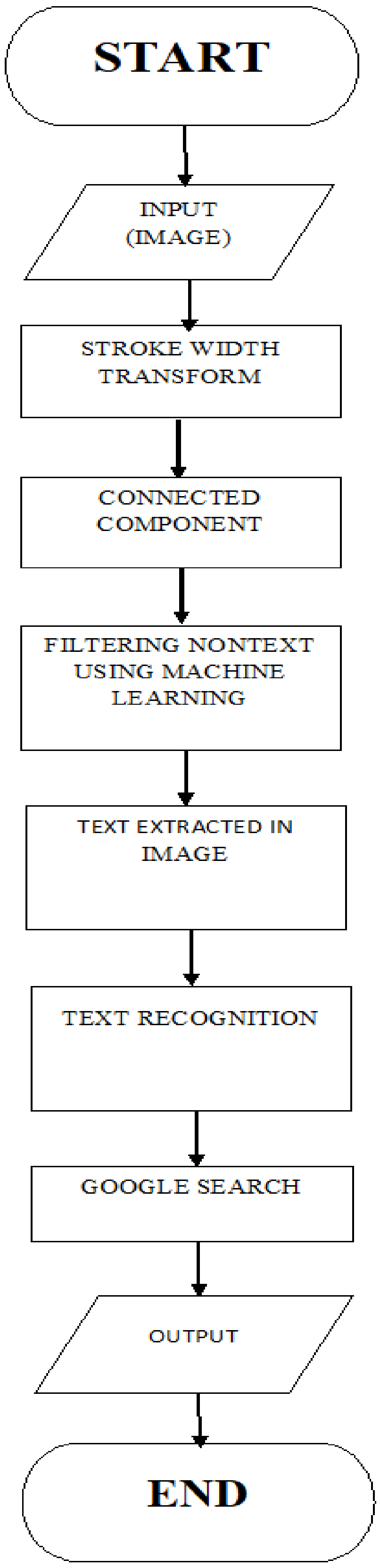


Figure 5.6 Flow Chart for the application

CHAPTER 6

IMPLEMENTATION

The implementation phase of any project development is the most important phase as it yields the final solution, solves the problem at hand. The implementation phase involves the actual materialization of the ideas, which are expressed in the analysis document and developed in the design phase.

6.1 Programming Language

The programming language used for the project is Python. Python is an open source programming and is an interpreted high-level programming language for general-purpose programming. It provides various libraries such as Tesseract – which is used for recognizing the text in various situations which might be text document, text in an image and so on.

It also provides the libraries such as Scikit – which helps in implementing the various machine learning algorithms much easier and provides the required files to support the functioning of it in a smooth way.

We also make use of JAVA programming language for developing the android application for our project.

6.2 Platform Used

Windows platform is used for this project and Enthought Canopy. interpreted high-level programming language for general-purpose programming. It provides various libraries such as Tesseract – which is used for recognizing the text in various situations which might be text document, text in an image and so on.

6.3 Stroke Width Transform

The Stroke Width Transform (SWT) is a local image operator which computes per pixel the width of the most likely stroke containing the pixel. The output of the SWT is an image of size equal to the size of the input image where each element contains the width of the stroke associated with the pixel. A stroke is defined to be a contiguous part of an image that forms a band of a nearly constant width.

Stroke width transform is applied on edge image. Therefore, first the edge image is generated using canny edge detector. Then for each pixel of that edge image, gradient direction dp as shown in Fig. 5 is considered. If p is a pixel lies on the stroke boundary, then calculate its gradients dp, it must be perpendicular to the direction of stroke. Follow the ray until another edge pixel q is found. Now at edge pixel q consider gradient direction dq. If dq is roughly opposite to dp (dp = - dq ʌ/6) then consider the segment [p, q] and assigned the width ||p-q|| to pixel p unless it already had a lower value[2] and the algorithm:

```
Input:    grayImage

Output:  swtImage

dp ← gradient direction at pixel p.

dq ← gradient direction at another edge pixel q.

edgeImage ← canny (grayImage, low_threshold, high_thrshold);

for all pixels € swtImage do

      s ← ∞

end for

for all pixelp € edgeImage do

      if (dq = - dp ʌ/6) then

                for all pixels € swtImage do

                        s ← width(||p-q||)

                end for

      end if

end for

return (swtImage)
```

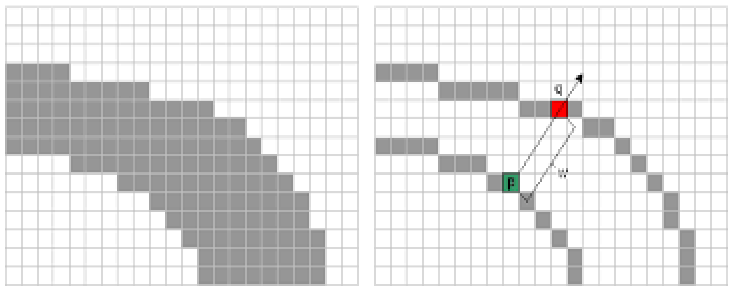


Figure 6.1 Stroke Width Principle

6.4 Connected Components Formation

In order to form a connected component in SWT image, there is a need to modify classical connected component algorithm. One of the important modifications to classical connected component algorithm is to group two neighbouring pixels in the SWT image, if they have similar stroke width [2]. In order to allow more elaborate fonts and perspective distortion, modification is to group two neighbouring pixels in the SWT image, if their SWT ratio does not exceed 3.0. This local rule guarantees that strokes with smoothly varying widths will also be grouped together. In order to accommodate both the cases, bright text on dark background and vice-versa, apply the algorithm twice, once along dp and once along –dp. Algorithm gives the modified connected component labelling algorithm for formation of connected component in the SWT image.

Input: swtImage

Output: Set of Connected Component

X-neighbours ← Set of neighbours that are connected to selected pixel.

For x=8, X-neighbors= {a, b, c, d}

Pass 1:

```
for all Pixelp € swtImage do
        for all Pixelp = 1 do
                case 1: if (all four neighbour == 0) then
                        p ←  new label;
```

```
            case 2: if (only one of the neighbour == 1) then
                        if (SWT ratio <=3.0) then
                        p ← neighbor’s label;
            case 3: if (more than one of the neighbour == 1) then
                        if (SWT ratio <=3.0) then
                        p ← one of the neighbours’ label    && Make a note of the
            equivalences.
      end for
end for
```

Sort the equivalent label pairs into equivalence classes and assign a unique label to each class.

Pass 2:

Replace each label by the label assigned to its equivalence classes.

return (count(label))

For display, the labels might be different gray levels or colors.

6.5 Support Vector Machine

Classification includes a broad range of decision-theoretic approaches to the identification of images (or parts thereof). All classification algorithms are based on the assumption that the image in question depicts one or more features (e.g., geometric parts in the case of a manufacturing classification system, or spectral regions in the case of remote sensing, as shown in the examples below) and that each of these features belongs to one of several distinct and exclusive classes. The classes may be specified a priori by an analyst (as in supervised classification) or automatically clustered (i.e. as in unsupervised classification) into sets of prototype classes, where the analyst merely specifies the number of desired categories. Classification and segmentation have closely related objectives, as the former is another form of component labeling that can result in segmentation of various features in a scene.

Image classification analyzes the numerical properties of various image features and organizes data into categories. Classification algorithms typically employ two phases of processing: training and testing. In the initial training phase, characteristic properties of typical image features are isolated and, based on these, a unique description of each classification category, i.e. training class, is created. In the subsequent testing phase, these feature-space partitions are used to classify image features.

The description of training classes is an extremely important component of the classification process. In supervised classification, statistical processes (i.e. based on an a priori knowledge of probability distribution functions) or distribution-free processes can be used to extract class descriptors. Unsupervised classification relies on clustering algorithms to automatically segment the training data into prototype classes.

Two main classification methods are:

- Supervised Classification: Supervised learning is the machine learning task of inferring a function from labeled training data. The training data consist of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances.
- Unsupervised Classification: Unsupervised classification is a method which examines a large number of unknown pixels and divides into a number of classed based on natural groupings present in the image values. Unlike supervised classification, unsupervised classification does not require analyst-specified training data. The basic premise is that values within a given cover type should be close together in the measurement space (i.e. have similar gray levels), whereas data in different classes should be comparatively well separated (i.e. have very different gray levels).

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

In order to train an SVM, a suitable kernel function needs to be selected and the kernel parameters and the trade-off parameter C need to be chosen. The quality of the classifications can be greatly affected by C, since it determines how severely classification errors must be penalized. A large C value may lead to over fitting problems, thus reducing the ability of SVM to generalize. The kernel and all these parameters are usually defined by cross-validation techniques.

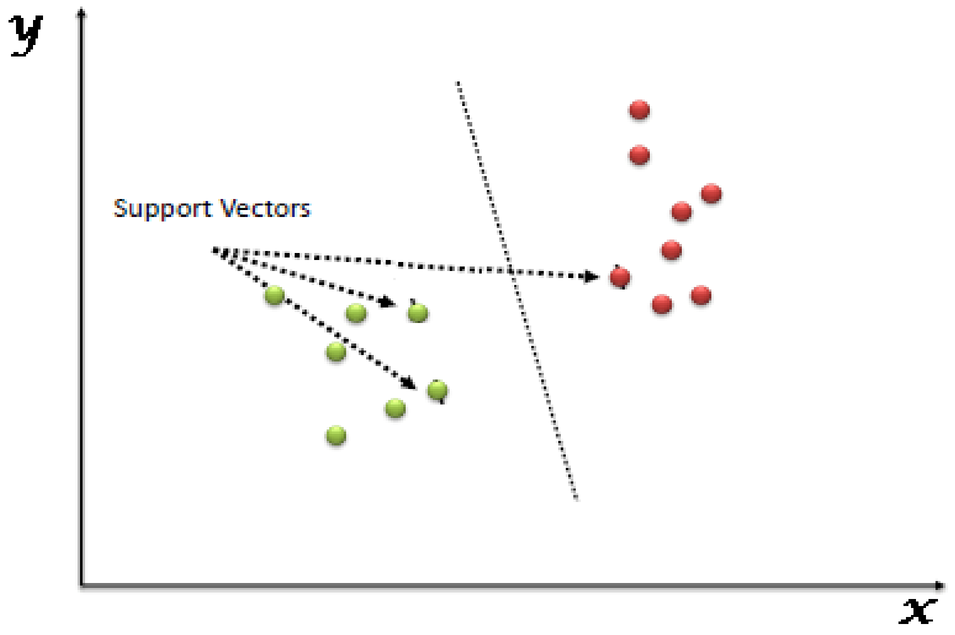


Figure 6.2 Support vectors of SVM classifier

We train the machine with the set of non-textual images. The algorithm trains the machine with the set of values according to a given classifier and then saves the value in a given text file which can be made use of for classifying the images during testing process.

Classifying data has been one of the major parts in machine learning. The idea of support vector machine is to create a hyper plane in between data sets to indicate which class it belongs to. The challenge is to train the machine to understand structure from data and mapping with the right class label, for the best result, the hyper plane has the largest distance to the nearest training data points of any class.

Classification of the images take place by making use of the trained data set. The training set is used to classify the textual and the non-textual contours in the processed image. Thus after classification we get more refined and accurate results for the image.

6.6 Feature Extraction

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. If there are n observations with p variables, then the number of distinct principal components is – min(n-1,p). This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors are an uncorrelated orthogonal basis set. PCA is sensitive to the relative scaling of the original variables.

Principal component analysis (PCA) is a technique used to emphasize variation and bring out strong patterns in a dataset. It's often used to make data easy to explore and visualize. First, consider a dataset in only two dimensions, like (height, weight). This dataset can be plotted as points in a plane. But if we want to tease out variation, PCA finds a new coordinate system in which every point has a new (x,y) value. The axes don't actually mean anything physical; they're combinations of height and weight called "principal components" that are chosen to give one axes lots of variation.

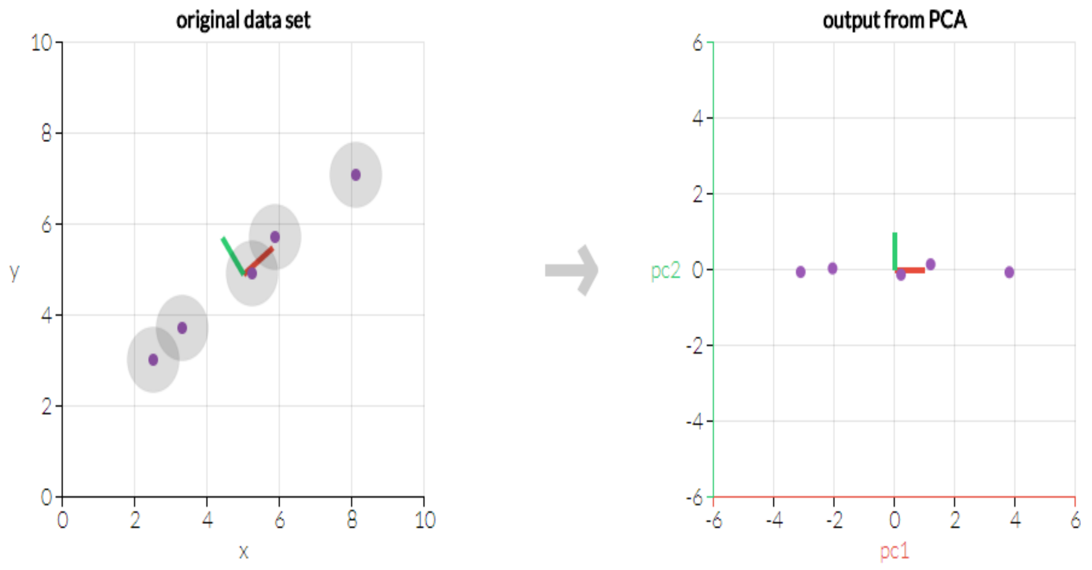


Figure 6.3 Principle Component Analysis

6.7 Google OCR - Tesseract

Tesseract is an OCR engine with support for unicode and the ability to recognize more than 100 languages out of the box. It can be trained to recognize other languages.

Tesseract is one of the most accurate open source OCR engines. Tesseract allows us to convert the given image into the text. Before going to the code we need to download the assembly and tessdata of the Tesseract. We can download the data from GitHub or NuGet.

After downloading the assembly, add the assembly in your project.

Then add the following package.

import pytesseract

Once this is done we can make use of the tesseract to recognize the given text in the image easily and output of that would be displayed on the screen. We can make use of this recognized text by processing it and ensuring we provide some useful information.

6.8 Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development.

The following features are provided in the current stable version: Gradle-based build support, Android-specific refactoring and quick fixes, Lint tools to catch performance, usability, version compatibility and other problems, ProGuard integration and app-signing capabilities, Template-based wizards to create common Android designs and components

A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations, Support for building Android Wear apps, Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine, Android Virtual Device (Emulator) to run and debug apps in the Android studio.

We create an app which can be made used for sending the image to the program through the app by creating a local server on our system. The image can be taken from our phones and then sent to the server by making use of the IP address of the given network. The app provides a user friendly interface for us and ensures an easy transfer of image to the program and helps us process it accordingly to provide the required output to the user.

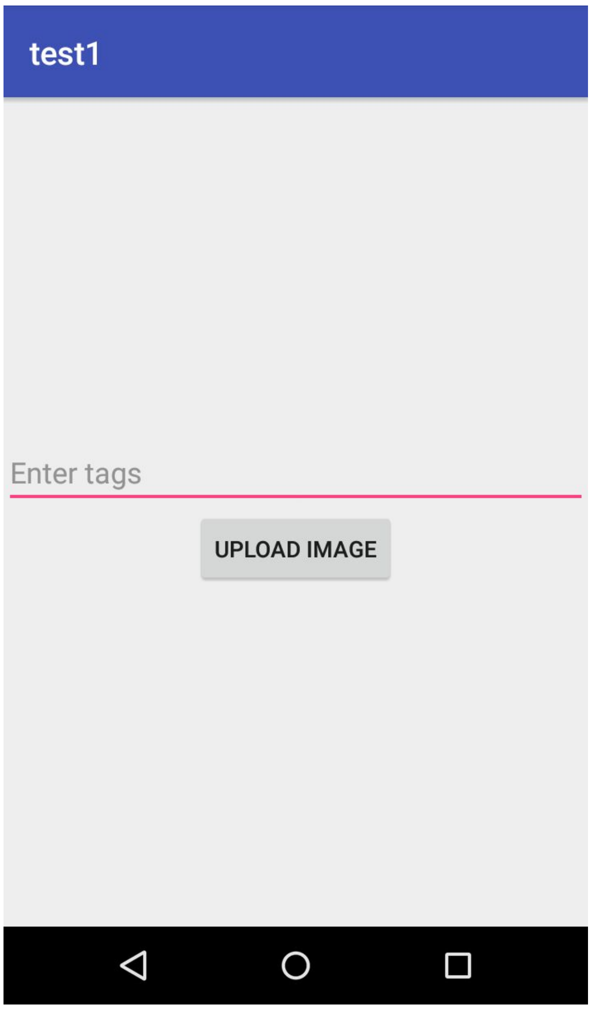


Figure 6.4 App User Interface

CHAPTER 7

EXPERIMENTAL RESULTS AND TESTING

The results are obtained can be conclusive proof that the proposed system has achieved higher accuracy while comparing it with just using the Tesseract OCR system. Most applications that utilize "Tesseract" module tend to use it directly since it is extremely accurate for scanning and recognizing the text present in documents. In case of text that is present in natural scenery, it doesn't tend be so simple since there are a lot of variable that interfere with the recognition procedure. In this paper we have used images from COCO [14], which contains a large number of images which is primarily used for training applications which contain sub-systems that perform object detection.

We compare the accuracy of recognition of text using just the Tesseract API verses utilizing our proposed methodology. In our paper we compare two images from the aforementioned dataset.

Formula to calculate Accuracy in OCR systems is given by Alexandrov [15].

$$G_{rec} = 100 \times \left\{ \left| \frac{n_c - n_e - n_{rej}}{n_c} \right| \right\}$$

Where, G_{rec} = Accuracy of the system, n_c = Number of characters in the system, n_e = Number of erroneous characters and n_{rej} = Number of characters rejected by OCR.

Test Cases

Case 1 –

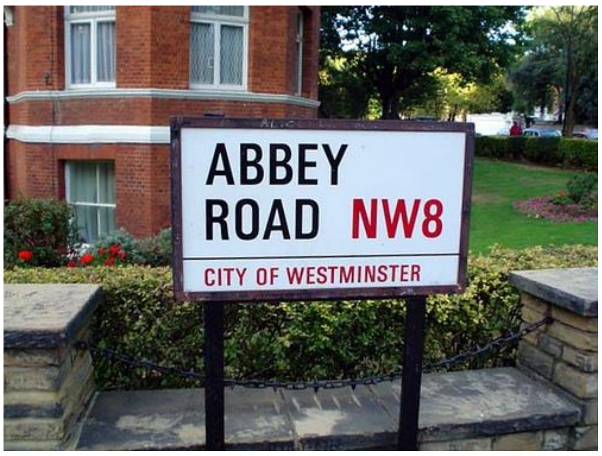


Fig 7.1 Original Image

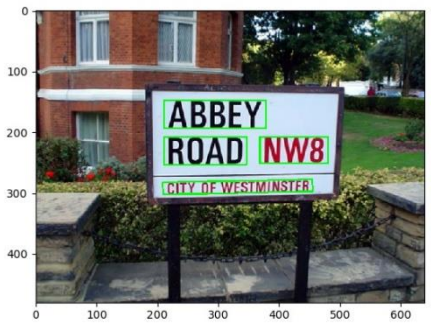
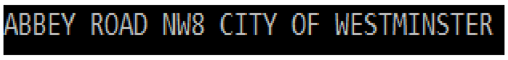


Figure 7.2 Detected Regions



```
ABBEY ROAD NW8 CITY OF WESTMINSTER
```

Figure 7.3 Detected Text

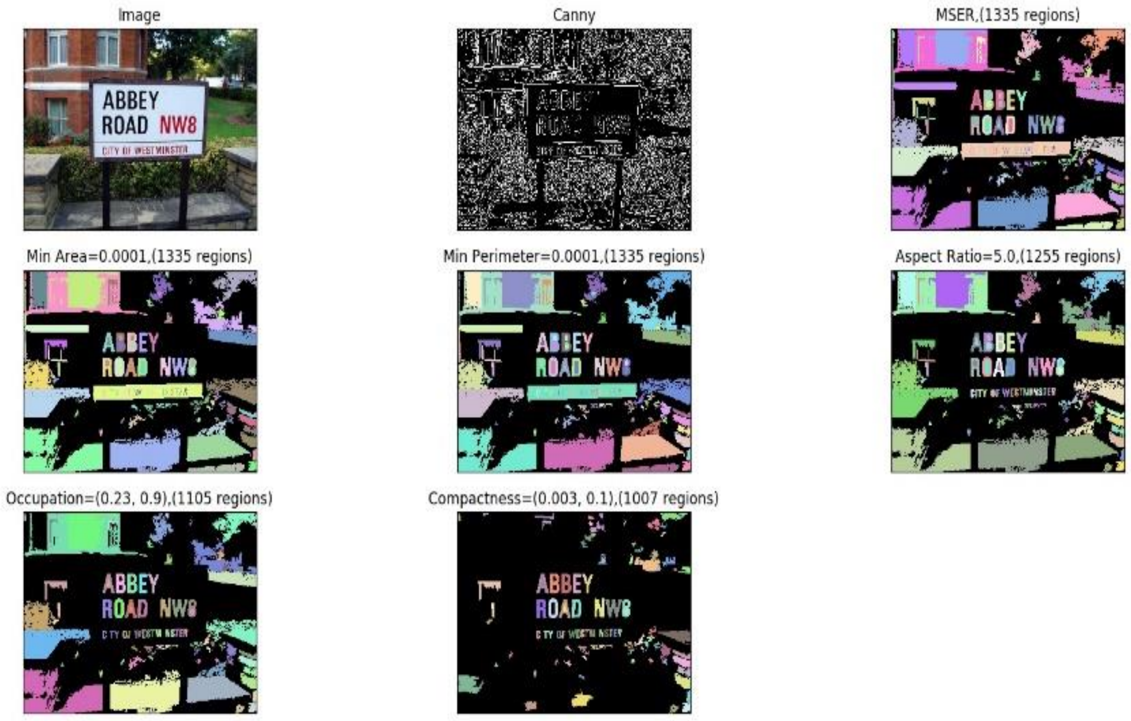


Image
Canny
MSER,(1335 regions)
Min Area=0.0001,(1335 regions)
Min Perimeter=0.0001,(1335 regions)
Aspect Ratio=5.0,(1255 regions)
Occupation=(0.23, 0.9),(1105 regions)
Compactness=(0.003, 0.1),(1007 regions)

Figure 7.4 First Stage of Pre-processing

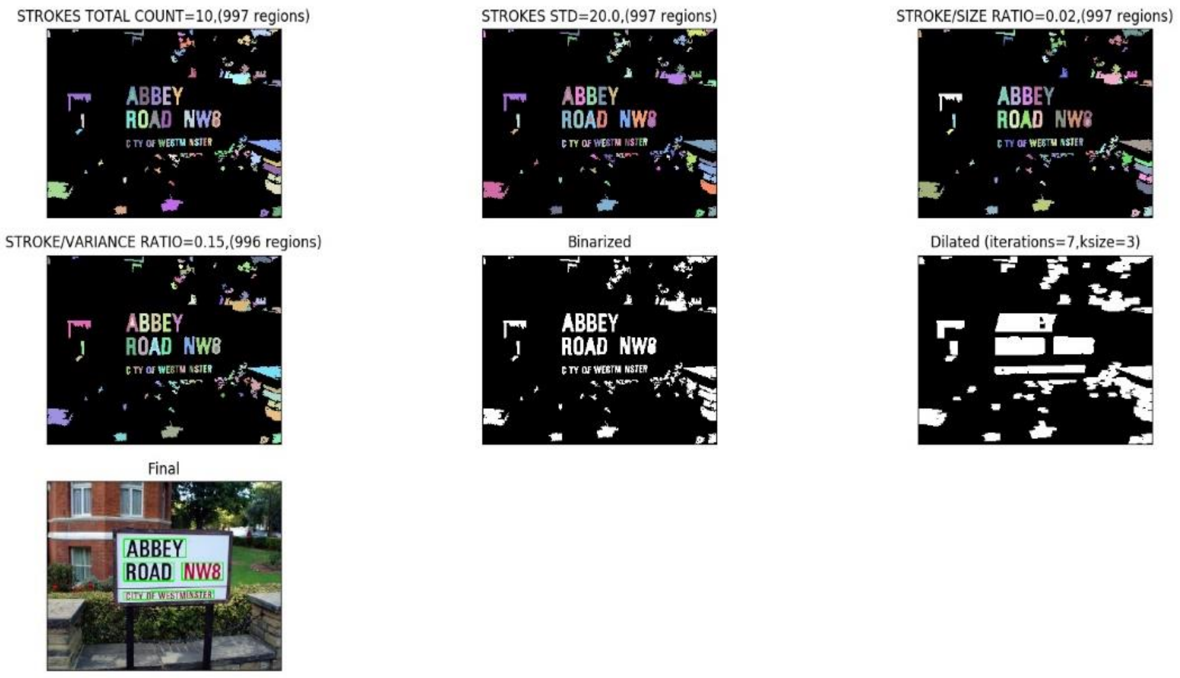
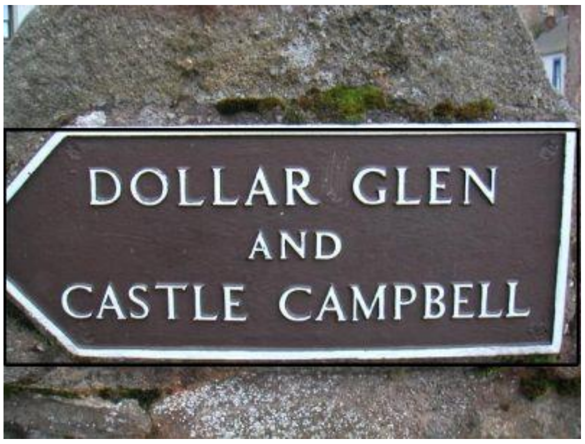

STROKES TOTAL COUNT=10,(997 regions)
STROKES STD=20.0,(997 regions)
STROKE/SIZE RATIO=0.02,(997 regions)
STROKE/VARIANCE RATIO=0.15,(996 regions)
Binarized
Dilated (iterations=7,ksize=3)
Final
ABBEY
ROAD NW8
CITY OF WESTMINSTER

Figure 7.5 Second Stage of Pre-processing

Case 2 –



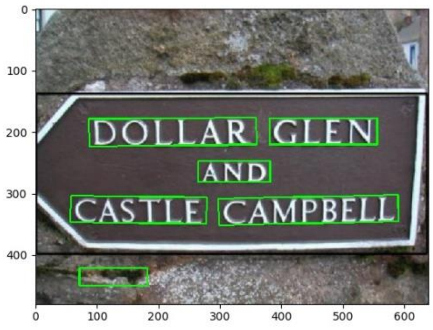
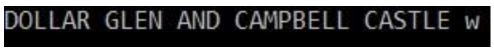


Figure 7.6 Original Image **Figure 7.7 Detected Regions**



```
DOLLAR GLEN AND CAMPBELL CASTLE w
```

Figure 7.8 Detected Text

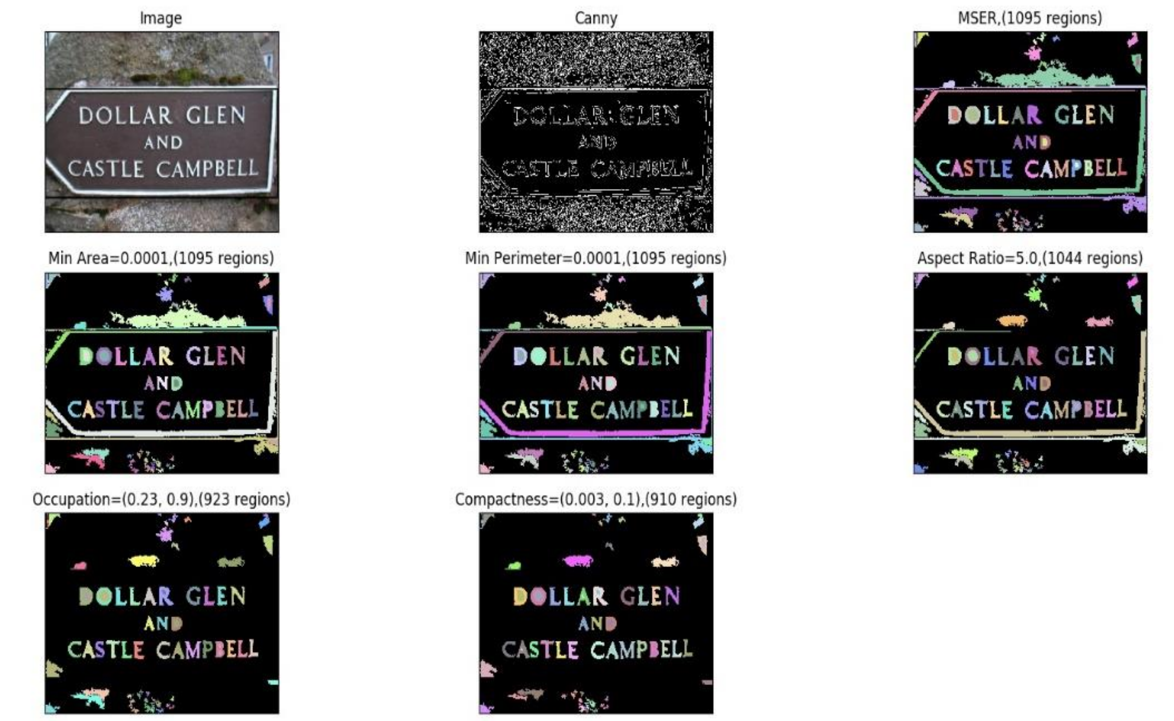


Image
DOLLAR GLEN AND CASTLE CAMPBELL
Canny
MSER,(1095 regions)
Min Area=0.0001,(1095 regions)
Min Perimeter=0.0001,(1095 regions)
Aspect Ratio=5.0,(1044 regions)
Occupation=(0.23, 0.9),(923 regions)
Compactness=(0.003, 0.1),(910 regions)

Figure 7.9 First Stage of Pre-processing

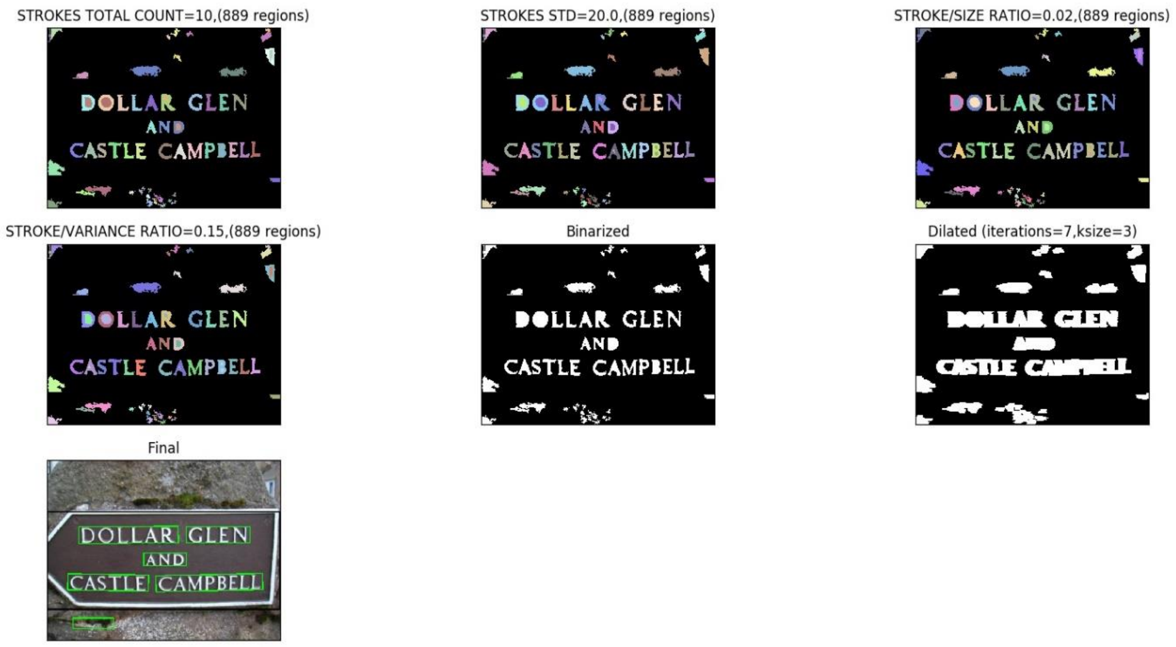
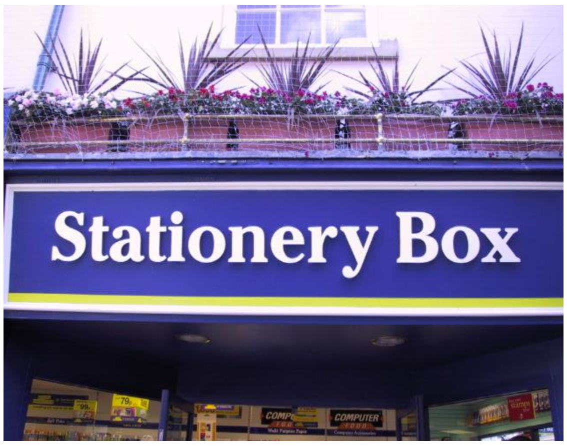

STROKES TOTAL COUNT=10,(889 regions)
STROKES STD=20.0,(889 regions)
STROKE/SIZE RATIO=0.02,(889 regions)
STROKE/VARIANCE RATIO=0.15,(889 regions)
Binarized
Dilated (iterations=7,ksize=3)
Final

Figure 7.10 Second Stage of Pre-processing

Case 3 –



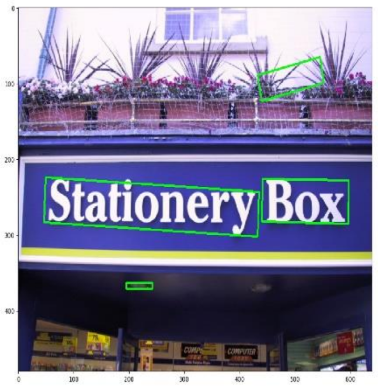
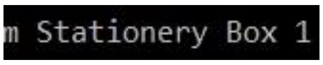


Figure 7.11 Original Image **Figure 7.12 Detected Regions**



```
m Stationery Box 1
```

Figure 7.13 Detected Text

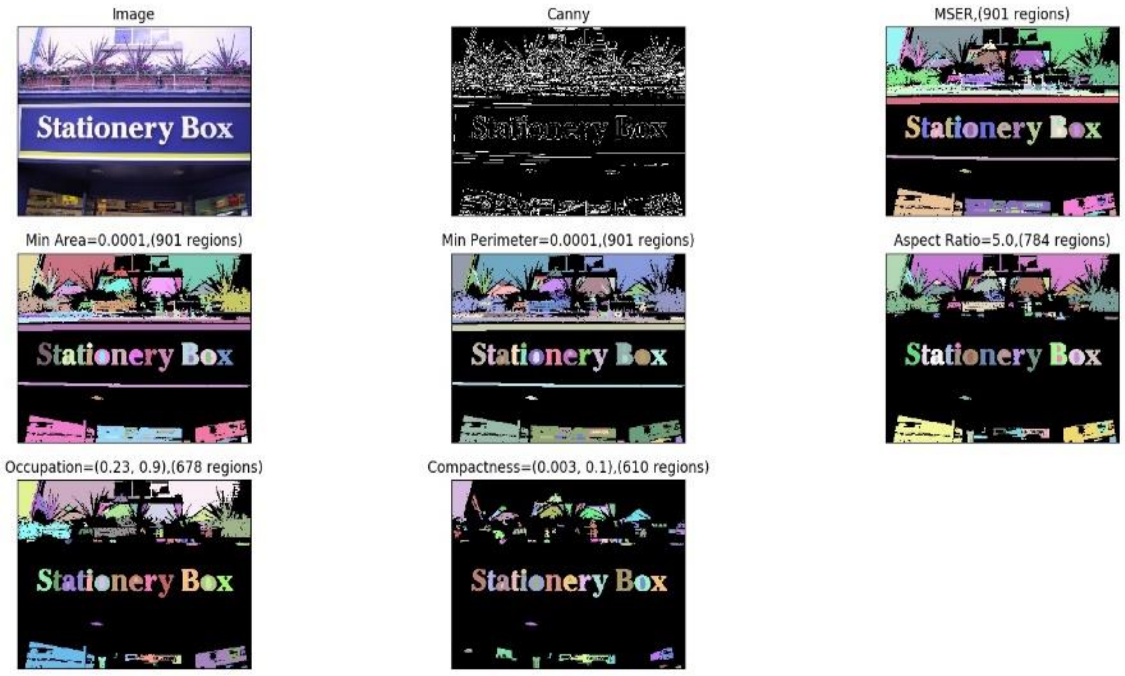


Figure 7.14 First Stage of Pre-processing

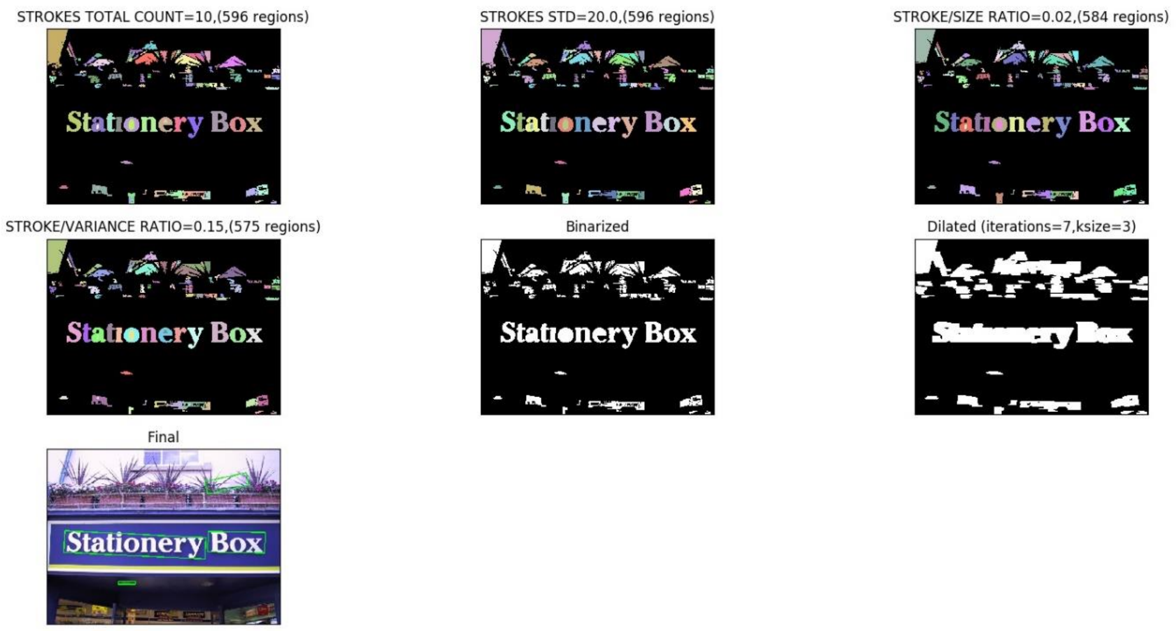

STROKES TOTAL COUNT=10,(596 regions)
STROKES STD=20.0,(596 regions)
STROKE/SIZE RATIO=0.02,(584 regions)
STROKE/VARIANCE RATIO=0.15,(575 regions)
Binarized
Dilated (iterations=7,ksize=3)
Final

Figure 7.15 Second Stage of Pre-processing

Case 4 –

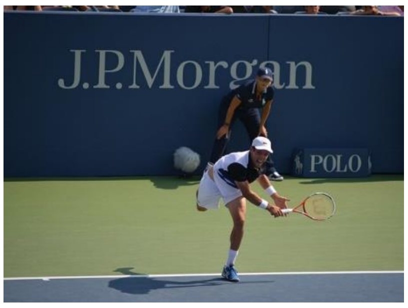


Figure 7.16 Original Image

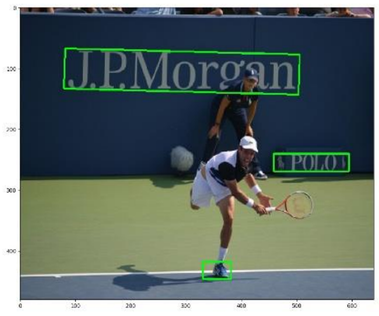
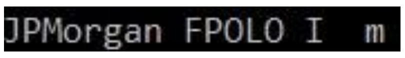


Figure 7.17 Detected Regions



```
JPMorgan FPOLO I  m
```

Figure 7.18 Detected Text

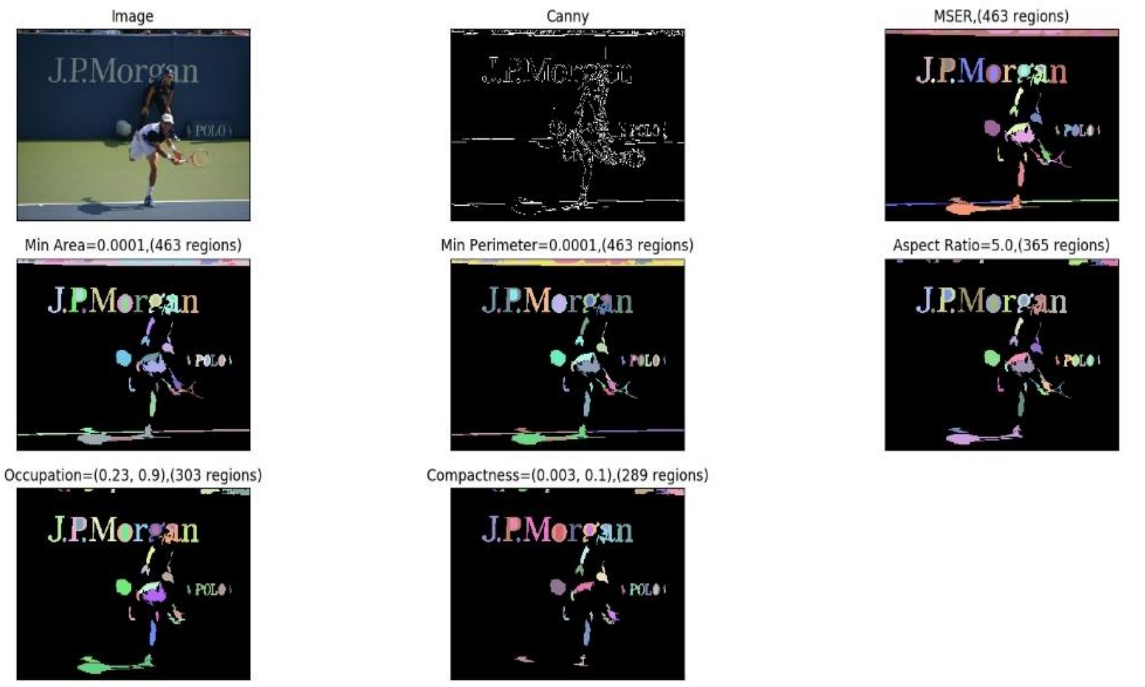


Image
Canny
MSER,(463 regions)
Min Area=0.0001,(463 regions)
Min Perimeter=0.0001,(463 regions)
Aspect Ratio=5.0,(365 regions)
Occupation=(0.23, 0.9),(303 regions)
Compactness=(0.003, 0.1),(289 regions)

Figure 7.19 First Stage of Pre-processing

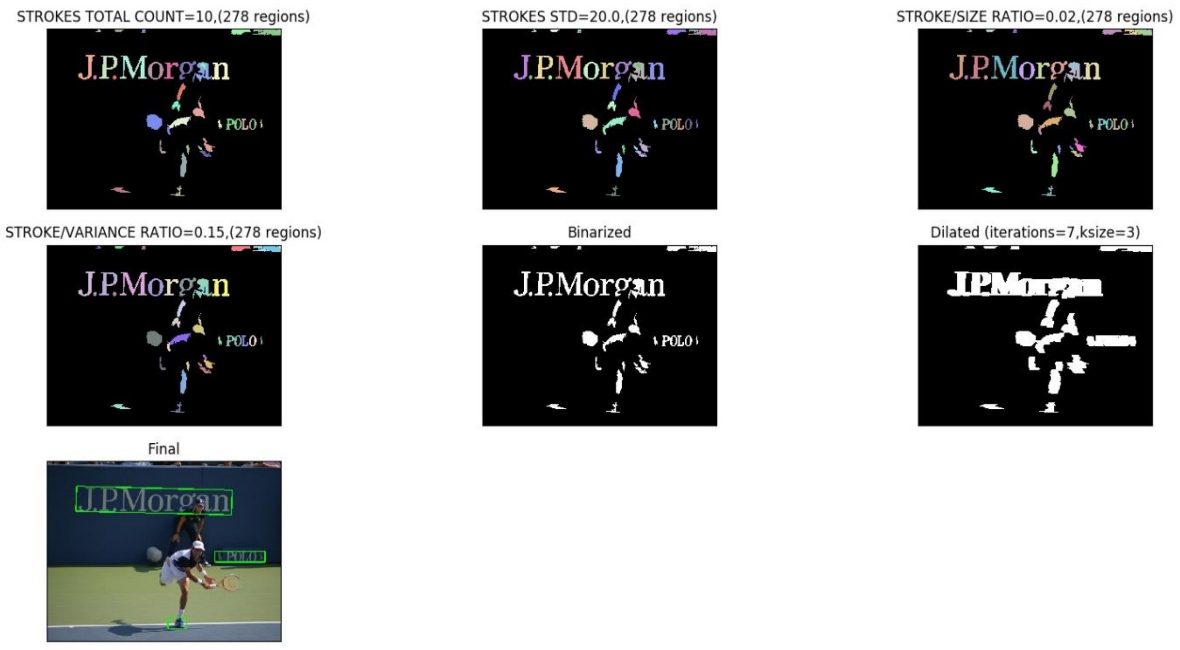

STROKES TOTAL COUNT=10,(278 regions)
STROKES STD=20.0,(278 regions)
STROKE/SIZE RATIO=0.02,(278 regions)
STROKE/VARIANCE RATIO=0.15,(278 regions)
Binarized
Dilated (iterations=7,ksize=3)
Final

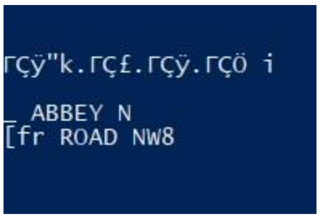
Figure 7.20 Second Stage of Pre-processing

7.1 Accuracy Tabulation and Comparison.

To determine the increase in accuracy of the proposed system we do a comparison with the Tesseract OCR Library.

The Tesseract output of all the cases mentioned above is given below –

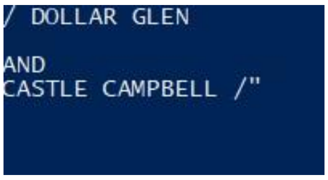
Case 1 –



```
rÇÿ"k.rÇ£.rÇÿ.rÇö i

_ ABBEY N
[fr ROAD NW8
```

Case 2 –

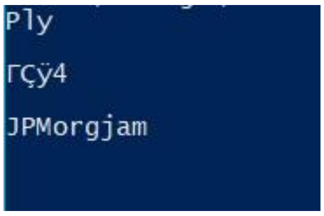


```
/ DOLLAR GLEN

AND
CASTLE CAMPBELL /"
```

Case 3 –

Case 4 –



```
Ply

rÇÿ4

JPMorgjam
```

Image	n_c	n_e	n_{rej}	G_{rec}
1	29	22	16	31.03%
2	28	3	0	89.28%
3	10	0	10	00.00%
4	11	8	4	09.09%

Table 7.1 Accuracy with Tesseract

Image	n_c	n_e	n_{rej}	G_{rec}
1	29	0	0	100.00%
2	28	1	0	96.43%
3	10	2	0	80.00%
4	11	3	0	72.72%

Table 7.2 Accuracy with the Proposed System

CHAPTER 8

CONCLUSION

An overview of the design and implementation of a real-time text recognition in natural scene images has been presented in this thesis. Scene text extraction is recent research area in the field of Computer Vision. It is challenging problem in the information processing field mainly due to, different variety of text patterns like fonts, colours, sizes, orientations; and presence of background outliers similar to text characters, such as windows, bricks. In this paper we have improved the overall accuracy of the system at the expense of execution time of recognition of text in the image given as the input. But, we consider the proposed system to be better since the accuracy of the system far exceeds that of the existing ones. First, text is extracted and later the text in the given image is recognized and used further. Pre-processing, utilization of SWT and SVM increases the execution time of the system. The process of Extraction is performed using stroke width transform (SWT) and connected component-based approach. The SWT operator is applied for each edge component instead of the entire image pixel. The local SWT also can be considered to split component which contains more than one text components. This may allow the detection image which contains touching characters.

After the text is extracted using stroke width transform and connected component we use support vector machine (SVM). Due to the complicated background and unpredictable text appearances scene text detection is still a challenging problem. We have presented in this paper an improved scene text detection method that makes use of machine learning classifiers. One for identifying the text component and other classification of text and non-text components in an image. We use Machine Learning Algorithms to refine the Images and extract information from them so that we can improve the accuracy of text detection and extraction systems. Support Vector Machine uses machine learning to remove the noise in the image. Machine learning helps and also reduces the time taken by the machine to remove the non-text elements in the given picture.

After the text is extracted the text has to be recognized for which we use Optical Character Recognition (OCR).Optical character recognition (OCR) for converting the detected text from an image to actual text. Optical character recognition (OCR) is performed by using

“Tesseract”, an open-source OCR module developed by Google. Tesseract will consider this as text and try to provide a result based on character that has the highest “confidence” value. After the text is recognized using Optical Character Recognition the obtained text is google searched to provide information on it.

8.1 Future Work

We have implemented Recognition of text in a scenic image for text only in English as of now. As for our future work, we are looking forward to implement a translation feature as well in the application. We are also looking forward to implementing it in many languages as well. So that it can extracted, recognized and given information on in the local language it is being used in We can also implement Fast Stroke Width Transform which increases the complexity of the system but also increases the speed at which the text is detected thereby increasing the speed of the system. We can also implement this process for texts in an image without proper orientation. So we can orient the text into a format which can be extracted by our machine. We are looking forward to reducing the time taken in the extraction of the text. Our main aim is integrate the whole system into one single android application.

REFERENCES

[1] Pooja Chavre and Dr. Archana Ghotkar, “Scene Text Extraction using Stroke Width Transform for Tourist Translator on Android Platform”, 2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), 2016.

[2] Nobuo Ezaki, Marius Bulacu, Lambert Schomaker, “Text Detection From Natural Scene Images - Towards a System for Visually Impaired Persons”, Proceedings of the 17th International Conference on Pattern Recognition (ICPR’04), September 2004.

[3] Wahyono, Munho Jeong, Kang-Hyun Jo, “Multi Language text Detection Using Fast Stroke Width Transform”, 2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV), May 2015.

[4] Arpit Jain, Xujun Peng, Xiaodan Zhuang, Pradeep Natarajan, Huaigu Cao, “Text Detection and Recognition in Natural Scenes and Consumer Videos”, 2014 IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP), July 2014.

[5] A. Ikica, and P. Peer, “SWT voting-based color reduction for text detection in natural scene image”, EURASIP Journal on Advances in Signal Processing, May 2013.

[6] K. I. Kim, K. Jung, and J. H. Kim, “Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm”, IEEE Trans. PAMI, 2003.

[7] B. Epshtein, E. Ofek, and Y. Wexler, “Detecting text in natural scenes with stroke width transform”, Proceeding of CVPR, 2010.

[8] C. Yao, X. Bai, W. liu, Y. Ma, and Z. Tu, “Detecting texts of arbitrary orientations in natural images”, Proceeding of CVPR, 2012.

[9] C. Yi and Y. Tian, “Text detection in natural scenes image by stroke gabor words”, Proceedings of ICDAR, 2011.

[10] W. Huang, Z. Lin, J. Yang, and J. Wang, “Text localization in natural images using stroke feature transform and text covariance descriptor”, Proceedings of ICCV, pp.1241-1248, 2013.

[11] B. Bai, F. Yin, and C.-L. Liu,” A fast stroke-based method for text detection in video, Proceedings of IAPR Document Analysis Systems”, pp.69-73, 2012.

[12] C. Yi and Y. Tian, “Localizing text in scene images by boundary clustering, stroke segmentation, and string fragment classification”, IEEE Trans. on Image Processing, Vol.21 No.9, pp 4256-4268, September2012.

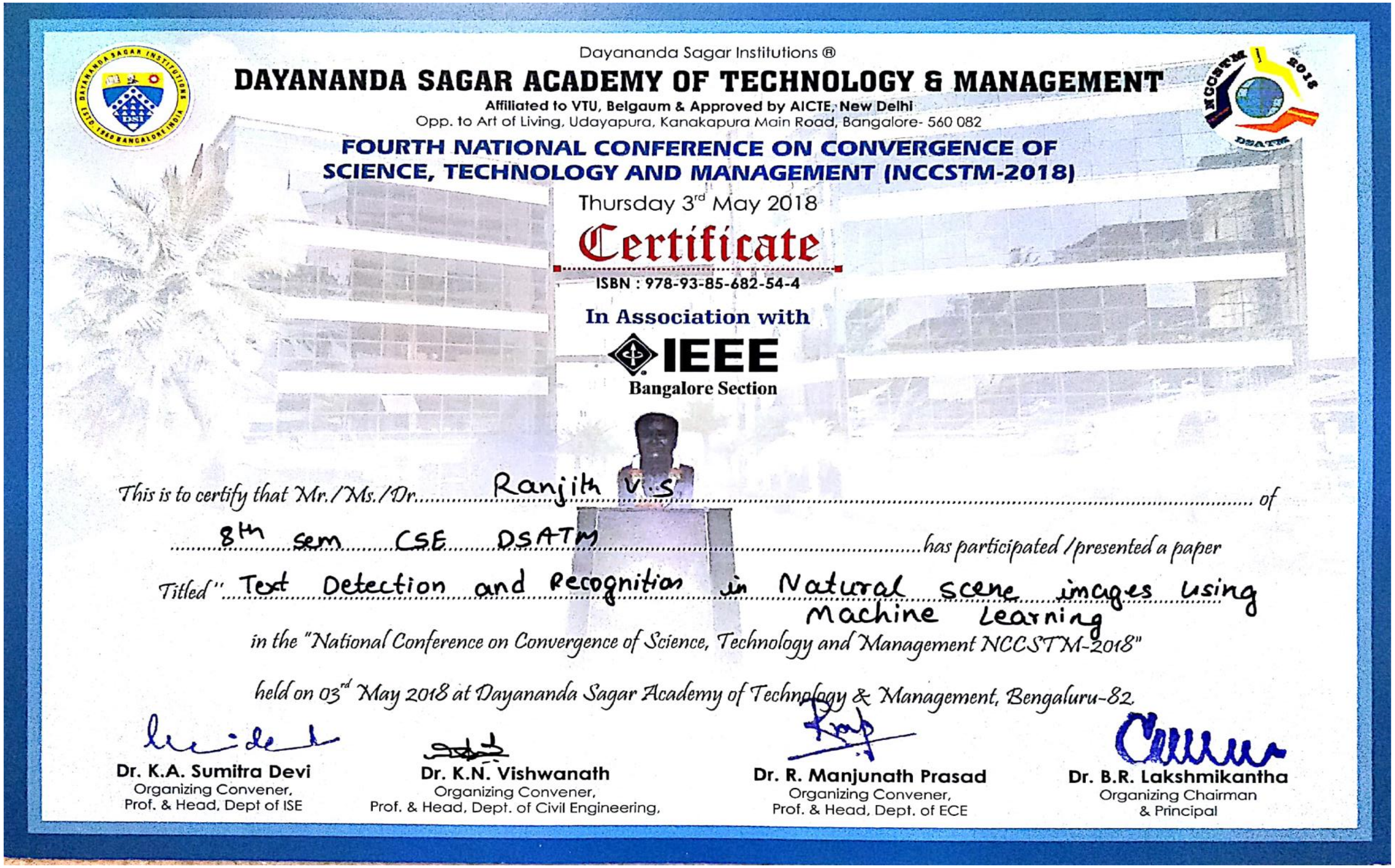
[13] A. Mosleh , N. Bouguila, and A.B. Hamza, “Image text detection using a band let-based edge detector and stroke width transform”, Proceedings of BMCV, 2012.

[14] http://cocodataset.org

[15] Ventzislav Alexandrov, “Error Evaluation and Applicability of OCR Systems”, International Conference on Computer Systems and Technologies – CompSysTech 2003.

APPENDIX A – CERTIFICATE

[1] "Text Detection and Recognition in Natural Scene Images Using Machine Learning" in the Fourth National Conference on Convergence of Science, Technology and Management (NCCSTM-2018) at Dayananda Sagar Academy of Technology and Management.



Dayananda Sagar Institutions ®

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & MANAGEMENT

Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi

Opp. to Art of Living, Udayapura, Kanakapura Main Road, Bangalore- 560 082

FOURTH NATIONAL CONFERENCE ON CONVERGENCE OF SCIENCE, TECHNOLOGY AND MANAGEMENT (NCCSTM-2018)

Thursday 3rd May 2018

Certificate

ISBN : 978-93-85-682-54-4

In Association with

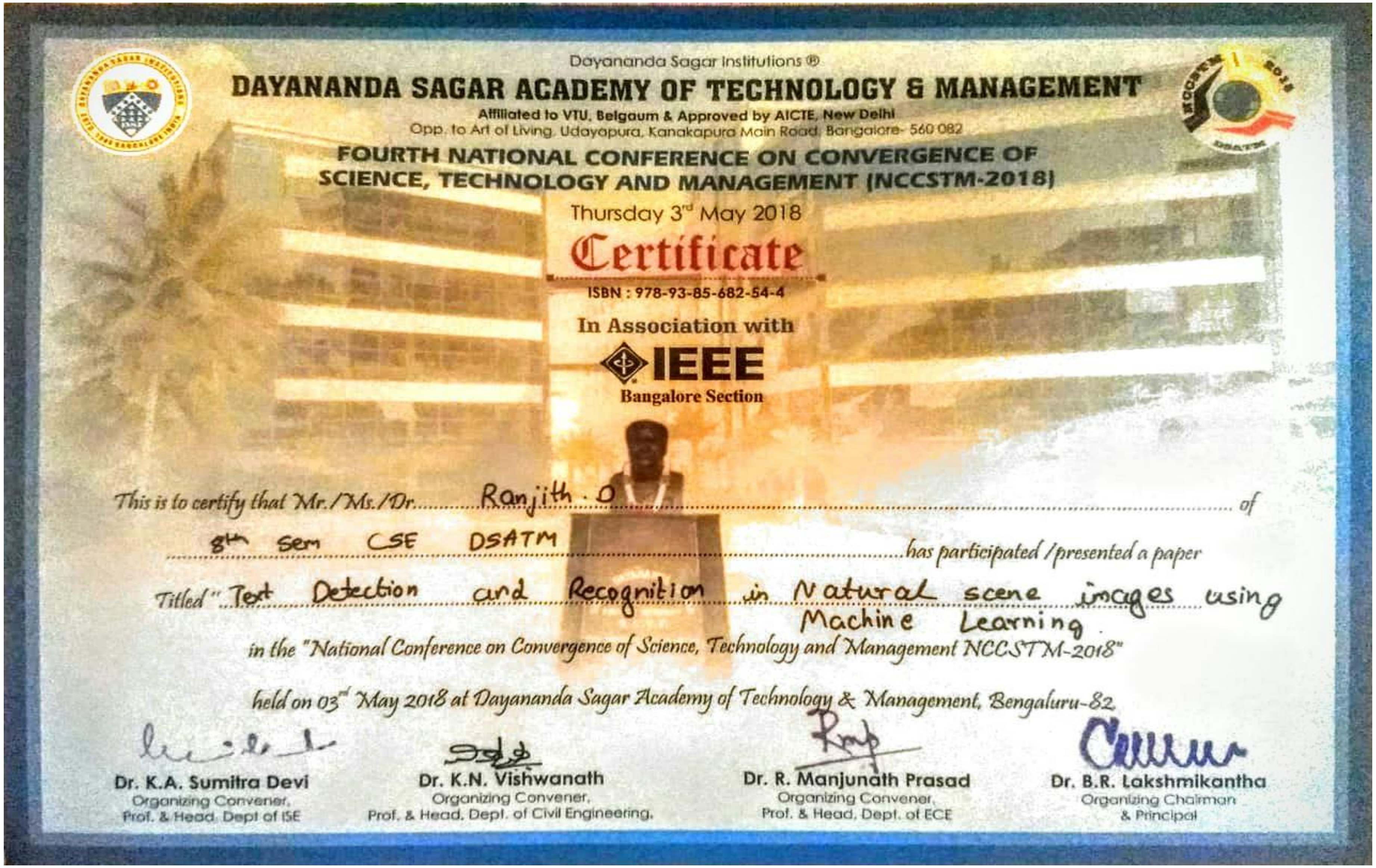
IEEE

Bangalore Section

This is to certify that Mr./Ms./Dr. Ranjith V.S *of* 8th sem CSE DSATM *has participated / presented a paper Titled* " Text Detection and Recognition in Natural scene images using Machine Learning *in the "National Conference on Convergence of Science, Technology and Management NCCSTM-2018"*

held on 03rd May 2018 at Dayananda Sagar Academy of Technology & Management, Bengaluru-82.

[illegible]	[illegible]	[illegible]	[illegible]
Dr. K.A. Sumitra Devi	**Dr. K.N. Vishwanath**	**Dr. R. Manjunath Prasad**	**Dr. B.R. Lakshmikantha**
Organizing Convener,	Organizing Convener,	Organizing Convener,	Organizing Chairman
Prof. & Head, Dept of ISE	Prof. & Head, Dept. of Civil Engineering,	Prof. & Head, Dept. of ECE	& Principal



Dayananda Sagar Institutions ®

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & MANAGEMENT

Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi

Opp. to Art of Living, Udayapura, Kanakapura Main Road, Bangalore- 560 082

FOURTH NATIONAL CONFERENCE ON CONVERGENCE OF SCIENCE, TECHNOLOGY AND MANAGEMENT (NCCSTM-2018)

Thursday 3rd May 2018

Certificate

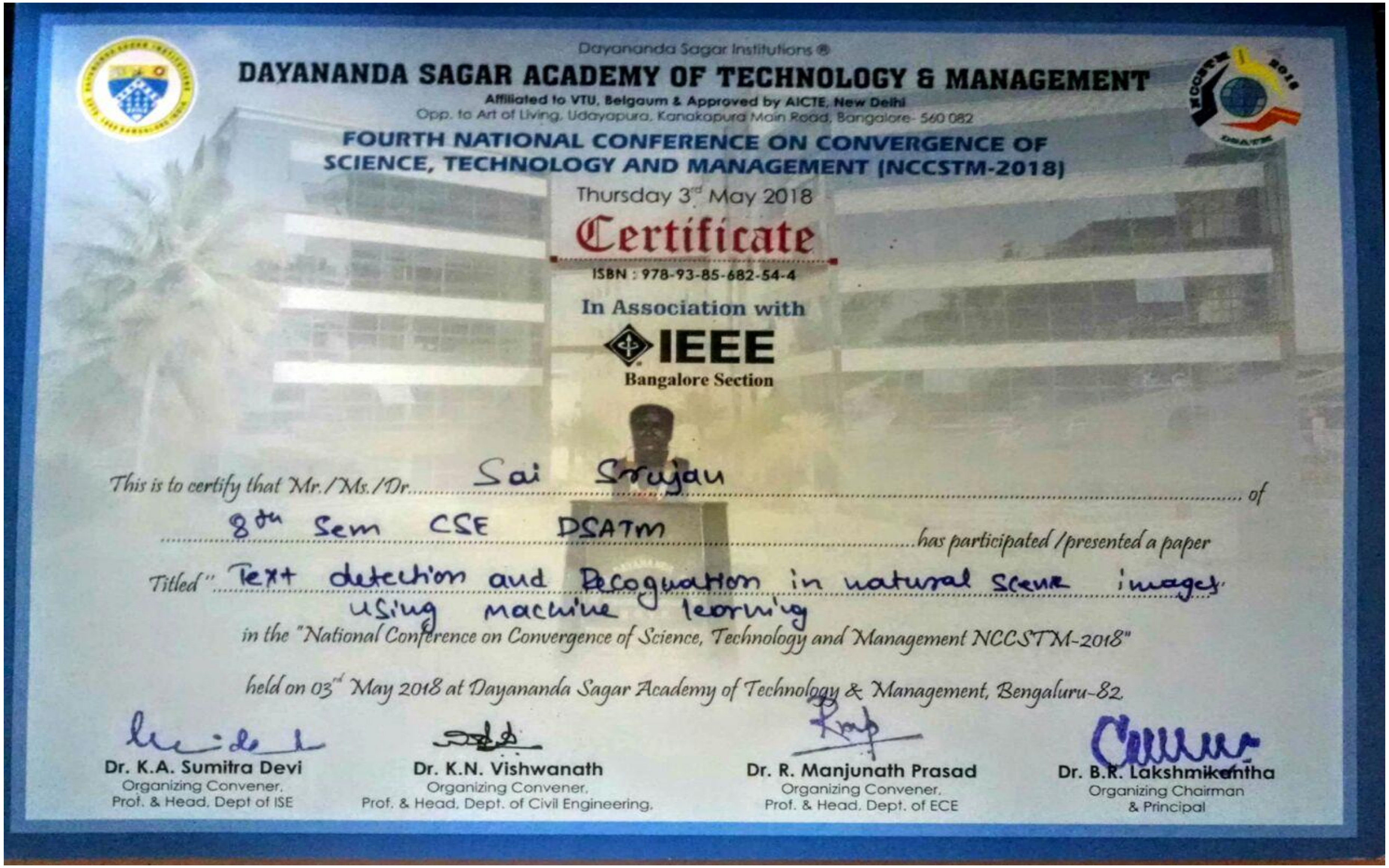
ISBN : 978-93-85-682-54-4

In Association with

IEEE Bangalore Section

This is to certify that Mr./Ms./Dr. Ranjith . O *of* 8th sem CSE DSATM *has participated /presented a paper Titled* "Text Detection and Recognition in Natural scene images using Machine Learning*" in the "National Conference on Convergence of Science, Technology and Management NCCSTM-2018" held on 03rd May 2018 at Dayananda Sagar Academy of Technology & Management, Bengaluru-82*

[illegible]	[illegible]	[illegible]	[illegible]
Dr. K.A. Sumitra Devi	Dr. K.N. Vishwanath	Dr. R. Manjunath Prasad	Dr. B.R. Lakshmikantha
Organizing Convener,	Organizing Convener,	Organizing Convener,	Organizing Chairman
Prof. & Head, Dept of ISE	Prof. & Head, Dept. of Civil Engineering.	Prof. & Head, Dept. of ECE	& Principal



Dayananda Sagar Institutions ®

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & MANAGEMENT

Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi

Opp. to Art of Living, Udayapura, Kanakapura Main Road, Bangalore- 560 082

FOURTH NATIONAL CONFERENCE ON CONVERGENCE OF SCIENCE, TECHNOLOGY AND MANAGEMENT (NCCSTM-2018)

Thursday 3rd May 2018

Certificate

ISBN : 978-93-85-682-54-4

In Association with

IEEE Bangalore Section

This is to certify that Mr./Ms./Dr. Sai Srujan *of*

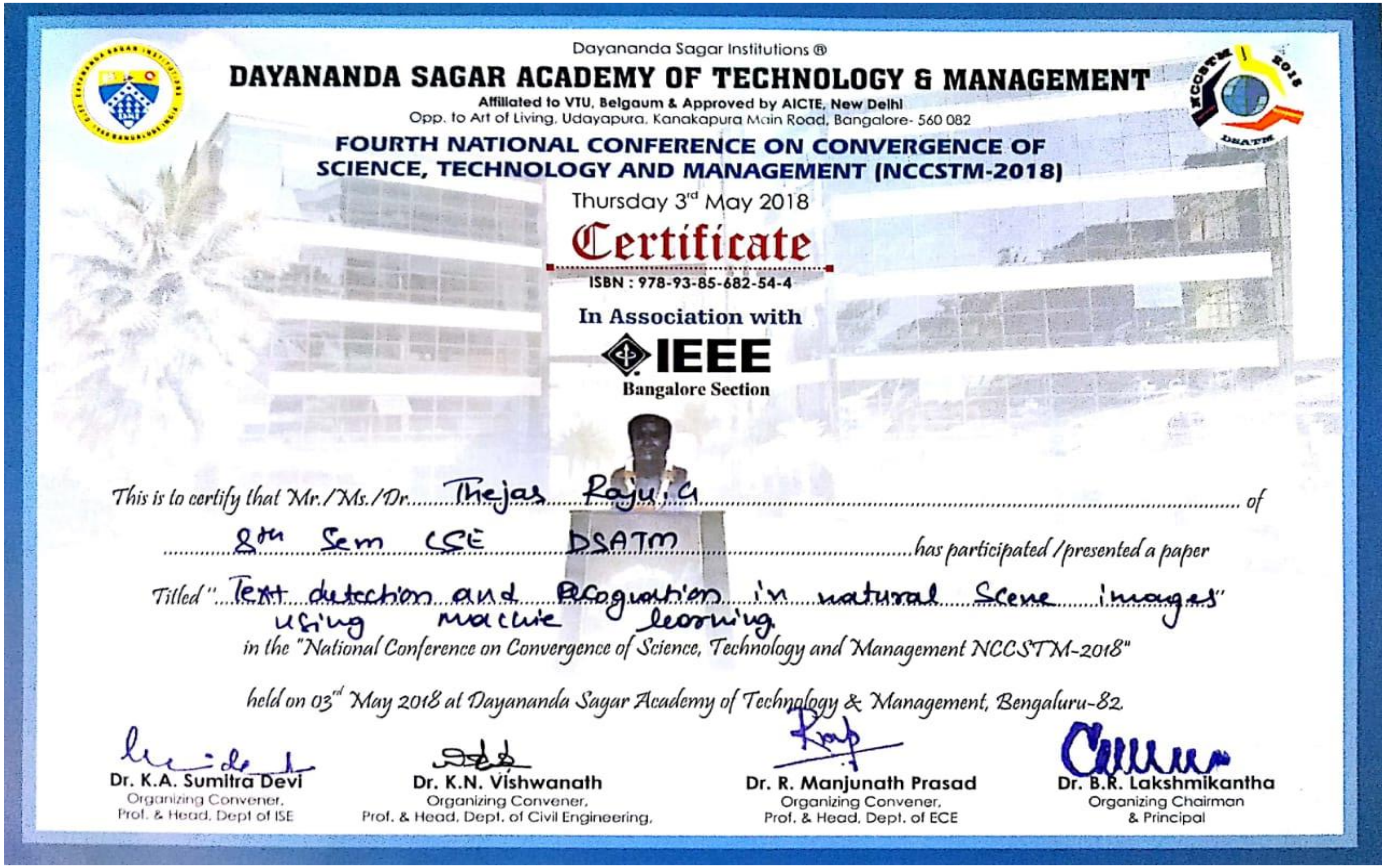
8th Sem CSE DSATM *has participated / presented a paper*

Titled" Text detection and Recognation in natural scene images using machine learning

in the "National Conference on Convergence of Science, Technology and Management NCCSTM-2018"

held on 03rd May 2018 at Dayananda Sagar Academy of Technology & Management, Bengaluru-82.

[illegible]	[illegible]	[illegible]	[illegible]
Dr. K.A. Sumitra Devi	**Dr. K.N. Vishwanath**	**Dr. R. Manjunath Prasad**	**Dr. B.R. Lakshmikantha**
Organizing Convener, Prof. & Head, Dept of ISE	Organizing Convener, Prof. & Head, Dept. of Civil Engineering.	Organizing Convener, Prof. & Head, Dept. of ECE	Organizing Chairman & Principal



Dayananda Sagar Institutions ®

DAYANANDA SAGAR ACADEMY OF TECHNOLOGY & MANAGEMENT

Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi
Opp. to Art of Living, Udayapura, Kanakapura Main Road, Bangalore- 560 082

FOURTH NATIONAL CONFERENCE ON CONVERGENCE OF SCIENCE, TECHNOLOGY AND MANAGEMENT (NCCSTM-2018)

Thursday 3rd May 2018

Certificate

ISBN : 978-93-85-682-54-4

In Association with

IEEE
Bangalore Section

This is to certify that Mr./Ms./Dr. Thejas Raju G *of*

8th Sem CSE DSATM *has participated /presented a paper*

Titled "Text detection and Recognition in natural Scene images using machine learning"

in the "National Conference on Convergence of Science, Technology and Management NCCSTM-2018"

held on 03rd May 2018 at Dayananda Sagar Academy of Technology & Management, Bengaluru-82.

[illegible]	[illegible]	[illegible]	[illegible]
Dr. K.A. Sumitra Devi	**Dr. K.N. Vishwanath**	**Dr. R. Manjunath Prasad**	**Dr. B.R. Lakshmikantha**
Organizing Convener, Prof. & Head, Dept of ISE	Organizing Convener, Prof. & Head, Dept. of Civil Engineering,	Organizing Convener, Prof. & Head, Dept. of ECE	Organizing Chairman & Principal