Sign Board Detection and Information extraction for MAVI (Mobility Assistant for Visually Impaired) project

ANAND AGRAWAL

A thesis submitted in partial fulfillment for the
degree of Master of Technology

in

Computer Science and Engineering
IIT DELHI

Supervisor: Prof. M.BALAKRISHNAN

JUNE 2017

Certification of Authorship

This is to declare that this thesis titled, **Sign Board Detection and Information Extraction for MAVI** being submitted by **Anand Agrawal** for award of **Master of Technology in Computer Science and Engineering** is an authentic work carried out by him under my guidance and supervision at the **Department of Computer Science and Engineering**.

Prof. M. Balakrishnan
Department of Computer Science and Engineering
Indian Institute of Technology Delhi

Dr. Chetan Arora
Department of Computer Science and Engineering
Indraprastha Institute of Information Technology Delhi

Acknowledgement

I would like to articulate my deep gratitude to my supervisor Prof. M. Balakrishnan who has always been my motivation for carrying out the thesis. I am highly indebted to him for believing in me and for his constant supervision and valuable guidance during the course of thesis

I would also like to thank co-supervisor Dr. Chetan Arora for his valuable insite and assistance regarding subject of Computer Vision.

I also extent my thanks to Mr. Rajesh Kedia, Mr. Anupam Sobti and Mr. Saurabh Agrawal for having useful discussion and providing ideas, Mr. Sharma for providing me with all the lab equipment and support.

Anand Agrawal

Abstract

Abstract

Sign Board detection and Information Extraction is one of the important task for guidance and safety of visually challenged people. Warning beforehand can be very helpful for the safety of visually impaired people. This thesis address Sign Board detection and information extraction from sign boards using image recognition Tesseract.

All the dataset are taken from IIT Delhi campus and nearby areas. The frames are taken by Pivot Head camera. All experiments and implementation are done using both c++, tesseract and OpenCV.

In recent times, there have been an increase in Optical Character Recognition (OCR) solutions for recognizing the text from scanned document images and scene-texts taken with the mobile devices. Many of these solutions works very good for individual script or language. But in multi-lingual environment such as in India, where a document image or scene-images may contain more than one language, tesseract has worked quite well. Hence, in order to recognize texts in the multilingual document image or scene-image, we have used tesseract and made modifications such as noise removal in image, processing skewed image with signboard at some angle and then feeding to the tesseract for better output accuracy.

All the dataset are taken from IIT Delhi campus and nearby areas. The frames are taken by PivotHead camera. In case of tesseract the output of signboard detection is processed and given as input to tesseract on multi-lingual OCR detection after processing. All experiments and implementation are done using both C++ and OpenCV

Contents

Chapter 1

Introduction

1.1 Introduction to MAVI SYSTEM

The two important problems faced by the visually impaired include Obstacle Detection and Independent Navigation. With the advent of technology, cost effective solutions like Smart Cane, have been developed to solve the obstacle detection problem. However independent Navigation especially in the Outdoor environment , is still an unresolved issue for the visually impaired. In general, Safety issues, Navigation problems in an unknown area and social inclusion are the major issues of concern . MAVI system attempts to solve these issues by integrating various modules which work to solve problems posed by outdoor navigation independently. MAVI stands for Mobility assistant for visually impaired . The overview of the MAVI system is shown in the below figure 1.1.

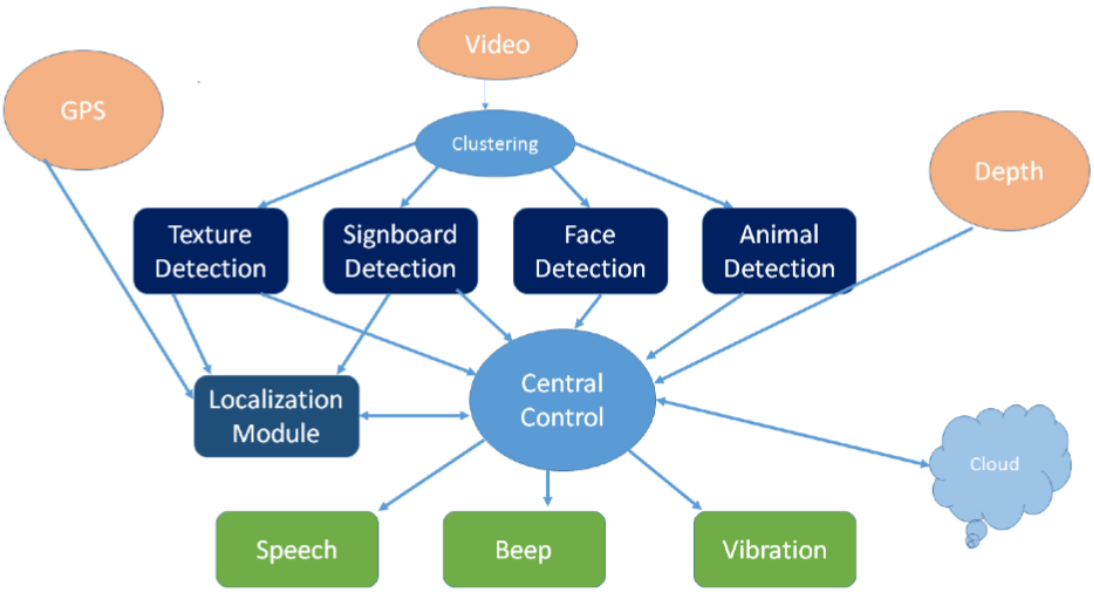


Fig.1.1 Overview of MAVI system

The MAVI system constitutes of a central controller that binds the four modules of MAVI system together. These modules are Texture detection,Animal detection, signboard Detection ,Face detection and recognition. The controller also has the information from the GPS+IMU localisation module and depth sensor. It also interacts with the server and MAVI app to provide information to the user.

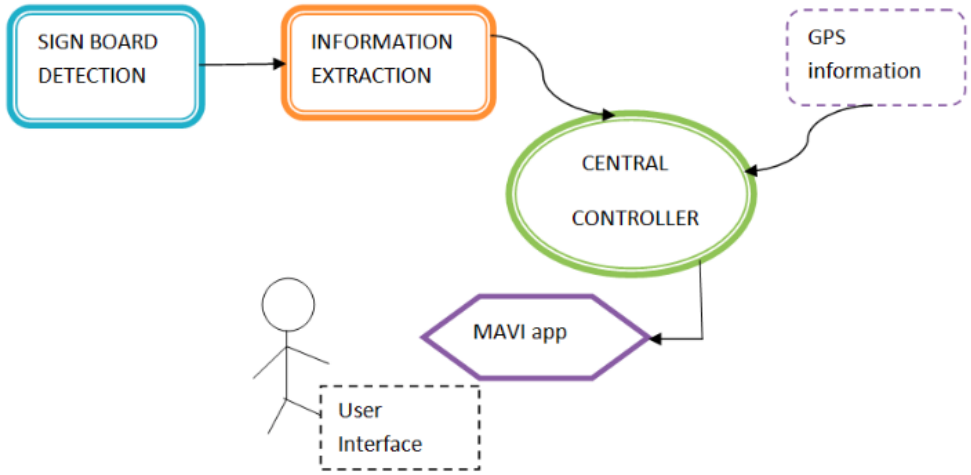


Fig.1.2 Overview of Sign Board Detection Module

1.2 Motivation and Objective

Eyes are one of the most important sense organ human body have to interact with their surrounding environment. Because of visual impairment, the ability for interacting with surrounding is very difficult and limited. i.e it become very difficult or almost impossible to walk, find places and social inclusion for visually challenged people. So, visual impairment is one of the severe type of disabilities a person must endure. In India mostly visually impaired people have to rely on traditional cane. The cane is very limited in terms of providing interaction and independence to its users. Now days Computer Vision is one of the biggest and most active research area and it can be very helpful for the visually impaired people as their eyes. So we team MAVI decided to design prototype that consist of Texture Detection , Pothole Detection, Face Detection, Signboard Detection and location information. Objective of this thesis is to develop signboard detection and information and extraction which aids the visually impaired in navigation.

1.3 Overview of signboards detection and info extraction

Signboards serve as navigation aids in an unknown location. The information present on them becomes extremely important for a person who is not the native of that place. For visually impaired people, this information remains unavailable unless a module is employed exclusively to provide the information. Hence signboard detection and Information extraction module focuses to assist the user in navigating seamlessly in an outdoor environment. The overall implementation of the Sign Board detection module is as shown above.

1.4 Thesis Contribution

This thesis address problem of signboard detection and information extraction using tesseract OCR and adding improvements to tesseract. This module is one of the four sensors of the MAVI. I've continued Dedeepya's work (Last year Student). She had worked on detection algorithm and ocr for image character recognition in matlab. The accuracy for the latter was too poor so tried to enhance the same but failed, so switched to Tesseract. Firstly i started with the conversion of code base from matlab to C++ as per Xilings requirement of C/C++. Later tried improvements in already implemented character recognition algorithm but realised its too low yet. So shifted to Tesseract OCR and tried compiling the code on ZedBoard. Finally

added improvements to Tesseract for skewed images. Time and distance analysis were done for Tesseract and signboard detection.

1.5 Thesis Outline

The body of the thesis is organised into 7 chapters out of which the first chapter is the introduction .The remainder of the thesis is organised as follows: The chapter 2 includes the introduction to OpenCV and briefly outlines the Detection Algorithms that are viable for the detection of sign boards. Thechapter 3 discusses the algorithm employed for signboard detection in MAVI system and its associated results, followed by chapter 4 which discusses the information extraction with the help of OCR Engine. The chapter5 describes software only implementation on Zed board and the performance estimation results for this implementation .Chapter6 is about the energy estimation for signboard detection module in various modes. Chapter 7 discusses the profiling results.Chapter8 is about the conclusion and future work.

Chapter 2

Background

2.1 Introduction to opencv

OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real time computer vision, developed by Intel .It is free for use under the open source BSD license. The library is cross-platform. It focuses mainly on real-time image processing. The library was originally written in C and this C interface makes OpenCV portable to some specific platforms such as digital signal processors. Wrappers for languages such as C hash, Python, Ruby and Java (using JavaCV) have also been developed. However, since version 2.0, OpenCV includes both its traditional C interface as well as a new C++ interface. This new interface seeks to reduce the number of lines of code necessary to code up computer vision functionality as well as reduce common programming errors such as memory leaks (through automatic data allocation and de-allocation) that can arise when using OpenCV in C. Most of the new developments and algorithms in OpenCV are now developed in the C++ interface .Unfortunately, it is much more difficult to provide wrappers in other languages to C++ code as opposed to C code; therefore the other language wrappers are generally lacking some of the newer OpenCV 2.0 features. A CUDA-based GPU interface has been in progress since September 2010.

2.2 Object detection algorithms

2.2.1 Template Matching:

Template matching is a technique in which a portion of the image or sub-image ,called template is searched in a given input image containing the sub-image to find its occurrence. Algorithm:

- It takes an source image and template image as input
- The identification of the matching region the template is compared with the source image by sliding it.
- The identification of the matching region the template is compared with the source image by sliding it.
- Sliding, refers to moving one pixel at a time from left to right, up to down.At each location the measurement of match is done by calculating a metric.
- For each location of T over I, the metric is calculated and stored in the result matrix (R).Here the match metric TMCCORRNORMED is used. method=CVTMCCORRNORMED .

2.2.2 Colour based detection using opencv

Detecting objects based on their colour is one of the prominent way of object detection. In this case the the colour of the object should significantly different from the colour of the background. Algorithm : 1.Creat an image window. 2.Creat track bars in the window for Hue,Saturation,Value. 3.Read the image in BGR/RGB

format. 4.Convert the image from RGB to HSV format. 5.Threshold the Image 6.Doing morphological opening/closing(optional) 7.Display the thresholded and original image.

2.2.3 Contour based object detection

OpenCV can be used for obtaining the vertices as a series of points of an object. So, a polygon can be determined by by getting the number of vertices present in it. For example a rectangle can be classified under an object with 4 vertices present in it. Also several features of the polygon such as convexity, flatness, concavity by comparing the position and distance between the vertices in an object Algorithm: 1.Create an image window. 2.Read the image. 3.Convert the image(if colour image) from RGB to HSV format. 4.Threshold the Image to get a binary image i.e. black(0) and white(1). 5.Doing morphological opening/closing(optional) 6.Search for contours in the thresholded image using OpenCV functions.

Chapter 3

Deep Neural Network on ZedBoard

3.1 Introduction

Nural Networks are one of the latest and growing technology in field of Machine learning and Artifical intelegence. Nural Netowrks are computing systems inspired by working of biological network in the brain. Neural network are found most useful in problems which are difficult to solve using traditional computiotinal algorithms. Many classification problems in area of vision and image processing can be solved using nural network for example image recognition. A classification problem involves a training and testing data set with class labels (In case of supervised learning) or without labels. And given a trainig data set we compute a function from trainig set to class. Thus nural network can be very useful in Face Detection, Texture and Pothole detection, Signboard detection and OCR and Animal detection modules of MAVI system. But neural network require to perform large amount of computation So, we wanted to test its performance on ZedBoard in order to find out if it can be any useful for us. We are more interested in memory consuption of neural network as ZedBoard has very limited memory available.

3.2 Cross Compilation of Caffe

Implementing Neural Network from scratch can be very time con- suming task So, I first explored already available tools for neural network implementaiton. Caffe is one of the popular open source deep learning framework Developed by Berkeley AI Research. It supports many different types of architecures geard towards image classification and image segmentation. In order to implement neural network in caffe we require defination of network in prototex format. So, I decided to use caffe for feasibility testing of neural network on ZedBoard. ZedBoard(Zynq Evaluation and Development Board) is an excellent developement kit based on Zynq All Programmable SoC (AP SoC). ZedBoard is ARM processor based. So, In order to compile for Zed- Board first we have generated arm binaries on Intel process using cross compiler tool chain for ARM. Then transfered binaries on Zed- Board

Caffe and it's all dependencies provide CMake build system for build- ing. CMake is cross-platform build system which use compiler- independent method and provide very easy way to handle depen- dencies. CMake seprates build from source so several builds are possible from same source directory. This is very useful and easy to handle in case when we need to compile of different platforms(In our case for Intel and ARM).

After compiling all dependencies, next step is to link those depen- dencies to the caffe itself and generate final executable image. The linker can either put all dependencies entirely into executable image or just remember their path on system and include dynamically at runtime. For Caffe I needed to compile following

dependencies for ARM:

Dynamic Linking: In this kind of linking linker takes path of shared libraries(.so on Linux, .dll on Windows) at the time of com- piling and link them while running the program by copying into RAM from storage and subsequently flling jump tables and reloca- tion pointers. This is useful when more then one program is using libraries, we can reduce storage uses by linking to same shared library in storage. And any change in library don't require re-compilation of executable.

Static Linking: This is useful when we don't have high-leve OS thus we can't have shared libraries to link in runtime. So, static libraries(.a in Linux) are copied into executable at the time of com- pilation of executable. In our case protability of executable on differ- ent platform become very easy as executable contain everyting that it needs to run. Size of executable will be more in this case due to attached libraries.

3.3 SqueezeNet Deep Compression on ZedBoard

I tested performance of 3 different networks on ZedBoard:

LeNet: LeNet is small convolution neural network for hand-writted digit recognition. I used MNIST dataset for hand-writted digits. Caffe provide pre-trained model using MNIST dataset so, we just need to feed in network defination, pre-trained model and a input image to caffe and it will output class of that image. ImageNet: Im- ageNet is very large visual database designed for use in visual object recognition softwares. ImageNet contain over 10 million annotated images with their classes.

Chapter 4

Algorithm used for signboard detection

The algorithm of signboard detection for MAVI is designed for the campus signboards of IIT Delhi as of now. The algorithm assumes that the boards are blue in colour with the text being written in white colour. The algorithm is constructed in two serial stages. The blue detection stage followed by the white identification stage. The algorithm is implemented in OpenCV 3.1.The input image is taken to be of vga resolution.(A 640x480 image is used.)

4.1 Overview of Algorithm

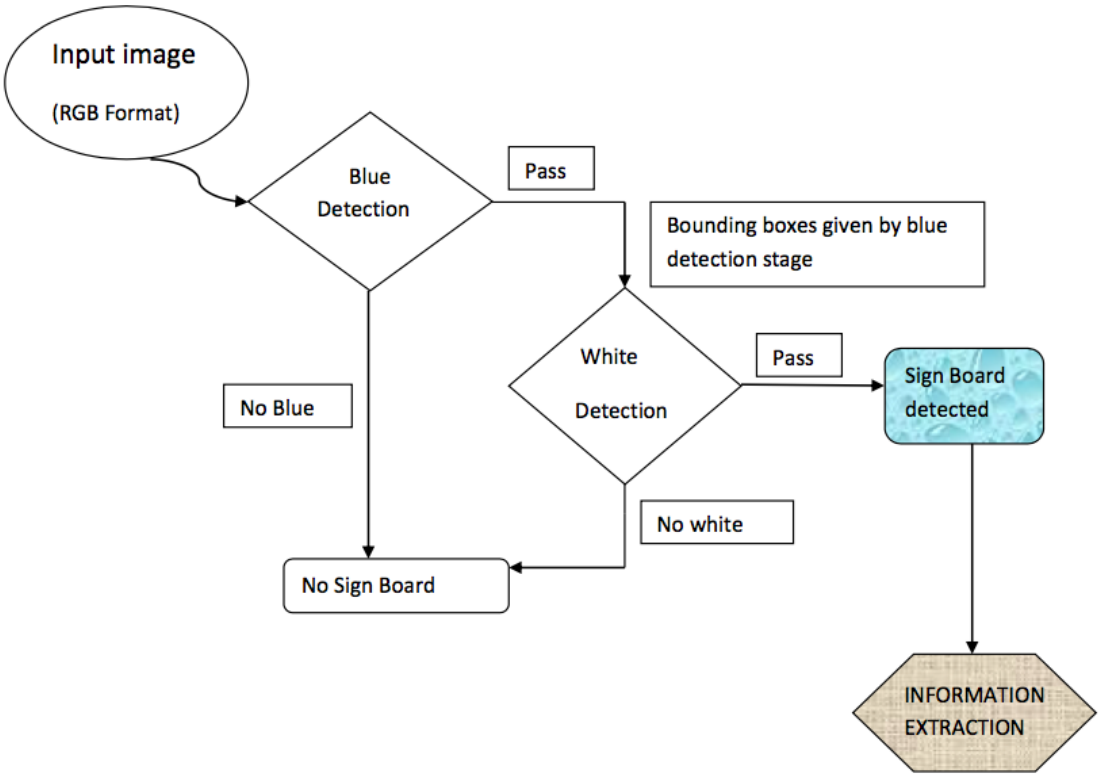


Fig. 3.1 Overview of Sign Board detection

The following figure 3.1 depicts the overview of algorithm.

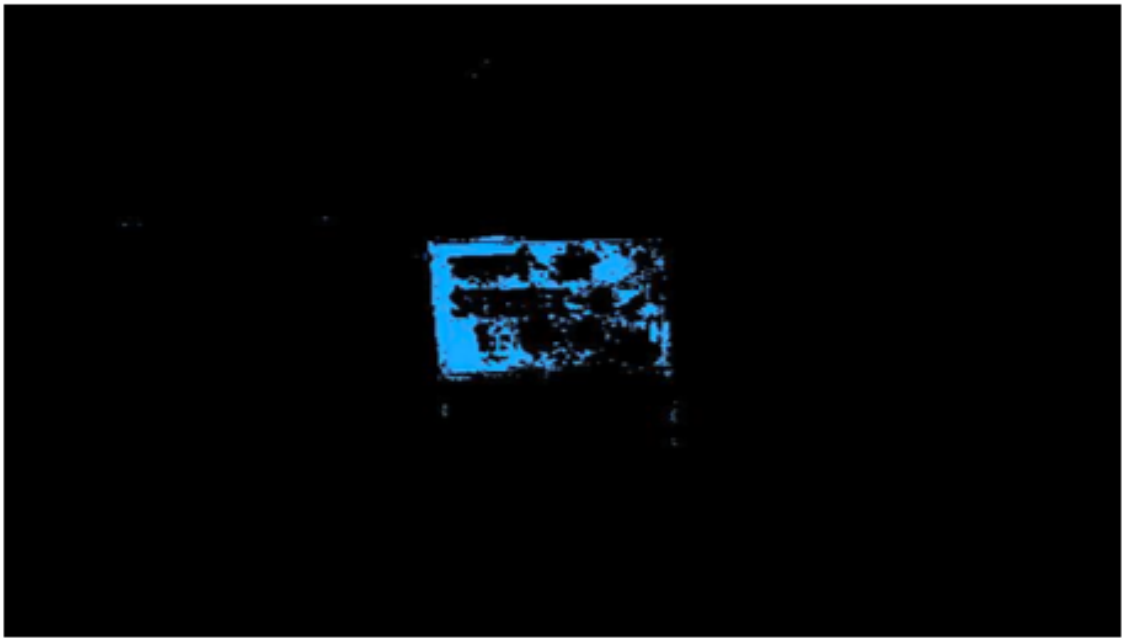
4.2 Stage 1:Blue detection stage

The steps performed in Blue detection stage are as follows:

- Read the input image in RGB format
- Transform the image from RGB colour space to HSV colour space using the OpenCV function CVBGR2HSV.
- Generate the mask for the Blue colour present in the image using the InRanges function of OpenCV and also performing BitWise AND operation to remove any residual blue.
- On the result obtained find the contours using FindContours function of OpenCV.
- While using this function , the contour retrieval mode is set to the CVRETRTREE so as to retrieve all of the contours and reconstruct a full hierarchy of nested contours.[]
- The contour approximation method used is CV CHAIN APPROX SIMPLE which compresses Horizontal, Vertical and diagonal segments and leaves only end points.
- The contours obtained are sorted in the descending order of their areas.
- These sorted contours are further approximated into closed closest regular polygon .This is accomplished by using the approxPolyDp function of OpenCV.
- The approxpolyDp function is used to approximate a curve or polygon with another curve or polygon with less vertices so that the distance between them is less than or equal to the specified precision.
- It uses Douglas-Peucker algorithm.
- Calculate the Up Right Rectangle on the obtained contour point set i.e on the approximated polygon try to fit a Rectangle.
- Group the rectangles which lie inside each other(if present) as a single rectangle.
- Return the two largest possible rectangles in the image.

These rectangular boxes are further passed on to the white detection stage for presence of text verification.The figures on the next page illustrate the implementation of the algorithm.


मार्ग-जी
STREET-G
1 TO 16



4.3 Stage 2:White detection stage

- The bounding boxes returned by the blue detect are passed on to this stage as input.
- Here we apply the flood fill technique so as to remove the boundary noise assuming that the noise is to the exterior of the board and at the border os the bounding box.

- On the resultant image we calculate the amount of white by counting the number of white pixels.
- Then a ratio of this amount of white to the total number of pixels in the image is obtained and compared against a predefined threshold.
- If the amount of the white in the image exceeds this preset threshold then the bounding box is passed out as a Possible Sign board.

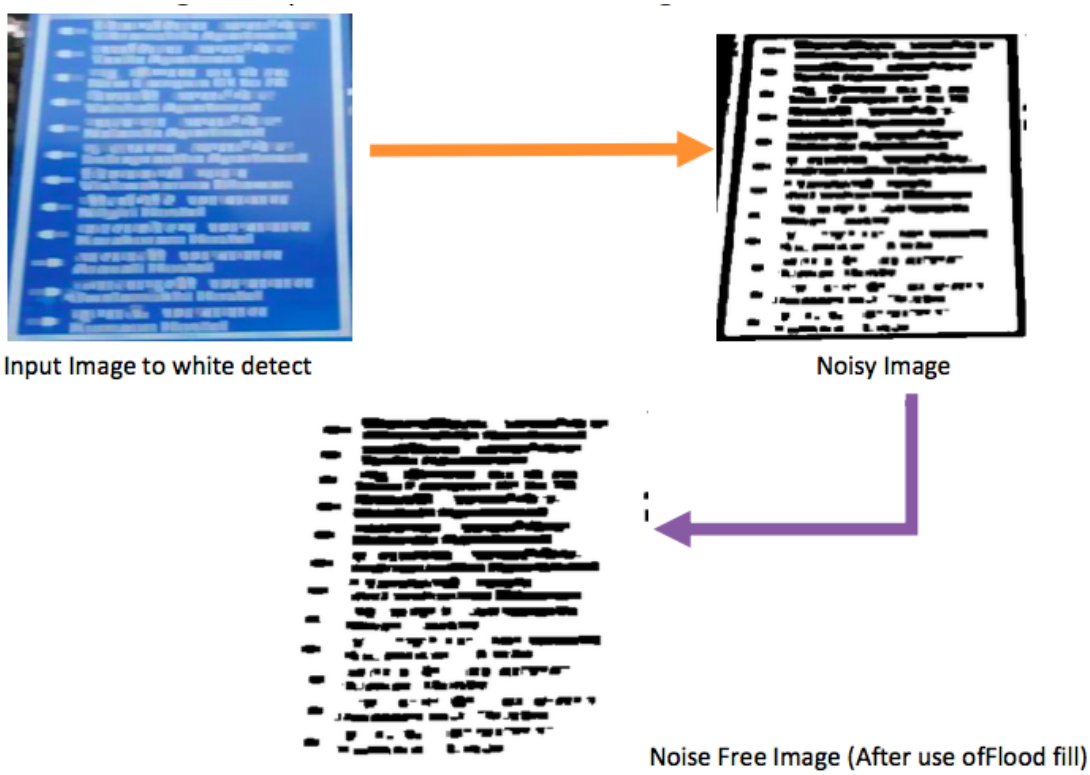


Fig.3.3.1.Demonstration of White Detection Stage

4.4 Experimental Results :

4.4.1 Accuracy of Sign Board Detection

Data Set Folder : Data set 1
Total no. of images = 600
No. of Images with sign boards = 347
No. of images without sign boards = 253

Type	Number	Percentage
True Positive	334	96.25%
True Negative	249	98.4%
False Positive	4	1.6%
False Negative	13	3.75%

Table 3.5.1. Accuracy of Sign Board Detection

4.4.2 Distance vs accuracy of Sign Board Detection

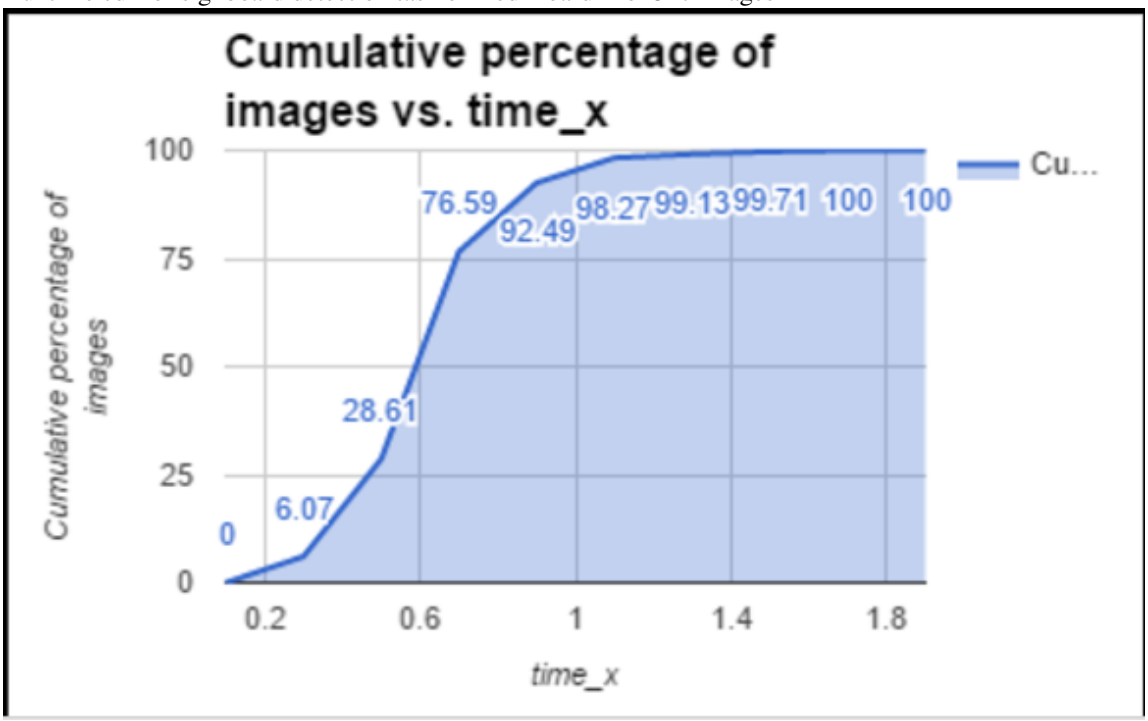
Data set employed : Data set 1

4.4.3 Time Analysis of Sign Board Detection

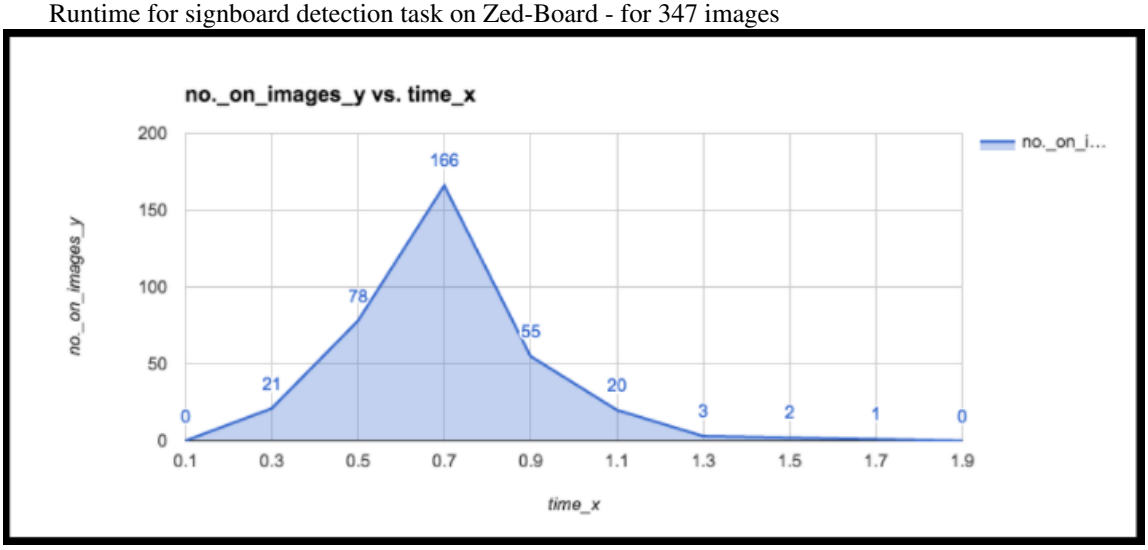
Total no. of images : 347
In Signboard Detection :
Average time taken : 0.079 secs (which had signboards) Average time taken : 0.053 secs (which did not have signboards)
In Tesseract:
On an average time taken for OCR recognition : 0.686 secs

4.4.4 Signboard Detection Time CDF

Runtime cdf for signboard detection task on Zed-Board - for 347 images



Runtime for signboard detection task on Zed-Board - for 347 images

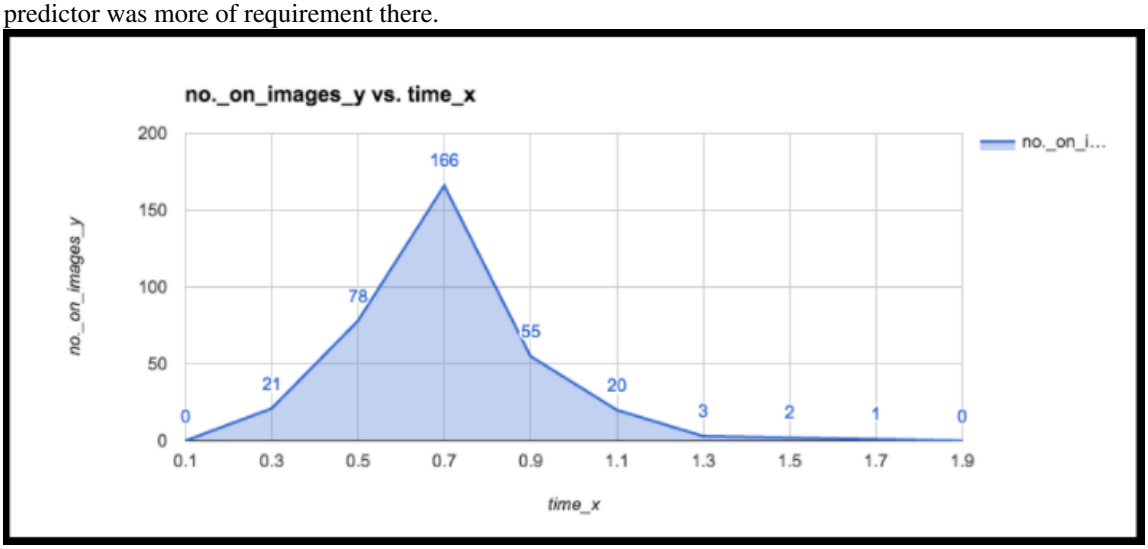


Chapter 5

Information Extraction

5.1 Conversion of Matlab code to C++

This chapter includes the process where matlab code was converted to c++ code. But the main problem faced was the library dependecies. Code had many dependencies on matlab libraries which were directly not available in c++, because of this reason we had to shift to a completely new OCR. Also the accuracy of the existing trained OCR was poor in the range of 60 to 65 percent, and so getting a new more accurate OCR predictor was more of requirement there.



5.2 Dependency on Matlab Libraries

In matlab, dependency on matlab libraries were too many which were not possible to be directly converted to C++ code for Xilings on Zed Board. Since the accuracy being low and conversion being impossible due to dependency on matlab libraries. So stopped working with the previously deployed information extracting OCR due to its non-improving accuracy.

5.3 OCR Information Extraction- Tesseract

I started working on other OCR's available which then resulted with Tesseract as its accuracy with the dataset of signboards turned out to be pretty good. Thus used it to test for time taken for output result in different languages as our signboards data samples had both English and Hindi characters. The accruacy was pretty high as stated below for a data set of 600 images.

5.4 OCR Stage

Optical character recognition (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).It is widely used as a form of data entry from printed paper data records, whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static-data, or any suitable documentation. It is a common method of digitizing printed texts so that it can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing text mining, machine translation, (extracted) text to speech and key data. OCR is a field of research in pattern recognition, artificial intelligence, and computer vision. In this work we are employing TESSERACT for recognising the English text and for Hindi text we are have implemented our own OCR.

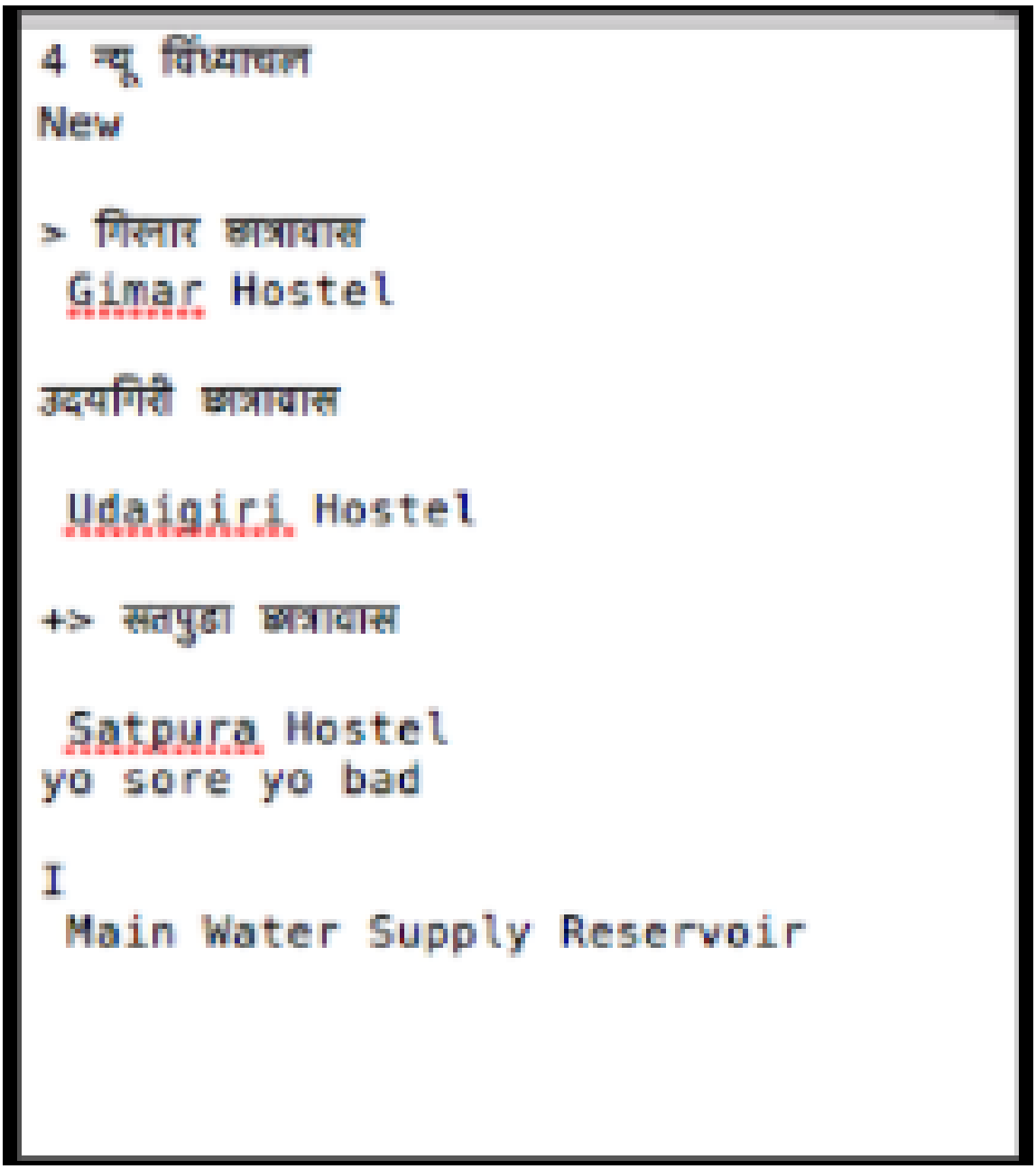
5.4.1 Tesseract

Tesseract is an engine for various operating systems. It is a free software, released under the Apache License Google Version 2.0 and development has been sponsored by Google since 2006. It was developed at Hewlett Packard Laboratories between 1985 and 1995.Tesseract OCR Engine works well for English text written in digitised format. However in case of the hand painted signboards the text identification is very inaccurate.

5.4.2 Multi Lingual

For Hindi text recognition also use of Tesseract is a viable option.
Result Sample Example
Input image to tesseract:

Output text file of tesseract



Input and output image of tesseract

विक्रमशिला
Vikramshila Apartment

तक्षशिला अपार्टमेन्ट
Taxila Apartment

न्यू कैम्पस 04 से 76
New Campus 01 to 76

वैशाली अपार्टमेन्ट
Vaishali Apartment

Nalanda Apartment

इन्द्रप्रस्थ अपार्टमेन्ट
Indraprastha Apartment

विश्वकर्मा भवन
Vishwakarma Bhawan

नीलगिरि छात्रावास
Nilgiri Hostel

कराकोरम छात्रावास
Karakoram Hostel

अरावली छात्रावास
Aravali Hostel

IE:
Jwalamukhi Hostel

कुमाऊँ छात्रावास
Cumaon Hostel

Chapter 6

OCR Software Only Implementation on Desktop

6.1 DATASET

Data Set Folder : Data set 1
Total no. of images = 600
No. of Images with sign boards = 347
No. of images without sign boards = 253
Dataset comprised of 600 images which included 347 images with signbaords and 253 images with no signboards. Almost all of the pictures has been taken in the campus. The accuracy, time and distance analysis has been done.
Features of Dataset : 600 images (347 with signboard vs 253 without signboard)
Image file (bounding box), Detection time, Extraction time, distance (0-1m, 1-3m, 3-5m, 5-7m, 7-9m, >9m), no. of signboards, GPS location, tesseract output file with file name.txt

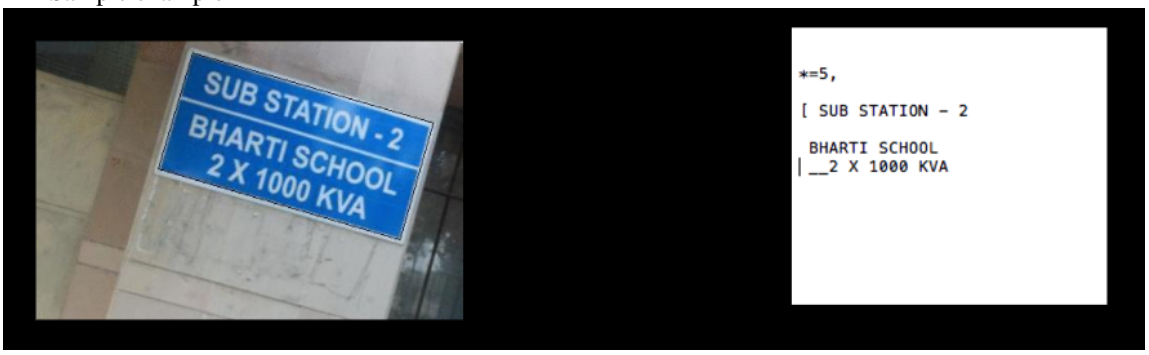
1	name	Detec	Extract	Disance	no.	GPS_lat	GPS_Long
2	truesignboard_DSC07046_resized.JPG	0.046352	0.393992	5	1	28.5455	77.18829
3	truesignboard_DSC07047_resized.JPG	0.038036	0.323306	1.5	1	28.5455	77.18829
4	truesignboard_DSC07052_resized.JPG	0.074933	0.636930	1.5	1	28.5455	77.18829
5	truesignboard_DSC07053_resized.JPG	0.085917	0.730294	1.5	1	28.54507	77.18947
6	truesignboard_DSC07056_resized.JPG	0.099299	0.844041	2	1	28.54552	77.18836
7	truesignboard_DSC07057_resized.JPG	0.12363	1.050855	3.5	1	28.54443	77.18189
8	truesignboard_DSC07062_resized.JPG	0.081939	0.696481	2	1	28.54443	77.18189

Sample Dataset

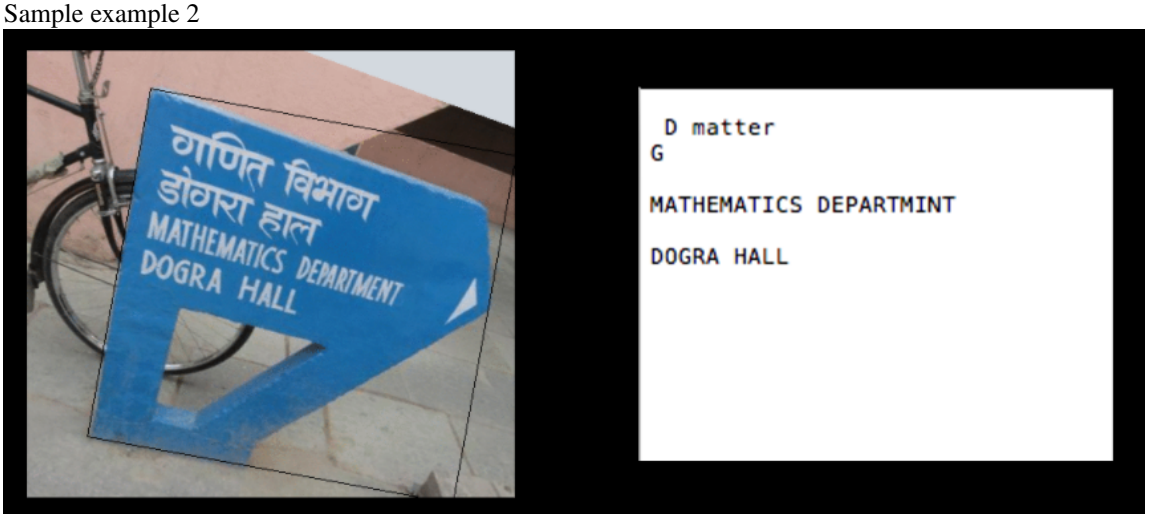
6.1.1 Skewed Image correction software only implementation

Improvisations in Tesseract
1. Accuracy too low for Skewed Images : When the images are inclined/skewed, the bounding box for signboard gave poor accuracy Implemented methods which exactly matches the dimensions of signboard (Image processing) Improved accuracy Effect on time for information Extraction : Average timing increased from 0.079 secs to 0 0.087 seconds (10

Sample example 1



Sample example 2



Chapter 7

Analysis - Time, Distance and Accuracy

7.1 Accuracy vs Distance

Distance of signboards from user vs accuracy
Fields in the table : Distance brackets,Count of sample sets in and No. of detected images

1	Distance range	Count	Accuracy
2	0.5	2	0
3	1	46	32
4	1.5	23	21
5	2	45	45
6	2.5	20	19
7	3	41	41
8	3.5	24	24
9	4	42	42
10	4.5	16	16
11	5	39	39
12	5.5	4	4
13	6	18	18
14	6.5	1	1
15	7	17	17
16	8	8	7
17	Grand Total	347	326

7.1.1 Maximum distance for which signboard is detected

For distance between1m and 8m : Total No. of Images(with sign board) = 347 (345 with distance greater than 1m) Detected = 326(94However for distance less than 1m (typically from 0.6m) the accuracy diminishes very much.

7.1.2 Maximum distance for which OCR detects accurately

Head-On (Average): Although for accuracy as above 2m is the maximum distance (depending on the size of the signboard/clarity of text)
Inclined : Average : 1.5m (maximum)

7.2 Tesseract Accuracy

7.2.1 Accuracy for OCR : Character by character

Out of total multiple test images i obtained from the video sample of pivot head camera i took 100 images for which i did the distance vs accuracy test analysis.
Hindi Only : 63.7
English Only : 94.4
Both : 81.7

7.2.2 Accuracy for OCR : Word by word

Out of total multiple test images i obtained from the video sample of pivot head camera i took 100 images for which i did the distance vs accuracy test analysis.
Hindi Only : 53.57
English Only : 85.3
Both : 70.1

7.3 Time Analysis

7.3.1 Time Profiling Results - Images With Sign Board

After calling both functions : i.e. Signboard detection part(blue part masking for a rectangular region followed by white part detection in it)
Count zeros the no. of times blue pixel has been occurred and if greater than 60 on masking then we say a signboard may be there and then white confirms it.
main : 40.33
Out of which : blue : 20white : 20
Rest of the time is taken by c++ internal libraries,openCV libraries

7.3.2 Time Profiling Results without Sign Board

main 51.7
Out of which : blue : 50.5white : 0
Rest of the time is taken by c++ internal libraries,openCV libraries

Chapter 8

Conclusion and Future Work

Signboard detection works completely fine and average time taken after improvisation is 0.87 sec.
Extraction : Tesseract accuracy is good enough but the time taken is quite bad as the average time on Zed Board = 6.86 secs
Google Vision Api and Hardware acceleration for OCR
Currently the algorithm works only for the sign boards with the text in white colour. This needs to be extended to all possible signboards by making the algorithm less sensitive to colour. The information extraction part is restricted in its accuracy to the digital sign boards as the OCR engines performance is very poor for hand written text. The noise removal algorithms include certain assumptions that in some cases lead to loss of information. The current state implementation does not have any provision for shadow removal and it needs to be taken care of in future. Also the hardware acceleration has to be performed.

Appendix A

References

- https://en.wikipedia.org/wiki/Opticalcharacterrecognitioncitenote-1.
- https://web.archive.org/web/20061026075310/http://google-code- updates.blogspot.com/2006 /08/announcing-tesseract-ocr.html
- https://en.wikipedia.org/wiki/Tesseract
- https://tesseract-ocr.googlecode.com/files/TesseractOSCON.pdf
- http://zedboard.org/sites/default/files/documentations/ZedBoardHWU Gv22.pdf
- http://zedboard.org/sites/default/files/documentations/GS-AES-Z7EV- 7Z020-G-14.1-V65B15D.pdf
- http://literature.cdn.keysight.com/litweb/pdf/59680162EN.pdf?id=10000 70110:epsg:dow