Signage board Information Recognition

CSCI55700 Project Report

Raghavendran Vijayan

April 23, 2017

1 Abstract

The purpose of the project is to try implementing concepts discussed in the classes and gain more practical experience on Image Processing. This project is about detecting traffic signs present on the sidelines of the road (static images considered) and alarming the driver. Though there are already number of systems in place detecting these traffic signs, this system make me try another approach and found it to be successful. A segmentation algorithm to identify and classify signs from the given image is proposed and implemented using MATLAB.

2 Introduction

In this project, we attempt to detect traffic signs present in the given static image and alarm the user. I have inspired and tried few concepts based on the lectures Edge, feature detection and Segmentation and binary images.

At this time, I have considered and developed system based on system followed in the United States. In this time span, we have a system, in place identifying the stop and yield sign. Since these signs are red in color, our model extracts the Red portions of the image and removes the blue and green portions from it for better performance. Then, our segmentation algorithm identifies shapes based on filling ratio of the identified segments. Then these identified shapes are shown in the original image and an alarm signal is played to alert the end user.

3 Literature Review

The paper [1] is the main motivation behind implementing the system. Their procedures include RGB thresholding and removing the areas whose filling ratio is too large. They have tested on dataset and have found the ratio of major to minor axis of the standard stop and yield signs found to be near 1. I have tried different images and have used 1.6 since it yields good results.

The paper [2] uses ring shaped template matching to detect and find the location of the signs present in static image. They have also listed how MATLAB helps in detecting these signs. Also, the difficulty in using RBG changes in light are discussed. If our algorithm combines the template matching feature, it couldve produced more accuracy.

The paper [3] proposes an idea to detect the traffic signs using color classification methods. The usage of Gaussian function to find the key points in the image is explained. It is a paper where they havent explained the implementation ideas (either proposed or delivered).

Among these papers, the first one is the backbone behind our proposed work flow in the system. The second one gives the solution to improve the accuracy.

4 Technical Description

4.1 Algorithm followed in our project

The following is the step by step execution details of our project.

1. Read the given image
2. Identify the R, G and B segments of the image and remove the blue and green portions from the red portion.
3. Convert the image into binary image
4. Remove the noises present in the binary image.
5. Perform connected component analysis using four connected neighborhood definition.
6. For every segment, find the minor and major axis.
7. If the ratio of the major to minor axis is found to be greater than 1.6, assume them as non-desired regions and remove considering them for future calculations.

8. Check for any other tiny region that are still continuing errors and remove them.
9. For every segment, find the points on all the four directions, where they begin initially.
10. Construct the points to draw the identified shape in the original image.
11. Check, if the area of the image is greater than our set threshold value 2000.
12. If it satisfies step 11, crop the points from the source image and play an alarm.

4.2 Work flow of the system

The following diagram will explain the work flow of the system in detail.

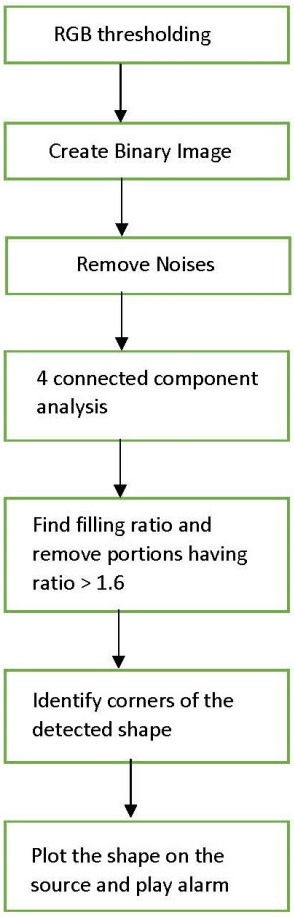

Figure 1: Work flow of the system

4.3 Implementation Details

I have implemented the whole system using MATLAB. There is no specific library or toolbox used for our project.

Following are the inbuilt functions I have used.

1. Imread and imshow() To read and show the processed Image
2. Imbinarize() To convert the RGB image to binary image
3. Bwareaopen() To remove areas that are having area lesser than our threshold set.
4. Regionprops() To measure Area, Major and Minor axis of the segment
5. Audioread() To read the alarm sound

6. Audioplayer() To create object for playing the sound

Following are some explanations, why we have our algorithm framed in this manner.

1. Since traffic signs are always red in color, its ideal to consider only the red portion of the image. For instance, there is no need to consider clouds and road present in the image to be detected.
2. After different tries, I have found the threshold value to remove noises present in the initial binary image to be 400. It eliminates majority of the noises in 9 out of 10 trials.
3. As per the proposal in the paper, we are calculating the ratio of the major to minor axis length. And, we are removing the segments with values greater than 1.6. Because standard signs are observed to have ratio less than 1.6. Also, it eliminates some bigger shapes that are red and not traffic signs.
4. Using different loops, we can figure out the places where the segment begins and ends.
5. To form the rectangle out of the identified points, we are manipulating values. Figure 2 shows the way how the points of the rectangle are calculated. They are found using the following formula
   $Top(Left)$
   $Row = top5;$
   $Column = right + 5;$

   $Top(-Right)$
   $Row = Top(Left);$
   $Column = left5;$

   $Bottom(-Left)$
   $Row = bottom + 5;$
   $Column = Top(Left);$

   $Bottom(-Right)$
   $Row = Bottom(Left);$
   $Column = Top(right);$

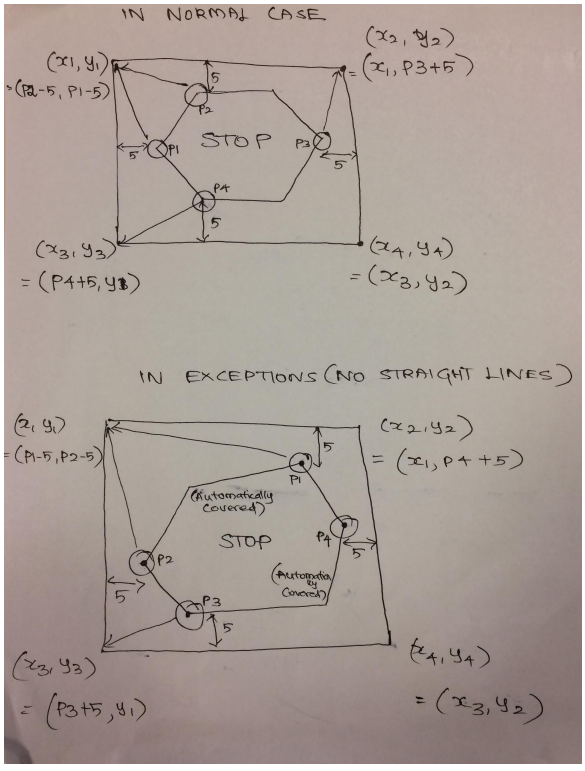

Figure 2: Corners of the rectangle are getting calculated

6. There are chances the areas within the segments are also considered as different segments. For example, the red part inside p and o characters of the stop may be treated as different segments, if it is a properly captured image. We can remove the case. Hence I am finding the area of the identified segment and capturing images that are having area greater than 2000(based on different trials).

4.4 Experimental Outcomes

The following sequence of images will show, How the signs are detected using this algorithm.

The sequence of images are obtained, when testing on "mixed.jpg".

Figure 3: Different transformations of the image

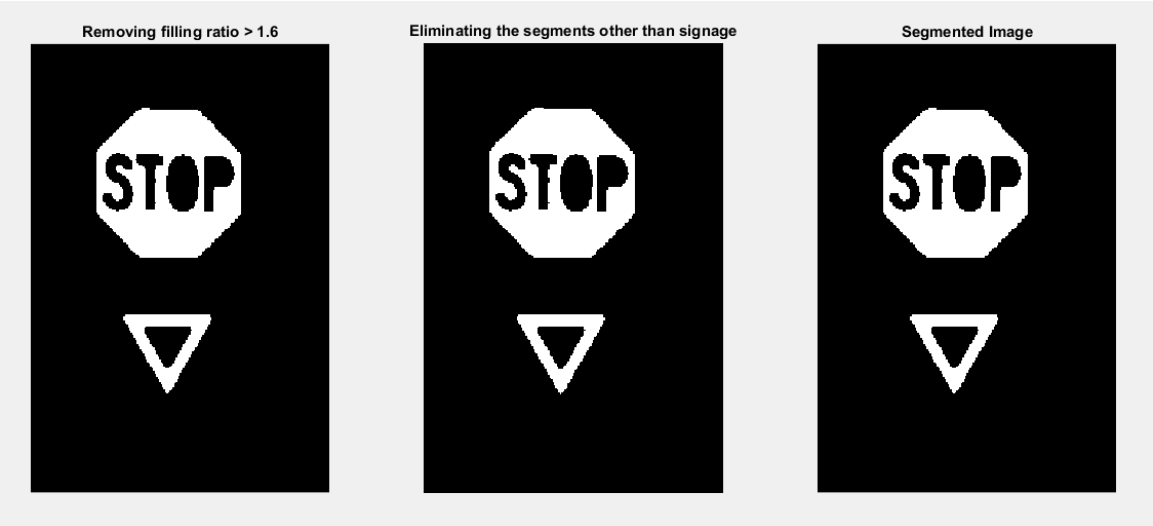

Figure 4: Obtaining Segmented Image

In this sequence of images, In the second image present in 3, The red portion has been extracted. If We notice well, There is something that is present between stop and yield sign is also captured. It gets removed in the 4th step after finding filling ratio for every segment. Since there is no noise present, The three images in second row will appear similar. Then the segments that are identified to be signs are plotted in original image and have shown separately too.

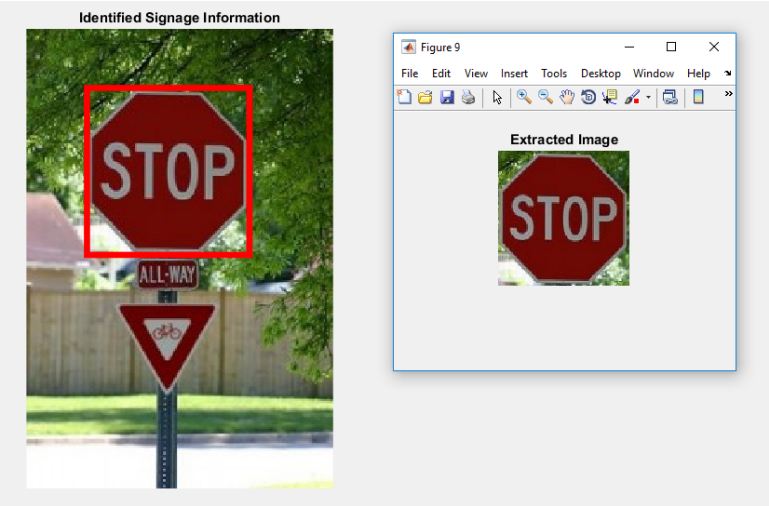

Figure 5: Stop sign detected

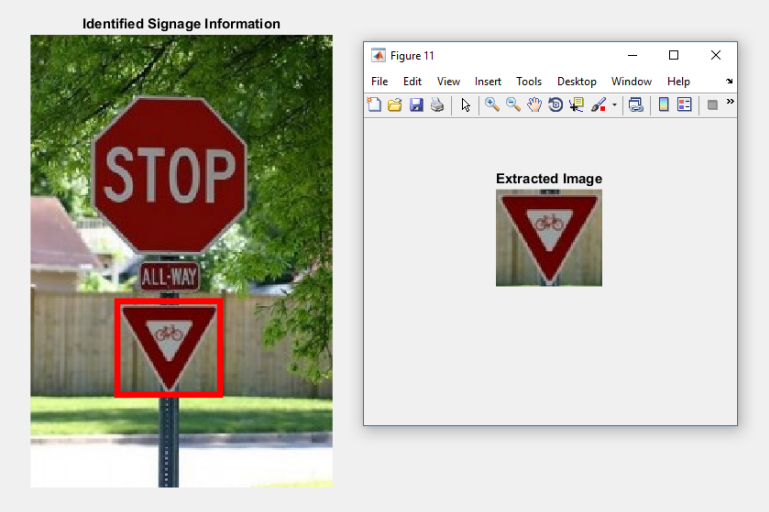

Figure 6: A symbol similar to yield is detected

Following are the results obtained when testing with "traffic3.jpg".

Figure 7: Different transformations of the image

Since, There is only one stop sign present, It is detected accurately. Though there are many other signage information and a red color car crossing the road, It detects only the stop sign.

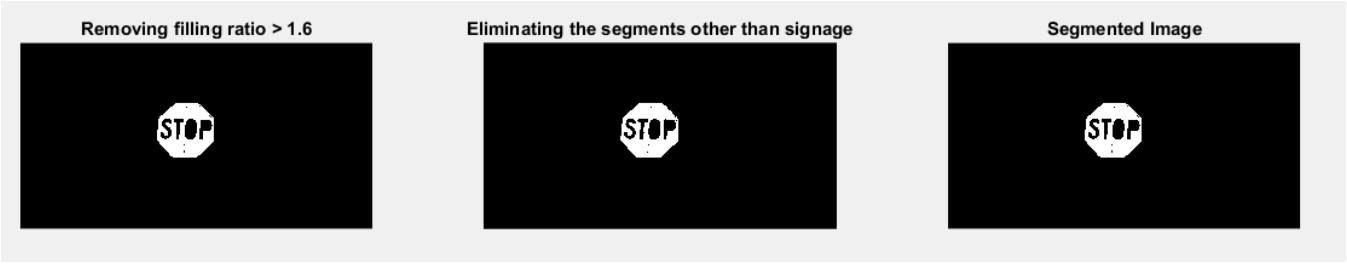

Figure 8: Obtaining Segmented Image

Figure 9: Stop sign detected

Lets see how It performs in case of "dual.jpg".

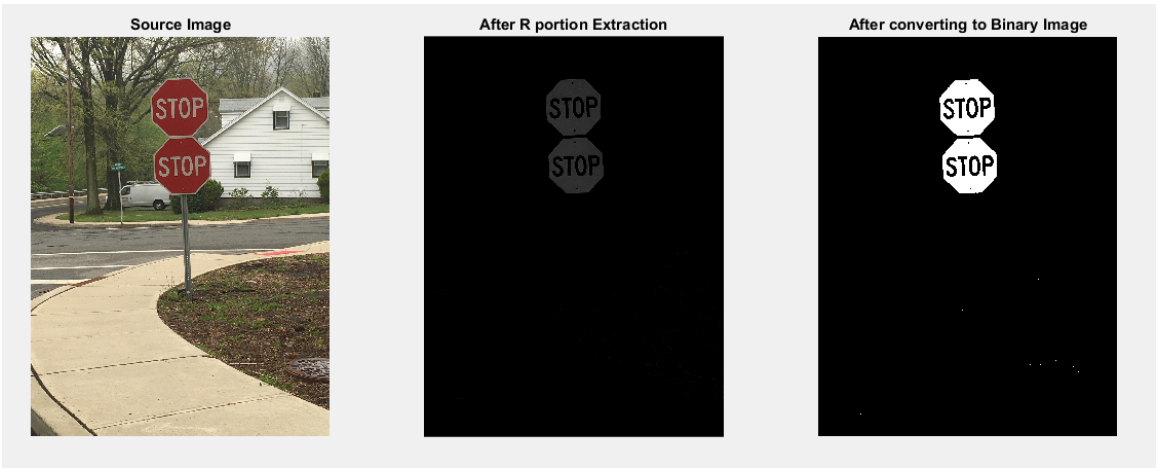

Figure 10: Different transformations of the image

Since the red parts present within the character P of the stop passes our different tests like filling ratio, small area elimination, It is also detected. Actually, It shouldnt be detected. Though there are places in the code, finding the area of the segments and checking if it is fit as sign or not, They are not working ideal in all cases. It is one of the cases where It failed to execute accurately.

Let's look at the image that is captured in the night time.

Here, the red color light present in the image is also detected in the first four portions. After eliminating areas that dont have area less than 500, It gets disappeared. Subsequently, those segments are not identified in the last step.

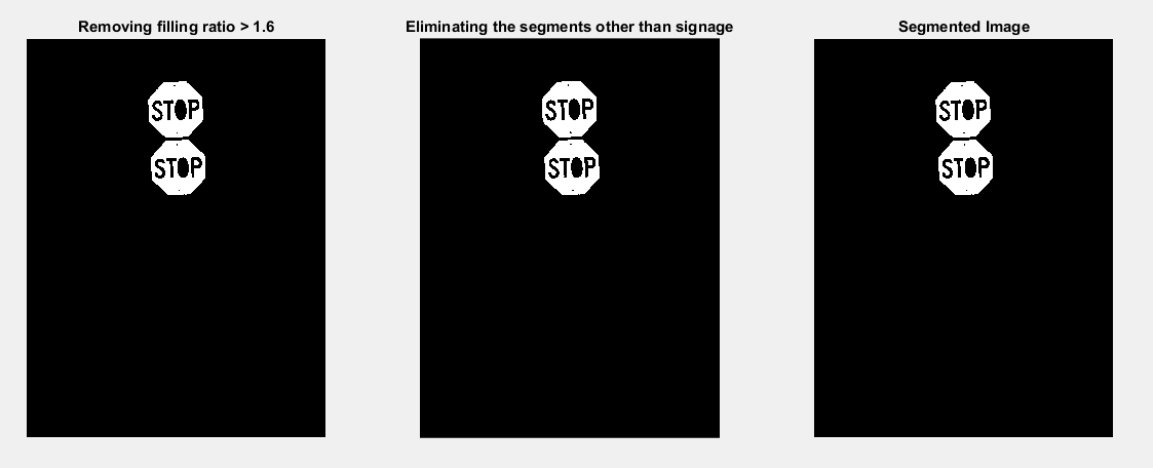

Figure 11: Obtaining Segmented Image

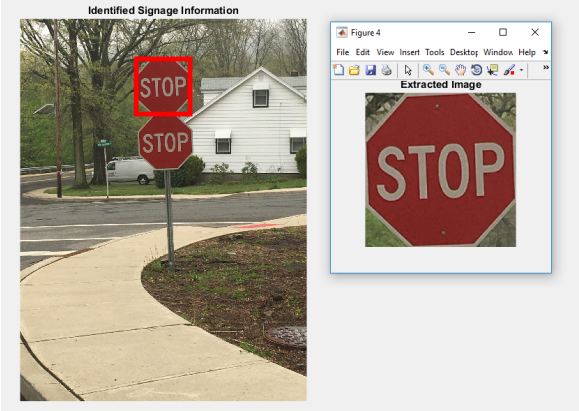

Figure 12: Stop sign detected

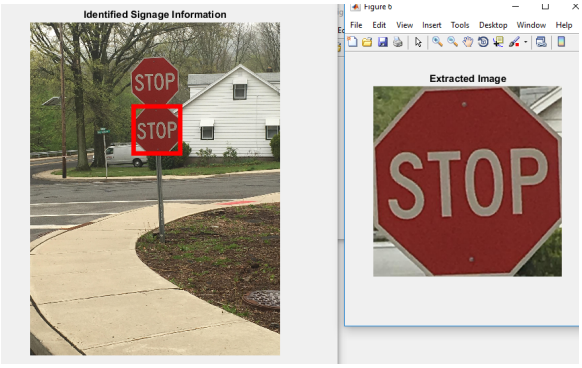

Figure 13: Stop sign detected

This will be an example where it is working with more failure rate. But, Still, it detects the presence of a stop sign.

I wish, more trials and observations can be explained when showing demonstration.

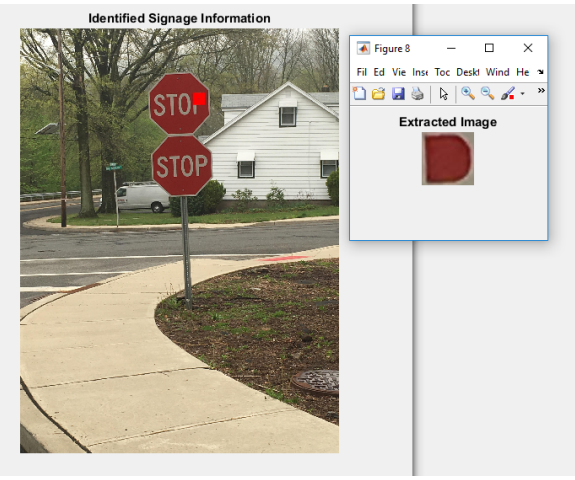

Figure 14: Red portion inside p is getting detected

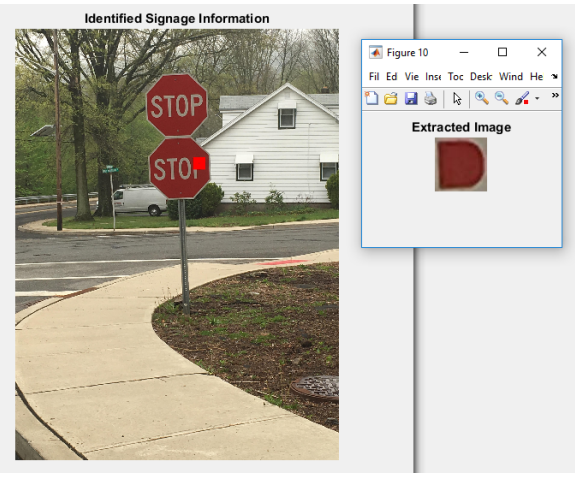

Figure 15: Red portion inside p is getting detected

Figure 16: Different transformations of the image

4.5 Other concepts tried

I have tried and worked out some other methods before arriving at this solution. I can show them during the project demo for reference.

1. SIFT Scale Invariant Feature Transform
   They detect local features present in the image. They extract the local points from the object and train

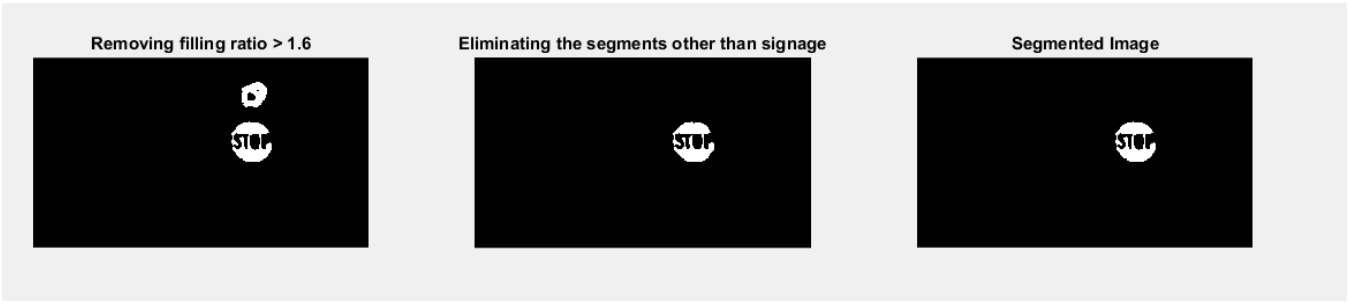

Figure 17: Obtaining Segmented Image

Figure 18: Stop sign detected

Figure 19: Different transformations of the image

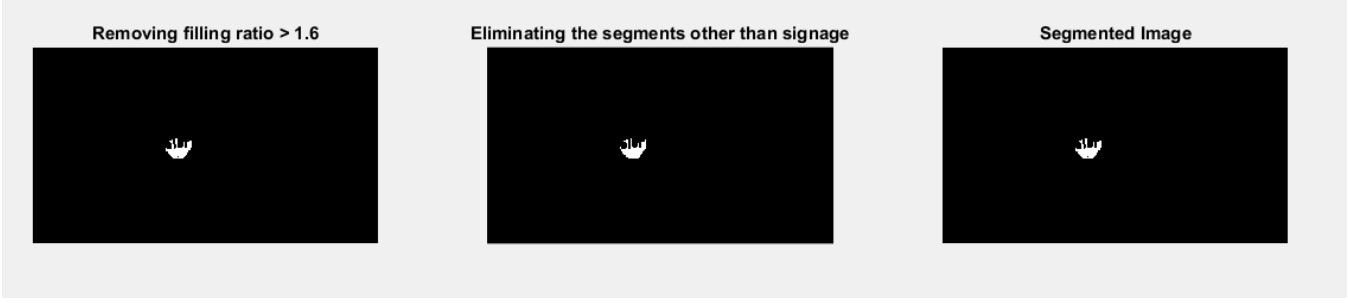

Figure 20: Obtaining Segmented Image

Figure 21: Stop sign detected partially

them to extract the same info from the other local test images. Like mentioned in the progress report 2, computed the interesting points of them using detectSURFFeatures inbuilt function. I extracted the features present in them and then, the matched features and points between them. those matched points are identified to be the points, where sign is located. They work fine half of the times. For the image, where it is inclined or viewed in different angle, It fails badly.

2. Instead of using bwlabel, I am trying to use the connected components analysis non-recursive algorithm implemented in the assignment 4. Since, I didnt successfully implement the pass 2 (equivalence class matching) portion, I have used the inbuilt function.

3. To compute the area of the segments, I have tried using the formula discussed in lectures. But, they seem to be not working properly in half of the cases. So, I have used the rectangles area formula.

4. I tried extracting characters using Optical Character Recognition. I am struck in between. So, I am not able to train and test the data completely.

5 Strengths and Weakness

5.1 Strengths

1. They detect the stop and yield sign with more accuracy irrespective of any situation. They can be inclined or the color in the board may be faded. But, they never cease to consider them for segmentation.

2. The system never considers any other noise as a segment and start marking as traffic sign.

3. It is less affected by any lighting conditions present in the image. We have shown, the algorithm working fine even in night.

5.2 Weakness

1. If there is more shadow or If the image is clicked during sun set, there are more chances, the stop sign will not carry red color. At these cases, the algorithm wont be effective since the first step in our algorithm segments based on color.

2. There is no machine learning portion present. As a result, we cannot be sure, the detected red color hexagon and triangle is always a stop and yield sign.

3. To adapt this system to work on real data, It iwill not be highly effective solution because of the time involved in calculations.

6 Future Work

1. Since this is my first course working and interpreting with images, it took time for me to understand and implement concepts. If provided more time, I will try Optical Character Recognition. It will assure cent percent accuracy in our algorithm.

2. I believe, I will implement this concept when giving demo. I am working on this now. Using the non-recursive connected components algorithm, I can try finding segments. If equivalence class portion is completed, I may not need using bwlabel function.

3. May be, instead of setting hardcoded threshold values, I may try local thresholding. This wouldve helped me gain hands-on experience for adaptive thresholding lecture slides.

7 Conclusion

The various processing methods in this project include RGB thresholding, 4 connected components analysis, segmentation of image and plotting the identified sign and alarming the end user. We have an algorithm created by us and have seen it successfully detecting and working. We have experimentally showed the results and I believe it works fine all the times and 3 out of 4 times, it works more accurately. To overcome the limitations, It requires training the data with good machine learning algorithm on the characters. The implementation of OCR will make it nearly cent percent accurate.

References

[1] Revathi A.S, Sanamdeep Singh Anand, and Tejaswin Gumber. Traffic Sign Detection Using MATLAB. In *International Journal of Science and Research (IJSR)*, volume 5, pages 1251–1254, Nov 2016.

[2] M. A. Garcia, M. A. Sotelo, and E. M. Gorostiza. Traffic Sign Detection in Static Images using Matlab. In *EFTA 2003. 2003 IEEE Conference on Emerging Technologies and Factory Automation. Proceedings (Cat. No.03TH8696)*, volume 2, pages 212–215 vol.2, Sept 2003.

[3] M. C. Kus, M. Gokmen, and S. Etaner-Uyar. Traffic Sign Recognition using Scale Invariant Feature Transform and Color classification. In *2008 23rd International Symposium on Computer and Information Sciences*, pages 1–6, Oct 2008.