STOP Sign Detection

Chaitali Kamble

ABSTRACT

Nowadays, Commutation is one of the essential part of human life. This commutation can be done by several means and driving a vehicle plays one of the measure part in the same. Thus, there are many research going on to make this driving easy and comfortable along with the security. Previously, the vehicles had manual gear shift eventually it converted into the automated gear shift. Nowadays, sensor plays a major role in driving and self drive car concept is emerging. To contribute the same, I have developed a STOP sign detection system. It provides drivers the relevant information about the road conditions. The aim of this project is to detect a STOP sign on a road. The interesting part of this project is learning various image processing techniques and implementing all of them under the hood. This report discusses various implementation techniques used in image processing with strong evidences based on the earlier research work in this area. Also, it discusses about the future work.

1. INTRODUCTION

The STOP sign detection has various research work associated with it. The dataset [3] considered for this project contains images of all United States stop signs. When these images are observed they all are not similar. The STOP signs in some of the images are very far from the observer hence, they appear small in size. Some images show varied lightening conditions. In fact, some of the images do not have appropriate red color color sign. To address these issues a research work has been carried out.

The images are divided into training and testing sets. Both the sets contain variety of images. This helped to get maximum accuracy in the implementation.

The subsequent sections in this paper show related research work, System overview, some of the challenges faced during implementation and evidences of the results.

2. RELATED WORK

Every project work needs some research work to understand other researchers though process. Hence, the research of three technical papers related to traffic sign detection is been carried out. Other supporting research papers [6], [7], [10], [8], [5], [4] are also considered.

The authors of the paper [12] detected traffic signs using Hough Transform technique. This approach followed certain steps such as color analysis and denoising, detection and then tracking with prediction for better accuracy. They applied these techniques to just detect triangular signs.

The authors of another research paper [11] detected the traffic sign and performed the classification for the speed limit. They realized that all the speed limit signs are circular and hence they used circular hough transform technique. In the step of detection of traffic signs, they are removing noise from the image and segmenting the image from background. They used Canny edge detector to find the edges in the image. The image is then segmented by extracting the digits and classified for speed limit sign detection.

The very last paper [9] talks about fast traffic sign detection and recognition under changing lighting conditions.They used circular Hough transform to detect the sign and then used Neural Networks to classify those signs. For tracking the traffic signs, they used Kalman filters.

The research work helped to decide the approach for this project implementation. The research carried out so far aims at specific business case and work can differ from one case to another.

3. SYSTEM OVERVIEW

The system built is to detect the STOP sign and has various stages or decision factors that can lead to the final aim of detecting the STOP sign. Initially, the system asks user about the method of user's choice to run the function. If user says that he wants to detect a STOP sign using MATLAB built in toolbox, then the built in toolbox function is called and the STOP sign is detected. But, if the user wants it using my implementation, then the test image passes through various stages such as extraction of red like sign, if not found then reading the STOP text in the image and matching the template. If a test image passes through any one of the above stages then the STOP sign is highlighted with a blue colored square box. The various stages involved in the System overview are explained in the implementation section.

4. IMPLEMENTATION AND RESULTS

4.1 Dataset

The dataset [3] used for the purpose of STOP sign detection is been referred from the Laboratory For Intelligent and Safe Automobiles (LISA). The LISA Traffic Sign Dataset [3] is a set of videos and annotated frames containing only US traffic signs. It has two sets, one set consists of just pictures and another set consists of pictures as well as videos. But, second set is still under construction. The dataset contains 47 US traffic sign types.It has 7855 annotations on 6610 frames. The images are obtained from various different cameras and even size of the images also varies. Each sign is annotated with sign type, position, size, occluded (yes/no), on side road (yes/no) in the CSV file.

This dataset is really very huge. Each traffic sign is different from other and hence needs individual processing. Therefore, for the purpose of this project, the interest is just shifted to STOP signs and other signs from the dataset are ignored.

In this project, STOP sign detection is performed in various ways. Hence, when MATLAB's built in classifier is used to detect STOP sign, it has it own built in dataset as well. For other techniques, the entire dataset of LISA is used for testing purpose.

4.2 Stop Sign Toolbox

The STOP sign toolbox [2] the built in toolbox provided in MATLAB for the purpose of detecting stop signs. It comes with a built in classifier named as train object detector. The file name for dataset is 'stopSignsAndCars.mat'. This file consists of the positive and negative instances of Stop signs with approrpiate labeling. Two separate vectors are constructed each one for positive instances and negative instances. MATLAB's built in function train cascade object detector is used to classify the instances based on HOG features.

trainCascadeObjectDetector ('stopSignDetector.xml', positiveInstances,negativeFolder, 'FalseAlarmRate',0.1, 'NumCascadeStages',5);

The newly trained classifier is used to detect the STOP sign in an image.

detector = vision.CascadeObjectDetector ('stopSignDetector.xml');

The Stop sign is detected using blue square bounding box around it.

4.3 Decision Tree

The STOP sign toolbox is using Classifier on the basis of Hough features. Hence, the another approach is the Decision tree which is based on several already learned techniques in image processing.

The various stages involved in the implementation are discussed below:

4.3.1 Noise Removal

The 'Guassian' filter of size 1 having standard deviation 0.4 is applied to remove the noise and keep the edges as it is.

function im_filtered = Remove_Noise(im)
filter_gauss = fspecial('gauss', 1, 0.4)
im_filtered = imfilter(im, filter_gauss, śame; ŕepl)́
end

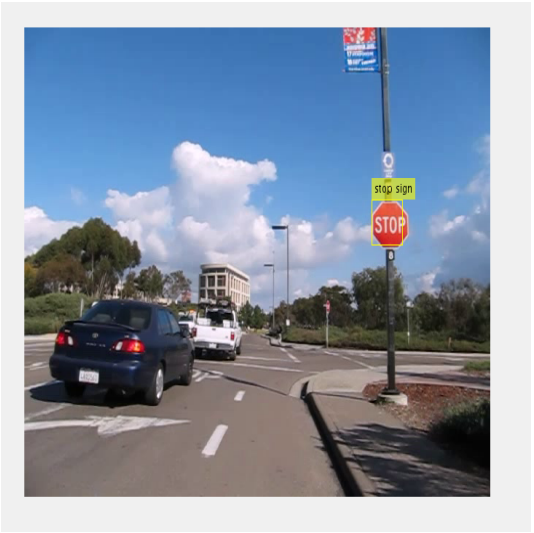

Figure 1: Stop Sign detected using MATLAB Tool-Box

4.3.2 Extract Red Objects

The very first task involved in the STOP sign detection is to identify the red like thing from an image. To identify the red like thing, the Macbeth color checker chart is used. The image is first converted into LAB color space as a trial color space. Then, the value of each color channel (in this case, L , a and b) is been checked where the STOP sign resides. The red like thing is then extracted from the image based on below condition.

RED OBJECT = (a $<$ 80) and (a $>$ 40) and (b $>$ 25) and (b $<$ 55)

where a is the first color channel in LAB color space and b is the second color channel in LAB color space. When this condition is actually applied on a input image, the red like objects are extracted from the image.

As there are two signs in original images, there are two extracted red like objects.

4.3.3 Dilate Red objects

The red objects found in the last step are not visible so as to detect their appropriate shape. They can be noise pixels as well. Hence, this image is subjected to dilation.

function im_dilate = Morphology(im)
Struct_ele = strel('disk', 9)
im_dilate = imdilate(im, Struct_ele)
end

After dilating the red like objects from the image, the red like blobs appear bigger in size.

4.3.4 Get appropriate red blobs

The dilated image is now subjected for clearing the borders from the image. Now to at least go with one of the red like blob, MATLAB's built in function region props is used used to get the Area and bounding box for the red like objects. The area parameter is important because it will help to remove red like noise from the image and bounding box will give the coordinates of the red blob. When tested on several images in dataset, it is found that if the area of the red like blob is below 500 then it is a noise and not a STOP

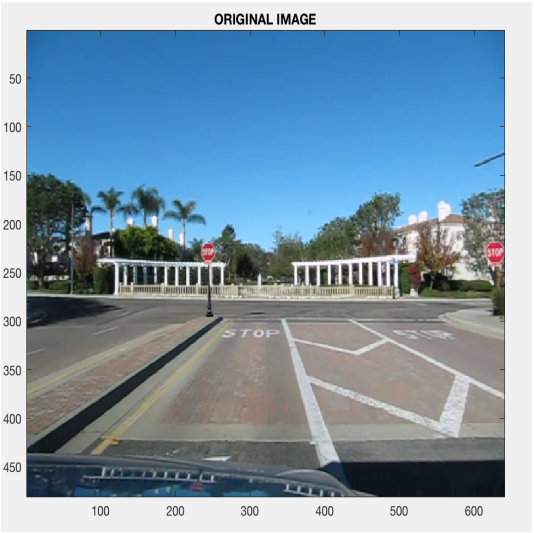

Figure 2: Original Image from the dataset

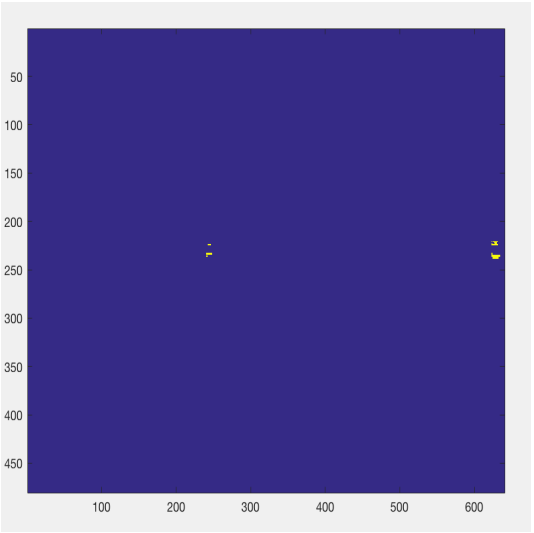

Figure 3: Red objects extracted from the image

sign. Hence, the images which has red like object having area greater than 500 are passed on to the next evaluation phase.

4.3.5 *Template Matching*

The bounding boxes of the previous images are considered and a template is created having the coordinates of the bounding box.
Template = im(round(Box(2):Box(2) + Box(4)),
round(Box(1): Box(1) + Box(3)),:);

The template is created based on the assumption that all the images in our dataset can correlate with this. This template is passed to perform correlation on the images so as to locate exact spot of the STOP sign.

If the Red like object is not detected in the first phase, it means that it might happen that the color of the STOP sign in an image must be different in shade than what we are expecting. Hence, this condition is checked from the nLabel parameter which is returned from the region props function. If nLabel is zero that says the sign is not detected properly and hence a new template is created to perform local correlation. This template is created from one of the images from the dataset. The maximum of all the pixels in the matched region are collected. This region is called as Region of interest. The STOP sign is located on the entire image surrounded by a square box.

4.3.6 *Locate Region of Interest*

The region of interest created from the earlier step is the region where the maximum correlation occurred. The pixels for this region is collected using max(I(:)). Now, to make the task simpler, these coordinates are checked whether they lie on left half or right half of the image. This step was important for the ocr detection in the next phase.
function ROI = GetRegionofInterest(im, x, y)
dims = size(im) ;
if x >= dims(1)/2 ;
then nstart = round(dims(1)/2);
and width = round(dims(1) - nstart);
else
nstart = 5;
width = round(dims(1)/2 - nstart);
end
if y >= dims(2)/2
then mstart = round(dims(2)/2);
and height = round(dims(2) - mstart);
else
mstart = 5;
and height = round(dims(2)/2 - mstart);
end

Newly formed region of interest is: ROI = [nstart, mstart, width, height];
end

4.3.7 *Optical Character Recognition*

The techniques used so far in the implementation were not leading towards the maximum accuracy of the results. Hence, Optical Character Recognition function is used to detect the 'STOP' text on the image.

MATLAB's built in function OCR(I) [1] returns an ocr text object containing optical character recognition information from the input image, I. The object contains recognized text, text location, and a metric indicating the confidence of the recognition result.

The drawback of this function is it always needs binary image. When the image is converted into binary format the STOP text is actually cluttered and hence, some morphological operations are performed to recognize the text from the image.

Initially, the function ocr was struggling to find the STOP word but when a region of interest (the one from template matching) is passed to the function, it detected the text correctly.

5. CHALLENGES

1. The image processing is itself a challenging task because the implementation technique discussed above

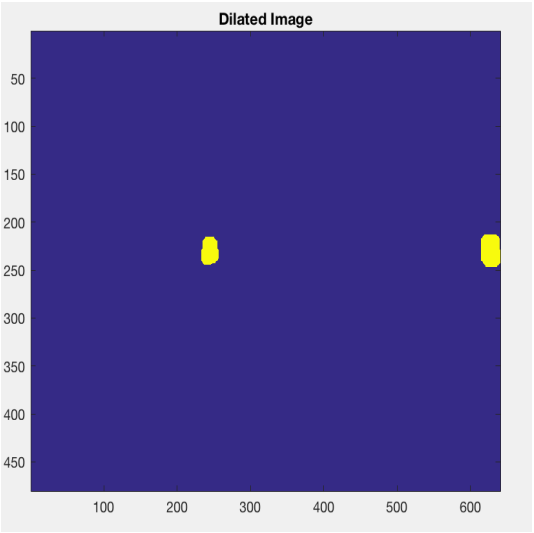

Figure 4: Dilated Red like objects

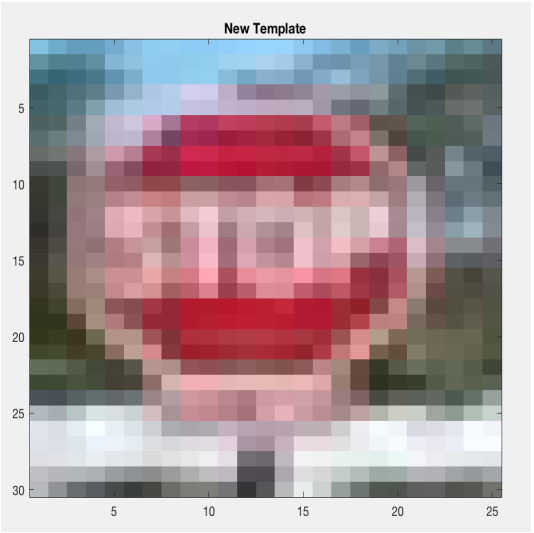

Figure 5: Template created with the help of bounding box

might just work for the selected set of images and may not work for other set of images.

2. The biggest challenge in this project is extracting red like objects from the images.Some images show STOP signs in dark red color and others just show signs in light color because of various lighting conditions. Hence,for the purpose of this project, the focus is just shifted to extracting red like object in the image. Sometimes, other objects such as advertisement hording also show similar red color objects but they have been taken care of in the next phase using area property.

3. The two templates created are just from the dataset [3] considered for the purpose of this project. But, if these template is used for other datasets, then there are high chances that the STOP sign may not be detected.

4. Finally, to improve the accuracy of the implementation, the Optical Character Recognition is used. The OCR technique has some of the issues such as, if the STOP sign is not visible properly then it will not detect it. Hence, the surrounding noise is been removed and STOP word is properly highlighted with the help of OCR. The accuracy produced by this technique is highly dependent on the dataset used for the detection.

5. Some of the images showed two STOP signs, but as we are just interested in finding at least one of the STOP sign then the another signs are ignored.

6. One of the challenge was creating a SVM classifier and its input parameters. As we had the limited understanding of the SVM classifier, the results could not be improved.

6. LEARNING LESSONS

Each task has some learning lessons. In this project, SVM classifier was a huge learning lesson. The inputs for the SVM classifier was a challenge and it did not worked out well. Hence, the focus shifted to using some other techniques. In computer vision, the problem can be solved by applying a combination of several techniques.The detection of the STOP sign is just not dependent on extracting red like objects, it should posses a combination of various other techniques as well. The tasks performed so far worked out for this dataset but may not work for other datasets.

7. FUTURE WORK

In future, various other techniques such as polygon detection and controlling other lighting conditions can be applied to improve the accuracy. If this system reaches at a particular accuracy such as approx 95 % then video processing can be used to detect the STOP signs in video frames. This system is then ready to go for real time STOP sign detection.The live demonstration of this system may need detecting STOP sign in small time frame. To reduce the detection sign already trained classifier such as SVM can be applied in near future.

8. REFERENCES

[1] Url for ocr function. `https://www.mathworks.com/help/vision/ref/ocr.html`.

[2] Url for stop sign detector toolbox. `https://www.mathworks.com/help/vision/ug/train-a-stop-sign-detector.html`.

[3] M. M. T. Andreas Mogelmose and T. B. Moeslund. Vision based traffic sign detection and analysis for intelligent driver assistance systems: Perspective and survey. *IEEE Transactions on Intelligent Transportation Systems*, 2012.

[4] S. C. C. Fang and C. Fuh. Road-sign detection and tracking. *IEEE Trans. on Vehicular Technology*, 52:1329–1341, 2003.

[5] H. Fleyeh. Traffic sign database.

[6] G. D. G. Adorni, V. Dandrea and M. Mordoni.

[7] A. Z. G. Loy. Fast radial symmetry for detecting

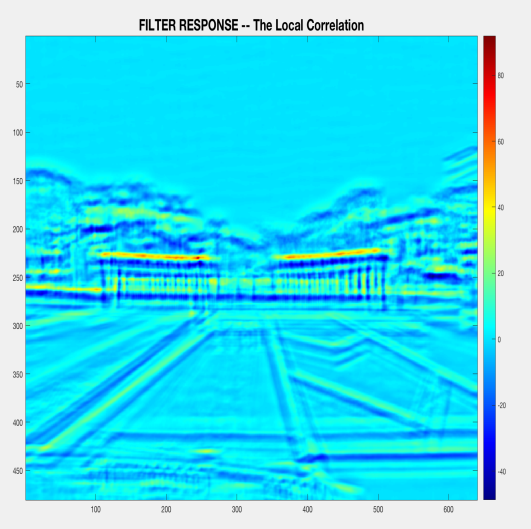

Figure 7: Filter Response when performed local correlation

Figure 6: Local average of input image

```
function RecognizeCharacters(im, ROI)
    Icorrected = imtophat(im(:,:,2), strel('disk', 15));

    marker = imerode(Icorrected, strel('line',10,0));
    Iclean = imreconstruct(marker, Icorrected);

    BW = imbinarize(Iclean);

    figure;
    imshow(BW);
    results = ocr(BW, ROI,  'TextLayout', 'Block');

    wordBox = locateText(results, 'STOP', 'UseRegexp', true);
    word = regexp(results.Text, 'STOP', 'match');
    if ~isempty(word)
        Iname = insertObjectAnnotation(im,  'rectangle', wordBox, word, ...
            'LineWidth', 1, 'Color', 'yellow', 'TextColor', 'black');
        figure;
        imshow(Iname);
        title('Sign Text');
    end


end
```

Figure 8: Optical Character Recognition function

points of interest. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25:959–973, 2003.

[8] T. K. J. Miura and Y. Shirai. An active vision system for real-time traffic sign recognition. *IEEE Intelligent Transportation Systems*, pages 52–57, 2000.

[9] M. A. S. M. A. Garcia-Garrido and E. Martin-Gorostiza. Fast traffic sign detection and recognition under changing lighting conditions. *IEEE Intelligent Transportation Systems Conference*, pages 811–816, 2006.

[10] N. Otsu. A threshold selection method from gray level histogram. *IEEE Trans. Syst.ManCybern*, SMC-9:52–66, 1979.

[11] H. F. R. Biswas and M. Mostakim. Detection and classification of speed limit traffic signs. *World Congress on Computer Applications and Information Systems (WCCAIS)*, pages 1–6, 2014.

[12] P. Yakimov and V. Fursov. Traffic signs detection and tracking using modified hough transformg. *12th International Joint Conference on e-Business and Telecommunications (ICETE)*, pages 22–28, 2015.

APPENDIX

A. USER INSTRUCTIONS

Following are the steps required to run the application.

1. Extract the contents from the zip file.
2. Environment: MATLAB R2017a.
3. Run the MATLAB function using command Kamble_Chaitali_Project <filenameofImage>.
4. The function will prompt you to either enter 'y' or 'n' for running the function in either of two ways (in built tool box or separate implementation). Enter the appropriate option as per your choice.
5. The output image will be displayed on the screen with highlighted STOP sign.