ENPM 673, Robotics Perception
Project 2: Traffic Sign Recognition.
Due on: 11:59:59PM on Monday – April 3, 2017

Prof. Yiannis Aloimonos

1 Traffic Sign Recognition - 100Pts

In this project we aim to do *Traffic Sign Recognition* which helps to take necessary actions for an *Autonomous Driving Car*.

2 Details

Traffic Sign Recognition can be staged into two sections *Traffic Sign Detection* and *Traffic Sign Classification*. This is one way to perceive the recognition pipeline. In Detection stage we aim to extract possible candidates/regions which contain some traffic sign (we do not care what the sign might be). In Classification stage, we go over each Region of Interest (RoI) and identify what sign it represents i.e., given that we know a set of traffic signs we are now classifying what this specific RoI represents.

Note: You are given images from a driving car, training/testing images for various signs and the output should be a video submission - DATASET.

2.1 Traffic Sign Detection - 50 Pts

Needless to say, there are a lot of ways this can be done. Let's see some of the common pipelines. Traffic Signs can be classified into two categories **RED** (Danger & Prohibitory) and **BLUE** (Mandatory) based on color (this is just one way as the true classes are Danger, Prohibitory and Mandatory - based on shape, color and content of signs).

In this section the objective is to extract the Region of Interest (i.e., around traffic sign), you can try to do so with simple thresholding in appropriate color space. Below are two ideas, one is using simple thresholding and other is using a much robust method to identify regions of *similar* intensity.

YOU CAN USE ANY OTHER APPROACH as you feel comfortable, which might not be given here. In any case, PLEASE SUBMIT A REPORT explaining your approach with relevant outputs after each step.

2.1.1 Thresholding in HSV Color Space

Here the idea is, for any traffic sign, say red, it will be of a typical color composition. HSV Color Space serves better to identify appropriate bands for H, S, V channels to model the color composition of traffic sign. Also it isolates intensity/brightness (unlike RGB) which helps for illumination invariant sign detection.

1. Denoise the image - (Noise Removal).
2. Model/Threshold in HSV color space (Ref. [1], [2]) to extract possible blobs for traffic signs.
3. Analyze the properties for each blob (e.g., size, aspect-ratio) to determine if it corresponds to traffic sign.
4. Extract the bounding box. Make sure the bounding box covers entire sign, if it doesn't then build an algorithm to stretch/shrink depending on blob/traffic-sign properties. The cues used for modifying bounding box SHOULD NOT BE hardcoded for any particular sign.

2.1.2 Using Maximally Stable Extremal Regions - MSER algorithm

Here we will not be discussing how MSER works, a trivial intuition is MSER gives regions of similar intensity given a grayscale image (Ref. [3]), and we know that traffic sign is mostly a *uniform intensity region* be it red or blue.

1. Denoise the image - (Noise Removal).
2. You might want to do contrast normalization over each channel (as mentioned in 3A of [3]) - (stretchlim)
3. Create appropriate grayscale image that best highlights the sign that you want to detect. Say, for red sign if we consider RGB color space, you can simply use Red channel as the grayscale image or use weighted combination of R,G,B channels to create grayscale image.

   Same goes for blue signs and **do experiment with other color spaces as well**. Objective is, say to detect red sign the generated grayscale image should be brighter (towards white) in the red region and darker elsewhere.
4. Normalize intensity of the image (Ref. [3])

$$C' = \frac{C}{R+G+B} \qquad \text{(simple method)}$$

$$C' = max(0, \frac{min(R-B, R-G)}{R+G+B}) \qquad \text{(useful for red signs)}$$

$$C' = max(0, \frac{B-R}{R+G+B}) \qquad \text{(useful for blue signs)}$$

5. Extract MSER regions from the image, this is the key part where you have to experiment with various parameters of MSER region extractor (**must refer** - 4.1). Your aim is to extract all the true sign regions while minimizing false positive regions.

6. Fit a bounding box to the sign, you might have to tweak the bounding box i.e., in case if MSER identifies an inner portion of a particular sign then you might want to scale up the bounding box so that it covers the whole sign. Build an algorithm to SCALE ONLY IF NECESSARY - probably using blob properties (e.g., size, aspect ratio).

Remember you need to create video of atleast 60 seconds duration at 30 fps i.e., your algorithm should work for atleast ∼1800 frames. So DO NOT hardcode parameters for specific frames.

2.2 Traffic Sign Classification - 50 Pts

You are given sample images for different signs, you can resize the image to a standard size say (64×64) and extract HOG features. Once you get the HOG features, train a multi-class SVM for various signs. You can test the classifier performance using test data (**must refer** - 4.2). This is independent of Traffic Sign Detection section.

To achieve a complete Traffic Sign Recognition pipeline, you can do one of the following

- Find a closest square to the bounding box extracted from Sign Detection stage, resize it to a standard size (used in Traffic Sign Classification say 64×64), extract HOG features and predict the sign using the trained SVM model.

- Fix an appropriate square size (depending on size of bounding box) , sweep the square over the bounding box and for each position of square over bounding box do previous step. For clear details look up 'PyImageSearch Blog' in 4.2

HOG feature is usually of very high dimension, though you can use them directly for SVM classification (Ref. [3]), you can also do dimensionality reduction over these feature vectors e.g., using PCA, LDA (Ref. [4]).

Note: You need to highlight the bounding box of traffic sign and paste the appropriate sign (from one of the sample training images) beside it. Traffic Signs to be detected are {45, 21, 38, 35, 17, 1, 14, 19}.

(a) 45 **(b)** 21 **(c)** 38 **(d)** 35

(e) 17 **(f)** 1 **(g)** 14 **(h)** 19

Figure 1: Different Traffic Signs

2.3 Few Other Tricks

In Traffic Sign Detection, the size of bounding box of detected sign can reveal information about the possible location of the sign i.e., if the bounding box is quite small then it could be that the sign is really far away then it should be closer to the horizon (or top) in the image. Else if it is bigger then it should be closer to the center line of the image. You can also observe that the traffic sign might not be appearing in, say, lower one-third of the image. There could many such cues from the images that you can exploit to remove false positives in Traffic Sign Detection.

Traffic Sign Classification is relatively straight forward pipeline and you might not need any tricks apart from experimenting with parameters. In case the HOG features are not robust to illumination in the given data set, you can try the normalization approach from [3] which is as follows you can find out the median of intensities and map it to 128 i.e., for red channel if the median is 80 then you can fit a linear mapping from 0-80 to 0-128 as follows $\frac{128*x}{80}$ and another linear mapping from 80-255 to 128-255 as follows $255 - \frac{(255-x)*127}{(255-80)}$.

3 Submission Guidelines

Your submission **SHOULD be a ZIP folder** (no other file extensions) with the naming convention `YourDirectoryID_proj2.zip` on to ELMS/Canvas. Additionally follow the guidelines given below:

1. You will have a parent directory `P2_Submission`.

2. Under `P2_Submission/TSR` you will have three sub-folders `code`, `input` and `output`.

3. You should also submit a report (`Report.pdf`) under `P2_Submission/TSR` folder.

Your output video should be *atleast* `60 seconds` of duration with 30 fps frame rate.

4 Useful Resources

Traffic Sign Recognition has become widely popular mainly because of German Traffic Sign Detection Challenge. Also there are many example scripts from MATLAB.

- You can check for any pipeline with expected results from here. Google for '<team_name > traffic sign detection' for relevant paper.

4.1 MSER Tutorials

- MATLAB tutorial for Text Recognition - link.
- MATLAB documentation for `detectMSERFeatures` - link.
- VLFeat Tutorial - link.
- MSER Theory Slides - link.

4.2 HOG - SVM

- MATLAB tutorial for Digit Classification using HOG features - link.
- VLFeat Tutorial - link.
- PyImageSearch blog - link.

NOTE: You can access research papers using an extension from UMD Library Database Finder. E.g., for IEEE Xplore paper, search for 'IEEE Xplore'

5 Acknowledgement

Traffic Sign Dataset used in this project is by courtesy of Radu Timofte from ETH-Zurich Vision Lab (Ref. [4]).

6 Collaboration Policy

You are allowed to discuss the ideas with fellow students, but you need to give credits in the report. But the code you turn-in should be your own and if you **DO USE** (try not to and it is not permitted) other external codes - do cite them and you might get partial credits. For other honor code refer to the University of Maryland Honor Pledge.

DISCLAIMER: *Details provided in the project's pipeline might not be fully complete, but in any case we might keep adding additional information as we find something relevant. You should take the effort to search online for any help regarding function usage, however, any concept related queries can be discussed in TA Office Hours.*

References

[1] Kishan S Athrey, Bharat M Kambalur, and K Krishna Kumar. Traffic sign recognition using blob analysis and template matching. In *Proceedings of the Sixth International Conference on Computer and Communication Technology 2015*, pages 219–222. ACM, 2015.

[2] Saturnino Maldonado-Bascon, Sergio Lafuente-Arroyo, Pedro Gil-Jimenez, Hilario Gomez-Moreno, and Francisco López-Ferreras. Road-sign detection and recognition based on support vector machines. *IEEE transactions on intelligent transportation systems*, 8(2):264–278, 2007.

[3] Samuele Salti, Alioscia Petrelli, Federico Tombari, Nicola Fioraio, and Luigi Di Stefano. A traffic sign detection pipeline based on interest region extraction. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–7. IEEE, 2013.

[4] Markus Mathias, Radu Timofte, Rodrigo Benenson, and Luc Van Gool. Traffic sign recognition—how far are we from the solution? In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8. IEEE, 2013.