ENPM673 - Perception for Autonomous Robots
Project 6 - Traffic Sign Recognition

Abstract

Traffic Sign Recognition is vital part of especially the autonomous car industry which leverages various image processing techniques to detect and recognize traffic signs. These traffic signs are usually analyzed using front-facing camera on the car which is the data set given. The traffic sign recognition can done in two stages as detection and classification.

1. Data Preparation

- From the given image data sequence, we first need to remove noise and smoothen the images.
- For this we tried common image smoothing techniques like Gaussian Blurring, Median Blurring which were good only in removing small quantities of noises.
- We used the OpenCV’s inbuilt fastNlMeansDenoisingColored() function to denoise the color images.
- The filter strength parameter h was set to be 10 as higher values removes better noise but loses more details of the images.

2. Traffic Sign Detection

In the traffic sign detection phase, we crop the regions of interest and given to the classifier for signal classification.

- The traffic sign boards are either in RED (Danger, Prohibitory) or BLUE (Mandatory). To get the necessary regions of interest, color segmentation would be the easiest tool to use and we will execute the following pipeline separately for red and blue signs separately.
- First, contrast is adjusted using normalization for which we stretch the image based on the lower and upper limit for each of the R,G,B channels and normalizing the contrast for the red and blue channels.

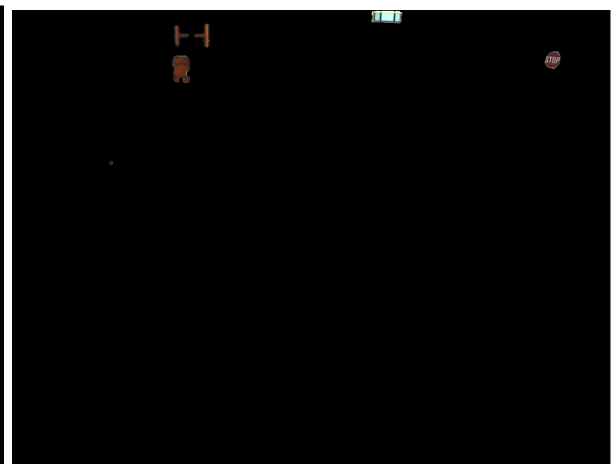
Contrast Normalized Blue ROI

Contrast Normalized Red ROI

2.1 MSER regions

- It was difficult to choose the thresholds for just the HSV approach owing to the brightness of the captured frames.
- The Maximally stable extremal regions (MSER) approach was seen to easily regard any flat areas like walls, posters, headlights as traffic signs as well. Hence a combination of the above two methods was chosen.
- We detect the MSER regions from the previously grayscale contrast normalized images with corresponding channel highlighted for each of Red and Blue signs.
- For extracting the MSER regions from the image, we experimented with various parameters of the MSER region extractor and finally arrived at the following parameters for the pipeline.
- _delta = 4 . For lower delta, we got lot of unwanted regions as well as noise and higher gave very less regions. This low delta was good enough to pick up even far away regions.
- _min_diversity = 0.8
- _max_variation = 0.2

MSER masked Blue ROI



MSER masked Red ROI

2.3 HSV Threshold

- There is a still a lot of noise which can be removed by combining the above obtained binary images with the HSV color thresholded binary images.
- The color thresholding range is selected such that the saturation and value such that only high intensity or bright colors are left off. The combined images have very less noise and accurate sign detected area.

HSV thresholded Blue ROI

HSV thresholded Red ROI

2.4 Bounding Boxes

- The traffic signs detected from above are marked by drawing bounding boxes on them and also cropped to be used for classification phase.
- Even after above steps, there was always some noise left in the image. We used further steps to further tweak and better the detection phase.
- An aspect ratio filter of 0.9 to 2.0 is enforced such that any blob more than this will be avoided. A minimum area limit of 1000 upto 30,000 to detect the proper bounding boxes.

Blue sign detected

Red sign detected

3. Traffic Sign Classification

With the detected traffic sign, we use a classifier to recognize what sign is present.

3.1 HOG-SVM Training

- For feature detection, Histogram of oriented gradients (HOG) was used for object detection which counts occurrences of gradient orientation in localized image portions.
- With the HOG features, we first train a Multivariate Support Vector Machine (SVM) where each of the feature has a label such that the classifier can identify which sign the HOG belongs to.
- The HOG pixel values are converted to a single column and used as input for SVM. This is repeated for all images in the training set to account for different viewing angles and orientations to better classify the signs.
- The training was tuned such that test phase rate as below.
    - Testing with training image - 98%
    - Testing with test image - 94%
- After the training data is obtained, we classify the different signs using SVM to predict the traffic sign class.

- Finally, after the sign is classified, we place the label/id of the detected sign beside the bounding box.

Sign `45` Classified

3.2 Tuning the SVM parameters:

C parameter: The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyperplane if that hyperplane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyperplane, even if that hyperplane mis-classifies more points.

Our analysis showed us that
The cost or penalty parameter C is responsible for hard / soft margins in SVM.
Too high C = overfitting possible in the training set.
Too low C = over generalisation / underfitting possible

Gamma: Gamma is the variance term
A small gamma means a Gaussian with a large variance

Also during the training we got a low training set error but high validation error.This was due to the high variance of the classifier and we reduced the gamma in those cases.

Nu: The parameter nu is the upper bound on the fraction of margin errors and lower bound of fraction of support vectors and given by nu = A+B/C where A and B are constants.

Furthermore, the tuned values for the proposed pipeline used are as follows

Poly degree = 2.5 C = 12.5 gamma = 0.03062

Code Download:
https://drive.google.com/drive/folders/1LrOhg105uIXVOgv6t_PSjwWlKS5g2V_p?usp=sharing

How to run code

1. Unzip the folder which has the code, input images.
2. Copy the TSR images folder such that the *.py files are in the same directory level.
3. Run ` denoising.py ` to prepare the images.
4. Each of the following code parts needs to be separately run.
5. For Traffic Sign Recognition proposed pipeline run
   ` python3 MSER_main.py `

Output Video:
https://drive.google.com/drive/folders/1LrOhg105uIXVOgv6t_PSjwWlKS5g2V_p?usp=sharing

Project Team

1. Nantha Kumar Sunder (UID: 116104777)
2. Nithish Kumar (UID: 116316958)
3. Rama Prashanth (UID: 116428941)

References

1. Computer Vision, A Modern Approach, Forsyth and Ponce , http://cmuems.com/excap/readings/forsyth-ponce-computer-vision-a-modern-approach.pdf
2. MSER parameters https://es.mathworks.com/help/vision/ref/detectmserfeatures.html

3. SVM Tuning https://link.springer.com/content/pdf/10.1007%2F978-3-642-22726-4_36.pdf
4. https://prateekvjoshi.com/2015/12/15/how-to-compute-confidence-measure-for-svm-classifiers/