Abstract

With the advent of self-driving cars and autonomous delivery robots, identification of street name signs and traffic signs has become a problem of paramount importance. In this report, we propose a lucid approach to tackle this problem. The proposed methodology is based on Dynamic Thresholding to recognize street signs effectively under varying conditions of illumination. The thresholding, however, is not performed on the RGB image space since the algorithm should be robust and invariant to factors such as illumination. For this reason, we choose the HLS (Hue,Luminance,Saturation) image space as our primary workspace and implement a robust algorithm that is reasonably invariant to illumination. Further, the thresholded images are made more accurate by employing strategic noise reduction techniques. This enables us to use a simple bounding box algorithm to extract the Region of Interest (RoI) after which a Convolutional Neural Network (CNN) is utilized to classify the traffic signs into pre-defined classes such as stop signs, speed limit signs, height clearance, etc.

The algorithms are then tested on videos generated in a highly dynamic environment with a significant amount of artificially induced noise such as camera shake. The videos were captured by us on the streets of New York City, and the algorithm is proven to be robust to videos with no staging or environment setup, to simulate its performance in the major target application areas such as autonomous vehicles and autonomous ground robots.

Contents

List of Figures

Chapter 1

Introduction

Street and traffic signs provide critical guidance information to both drivers and autonomous vehicles alike. Recognition of these signs, especially in large cities, is complicated due to the cluttered environment. Moreover, the inaccuracy and unavailability of GPS in some regions necessitates another method for positioning and guidance.

In big cities, GPS data is annotated with traffic and street signs, whereas in semi-urban and rural areas this is not the case and hence, guidance solely depending on GPS is inefficient and could potentially be dangerous in certain situations. On the other end of the spectrum, GPS itself is fallible in cities with tall buildings like New York due to a phenomenon known as ”Urban Canyons”, where the high-rise buildings shield the Line of Sight (LoS) to tracking satellites, making position triangulation highly sporadic.

Skyscrapers also pose another major problem with positioning using GPS, known as the 'Multi-path effect', in which the signals reflect off the surfaces of these tall structures causing delayed measurements and inaccuracy in positioning due to changes in the apparent LoS. Currently, autonomous vehicles use vision-based systems solely for the identification of traffic signs like stop signs or speed limit boards. However, due to the limitations faced by GPS as described above, vision-based systems could be incorporated into the positioning aspect as well, by detecting street name signs for increased accuracy.

Another potential application lies in positioning for autonomous ground robots such as delivery bots. With increasing levels of traffic and congestion in large cities, e-retailers and delivery services like Amazon and FedEx are heading towards autonomous wheeled ground delivery bots for faster and logistically efficient deliveries. These wheeled bots travel on sidewalks instead of roads and identification of street names

and signs could provide the system with vital information in case of GPS malfunctions. Again, due to the problems faced by GPS in cities, the incorporation of a vision-based system for street and traffic sign identification would increase the accuracy of positioning for all such systems.

This report includes the literature review which consists of a summary of the existing methods and algorithms to tackle this problem. The identification method is chosen after critical deliberations over multiple methodologies and is proposed in this report and is backed by strong test cases. It also delineates noise reduction techniques employed such as area filtering in Sec.3.2. The classification process using Convolutional Neural Networks (CNN) is explained in Sec.3.3. The report also discusses the results and analysis of the proposed algorithm and is concluded with scope for future work in this regard.

Chapter 2

Literature Review

2.1 Existing Approaches

Tackling the problem of detecting street and traffic signs has been a topic of interest in recent times. Methodologies such as a dynamic optimized Hue, Saturation, Values (HSV) sub-space thresholding, based on S and V channel reference values is proposed in [1]. Colour segmentation is achieved by applying standard HSV colour filtering, generating sub-images to calculate seed pixels, and aggregating pixels depending on the seed saturation values by applying a region growing algorithm. Shape detection is achieved by using similarity coefficients between the segmented region and the sample images for road signs. A segmentation rate of 94.6% for red circular signs, 86.3% for red triangular signs, and 95.7% for blue circular signs is achieved.

Segmentation using this method is performed by introducing a dynamic threshold in the pixel aggregation process on HSV colour space. The main purpose of dynamic thresholding is to reduce hue instability in real scenes depending on external brightness variation. They used a neural network to classify the candidate sign regions according to the information inside it. Classification is carried out by a feed-forward neural network classifier, where a 36×36 pixels candidate is fed to the neural network input. A classification rate of 84% for red circular signs, 88% for red triangular signs, and 100% for blue circular signs were achieved respectively.

A comprehensive approach to the recognition of traffic signs from video input is proposed in [2]. A trained attentive classifier cascade is used to scan the scene in order to quickly establish regions of interest (ROI). Sign candidates within the ROIs are captured by detecting the instances of equiangular polygons using a Hough Transform-style shape detector. To ensure a stable tracking of the likely traffic signs, especially in a clut-

tered background, they propose a Pixel Relevance Model, where the pixel relevance is defined as a confidence measure for a pixel being part of a sign's contour. The relevance of the hypothesized contour pixels is updated dynamically within a small search region maintained by a Kalman Filter, which ensures faster computation. Gradient magnitude is used as a piece of observable evidence for this update process. In the classification stage, a temporally integrated template matching technique based on the class-specific discriminative local region representation of an image is adopted. They have evaluated the proposed approach on a large database of 135 traffic signs and numerous real traffic video sequences. A Recognition accuracy of over 93% in near real-time has been achieved.

Use of Convolutional Networks (ConvNets) for the task of traffic sign classification is a common and effective approach as proposed in [3]. ConvNets are biologically-inspired multi-stage architectures that automatically learn hierarchies of invariant features. While many vision approaches use handcrafted features such as HOG or SIFT, ConvNets learn features at every level from data that are tuned to the task at hand. The ConvNet architecture was modified by feeding 1st stage features in addition to 2nd stage features to the classifier. The system yielded an accuracy of 98.97% compared to the human performance of 98.81%, using 32×32 color input images and produced an accuracy of 99.17% for grayscale images instead of color. Interestingly, random features also yielded a 97.33% accuracy.

2.2 Drawbacks of Existing Approaches

- Colour based thresholding in the sRGB space poses many problems, the most significant of which being the absence of invariance to ambient conditions such as illumination and saturation produced by standard camera pre-processing algorithms. Using thresholding over RGB space will not serve the purpose of detecting signs from heavy noise carrying video sequences captured in highly dynamic environments.

- Use of HSV space does yield better results compared to RGB space thresholding in terms of being invariant to saturation values in frames. However, this method is still not invariant to illumination changes.

- Neural networks are very robust in detecting and classifying road signs but they are not quite as accurate when it comes to identifying street name signs due to the lack of an existing training dataset.

- Creating a specific dataset for this purpose is an arduous and computationally expensive task with a small reward to effort ratio.
- Shape-based template matching is a proposed method and can be introduced to improve accuracy. However, shape-based matching serves no purpose due to a large number of potential false positives such as shop name boards. Usage of this on top of dynamic thresholding, on the other hand, becomes highly redundant.
- Using feature extractors such as SIFT, SURF, FAST, etc. are not optimal due to the unavailability of specific standard image sets to generate putative matches and identify RoIs. These algorithms are also computationally more expensive compared to our proposed method.

After considering these methods and deliberating over their commensurate advantages and disadvantages, we decide to implement the approach presented in Sec.3. The proposed system was found to be within acceptable detection probabilities, with far less computational intensity thereby striking a balance between accuracy and amount of computation required.

Chapter 3

Methodology

3.1 Dynamic Thresholding

3.1.1 The HSL Image Space

Separating the colour from brightness information is very difficult in an RGB image. Instead, the HLS (Hue, Luminance, Saturation) colour space can be used(HSL and HLS used interchangeably). Here, the colour information (Hue and Saturation) can be separated from the overall brightness intensity values for the image, thus making it more robust to variation in lighting conditions. This makes it useful for colour segmentation.

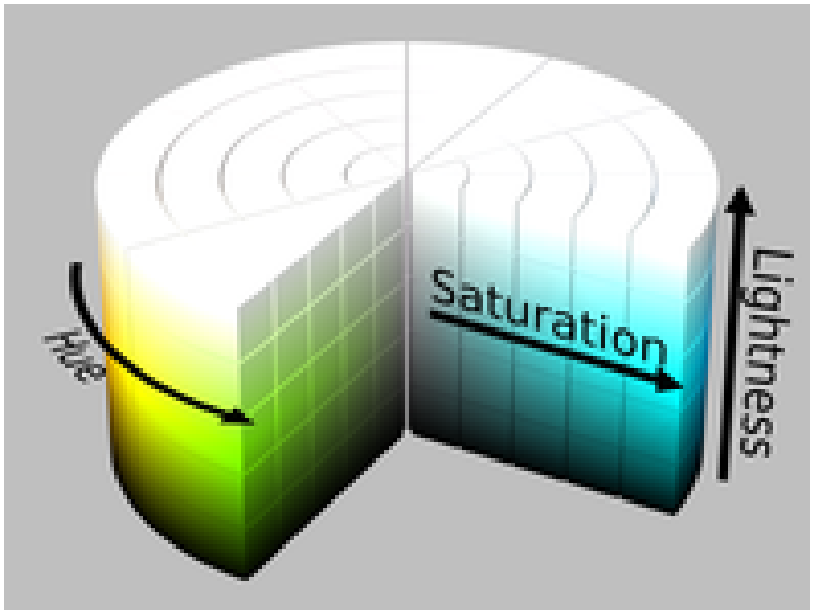

Figure 3.1: HSL Colour Space

In the HLS colour space, the Hue component is used to describe the colour and is represented by angles between $[0°, 360°]$. The saturation denotes the distance from the

center of the Hue wheel (Fig.3.2) and ranges between 0 and 1. Maximum saturation of 1 is obtained at the border of the hue circle and the minimum value 0 represents the center of the circle. Luminance/Intensity values range between $[0, 1]$. Here, black is represented by 0 and white is represented by 1. The RGB to HLS conversion is as follows:

$$V_{max} = max(R, G, B) \tag{3.1}$$

$$V_{min} = min(R, G, B) \tag{3.2}$$

$$L = (V_{max} + V_{min})/2 \tag{3.3}$$

$$S = \begin{cases} (V_{max} - V_{min})/(V_{max} + V_{min}) & if\ L < 0.5 \\ (V_{max} - V_{min})/[2 - (V_{max} + V_{min})] & if\ L \geq 0.5 \end{cases} \tag{3.4}$$

$$H = \begin{cases} 60(G - B)/(V_{max} - V_{min}) & if\ V_{max} = R \\ 120 + 60(B - R)/(V_{max} - V_{min}) & if\ V_{max} = G \\ 240 + 60(R - G)/(V_{max} - V_{min}) & if\ V_{max} = B \end{cases} \tag{3.5}$$

$$If\ H < 0\ then\ H = H + 360. \tag{3.6}$$

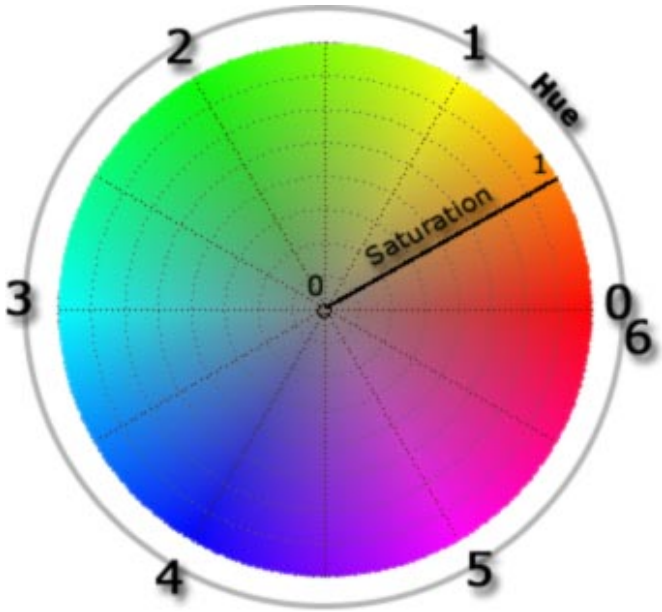

Figure 3.2: Hue Circle

3.1.2 HSL based Dynamic Thresholding

A thresholding algorithm is needed in order to segment a street sign from the background in a given image. Variation in lighting conditions must be considered while designing a system that will be capable of performing such segmentation. For calculating the threshold under such cases, the following approach can be used.[4]

Convert the RGB image into HLS colour space. Normalize the Hue, Saturation, and Luminance channels to $[0, 255]$. Calculate the normalized global mean $(Nmean)$ for the image in the L channel (Illumination intensity channel). The reference colour and the unknown colour can be represented as vectors in the hue circle (Figure 3). In the hue circle, the hue values are represented as angles and saturation values correspond to the radius (or the length of the vector) for a given hue angle.

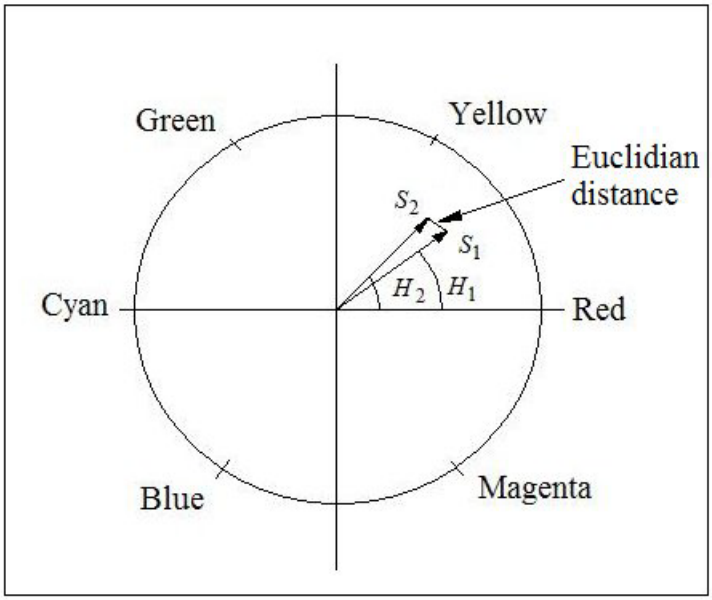

Figure 3.3: Vector representation of colours on Hue circle

A pixel having hue and saturation values as $H1$ and $S1$ in a given image can be represented as a vector. Similarly, the reference colour for the street sign can be represented by a vector having hue and saturation values as $H2$ and $S2$. The distance between these 2 vectors on this circle can be used as a metric for evaluating the similarity between the 2 colours. The Euclidean distance between the 2 vectors will be:

$$d = [(S2\cos(H2) – S1\cos(H1))^2 – (S2\sin(H2) – S1\sin(H1))^2]^{1/2} \tag{3.7}$$

For different values of illumination intensities, the hue and saturation values change

as seen in Fig.3.1. In case the illumination for an image is high, even if the image has a street sign, their corresponding hue and saturation values will be high. The threshold must take this into account. It can be calculated using the normalized global mean.

$$threshold = e^{(-Nmean)} \tag{3.8}$$

The street signs are designed in a way that they are easily recognizable in most cases. In the case of high illumination, the threshold is lower because the normalized global mean is lesser. Thus, the globalized mean will be related to the Euclidean distance between 2 vectors and this will allow the luminance of the image to determine the relationship between an unknown and reference colour. $e^{(-Nmean)}$ is used because the colours are easily distinguishable when they are present near the bottom of the HLS space representation, as seen in Fig.3.1. If the luminance of the image is high, and a threshold of $e^{(Nmean)}$ is used instead, results of segmentation are poor.

To check if the colour of a pixel in the given image is similar to the reference colour, the algorithm will compare the Euclidean distance with the threshold for all the pixels in the image. If the Euclidean distance is lesser than the threshold, the pixel will be considered as an object pixel otherwise it would be categorized as background. The output will be a binary image containing pixels with a colour similar to the reference colour.

3.2 Noise Reduction

Once the image has been thresholded using the proposed algorithm, the output still contains multiple instances of salt and pepper noise. To remove this noise, we first take the negative image of the binarized thresholding output image, just as a convention. After this, we employ area filtering to get rid of the noise in a very computationally cheap manner.

To implement this technique, we make certain key assumptions. We first assume that in the binary image, the area of noisy regions is always less than the detected signboards, indicating the robust performance of the thresholding algorithm. This assumption is corroborated by examining the thresholding algorithm over a set of test images extracted at random from the test videos. The second key assumption made is that, at any given point, our system needs to read up to 3 boards at a time to have sufficient information for triangulating its position accurately.

With these assumptions in place, we start the area based filtering process. Initially, we start a pixel-wise search and start counting the pixels, if there has been a change from 0 to 1 in pixel values. Recall, in the negative image of our binarized thresholding output, $1s$ correspond to regions of interest. The count is updated for every neighbouring pixel with a value of 1. The counting is stopped when there is a pixel value change from 1 to 0 and the net count is stored as the area of the corresponding object.

As we move down the rows, the area of any object whose upper neighbours have the same pixel values, are added to the area of the upper object and the area of the current object is initialized back to 0. By this technique, we end up with an array of regional areas corresponding to the RoI in our image, and making use of our assumptions, we only keep the regions whose area is within the 3 largest generated area values. The robustness of this algorithm is displayed in Fig.3.4

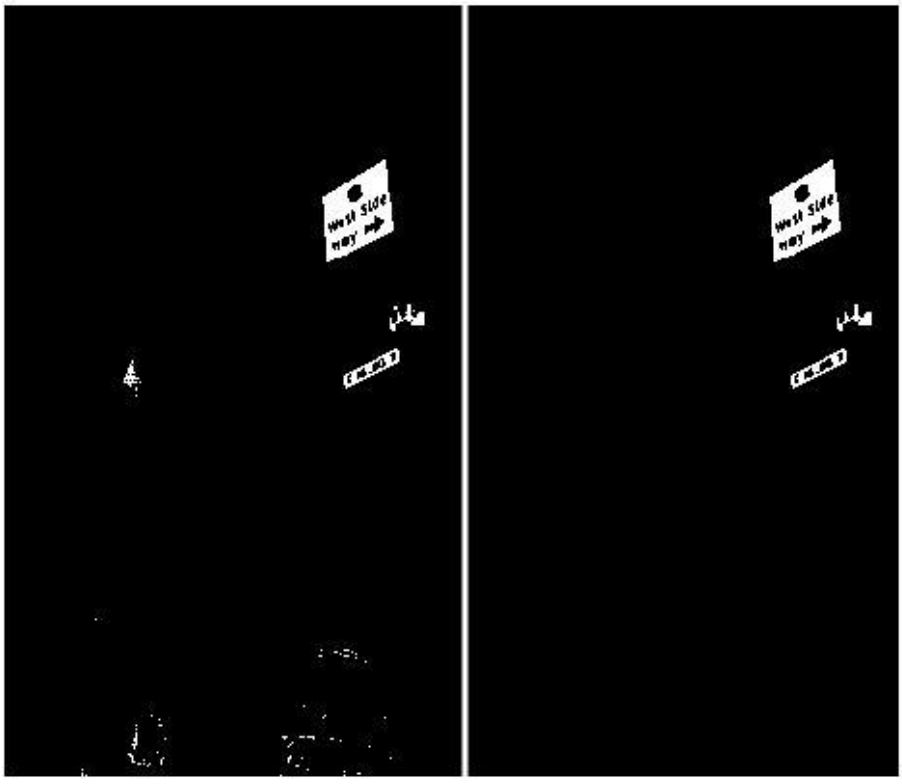

Figure 3.4: Dynamic thresholding in the HSL space for red and green

3.3 Classification of detected traffic signs

To classify detected traffic signs, we employ the use of CNNs, due their robustness in studying and learning image based features.

The most common architecture for image processing, using CNNs, is the VGG16 architecture shown in Fig.3.5. There is an input layer, followed by pairs of Convolution layers and Pooling layers stacked together, depending on how deep the desired model is. The final convolution layer is followed by a flattening layer, which is in turn followed by a series of fully connected layers that plug out the output nodes.

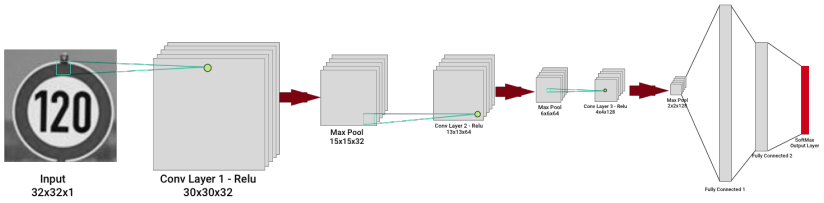

Figure 3.5: VGG16 model architecture

The convolution layers perform kernel convolutions on the input image, to extract certain properties of the image. For e.g., a Gaussian kernel performs image smoothing on being convoluted with the input image, a Sobel filter is sensitive to edges in the image and so on. Since the image is in the RGB format, there are 3 kernels, one per channel and the size of each kernel used is 3×3.

After each of the convolution layers, a pooling layer is added to group together features learned in that layer. Max Pooling is used to take into account the strongest of the learned features and reject the rest. This output is then passed as an input to the next convolution layer and the number of such layers depends on the depth of the desired model.

Each convolution layer may be associated with a padding that defines how the kernel is convoluted with corner and edge pixels, and an activation function that deals with altering the outputs such that the net output encompasses important learned features. It dies this through deciding which neurons fire in each layer and thereby controlling which weights are updated in each layer.

The net output of all the convolution layers however, is not flat since the image is a multi-channel matrix. For this reason, the output of the convolution layers is flattened before passing it as an input to the fully connected layers. These layers perform in exactly the same manner as a simple Deep Neural Network (DNN).

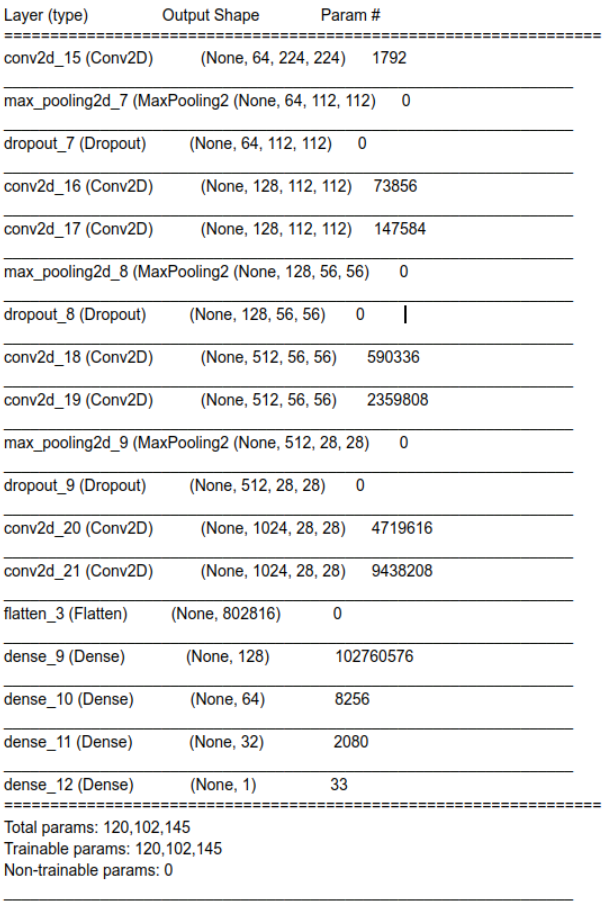

Figure 3.6: CNN model architecture

Simple DNNs are composed of an input layer, followed by stacks of fully connected, hidden layers of neurons that finally feed out to the output layer. Each input node is passed through each node of the first hidden layer and the corresponding weights extract the input feature vectors. The output of each node of the first hidden layer is fed as an input to each node of the second hidden layer and this process is repeated for each layer.

The weights are tracked and a bias may be introduced to lead the net output to converge to a particular vector. The output nodes are then back-propagated, depending on the error function associated with the output vector of the first pass. This permits efficient updating of weights making the model more accurate for applications such as clustering or classification.

When an optimal model has been trained, it can be tested with a different set of inputs and the accuracy of classification is generally commensurate to the training accuracy, since back-propagation is a strong weight optimization algorithm.

The major drawback with using simple DNNs however, is that training large data sets drastically affects the computation cost, since each input node is passed through all hidden layer nodes in the fully connected setup. This is where CNN has a comparitive advantage as it shares its weights, allowing it be much deeper, with the same number of tune-able parameters. It becomes evident that CNN stacks followed by fully connected layers perform better.

This CNN based model architecture, shown in Fig. 3.6 is robust to learning image features and the depth of the model ensures feature maps with higher level features learned. These higher level features are crucial when it comes to classifying images with features that are virtually indiscernible on a cursory glance. The lower level features help identify basic discrepancies that are equally important for classification.

Chapter 4

Results

To test the proposed methods, videos of the streets of New York were taken using a smartphone camera. Dynamic thresholding was performed in the HSL space with appropriate values for green and red, results of which are shown in Fig.4.1.

Further, for noise reduction, area filtering as explained above is executed on the thresholded image which is shown in Figs.4.2. 4.3 and 4.4 display thresholded, noise reduced images with bounding boxes over the street signs.

The computation details are as follows:

```
These are results based on running the algorithm over a 2.5GHz
Intel MLK CPU, with no GPU support, averaged over 440 frames
```

- Computation Time for Detection Algorithm $= \ 15ms/frame$
- Computation Time for Detection and Classification (CNN) Algorithms run successively $= \ 240ms/frame$

Figure 4.1: Dynamic thresholding in the HSL space for red and green

The classification results from the CNN model is shown in Fig.4.5.
The evolution of loss and accuracy of parameters is shown in Figs.4.6 and 4.7.

Figure 4.2: Noise reduction by area filtering

Figure 4.3: Bounding boxes over noise reduced street signs- Test Image

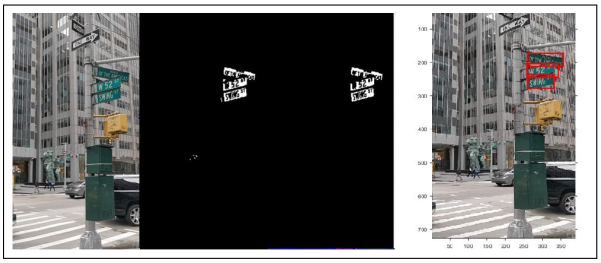

Figure 4.4: Bounding boxes over noise reduced street signs- Test Image

```
Train on 23525 samples, validate on 7842 samples
Epoch 1/10
23525/23525 [==============================] - 1240s 53ms/step - loss: 2.3880 - acc: 0.3497 - val_loss: 0.8779 - val_acc: 0.7339
Epoch 2/10
23525/23525 [==============================] - 1285s 55ms/step - loss: 0.8086 - acc: 0.7483 - val_loss: 0.2352 - val_acc: 0.9282
Epoch 3/10
23525/23525 [==============================] - 1224s 52ms/step - loss: 0.3228 - acc: 0.8983 - val_loss: 0.1036 - val_acc: 0.9686
Epoch 4/10
23525/23525 [==============================] - 1222s 52ms/step - loss: 0.1631 - acc: 0.9494 - val_loss: 0.0627 - val_acc: 0.9819
Epoch 5/10
23525/23525 [==============================] - 1221s 52ms/step - loss: 0.1022 - acc: 0.9680 - val_loss: 0.0526 - val_acc: 0.9853
Epoch 6/10
23525/23525 [==============================] - 1235s 53ms/step - loss: 0.0780 - acc: 0.9761 - val_loss: 0.0511 - val_acc: 0.9839
Epoch 7/10
23525/23525 [==============================] - 1222s 52ms/step - loss: 0.0601 - acc: 0.9821 - val_loss: 0.0411 - val_acc: 0.9878
Epoch 8/10
23525/23525 [==============================] - 1218s 52ms/step - loss: 0.0463 - acc: 0.9850 - val_loss: 0.0376 - val_acc: 0.9911
Epoch 9/10
23525/23525 [==============================] - 1222s 52ms/step - loss: 0.0408 - acc: 0.9870 - val_loss: 0.0325 - val_acc: 0.9913
Epoch 10/10
23525/23525 [==============================] - 1249s 53ms/step - loss: 0.0308 - acc: 0.9902 - val_loss: 0.0517 - val_acc: 0.9885
7842/7842 [==============================] - 76s 10ms/step
(AutoDrive) pratiks-MacBook-Pro:CV_ProjectGTSRB pratcr7$
```

Figure 4.5: CNN model results per Epoch

Figs.4.8, 4.9, 4.10, 4.11 and 4.12 show screenshots taken from the dynamic thresholding algorithm applied to test videos captured on the streets of New York City.

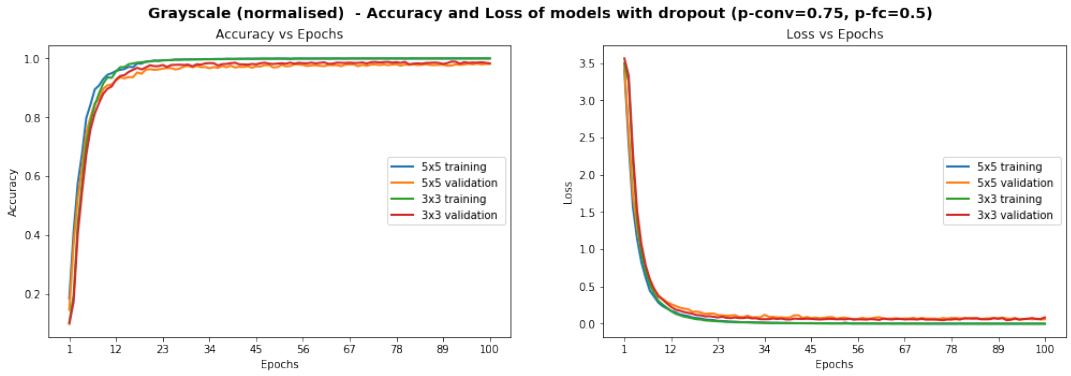

Figure 4.6: CNN model accuracy and loss values over number of Epochs

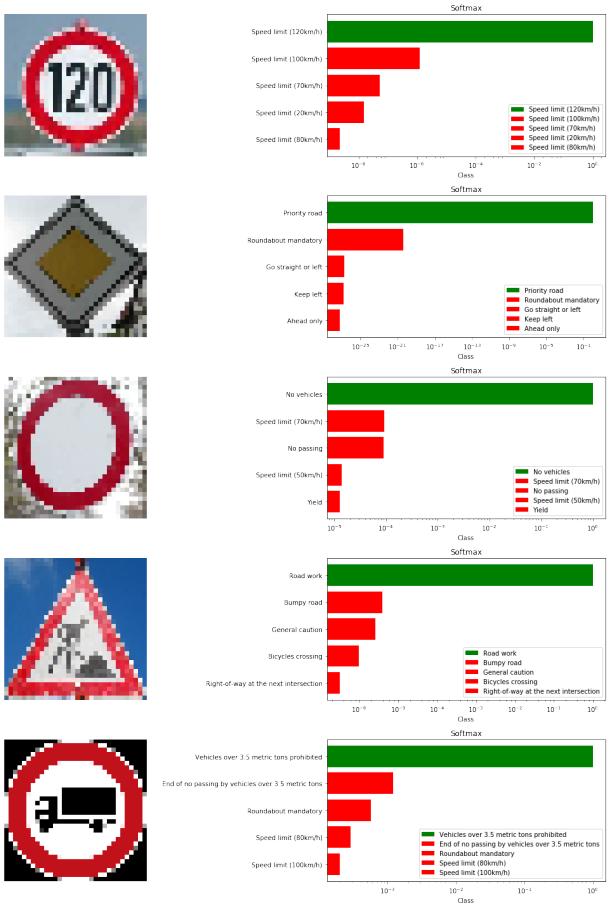

Figure 4.7: CNN model prediction results

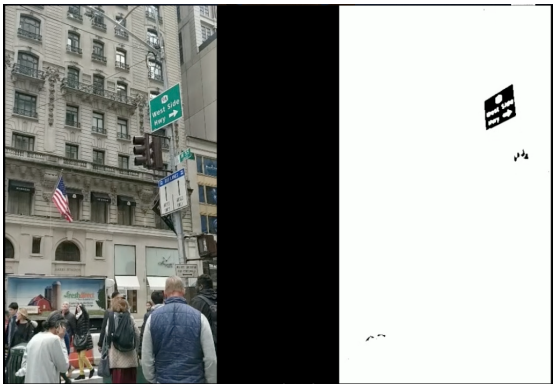

Figure 4.8: Screenshot from test video

Figure 4.9: Screenshot from test video

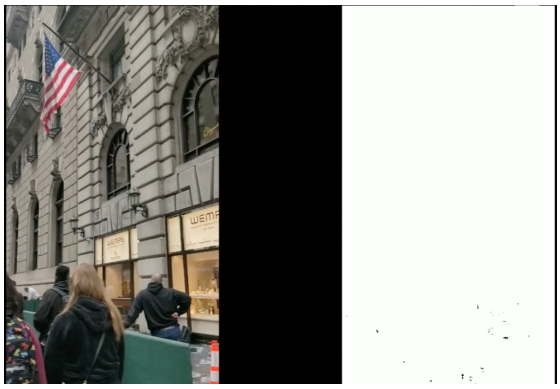

Figure 4.10: Screenshot from test video

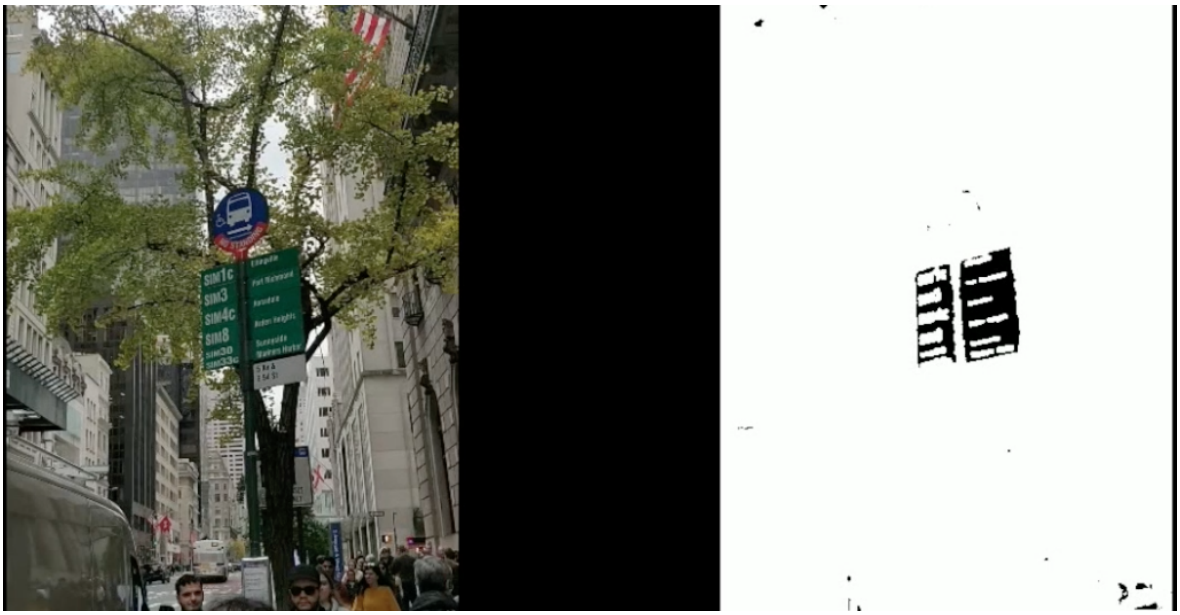

Figure 4.11: Screenshot from test video

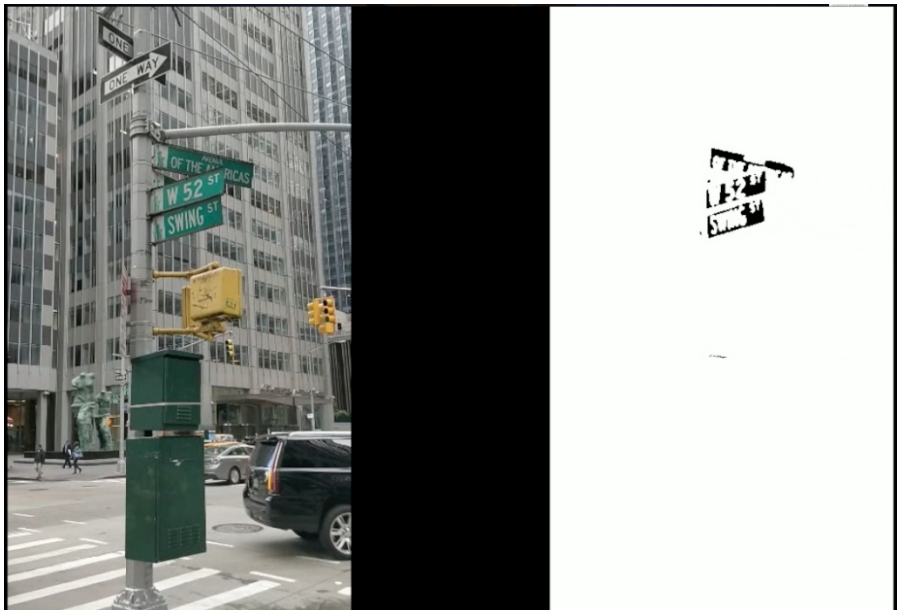

Figure 4.12: Screenshot from test video

Chapter 5

Analysis

The thresholding algorithm is found to be very robust to highly tricky scenarios involving multiple ambient factors and disturbances such as camera shake, people and vehicles moving across the frames in an uncontrolled manner, etc. This is quite accurately represented in Figs.4.8 and 4.12.

In the case of frames where there are other green objects that take up significant space in the frames, our algorithm is just as robust as demonstrated in Figs.4.11 and 4.12.

Fig.4.9 represents one of the drawbacks that arise due to the imaging device used. All test videos were shot on a smartphone camera (Model name: OnePlus 5T). This camera has an in-built post-capture processing algorithm that is used to counter changing exposure values. This happens when the camera sensor is suddenly pointed at a bright light source such as the sky in this case. When this happens, as is seen in Fig.4.9, the smartphone automatically drops the S and L channel values for all pixels in the frame according to a predefined Auto Exposure (AE) adjustment algorithm. Unless we know the exact algorithm, which is proprietary of the manufacturer, it is impossible to account for this sudden change in the S channel values, owing to which the sky crosses the detection threshold and is included in the output for that frame.

However, this problem can be easily evaded by using a standard imaging device with no post-capture image processing algorithms applied to the frames.

Chapter 6

Conclusion

This work proposed a dynamic thresholding algorithm for the identification of street and traffic signs based on the Hue, Saturation and Luminance values. Justification for the usage of the said color space is presented and further noise reduction was performed. For the classification of the various traffic signs, a CNN was utilized. Results were presented to justify the effectiveness of the proposed methods. However, certain issues still persist which pave the way for future work.

First, is utilization of a smartphone camera with inbuilt Auto Exposure Correction and Electronic Image Stabilization which led to excessive noise when the image features were morphed. The process of noise reduction includes the assumption of limiting maximum detected features from the thresholded image leading to loss of information. Another issue for which the reasons remain unknown is the detection of highly reflective surfaces by the proposed algorithm.

6.1 Scope for Future Work

Future work with regard to closing the loop on this project would involve:

- Use of a robust region selection algorithm to determine which of the boards is of higher priority and read those first
- Perspective transformation is to be performed while preserving text related features
- Developing a robust OCR algorithm to read sign boards
- Comparing the read text from the boards and cross referencing it with non-GPS based map data, to triangulate current location / learn data that is under annotated on available digital maps

References

[1] Vitabile, Salvatore, et al. ”Road signs recognition using a dynamic pixel aggregation technique in the HSV color space.” Proceedings 11th International Conference on Image Analysis and Processing. IEEE, 2001.

[2] Ruta, Andrzej, Yongmin Li, and Xiaohui Liu. ”Detection, tracking and recognition of traffic signs from video input.” 2008 11th International IEEE Conference on Intelligent Transportation Systems. IEEE, 2008.

[3] Sermanet, Pierre, and Yann LeCun. ”Traffic sign recognition with multi-scale Convolutional Networks.” IJCNN. 2011.

[4] H. Fleyeh, ”Color detection and segmentation for road and traffic signs,” IEEE Conference on Cybernetics and Intelligent Systems, 2004., Singapore, 2004, pp. 809-814.

[5] H. Fleyeh, ‘Traffic and Road Sign Recognition’, PhD dissertation, Edinburgh, 2008.