Detection and Recognition of Alert Traffic Signs

Chia-Hsiung Chen, Marcus Chen, and Tianshi Gao[1]
Stanford University
Stanford, CA 94305
{echchen, marcuscc, tianshig}@stanford.edu

Abstract

Traffic signs provide drivers important information for safety and efficient navigation. Automatic detection and recognition of traffic signs inevitably become more popular. In this paper, an efficient algorithm/platform is presented to achieve automatic alert traffic signs detection and recognition. Histogram of Gradient (HOG) is adapted to extract features and an over-complete set of 1680 features is designed. A cascade classifier for each sign is trained and built with Support Vector Machine (SVM) as the single stage classifier. To encode the color information, features from different layers in RGB space are combined into a single vector as the feature descriptor. Furthermore, color segmentation is performed to reduce the search regions and a specially designed integral image is used to extract features in a look-up manner. Experimental results show that our system can achieve invariance to illumination, scale and pose. The smallest detectable size is 14×14. The average detection rate is around 98% and the false positive rate is around 1.6%. The processing time for a typical 640×480 image is around 7-9 seconds.

1. Introduction

Alert traffic signs are usually bright in color in hope to attract drivers' attention. However, drivers may fail to notice, especially when they are facing glaring light or when the signs are too small, etc. As such, accurate detection of these signs under different illumination, pose, and scaling becomes a main issue.

Symmetry of traffic signs can be exploited for detection. However, it is subjected to the problem of illumination and pose variance as edges may be blurred in images and symmetry may be skewed due to relative poses of the camera.

SIFT [1] can capture general features with decent efficiency. However, the recognition procedure using SIFT requires counting the number of matched points. If the target sign is too small, SIFT can only extract a few key points. Even when the matching is perfect, the number of matched points is still small. Due to this disadvantage, it is hard and ineffective to use such a counting-based method for detection and recognition tasks.

Edge histogram information is used in traffic sign detection in [2]. However, this approach used in this paper is specially designed for certain shaped signs, and cannot easily be extended to more general signs easily.

In [3,4], the authors present an algorithm for human being detection, using Viola's face detection training algorithm and HOG to extract features. The results seem to promise a possible solution to our problem. However, the features are designed mainly for the human body and are not readily to be used in the traffic sign detection, especially since they do not take the color information into consideration.

Based on the analysis above, in our approach we adapted HOG to extract features and designed a 1680 over complete set of features. From the set of features that best describe and distinguish the signs are selected. The color information is also encoded indirectly into the features, and a cascade classifier for each sign is trained and built using SVM as the single stage classifier. Furthermore, to speed up the processing time, color segmentation is performed to reduce the search regions and a specially designed integral image is used to extract features in a look-up manner.

The remaining of this paper is organized as follows: Section 2 describes an overview of our approach, while Section 3 and 4 introduces the feature design, representation and learning procedure. In Section 5 our experimental setup and results are shown, and Section 6 concludes this paper.

2. Overview of our System

As shown in Figure 1, the input image is first scaled down to our pre-defined size, passed through color segmentation and region reduction, and then candidate regions are fed into our detection and classification system. In the color segmentation and region reduction step, we narrow down the search region by filtering out negative regions that do not contain the desired colors and that fail in the gradient magnitude test. The region with desired colors (red and white in our case) and that passed gradient magnitude test, forms a block, and is subsequently sent to the cascade classifier. In each cascade classifier, desired

[1] The names of the authors are ordered alphabetically.

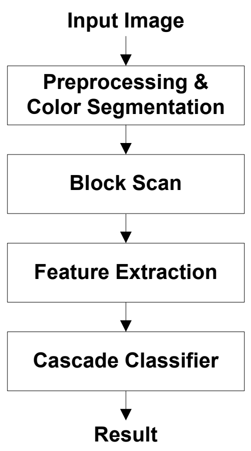


Figure 1: Overview of the algorithm.

features are extracted for classifying. Blocks that fail the tests at any stage in the cascade classifiers are declared as negative and no further tests are required. On the other hand, those blocks that pass the tests through all stages in the cascade classifier are declared as positive.

3. Feature Design and Representation

Based on the observation that alerts traffic signs have very bright color, shape, and pattern attributes, we designed our features to catch this information.

3.1. Color Encoding

The color of the alert traffic signs is a valuable clue in sign detection. However, since traffic signs are exposed under different lighting conditions, it is hard to achieve illumination invariant detection using the absolute color values directly. Therefore, we'd like to encode the color information indirectly, and take advantage of the relative appearance instead of the absolute values on different color layers.

Take a stop sign for example. On the R layer in RGB space, since both white and red pixels color have high intensity values, the magnitude of the gradient on this layer is small. However, on the G and B layers, red pixels have low value while the intensity value of white pixels are still high; thus the magnitude of the gradient on those edges between white and red pixels is high. As a result, the gradient of the same pixel on different layers may exhibit very different values, and these values are correlated. Therefore, we combine the gradient values on the three layers into a vector and normalize it using the sum of the gradient magnitude over the three layers.

3.2. HOG Based Feature

Like SIFT, we adapted the HOG to extract features. There are two benefits of using HOG-based methods. First,

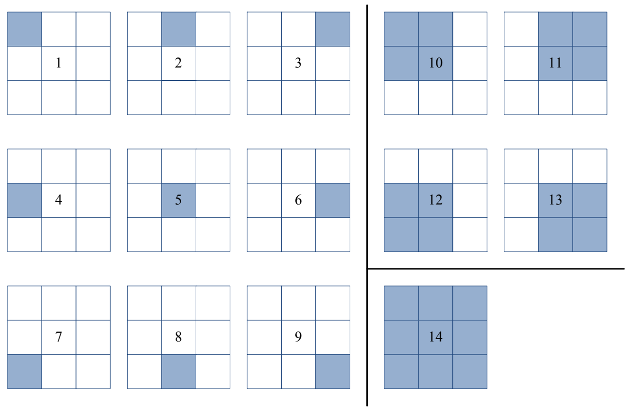


Figure 2: Sub-blocks of the image.

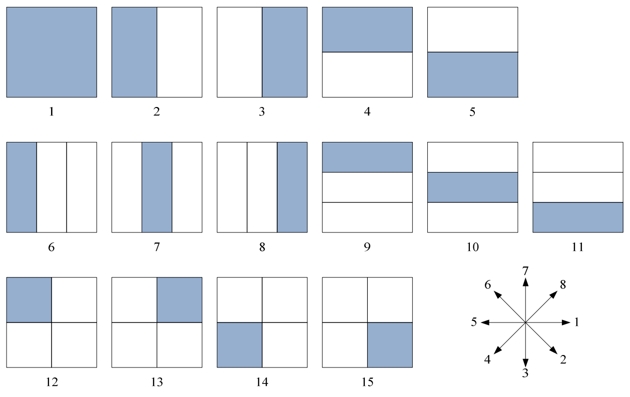


Figure 3: Templates and orientation bins.

traffic signs have distinctive shapes, and the patterns on different layers may have strong edges. Therefore, the gradient can efficiently capture these features. Second, using HOG can help to achieve scale invariance.

In addition, inspired by feature selection in face detection, we designed an over complete set of the Haar-like features, from which we can select features that can best describe and distinguish different signs. The details are described in the following subsections.

3.2.1 Feature Definition

First, we divide the image into 3-by-3 square regions and define 14 sub-blocks shown as the shaded regions in Figure 2. Second, we design 15 templates and divide the orientation range [0 2π] into 8 bins as shown in Figure 3. Each template is a square block with a shaded cell.

Given a color image, the feature is evaluated on different layers at the *i*-th sub-block, using the *j*-th template and the *k*-th orientation. Each feature is denoted by $F(i,j,k) = [f_r(i,j,k), f_g(i,j,k), f_b(i,j,k)]^T$, where the three components correspond to the three RGB layers. Denote the *i*-th sub-block by B_i and the shaded cell region in *j*-th template by C_j. Then for $f_r(i,j,k)$, we first calculate the gradient at each pixel (x, y) on R layer as:

$$G_{rx}(x,y) = [-1\ 0\ 1] * I_r(x,y)$$
$$G_{ry}(x,y) = [-1\ 0\ 1]^T * I_r(x,y)$$

the strength of the gradient at (x, y) is

$$G_r(x,y) = \sqrt{G_{rx}(x,y)^2 + G_{ry}(x,y)^2}$$

and the orientation is:

$$\theta_r(x,y) = \tan^{-1}\left(\frac{G_{ry}(x,y)}{G_{rx}(x,y)}\right)$$

Remember that we divided the orientation range [0 2π] into 8 bins, and denoted the value of the *k*-th bin to be

$$\psi_{rk}(x,y)\begin{cases} G_r(x,y), & if\ \theta_r(x,y) \in bin_k \\ 0, & otherwise \end{cases}$$

The feature value for $f_r(i,j,k)$ is defined as

$$f_r(i,j,k) = \frac{\sum_{(x,y)\in C_j} \psi_{rk}(x,y)}{\sum_{(x,y)\in B_i}\left(G_r(x,y) + G_g(x,y) + G_b(x,y)\right)}$$

where $G_g(x,y)$ and $G_b(x,y)$ are the magnitudes of the gradient on the G and B layers, respectively.

Similarly, we have

$$f_g(i,j,k) = \frac{\sum_{(x,y)\in C_j} \psi_{gk}(x,y)}{\sum_{(x,y)\in B_i}\left(G_r(x,y) + G_g(x,y) + G_b(x,y)\right)}$$
$$f_b(i,j,k) = \frac{\sum_{(x,y)\in C_j} \psi_{bk}(x,y)}{\sum_{(x,y)\in B_i}\left(G_r(x,y) + G_g(x,y) + G_b(x,y)\right)}$$

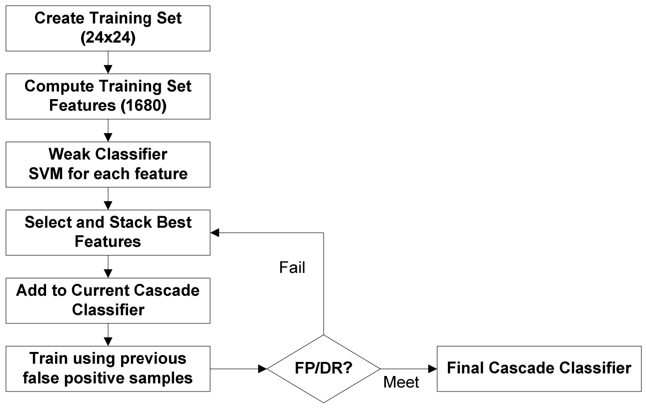
Finally, we have

$$F(i,j,k) = [f_r(i,j,k), f_g(i,j,k), f_b(i,j,k)]^T$$

Since there are 14 sub-blocks, 15 templates, and 8 orientations, we have a total of 14×15×8 = 1680 features.

3.2.2 Feature Evaluation

Integral images are used extensively in our algorithms and thus cut down repeated computation. The pre-computation enables fast computation in pre-processing and also features extraction steps. The following formula shows the computation of integral image. Any reading in integral image represents the sum from (1,1) to this point. As such, summation of a region C can then be done by using only 4 arithmetic computation of readings on the four corners of this region.



Create Training Set (24x24)
Compute Training Set Features (1680)
Weak Classifier SVM for each feature
Select and Stack Best Features
Add to Current Cascade Classifier
Train using previous false positive samples
FP/DR?
Fail
Meet
Final Cascade Classifier

Figure 4: Flow chart of training phase

$$I_{Gk}(x,y) = \sum_{0\le x'\le x, 0\le y'\le y} \psi_k(x',y')$$

$$\sum_{(x,y)\in C} \psi_k(x,y) = I_{Gk}(x_c-1, y_c-1) + I_{Gk}(x_c+w_c-1, y_c+h_c-1) - I_{Gk}(x_c-1, y_c+h_c-1) - I_{Gk}(x_c+w_c-1, y_c-1)$$

4. Learning Methods

The set of 1680 features is over-complete. Therefore, we need to select features for each of the signs that can best describe and distinguish them. Moreover, the process of selecting features interweaves with the process of building up the classifier. The overview of the training phase is shown in Figure 4. Our goal is to build a cascade classifier for each of the signs. The details are as follows:

First, for each sign we created a training set in which images of the currently training sign are positive samples while all other signs and random selected images are negative samples. After that, the 1680 features for each of the samples are computed.

Second, for each of the features, we use Support Vector Machine with RPF kernel to classify all the samples. Since the number of the negative samples is about 4 times more than that of the positive samples, we adjusted the penalty parameters of the error term for the two classes to deal with this unbalanced situation. Furthermore, we used different sets of penalty parameters with different emphasis on the positive and negative samples to trade off the detection rate and false positive rate.

Third, after getting the classification results using each feature, we selected several top features with the highest detection rate and a relatively smaller false positive rate. We enumerated different selections and combinations of these features and for each combination we stacked the selected features as a higher dimension feature and then use the new single feature to classify the training set. The new

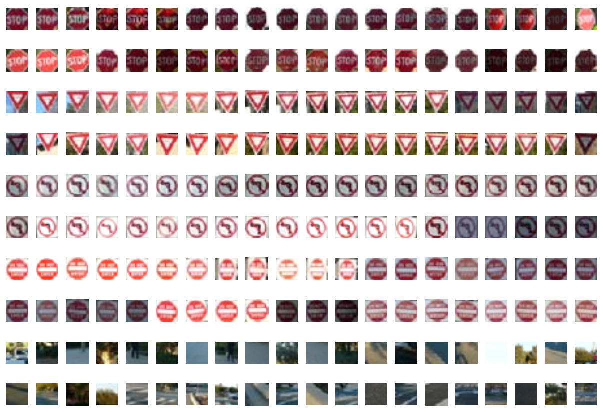


Figure 5: Example of sign/non-sign images in our data set.

feature with the highest detection rate and a relatively smaller false positive rate is selected as a first stage single classifier.

Fourth, having obtained the first stage single classifier, we used another validation set to test the current classifier to make sure it has a very high detection rate ($> 98.5\%$) and recorded the false positive samples. Then, we combined false positive samples from both the training set and the validation set as a new negative sample set. This new negative sample set and the positive samples from the training set are combined into a new training set.

Finally, using the new training set, we repeated the second to fourth steps to form another new single stage classifier which is cascaded after the current classifier. This procedure is repeated until the cascade classifier achieves the predefined false positive rate while having a detection rate higher than a predefined lowest detection rate.

5. Experimental Results

The alert traffic sign detection algorithm was extensively examined in this research. Here we present results on several sequences of images. The input image is first scaled down to 640x480, and then passed into our alert traffic sign detection system. In these sequences, some signs appear alone, while multiple-sign scenarios are also included. These sequences also contain different lighting, scale, and pose conditions for various signs, and are suitable for examining the robustness of our algorithm.

5.1. Data Set and Training

In this presentation, we created a traffic sign training data set with 1921 sign/non-sign samples from 800+ photos we took around Stanford campus. The data set consists of several image sequences, which include traffic signs with different poses and lighting conditions, and with size ranges from 16 to 220 pixels. These signs are hand labeled and scaled to 24×24 by bilinear interpolation. Non-sign samples are randomly cropped from these input images, with various sizes, and are also scaled to 24×24 by bilinear interpolation. The final training set includes 161 stop signs, 72 yield signs, 73 no left turn signs, 83 do not enter signs, and 1532 non-sign images. For each sign, the training set is divided into two classes, where the positive samples are the current target sign images while the negative samples are all other signs and randomly selected non-sign images. Some of the training set images are shown in Figure 5.

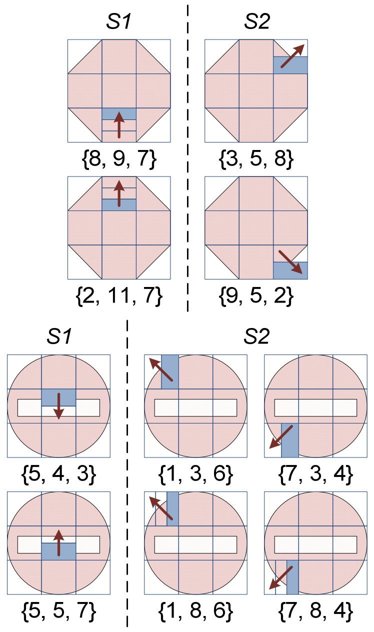


Figure 6: The first two stages of features for stop and do not enter sign, respectively.

In Section 3.2, we have demonstrated how features are represented, and in Section 4, we have demonstrated how to learn and iteratively combine or stack features from the feature pool of 1680 features. The general rule is to select features with a high detection rate and a low false positive rate. These two guidelines should be considered as a whole, not individually. In Figure 6, the selected features for the first two stages in the cascade classifiers for the stop and do-not enter signs are shown, respectively.

5.2. Cascade Sign Detectors

The cascade classifiers for different signs are combined serially to form the final alert traffic sign classifier. As shown in Figure 7, an input image is first examined by the the stop sign classifier. If the image passes through all stages of stop sign cascade classifier, we label it as a possible stop sign region. Otherwise, we pass it into the do not enter sign classifier, and the same procedure goes on for the no left turn and yield classifiers. If we find out that the current image does not belong to any of these signs, we declare it as a non-sign region.

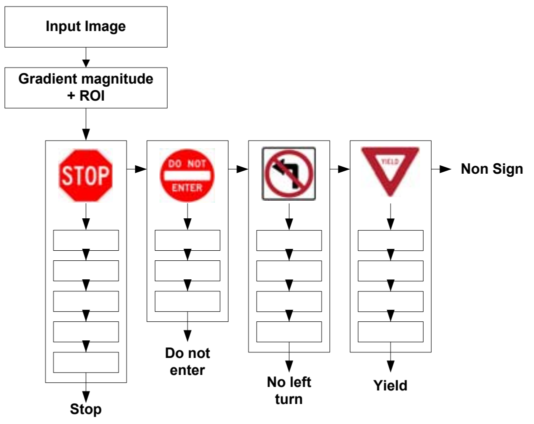


Figure 7: Cascade sign detectors

5.3. Speed Consideration

5.3.1 Gradient Magnitude

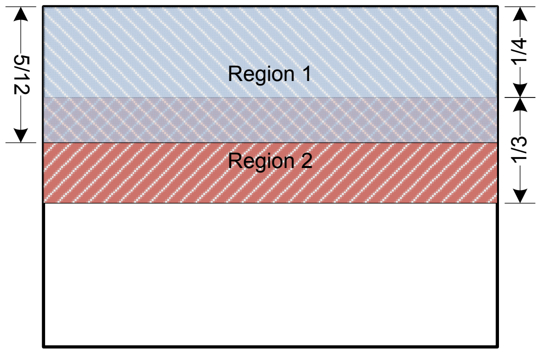
Gradient magnitudes help to quickly narrow the search region. The alert signs have a high contrast in RGB space; for example, white is [255, 255, 255] as compared to red [255, 0, 0]. As such, an edge in RGB images will have no edge in the red layer, but will have edges in the other two layers. Using this fact, we design a hyper-plane with [-1 ½ ½] as the direction vector. The sum of the gradient magnitudes of three layers effectively separate signs from non-signs with 99.4% positive detection and 20% of false positive. This greatly reduces the search region and improves computational efficiency. The hyper-plane coefficients are intuitive, since they punish the gradient in the red layer and reward the green and blue layers. Another thing to note is that absolute gradient magnitudes are used instead of normalized magnitudes. This is more effective to eliminate negative images with noise since normalized ones could give a large magnitude ratio even though the original magnitudes are not large. This hyper-plane method is scale invariant.

Clearly, un-normalized magnitudes will also give a fairly large amount of false positives, especially it also gives positive results for larger regions that contain desired signs. To further tackle this problem, we introduce another detection feature that exploits the raw colors of signs by using color segmentation.

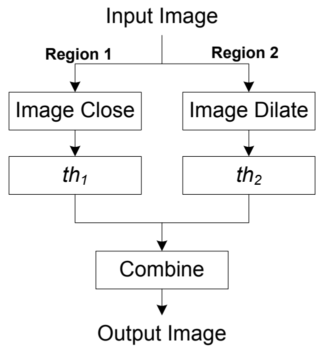
5.3.2 Color Segmentation

In addition to the integral image technique discussed in Section 3.2.2, we also incorporated a red-white color segmentation to further reduce the amount of calculation.

The color segmentation is designed based on two observations. First, in practical settings traffic signs rarely show up on the bottom part of an image, and therefore, we can ignore the bottom part directly. Second, signs on the top part of an image tend to be a lot larger than signs in the middle part of the image. Based on these observations, we divide the top half of the image into two sub regions, and apply different decision rules to eliminate the non-sign region while preserving our region of interest. The two regions we employed in our system are as shown in Figure 8(a). Note that we deliberately overlap the two regions in order to decrease the possibility that the signs are cut in half, and to ensure that we do not lose any regions with signs during the color segmentation process.



(a)



(b)

Figure 8: Color segmentation (a) two regions for different consideration, and (b) segmentation algorithm.

We operate our red-white color segmentation on the YCbCr color space [5]. The thresholds are chosen based on the statistical mean and standard deviation of the CbCr plane. To better differentiate signs in region 1 with extremely bright sky and buildings, we perform image closing on the color thresholded image. For region 2, however, we need to intensify the response for small signs. Thus, we perform dilation on the color thresholded region 2 image. The color segmentation flow is shown in Figure 8(b).

5.4. Experiments on Real-World Test Set

The proposed algorithm is implemented in Matlab, and is tested on a test set of 499 images of size 640×480. This test set contains several image sequences with single and multiple signs under different lighting, scale, and pose conditions. The four alert traffic signs detectors are tested independently in order to examine the efficiency and correctness of our design. Figure 9 shows some examples of detection results, and Table 1 gives the performance of the proposed algorithm. From Table 1, we can see that our system has an average run time around 7~9 seconds for 640×480 images, and is able to achieve a very high detection rate (DR), while keeping a low false positive rate (FP).

TABLE 1. PERFORMANCE OF THE PROPOSED ALGORITHM.

	Avg Time	Smallest Size	DR	FP
Stop	9.21	18x18	98.80%	0.20%
Yield	7.34	14x14	100.00%	1.20%
No left turn	8.05	14x14	92.86%	4.81%
Do not enter	7.96	20x20	100.00%	0.20%

6. Conclusion and Discussion

In this paper, we have demonstrated an efficient alert traffic sign detection and recognition system with novel robust feature extraction and representation. Our experiment shows that the system can achieve a high detection rate of 92%~100% (average of 98%) and a false positive rate of 0.19%~5% (average of 1.6%) for different signs. The system can also achieve illumination, scale, and pose invariance. With an over-complete set of features, our system is able to fully extract distinguishable features for different traffic signs, and therefore can be easily extended to accommodate new signs. The processing time in Matlab for a 640x480 image is around 7-9 seconds. When used with tracking, the correlation between successive frames can be exploited, and real time traffic sign detection can be achieved.

References

[1] D. Lowe. Distinctive Image Features from Scale-invariant Keypoints. International Journal of Computer Vision, 60(2):99-110, 2004.

[2] B. Alefs, G. Eschemann, H. Ramoser, C. Beleznai. Road Sign Detection from Edge Orientation Histograms. IEEE Intelligent Vehicles Symposium, 2007.

[3] H.X. Jia, Y.J.Zhang. Fast Human Detection by Boosting of Orientated Gradient. Fourth International Conference on Image and Graphics.

[4] Q. Zhu, S. Avidan, M. Yeh, and K. Cheng. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients. IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2006.

[5] Y.B. Damavandi, K. Mohammadi. Speed Limit Traffic Sign Detection and Recognition. IEEE Conference on Cybernetics and Intelligent Systems, 2004.

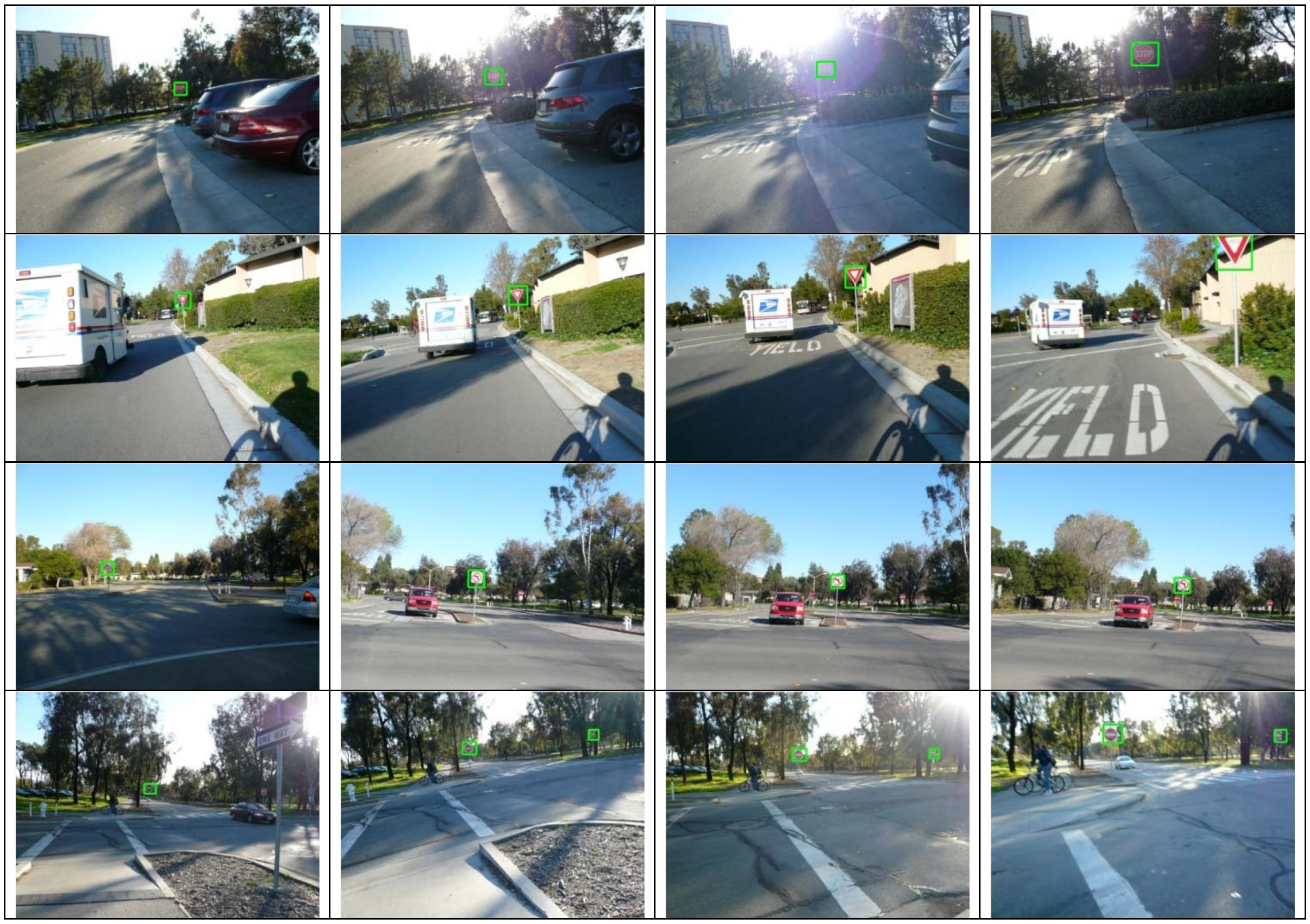


Figure 9: Examples of detection results.
These results are selected to demonstrate our system achieves illumination, scale and pose invariance.