Traffic Sign Detection Using a Cascade Method With Fast Feature Extraction and Saliency Test

Dongdong Wang, Xinwen Hou, Jiawei Xu, Shigang Yue, *Member, IEEE*, and Cheng-Lin Liu, *Fellow, IEEE*

***Abstract*— Automatic traffic sign detection is challenging due to the complexity of scene images, and fast detection is required in real applications such as driver assistance systems. In this paper, we propose a fast traffic sign detection method based on a cascade method with saliency test and neighboring scale awareness. In the cascade method, feature maps of several channels are extracted efficiently using approximation techniques. Sliding windows are pruned hierarchically using coarse-to-fine classifiers and the correlation between neighboring scales. The cascade system has only one free parameter, while the multiple thresholds are selected by a data-driven approach. To further increase speed, we also use a novel saliency test based on mid-level features to pre-prune background windows. Experiments on two public traffic sign data sets show that the proposed method achieves competing performance and runs 2∼7 times as fast as most of the state-of-the-art methods.**

***Index Terms*— Traffic sign detection, cascade system, fast feature extraction, saliency test.**

I. Introduction

TRAFFIC sign detection plays an important role in intelligent transportation such as driver assistance systems, road maintenance and automated driving. Although signs are designed with distinct color and simple shape, automatic detection is still challenging in complex scenes, because the background and illumination are changing, signs may be distorted in color and shape, and sometimes, partially occluded. In addition, the image undergoes motion blur when the vehicle moves fast. A traffic sign detection method should be designed to overcome these problems to achieve high accuracy and reliability. Moreover, detection should be fast to satisfy real-time applications such as driver assistance systems.

Manuscript received February 24, 2016; revised August 12, 2016 and January 25, 2017; accepted March 7, 2017. This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2012CB316302, in part by the National Natural Science Foundation of China under Grant 61271306, in part by the Strategic Priority Research Program of the Chinese Academy of Sciences under Grant XDA06040102, and in part by the EU FP7 Project HAZCEPT under Grant 318907. The Associate Editor for this paper was J. M. Alvarez.

D. Wang and X. Hou are with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: ddwang@nlpr.ia.ac.cn; xwhou@nlpr.ia.ac.cn).

J. Xu and S. Yue are with the School of Computer Science, University of Lincoln, Lincoln LN6 7TS, U.K. (e-mail: jxulincoln@gmail.com; syue@lincoln.ac.uk).

C.-L. Liu is with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: liucl@nlpr.ia.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2017.2682181

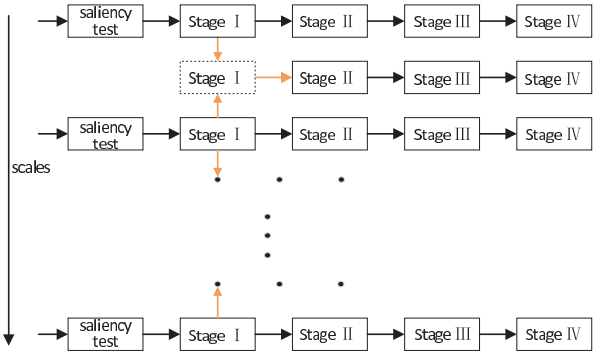

Fig. 1. Proposed traffic sign detection system HHVCas. After pre-pruning by saliency test, HHVCas has four stage classifiers. Stage I rejects windows using a linear SVM classifier on compressed integral HOG feature and neighboring scale awareness. Stage II employs a LDA classifier on integral HOG feature, and Stage III uses a LDA classifier on HOG feature. Stage IV uses a nonlinear SVM on color HOG features.

Traffic sign detection has been studied intensively in the past decades and many approaches have been proposed. Early methods usually exploited the color or geometric information of traffic signs [1], [2]. Since the famous Viola-Jones detector [3] was successfully used in face detection, sliding window and machine learning based methods have become prevalent. Recently, some sliding window based methods [4]–[6] achieved leading performance in the competition of Germany Traffic Sign Detection Benchmark (GTSDB) [7]. Nevertheless, these methods are computationally expensive.

We aim to design a fast traffic sign detection system to maintain the performance advantage of sliding window based methods with significant speedup. There are three main contributions in this work. First, we propose a cascade framework with neighboring scale awareness for fast traffic sign detection. The system has only one free parameter to control the tradeoff between detection speed and accuracy, while the multiple thresholds are selected by a data-driven approach. Second, we design an approximation approach for fast feature extraction, which leads to additional speedup. Third, we propose a novel saliency test based on mid-level features, which is demonstrated to be robust and effective in pre-pruning windows.

Our detection system consists of four cascaded stages where different Histograms of Oriented Gradient (HOG) feature variants are used, as shown in Fig. 1. We name the system as a Hybrid HOG Variants Cascade (HHVCas). The HHVCas detector works by evaluating multi-scale hypothesized windows hierarchically: each stage rejects a portion of non-sign windows and the surviving windows are further evaluated in the next stage with a stronger classifier. We use linear

classifiers for the first three stages and a nonlinear classifier for the last stage. The used features also have increasing computation complexity or dimensionality from stage to stage. The early stages with fast and simplified features run fast to eliminate apparent non-sign windows while preserving signs with high recall rate. The latter stages, based on more representative features that are computed more accurately with more information, provide better discrimination. The saliency test before the cascade can preclude a portion of windows from evaluation by the cascaded classifiers.

Our experimental results on the GTSDB dataset show that the proposed HHVCas detector can achieve competitive performance compared with state-of-the-art methods and runs 2∼7 times as fast. Compared to the recent method [8] which provide high accuracy and speed, our method relies on little color information so that it is less sensitive to illumination. In addition, it involves fewer artificial parameters, and thus has the potential of better generalization. We also demonstrated the promise of the proposed method on the Swedish Traffic Signs Dataset (STSD) [9].

A preliminary version of the proposed method was presented in a conference paper [10]. Since then, the work has been extended in several ways:

- The method is simplified by eliminating the utilization of multi-resolution models in the first two stages, which effects in reducing artificial parameters.
- A data-driving approach is proposed to optimize the thresholds in the system, leaving only one free parameter to select.
- Experimental evaluation is enhanced with detailed analysis and an additional dataset.

The rest of this paper is organized as follow. Section II reviews the related previous works. Section III describes the proposed detection method in detail. Section IV presents the experimental results and discussions, and Section V gives concluding remarks.

II. Related Work

Traffic sign detection methods proposed so far fall into three categories: segmentation-based, shape-based and sliding window based. Segmentation-based methods commonly use color information to classify pixels for extracting candidate signs [11]–[14], or use color in preprocessing to eliminate irrelevant scene regions. To overcome the color sensitivity to illumination, the RGB color space is transformed [12] or converted to other color spaces such as HSV/HSI [13], [15], Lab [14] and CIEACM97 [11]. A comprehensive evaluation of color-based segmentation algorithms can be found in [16]. Some methods extract candidate traffic signs as Maximally Stable Extremal Regions (MSERs) when using thresholds at several levels [8], [17]. Salti *et al.* [18], [19] used the MSER technique to extract regions that exhibit a uniform value of distinctive sign color, and used the Wave Equation algorithm to detect geometrically symmetric regions. The obtained Regions of Interest (ROIs) were further verified by Support Vector Machine (SVM) classifiers and other pruning techniques.

Many methods have exploited the circular or polygonal shape of traffic signs. Barnes and Zelinsky [20] detected speed limitation signs using a Fast Radial Symmetry Transform (FRST), which extracts signs by examining the peaks in a parameter space voted by edge points like that in circular Hough transform. Loy and Barnes [21] proposed an extended FRST to detect equiangular polygonal signs by considering the symmetry of target polygons. Höferlin and Zimmermann [22] localized potential signs using SIFT, as a complement of FRST. García-Garrido *et al.* [23] located circular signs using FRST as well, and detected polygonal signs by locating lines with Hough transform. Other Hough-like methods include Vertex and Bisector Transform [24], Bilateral Chinese Transform [25], Single Target Vote for Upright Triangles [2], Single Target Vote for Upright Ellipses [2] and RANSAC for Symmetric Lines Detection [26]. Some methods [27], [28] simplify the sign contours using a constrained combination of simple linear structures which are coded by Local Contour Patterns descriptor.

Some shape based methods use classifier or shape matching to verify sign hypotheses proposed by simple features or image segmentation. Landesa-Vázquez *et al.* [28] refined the hypotheses using a cascaded AdaBoost detector [3] where the weak learners are based on intensity comparison between pixels. Liang *et al.* [6] applied shape-specific templates to search potential signs on a transformed image where each RGB triple was projected to a scalar value, and then used SVM classifiers to refine hypotheses. Timofte *et al.* [29] exploited additional multi-view 3D information captured by multiple cameras to improve detection. Candidates extracted in single views were verified by a cascaded AdaBoost classifier [3] and combined to generate 3D hypotheses.

Sliding window based methods have been widely adopted in object detection, mostly using the cascaded AdaBoost classifier [3], where the weak learners often use Haar-like features [30]–[32]. Bahlmann *et al.* [33] proposed color parameterized Haar-like features for traffic sign detection. Other features used include the Edge Orientation Histograms [34], quantum features [28] and Local Rank Pattern [35]. Some methods [36]–[38] use simplified versions of HOG for constructing weak learners, where the gradient orientation is discretized by several comparisons in horizontal and vertical gradients. Specifically, Pettersson *et al.* [36] built HistFeat features which are 2D tables derived from pairs of orientation bins. Overett *et al.* [37] proposed LiteHOG and LiteHOG+ features by projecting multiple orientations into a single scalar with Fisher Discriminant Analysis. Mathias et al. [4] adopted depth-2 decision trees as weak learners based on integral channel features. Møgelmose *et al.* [39] employed the same method to detect US traffic signs. Liu *et al.* [40] proposed two variants of Local Binary Pattern and a split-flow cascade tree structure to detect multiple types of signs, where a Common Finder AdaBoost is designed to find the common features that are shared by signs of different types. Instead of the AdaBoost cascade, Wang *et al.* [5] designed a two-stage detector in coarse-to-fine manner, with a Linear Discriminant Analysis (LDA) classifier and a nonlinear SVM in two stages. This approach reported appealing performance, but the high computational cost remains an issue.

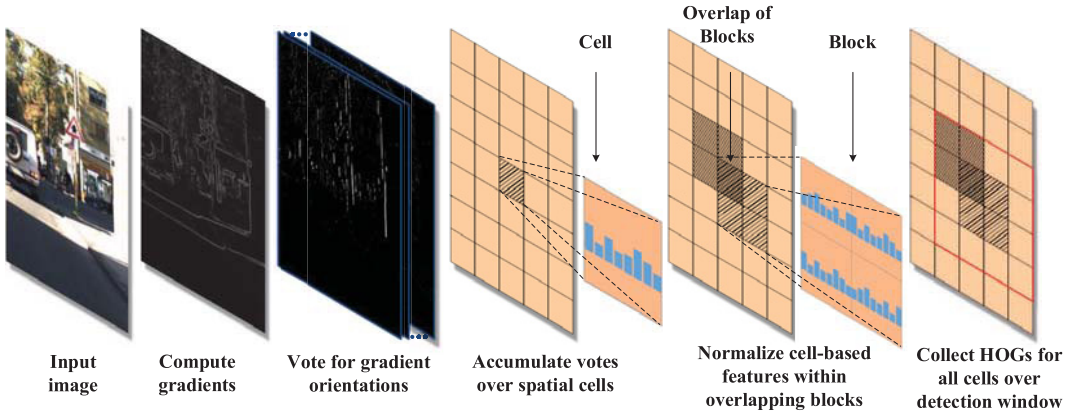

Fig. 2. Steps of HOG computation [44]. First, the gradient of each pixel in image is computed and quantized to N orientations by bilinear interpolation. The image plane is then partitioned into a dense grid of rectangular cells, where pixel-level features (orientation channel values) are accumulated to obtain cell-based histograms of oriented gradients. Cell-based features are normalized within overlapping blocks, and the normalized features of all cells in a window are concatenated into a feature vector.

It is worthy of mentioning that for generic object detection, recent methods based on deep convolutional neural networks (CNNs) [41]–[43] have reported superior performance. They explore learned features in deep neural networks and use GPUs to satisfy the very high computation demand. These methods reveal some insights for traffic sign detection in the future, but to reduce the computation cost remains an issue.

Our proposed method detects traffic signs from sliding windows using a cascade framework like the Viola-Jones detector [3] to achieve fast detection. The key difference from previous methods lies in that we use strong classifiers in each stage of our system to achieve better tradeoff between speed and accuracy. Compared to the method of [5], our system uses more stages in the hierarchy for faster detection while maintaining high accuracy by using strong classifiers.

III. Traffic Sign Detection

The proposed HHVCas detector (Fig. 1) consists of four cascaded stages for coarse-to-fine sliding window evaluation in addition to a saliency test stage for pre-pruning windows. The Stage I rejects windows using a linear SVM classifier based on compressed integral HOG feature. The Stage II employs a LDA classifier on integral HOG feature, which is more representative than the one in the preceding stage and can prune more disturbing windows. The surviving windows are fed into the Stage III which uses a LDA classifier on HOG, which is stronger than the integral HOG. The Stage IV uses a nonlinear SVM with color HOG feature [5] to make final decisions. The Stage I also exploits the correlation between the windows of neighboring scales to reduce the computation of window evaluation. The cascade involves several thresholds for window rejection, which are jointly optimized on a training dataset, and only one free parameter is remaining to be selected artificially for controlling the tradeoff between the detection performance and speed.

In the following, we first describe the feature extraction techniques for the cascade, then illustrate the techniques of neighboring scale awareness, parameter optimization, and saliency test in the sequel.

A. Fast Feature Extraction

Our HHVCas system uses several different HOG variants, including integral HOG [45] and its compressed version, HOG [46] and color HOG [5].

1) HOG: The steps of HOG computation are depicted in Fig. 2. First, the gradient of each pixel in the image is quantized according to orientation. The image plane is then partitioned into a dense grid of rectangular cells. In each cell, the pixel-level features (values of N orientation channels) are accumulated to obtain cell-based histograms of oriented gradients, where each pixel contributes to the cells around it by bilinear interpolation. The cell-based features are further normalized in larger spatial regions called blocks. Typically, blocks include 2×2 cells and overlap by one cell. Hence, each cell is normalized by four factors corresponding to four blocks which it belongs to, producing a $4 \times N$-dimensional feature vector for the cell. In a detection window, the cell-based features are concatenated into a long feature vector for evaluation. More details of HOG computation can be found in [46].

2) Integral HOG: The integral HOG is different from the HOG only in the step of cell-related accumulation: each pixel contributes to the nearest cell only, or saying, the cell-based features are formed by hard partition, unlike the soft assignment (bilinear interpolation) of HOG. The hard cell assignment makes the integral HOG easily computed via the integral images of oriented gradients. In our implementation, we also perform normalization on per-cell aggregation as in HOG. Due to the hard cell assignment, the integral HOG is less discriminative than the HOG.

3) Compressed Integral HOG: The high dimensionality of integral HOG leads to expensive window evaluation, so we introduce a condensed version. Unlike in integral HOG that the N-dimensional histograms of each cell are normalized by four different factors to form a $4 \times N$-dimensional vector, we can obtain a compressed vector of $4 + N$-dimension, by summing over both the four normalized values for a fixed orientation and the N orientations for a fixed normalization factor. This technique was firstly proposed by Felzenszwalb *et al.* [47] for

TABLE I
THE PARAMETERS OF HOG VARIANTS

	Window size (pixels)	Cell size (pixels)	Block size (cells)	Block overlap (cells)	Orientation bins	Feature dimensions
HOG	20×20	4×4	2×2	1	8	800
color HOG	40×40	8×8	2×2	1	8	2400
Integral HOG/compressed	$(20\times20)/1.08^2$ 20×20 $(20\times20)\times1.08^2$	$(4\times4)/1.08^2$ 4×4 $(4\times4)\times1.08^2$	2×2	1	8	800/300

the HOG feature, which leads to little loss in discriminability. In our HHVCas system, the compressed integral HOG feature is used in the first stage for quick rejection of windows, and is expected to complement the integral HOG feature used in the second stage.

4) Color HOG: For each color channel of image, HOG feature is calculated for each cell as in the above procedure, and histograms of different channels of all cells in a detection window are concatenated into a long feature vector.

The parameters for the HOG variants used in our system are summarized in Table I. Since each window is partitioned into 5×5 cells, the feature dimensionality is 800 for HOG and integral HOG, 2400 for color HOG, and 300 for compressed integral HOG.

In our system, the integral HOG feature is calculated at multiple scales for Stage I and Stage II evaluation. To construct such an integral HOG pyramid is computationally expensive due to the calculation of oriented gradients for each pixel. Inspired by the method in [48], we propose a fast strategy by sharing orientation channels among neighboring scales. Let I denote an $m \times n$ image, I_s denote the scaling of I with a factor s, $R(I, s)$ specify the sampling of I with factor s, and F denote the maps of extracted features of an image. Suppose we have computed $F = \Omega(I)$, e.g. N gradient orientation maps. The scaled maps F_s can be obtained by

$$F_s = \Omega(I_s) = \Omega(R(I, s)). \tag{1}$$

Alternatively, Dollár *et al.* [48] proposed the approximation

$$F_s \approx R(F, s) \cdot s^{-\lambda_\Omega}, \tag{2}$$

where λ_Ω is a feature-related parameter. Equation (2) shows that the N orientation maps of I_s can be approximated by those maps of I. Dollár *et al.* [48] adopted this strategy for the fast calculation of integral channel features.

For calculating the integral HOG feature of I_s, we can first obtain the oriented gradient maps F_s using the general method (1) or the approximation (2). The latter saves much time by avoiding the direct gradient computation from I_s, yet still suffers from the overhead of resampling and summation in each cell. We propose a further acceleration technique by considering the relation between F and F_s with scaled cell $w = w_s \cdot s$:

$$\frac{1}{|w_s|} \sum_{i,j \in w_s} F_s(i, j) \approx \frac{1}{|w|} \sum_{i,j \in w} F(i, j) s^{-\lambda_\Omega}. \tag{3}$$

This shows that the summation on F_s with cell w_s can be obtained from F with scaled cell w, and vice versa.

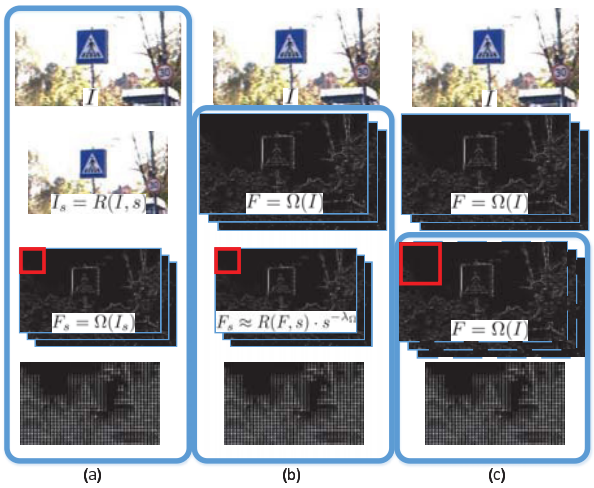

Fig. 3. Strategies for calculating integral HOG of different scales. (a) Ordinary method: oriented gradient maps of different scales are calculated independently. (b) Approximation of [48], oriented gradient maps of I_s are obtained by resampling those of I. (c) Integral HOG for I_s are approximated from the N orientation channels of I with a scaled cell $w = w_s \cdot s$.

In practice, the scaling effect of summation is canceled out when performing local normalization across cells. So, the scaling factor $s^{-\lambda_\Omega}$ of summation can be simply omitted and the cell summation of a neighboring scale is directly taken. Fig. 3 shows the three strategies. Column (a) shows the ordinary strategy of image scaling followed by feature map calculation. Column (b) is the approximation via equation (2). The two methods both calculate F_s explicitly. Column (c) shows the proposed approximation: use the N orientation channels of I directly for a different scale. This leads to the same features as proposed in [48] and is more efficient.

By the above approximation, the loss of feature representation is negligible for small scaling factor s, but is considerable when s is large. Therefore, we calculate the maps oriented gradients on a set of sparse scales for an integral HOG pyramid, and share the maps among neighboring scales only. As illustrated in Fig. 4, the oriented gradients are calculated every three scales and shared locally. The window/cell sizes of the three scales differ by scaling factor $s = 1.08$, as shown in Table I.

In summary, HOG feature is prevalent in the community of computer vision for its good representation, but the disadvantage is that this feature is time consuming for computation. We use its simplified variants in the first stages to reject most hypotheses. Then HOG and color HOG of surviving hypotheses are extracted and evaluated in the last stages.

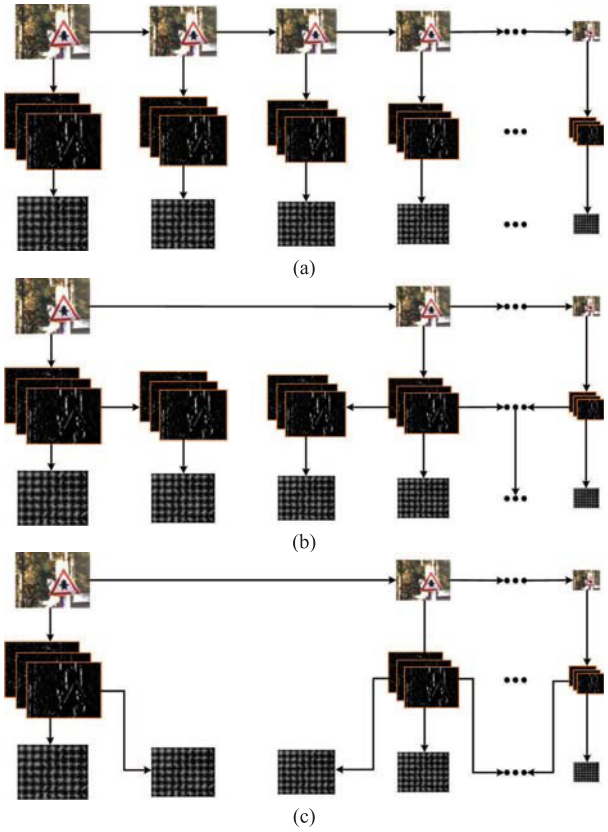

Fig. 4. Strategies of constructing an integral HOG pyramid. (a) Ordinary method: oriented gradient maps and features of difference scales are calculated independently. (b) Method of [48]: orientation maps are obtained by scaling those of a neighboring scale. (c) Take the channel maps of a neighboring scale and use scaled size of window/cell.

The parameters of HOG variants are referred partly to the work of Wang *et al.* [5]. We set the windows size in the first two stages as 20×20 to detect the smallest signs in re-scaled images. In stage III, the window size remains 20x20 for incurring minimum cost in calculating HOG feature. The last stage handles big windows (40×40) for exploiting more discriminative information in high-resolution image, while the increase of computation cost in this stage is moderate because of small number of surviving windows. As illustrated in Table I, the increasing dimensionalities of features in stages (300, 800, 800 and 2400) correspond to increasing evaluation time. Though the dimensionalities of feature vectors in stage II and III are the same, the integral HOG in stage II is much more efficient in computation.

B. Neighboring Scale Awareness for Speedup

In addition to saving feature calculation by approximating feature maps from neighboring scales as described, we also use neighboring scale awareness to save window evaluation. It has been observed that responses of a detector at nearby positions (in the same or neighboring scales) are correlated [49], [50]. We speed up detection by exploiting the correlation between the detection windows in neighboring scales. Let x be a hypothesis window at scale s in search space. Let $\mathcal{N}(x)$ be x's neighbors in adjacent scales. We consider our HHVCas as a four-stage detector, in which the per-stage classifier is H_k ($k \in \{1, \ldots, 4\}$). For any window x at scale s, we do not compute the score $H_1(x)$, but instead estimate from the scores $H_1(x'), \forall x' \in \mathcal{N}(x)$. Because $H_1(x)$ is correlated with (actually similar to) the scores of its neighboring windows, we can reject x if $H_1(x')$ for all $x' \in \mathcal{N}(x)$ fall below a threshold θ^S, otherwise the window x is retained and fed into the next stage classifier H_2. Since the scores of windows in neighboring scales are used, we refer to this technique as neighboring scale awareness. We apply this technique only in the first stage for scoring H_1, as show in Fig. 1. The subsequent stages do not use neighboring scale awareness because they need higher accuracy and encounter far fewer windows than the first stage.

C. Parameter Optimization

The HHVCas detector involves thresholds for both per-stage classifier rejection and neighboring scale awareness based pruning. We optimize the thresholds jointly using an unsupervised data-driven optimization approach, as inspired by the work of [50] for a soft cascade.

We consider the thresholds for the first three stages H_k in HHVCas, since the threshold in the last stage is variable for tradeoff the precision and recall rate. There are two types of thresholds: per-stage rejection threshold θ_k^R, and neighboring scale pruning threshold θ^S in Stage I. The multi-stage rejection thresholds are initially selected conservatively according to the performance on a training image set, letting most positive windows retained. These initial thresholds are denoted by θ_k^* and called base thresholds.

Using the HHVCas with base thresholds to evaluate an image set, we collect the detected windows $\mathcal{X}$ as *quasi-positives*. The fraction of quasi-positives $\mathcal{X}$ rejected by stages is called Quasi Miss Rate (QMR), given a set of thresholds. If the QMR at each rejection stage is $\leq \gamma'$, the overall QMR of the cascade detector will be $\leq \gamma = 1 - (1 - \gamma')^K$. Let $\mathcal{X}_1 = \mathcal{X}$ be the initial set of quasi-positives and define $\mathcal{H}_1 = \{H_1(x) | x \in \mathcal{X}_1\}$. The first rejection threshold θ_1^R is obtained as:

$$\theta_1^R = \lfloor \mathcal{H}_1 \rfloor_r - \epsilon, \quad \text{where } r = \lfloor \gamma' \cdot |\mathcal{H}_1| \rfloor, \tag{4}$$

where $\gamma' = 1-(1-\gamma)^{1/K}$, $\lfloor \mathcal{H} \rfloor_r$ denotes the r^{th} smallest value in $\mathcal{H}$ and $\epsilon = 10^{-5}$. For other stages $1 < k \leq K$, we define $\mathcal{X}_k = \{x \in \mathcal{X}_{k-1} | H_{k-1}(x) > \theta_{k-1}^R\}$ and $\mathcal{H}_k = \{H_k(x) | x \in \mathcal{X}_k\}$. We can then obtain θ_k^R as:

$$\theta_k^R = \lfloor \mathcal{H}_k \rfloor_r - \epsilon, \quad \text{where } r = \lfloor \gamma' \cdot |\mathcal{H}_k| \rfloor. \tag{5}$$

The neighboring-scale awareness module is optimized by considering H_1. For each quasi positive x, let $\mathcal{N}(x)$ be its neighbors in neighboring scales. We collect $\mathcal{X}^S = \{x'_m | H_1(x'_m) \geq H_1(x') \wedge H_1(x'_m) > \theta_1^*,\ x'_m, x' \in \mathcal{N}(x)\}$. Let $\mathcal{H}_1^S = \{H_1(x'_m) | x'_m \in \mathcal{X}^S\}$. We set θ^S by:

$$\theta^S = \lfloor \mathcal{H}_1^S \rfloor_r - \epsilon, \quad \text{where } r = \lfloor \gamma' \cdot |\mathcal{H}_1^S| \rfloor. \tag{6}$$

It is easy to see that with the above thresholds θ_k^R and θ^S, the cascade detector with neighboring scale pruning has QMR at most γ. In the first stage, whether to prune a quasi-positive

x or not is determined by scoring H_1 or by looking at the scores of its neighbors, and the two cases are equivalent based on Equations (6) and (4). This makes $|\mathcal{X}_2| \geq |\mathcal{X}_1 \cdot (1-\gamma')|$. For subsequent stages, we have $|\mathcal{X}_k| \geq |\mathcal{X}_{k-1} \cdot (1-\gamma')|$. Therefore, $|\mathcal{X}_{K+1}| \geq |\mathcal{X}_1 \cdot (1-\gamma')^K|$. The overall fraction of pruned quasi-positives is at most $1-(1-\gamma')^K = \gamma$. If there are too many quasi-positives per image, we can conduct this procedure several rounds to obtain optimal θ_k^R and θ^S iteratively. Typically, two rounds is enough to get stable estimates: a QMR is picked at the first round to prune a fraction of quasi-positives, and then the obtained $\mathcal{X}_{K+1}$ serves as the initial set of quasi-positives for the second round.

D. Saliency Test

Preceding the cascade detector with a pre-pruning module based on saliency test can further speed up detection. We propose a robust saliency test based on mid-level features (such as HOG) instead of on low-level features in common saliency-based detection. This is intuitive that a mid-level representation is more discriminative than a low-level one to locate candidate sign regions, while a low-level representation may not prune non-sign regions reliably though it runs fast.

For easy implementation, we adopt the simple center-surround saliency [51], which is based on the assumption that saliency reflects the local contrast of an image region with respect to its neighborhood. In this case, the saliency of a region is computed as the distance between the average feature in the region and the average feature over its neighborhood. Let v denote a feature vector, w_0 and w_1 denote a center and a surround region centered at pixel (i, j), respectively. Let $|\cdot|$ be the area covered by a region, and $D(\cdot)$ denote the distance between two vectors, then the saliency value $V(i, j)$ can be computed by

$$V(i,j) = D\left(\frac{1}{|w_0|}\sum_{p\in w_0} v_p, \frac{1}{|w_1|}\sum_{q\in w_1} v_q\right). \tag{7}$$

Typically, multi-scale saliency is computed using several surrounding window sizes and aggregating the multiple saliency values:

$$V(i,j) = \sum_{\mathcal{S}} V_s(i,j), \tag{8}$$

where $\mathcal{S}$ is the set of surrounding window sizes. Binarizing the saliency map $V(i, j)$ gives a mask of image, with zero denoting non-saliency pixels.

In our method, we calculate two cell-level saliency maps based on two types of mid-level features: compressed HOG and non-normalized HOG (without block-based normalization). In the input image, each cell (size 8×8) has a compressed HOG feature and a non-normalized HOG feature, both are N-dimension. The compressed HOG is obtained from the normal HOG feature of one cell by summing the four normalized values of each orientation. The non-normalized HOG is obtained by summing the orientation values over the pixels in a cell. It is not robust to illumination change, but helps eliminate low-contrast regions, which are unlikely to contain signs. The two cell-level saliency maps based on compressed HOG and non-normalized HOG are denoted as V_h and V_{nH}, respectively. The center region is the 8×8 cell, while the surrounding region has three sizes of 3 cells, 5 cells and 7 cells wide. The two cell-level maps are smoothed using a Gaussian filter ($\sigma = 0.5$) and then re-scaled to the same size as the input image. We apply thresholds T_h and T_{nH} on V_h and V_{nH}, respectively, to get two binary masks. The two masks are then fused into one by AND operation, i.e., pixels that are salient in both maps can survive. For a detection window, it is expected to contain a sign if the fraction of salient pixels is above a threshold T_{area}.

The cell size 8×8 was selected empirically based on the assumed minimum sign size 20×20 in detection. If the cell size is as large as the sign size, the HOG in a cell will be less discriminative to differentiate between signs and background regions. In contrast, a partial region of a sign which has dominant orientation is more likely to be salient from background. On the other hand, too small cell size would result in big HOG maps, thus leads to expensive saliency computation. Empirically, the cell size can be set in between 6 and 10, and specifically, set as 8×8 in our experiments. The setting of thresholds T_h, T_{nH} and T_{area} is specified later in the experimental section.

Examples of saliency test are shown in Fig. 5, where non-salient pixels are displayed in black. It is seen that the sign regions are well preserved while some image regions are eliminated.

E. Summary of Detection Process

Since the detector involves multiple steps and techniques, we summarize the processing steps in sequential as follows.

- Step 1: Saliency mask generation. For an input image, calculate the mask image from two saliency maps, based on the compressed HOG and non-normalized HOG, respectively. The saliency mask labels saliency for each pixel in the input image.
- Step 2: Feature pyramid construction. Build two feature pyramids with scaling factor 1.08 for integral HOG and compressed integral HOG using the proposed fast feature extraction technique. To save computation, oriented gradient channels are computed once every two scales and are shared locally among neighboring scales.
- Step 3: Saliency test for every other scale. On a scale, saliency test is adopted to pre-prune background windows. The candidate window (size 20×20) is back-mapped to the input image to obtain the corresponding patch, of which the proportion of salient pixels is calculated through the integral image of the saliency mask. If the proportion of salient pixels in the patch is lower than threshold T_{area}, the window is pruned, otherwise, the window is fed into the cascade detector.
- Step 4: Speedup by neighboring scale awareness. In Stage I of the cascade, window evaluation by linear SVM on compressed integral HOG is performed for one scale of every two. For another scale without Stage I evaluation, neighboring scale awareness is used to prune candidate windows according to the scores of neighboring

Fig. 5. Examples of saliency test. In each column, the first row shows the original image, and the second row has non-salient pixels displayed in black.

Fig. 6. Sign classes in GTSDB: (a) prohibitory, (b) danger, (c) mandatory, (d) other signs which are not evaluated in detection.

Fig. 7. Sign classes in STSD: (a) prohibitory, (b) mandatory, (c) some signs that are prone to be confused with the two specified categories.

scales. Windows surviving Stage I are fed into Stage II and Stage III for further evaluation.

- Step 5: Accurate detection in Stage IV. Windows surviving Stages I, II and III are verified in Stage IV using nonlinear SVM on color HOG. The 20×20 window is back-mapped to the input image to re-scale the corresponding region into a 40×40 window for extracting color HOG, and given final score by nonlinear SVM.
- Step 6: Duplicate detects suppression. Overlapping detects in different scales are merged by non-maximum suppression.

IV. Experiments

We evaluate the performance of the proposed HHVCas detector and saliency test on two public datasets with comparison to two baseline detectors. On the dataset GTSDB, we also compare the performance with the state-of-the-art results reported in the literature.

A. Datasets

The dataset of German Traffic Sign Detection Benchmark (GTSDB) [7] consists of 600 training images (containing 846 traffic signs) and 300 test images (360 traffic signs). All the images are in high resolution of 1360×800 size, and the size of signs varies from 16 to 128 in terms of the longer side. The types of traffic signs are divided into three major categories and some minor categories. According to standard practice, three categories (prohibitory, danger and mandatory signs) are used to evaluate detection methods. The sign classes are shown in Fig. 6.

The Swedish Traffic Sign Dataset (STSD) was previously used for evaluating traffic sign recognition [9]. It has 3,777 annotated images. Like the partition in GTSDB, we randomly split the images into a training set and a test set in 2:1 ratio, and take the prohibitory and mandatory categories in evaluation (Fig. 7), while the signs of danger category are not explicitly labeled in STSD. The very small signs with longer side less than 16 pixels are excluded from evaluation. In total, there are 1,498 traffic signs for training and 786 signs for testing, respectively. In Fig. 7, it can be seen that detection in STSD is more difficult since the targeted sign categories are similar to some signs that are excluded from evaluation.

B. Detector Settings

1) HHVCas: In the HHVCas detector, the parameters for HOG variants are described in Table I. Integral HOG pyramids are built with a scaling factor of 1.08. Oriented gradient channels are computed once every 3 scales and are shared locally among neighboring scales (Fig. 4). Note that this sharing leads to three windows sizes as depicted in Table I.

2) Baseline Detectors: For comparison with the proposed method, we implemented two baseline detectors based on sliding window. One is a coarse-to-fine detector named as HOG_LDA_SVM following the method of [5]. It evaluates densely sampled windows in multiple scales first using a LDA classifier on HOG feature, and the windows surviving the first stage are verified using an intersection kernel SVM (IKSVM) on color HOG. Compared with the original detector [5], the baseline HOG_LDA_SVM omits the rectifying step for danger category and trains a single classifier for mandatory category as opposed to several classifiers for each specific type in this category. To detect signs of variable sizes, the input image is re-scaled with a factor of 1.08 and the parameters of HOG and color HOG features are the same as in [5]. These settings are also the same as in our HHVCas.

The other detector named as ICF_AdaBoost is a soft cascade of boosted classifiers using integral channel features (ICFs), as described in [4]. Mathias *et al.* [4] applied ICF_AdaBoost on images of a dozen of aspect ratios by scaling the input image, and used a GPU to satisfy the heavy computation overhead. To be comparable with other detectors, we only use ICF_AdaBoost on the input image without changing aspect ratio, and report experimental results on a CPU. The feature channels include gradient magnitude, six oriented gradients and 3 LUV color channels. Each weak learner is a two-depth tree, where decisions are made by selecting the best channel-related rectangle region. By randomly choosing channel features and rectangle regions, a pool of 30,000 features are generated. The final detector is obtained by four rounds of training with increasing numbers of weak learners (50, 100, 200 and 400). The ICF_AdaBoost works with a 20×20 sliding window and a sliding step of 4 pixels. The same scale factor as the HOG_LDA_SVM is used to construct an image pyramid.

We implemented the detectors by programming in C++ and optimized the codes using SIMD technique. Experimental results were obtained on a single CPU of a PC with i7 3.6GHz core. For accelerating the HHVCas detector in implementation, we replaced float multiplications with integer multiplications in the first two stages.

C. Training

For training the HHVCas detector, the positive sample set is generated by cropping signs from training images with certain margin pixels and jittering by random translation, rotation and scaling. Negative samples are collected by randomly cropping square background regions which have at most 0.3 overlap with the ground truth signs. These two sets are used to train the linear SVM in Stage I. For the subsequent classifiers, we collect training samples by performing detection on the training images with the preceding classifiers. The detected signs that have large overlap (at least 0.7) with an annotated sign are used as additional positives, and the false detects are added to the negative set. For rejection threshold optimization, the linear SVM has an initial threshold 0, and the LDA classifiers for Stages II and III take the minimum scores of positives as thresholds. These thresholds are referred to as base thresholds θ_k^*. The optimal thresholds θ_k^R and θ^S are then determined by the procedure described in Section III-C. Once the first three stages are fixed, we bootstrap the IKSVM in the last stage by collecting hard negatives iteratively, as described in [5]. In the iteration, the IKSVM starts with randomly sampled positives and negatives and threshold -1. The negative set is augmented by the obtained false alarms on training images and the IKSVM is re-trained. This procedure repeats until the HHVCas detects no false alarms or a maximum number of iterations (6 in our experiments) is reached.

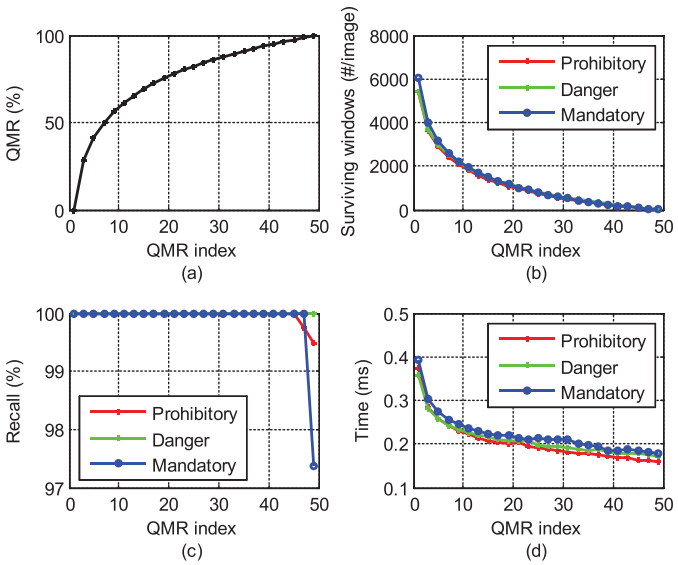

Fig. 8. Evaluation of different QMRs. (a) evaluated QMRs, (b-d): number of surviving windows, recall rate and testing time corresponding to different QMRs.

D. Results on GTSDB

1) Optimal Pruning Thresholds: As described in Section III-C, the rejection thresholds of the first three stages of HHVCas detector are controlled by a quasi-miss rate (QMR). Using the base thresholds as specified in the previous section IV-C, we obtained about 5,000 quasi-positives per image on average. For thresholds optimization, we evaluated different QMRs in terms of the number of surviving windows, recall rate and testing time for each category of signs on the GTSDB training set. We used 50 QMRs in $[0, 1]$ according to a logarithmic distribution, with 25 of them pointed in Fig. 8(a). The numbers of surviving windows, recall rates and testing times corresponding to different QMRs are shown in Fig. 8(b-d). It can be seen that high QMRs prune a large fraction of windows and lead to fast detection speed. The recall rates keep high when QMRs increase within a large range, but decrease in case of excessive QMRs. Based on these statistics, we select several values empirically: the 43th, 44th and 42th QMRs (0.9614,

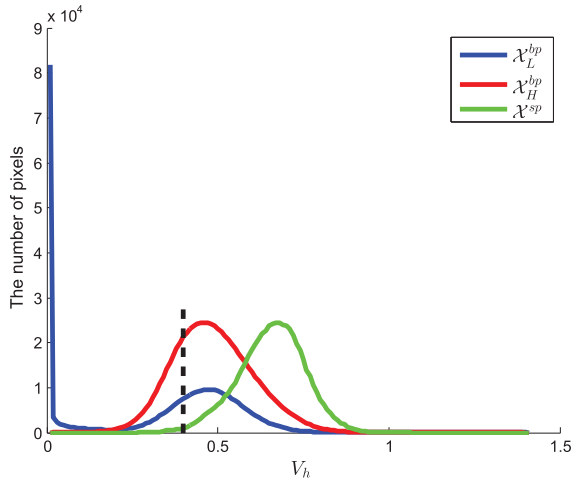

Fig. 9. Histograms of V_h for three sets of pixels. There are some interesting observations: the values of saliency V_h varies within a fairly small range, from 0 to 1.4; a large number of background pixels $\mathcal{X}_L^{bp}$ can be rejected by applying T_{nH} on V_{nH}; the remaining background pixels $\mathcal{X}_H^{bp}$ and sign pixels $\mathcal{X}^{sp}$ are both concentrated and present a separable distribution, which implies that we can apply a threshold reliably on V_h for additional pruning; combined with V_{nH}, a setting of $T_h = 0.4$ eliminates more than 40% background pixels, yet rejects only 0.09% sign pixels, as shown by the black dotted line.

0.9673, 0.9554) for prohibitory, danger and mandatory signs, respectively. Corresponding to the selected QMRs, the stage thresholds of the HHVCas detector are used accordingly.

2) Tuning Saliency Test: To determine the thresholds T_h and T_{nH}, we compute two saliency maps V_h and V_{nH} on the training images. According the pixel values of V_{nH} on training images, we observe that all the traffic sign pixels in the training images are retained with a threshold $T_{nH} = 0.0012$. To select T_h, we analyze the distributions of V_h on the background pixels and sign pixels in training images. For sign pixels $\mathcal{X}^{sp}$, we only count the central rectangle area (0.8 of total area) of prohibitory and mandatory signs, while danger signs are not counted because they are triangular. For background pixels $\mathcal{X}^{bp}$, we count all the pixels in training images, where the sign pixels have little influence because they occupy a very small proportion. We divide $\mathcal{X}^{bp}$ into two subsets according to T_{nH}: $\mathcal{X}_L^{bp} = \{p | p \in \mathcal{X}^{bp}, V_{nH}(p) < T_{nH}\}$ and $\mathcal{X}_H^{bp} = \{p | p \in \mathcal{X}^{bp}, V_{nH}(p) \geq T_{nH}\}$. We build three histograms of V_h for the pixels in $\mathcal{X}^{sp}$, $\mathcal{X}_L^{bp}$ and $\mathcal{X}_H^{bp}$, as shown in Fig. 9. It is seen that the histograms of saliency V_h of background pixels $\mathcal{X}_H^{bp}$ and sign pixels $\mathcal{X}^{sp}$ are dispersed. The histograms show that a threshold $T_h = 0.4$ makes most sign pixels retained (only 0.09% rejected) while more than 40% background pixels can be rejected.

Rejection for an image patch is made by examining the fraction of salient pixels with a threshold T_{area}. For dense patches in an input image, we can calculate this fraction easily through the integral image of the saliency mask. This saliency-based pruning is not suitable for danger signs, because they are triangular and have many background pixels confused in a square search window. For selecting the threshold T_{area}, we test a number of values from 0.8 to 0.9, and found that there is hardly speedup for a value larger than 0.82. When applying saliency test with $T_{area} = 0.82$ to the training images, about 43% pixels and 63% background windows (size from 20×20 to 128×128 pixels) can be rejected without eliminating sign regions.

3) Runtime Analysis: To justify the effects of neighboring scale awareness, we show the detection times of HHVCas without and with this technique in Table II. The time consumed by Stage I and II is divided into two parts, for feature extraction and window evaluation, respectively. Feature extraction in Stage I and II takes about half of the overall detection time, and this time cannot be reduced by window pruning. If we do not use saliency test, the time for window evaluation in the first two stages decreases by about one-third when using neighboring scale pruning. It reduces the total detection time and has little influence on detection performance.

For effects of saliency test in Table II, the time for window evaluation in stage I and II decreases by a half when using saliency-based pruning. This implies that large amount of windows are eliminated by saliency test. The decrease is still significant in the presence of neighboring scale awareness. Processing times in the last two stages are also reduced, but not significantly, because the windows tested in Stages III and IV are fewer and harder. Overall, saliency test consumes additional $22ms$ and saves about $40ms$ for detection by combining with neighboring scale awareness.

4) Detection Results: Table III shows the detection performance in terms of Area Under precision-recall Curves (AUC) and runtime time of the proposed HHVCas detector versus the two baseline detectors on the test images of GTSDB, and Fig. 10 shows the precision-recall curves. It can be seen that our HHVCas detector achieves higher performance than the baseline methods on all the three sign categories. More importantly, it consumes much lest detection time. Using saliency test with the three detectors, the detection speed is improved while the detection performance is preserved. Saliency test even improves the detection performance for the mandatory category, because it precludes background patches that disturb the cascade detector. Saliency test is not used for signs of danger category, which are triangular.

In Table IV, we compare the performance of HHVCas with previous methods that participated in the competition GTSDB and a recently proposed method. High detection performance has been achieved by the top-ranked methods [4]–[6], [18], [19]. Unfortunately, these methods are not applicable for real-time detection on CPU because of the low speed. Note that the time reported by the method [4] was evaluated on GPU. In contrast, our HHVCas detector achieves comparable performance, while running 2∼7 times as fast as most of the previous methods. The recent method of [8] also aims at fast detection. It detects signs of three categories simultaneously, so, the given time is the total time. We can see its performance is promising and the speed is even faster than our HHVCas detector. If we apply HHCVas for three categories together, the detector can use the common integral HOG pyramids across multiple categories and takes about $300ms$ in total. In comparison with the method of [8], our HHVCas achieves higher performance on two of the three categories and is inferior on the mandatory category. It is noteworthy that the method of [8] employs a color probability model to transform

TABLE II

PERFORMANCE AND RUNTIME ON GTSDB WITH/WITHOUT NEIGHBORING SCALE AWARENESS AND SALIENCY TEST

	Neighboring scale awareness (Y/N)	Saliency (Y/N) (ms)	Stage I and II (ms)		Stage III (ms)	Stage IV (ms)	Total time (ms)	AUC (%)
			integral HOG	SVM+LDA				
Prohibitory	N	N(0)	99	56	27	17	199	99.83
	N	Y(19)		26	31	13	188	99.83
	Y	N(0)		45	21	17	184	99.83
	Y	Y(19)		24	17	13	**172**	**99.83**
Danger	N	N(0)	99	58	24	16	197	98.11
	Y	N(0)		43	22	18	**182**	**98.11**
Mandatory	N	N(0)	99	59	26	19	203	95.41
	N	Y(19)		24	27	14	183	**96.62**
	Y	N(0)		42	26	20	187	95.41
	Y	Y(19)		23	17	14	**172**	96.58

TABLE III

COMPARISON BETWEEN HHVCAS AND THE TWO BASELINES ON GTSDB WITH/WITHOUT SALIENCY TEST

		Prohibitory		Danger		Mandatory	
		AUC (%)	Time (ms)	AUC (%)	Time (ms)	AUC (%)	Time (ms)
Without saliency	HHVCas	**99.83**	184	**98.11**	**182**	95.41	187
	HOG_LDA_SVM	99.68	528	97.90	534	91.50	535
	ICF_AdaBoost	97.51	2399	89.61	2369	85.57	2423
With saliency	HHVCas	**99.83**	**172**	-	-	**96.58**	**172**
	HOG_LDA_SVM	99.68	437	-	-	91.68	437
	ICF_AdaBoost	97.53	1326	-	-	85.64	1328

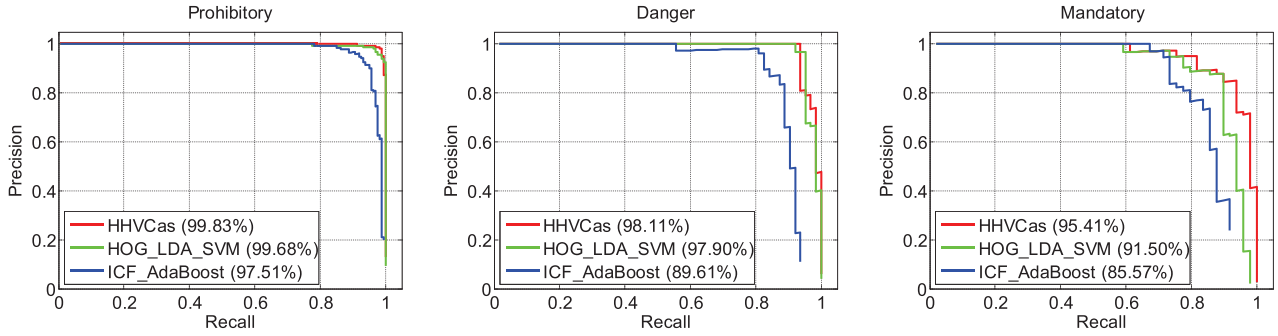

Fig. 10. Precision-recall curves on GTSDB.

TABLE IV

PERFORMANCE AND RUNTIME ON GTSDB WITH COMPARISON TO STATE-OF-THE-ART

Team/Method	Prohibitive		Danger		Mandatory		
	AUC	Time (ms)	AUC	Time (ms)	AUC	Time (ms)	
wgy@HIT501 [5]	100%	∼1122	99.91%	∼1179	100%	∼1232	
visics [4]	100%	∼400*	100%	∼400*	96.98%	∼400*	*GPU
LITS1 [6]	100%	400∼1000	98.85%	400∼1000	92%	400∼1000	
BolognaSVLab [18]	99.98%	∼1667	98.72%	∼794	95.76%	∼571	
NII-UIT	98.11%	-	-	-	86.97%	-	
wff	-	-	99.78%	-	97.62%	-	
milan	-	-	96.55%	-	95.76%	-	
WaDe+MSER [19]	99.99%	∼1667	99.79%	∼794	96.74%	∼571	
CPM_MSER [8]	99.29%	∼162*	97.13%	*	98.17%		*Total time
HHVCas	99.83%	∼184	98.11%	∼182	95.41%	∼187	
HHVCas+Saliency	99.83%	∼172	-	-	96.58%	∼172	

color images into probability maps, where ROIs are extracted by a MSER detector. Thus, it relies on reliable color to certain extent, which is not available in bad illumination conditions. In addition, the MSER extractor has several tunable parameters which are influential to detection performance. On the other hand, our HHVCas exploits little color information and has

TABLE V
PERFORMANCE AND RUNTIME ON STSD WITH/WITHOUT NEIGHBORING SCALE AWARENESS AND SALIENCY TEST

	Neighboring scale awareness (Y/N)	Saliency (Y/N) (ms)	Stage I and II (ms)		Stage III (ms)	Stage IV (ms)	Total time (ms)	AUC (%)
			integral HOG	SVM+LDA				
Prohibitory	N	N(0)	114	59	29	22	224	**86.81**
	N	Y(23)		34	31	17	219	86.48
	Y	N(0)		51	26	26	217	86.75
	Y	Y(23)		29	22	19	**207**	86.42
Mandatory	N	N(0)	114	62	28	31	235	81.39
	N	Y(23)		35	29	25	226	**81.44**
	Y	N(0)		48	30	39	231	81.26
	Y	Y(23)		28	23	29	**217**	81.32

TABLE VI
COMPARISON BETWEEN HHVCas AND THE TWO BASELINES ON STSD WITH/WITHOUT SALIENCY TEST

		Prohibitory		Mandatory	
		AUC (%)	Time (ms)	AUC (%)	Time (ms)
Without saliency	HHVCas	86.75	217	81.26	231
	HOG_LDA_SVM	**86.78**	618	80.50	622
	ICF_AdaBoost	85.01	2823	80.99	2750
With saliency	HHVCas	86.42	**207**	**81.32**	**217**
	HOG_LDA_SVM	86.06	508	80.53	510
	ICF_AdaBoost	84.73	1705	81.05	1613

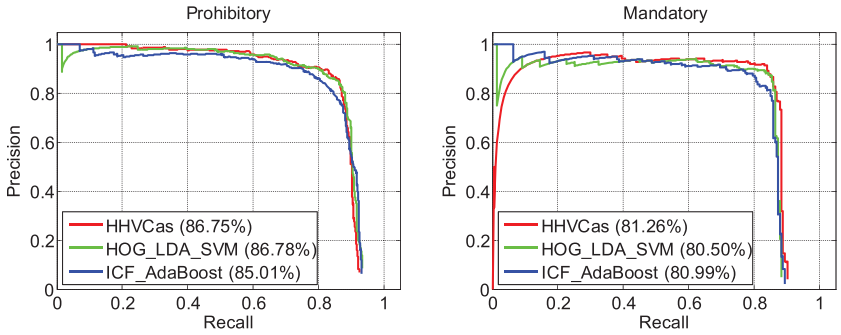

Fig. 11. The precision-recall curves on STSD.

only a single free parameter. For detecting three categories, the HHVCas has three parameters, but they can be selected independently.

E. Results on STSD

To apply the HHVCas detector on the STSD, we optimize the thresholds on the training set using the procedure in Section III-C. The details are omitted here for saving space. For saliency test, we use the same thresholds T_h, T_{nH} and T_{area} estimated on the GTSDB, because these parameters are less dependent on training data.

1) Runtime Analysis: Table V shows the detection times of HHVCas without and with neighboring scale awareness. We can see the processing time on images of STSD is longer than that on GTSDB, because the images in STSD have higher resolution and the size of signs varies in a larger range. The time for window evaluation in the first two stages can be reduced by using neighboring scale pruning, as is like on GTSDB, and this pruning technique deteriorates the detection performance very slightly. We can see the processing time on images of STSD is longer than that on GTSDB, because the images in STSD have higher resolution and the size of signs varies in a larger range. Again, it is demonstrated that saliency test reduces the overall detection time, particularly reducing the number of windows in Stage I and II.

2) Detection Results: The performance on STSD using the HHVCas detector and two baselines is illustrated in Table VI and Fig. 11. It is seen that the performance of all detectors on STSD is worse than that on GTSDB. This is partly due to the confusion between the targeted signs and the other signs excluded from evaluation. Another reason is that many signs, especially the "STOP" sign in prohibitory category, undergo perspective deformation in plane. Table VI shows that our HHVCas detector is faster than the baselines, while its detection performance is superior or comparable to the baselines. For all the detectors, saliency test improves the speed while maintaining the detection performance. The STSD was previously used for evaluating traffic sign recognition [9]. Therefore, those results are not comparable with our results of detection.

The proposed method mainly focuses on traffic signs in frontal view as done in the previous works. So, it lacks the flexibility of detection in scenes with substantial view variation. This remains a research issue in future works. In addition, the performance of our method could be improved

by replacing the last stages with deep neural networks (DNNs) if implemented on GPU, since learned features by DNNs show great promise in recent works of object detection and recognition.

V. Conclusion

We propose a cascade detector called HHVCas for fast traffic sign detection. It uses multiple stage classifiers in coarse-to-fine manner. To evaluate a large number of windows at the first two stages, we design fast feature extraction techniques and use linear classifiers. The Stage III and Stage IV use features of increasing dimensionalities. The Stage I also use neighboring scale awareness to save the computation of window evaluation. The rejection thresholds of HHVCas are optimized jointly by a data-driven approach. In addition, a novel saliency test based on mid-level features is introduced to pre-prune sliding windows while maintaining detection accuracy. Experiments on the GTSDB dataset show that our HHVCas achieves competitive performance in comparison with state-of-the-art methods, while running 2∼7 times as fast as most of them. Compared with a very recent fast method, the HHVCas relies on little color information and has fewer free parameters.

References

[1] A. Møgelmose, M. M. Trivedi, and T. B. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1484–1497, Dec. 2012.

[2] S. Houben, "A single target voting scheme for traffic sign detection," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2011, pp. 124–129.

[3] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2004.

[4] M. Mathias, R. Timofte, R. Benenson, and L. Van Gool, "Traffic sign recognition—How far are we from the solution?" in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2013, pp. 1–8.

[5] G. Wang, G. Ren, Z. Wu, Y. Zhao, and L. Jiang, "A robust, coarse-to-fine traffic sign detection method," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2013, pp. 1–5.

[6] M. Liang, M. Yuan, X. Hu, J. Li, and H. Liu, "Traffic sign detection by ROI extraction and histogram features-based recognition," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2013, pp. 1–8.

[7] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German traffic sign detection benchmark," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2013, pp. 1–8.

[8] Y. Yang, H. Luo, H. Xu, and F. Wu, "Towards real-time traffic sign detection and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 2022–2031, Jul. 2016.

[9] F. Larsson and M. Felsberg, "Using Fourier descriptors and spatial models for traffic sign recognition," in *Proc. Scandin. Conf. Image Anal.*, 2011, pp. 238–249.

[10] D. Wang, S. Yue, J. Xu, X. Hou, and C.-L. Liu, "A saliency-based cascade method for fast traffic sign detection," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2015, pp. 180–185.

[11] X. W. Gao, L. Podladchikova, D. Shaposhnikov, K. Hong, and N. Shevtsova, "Recognition of traffic signs based on their colour and shape features extracted using human vision models," *J. Vis. Commun. Image Represent.*, vol. 17, no. 4, pp. 675–685, Aug. 2006.

[12] W. Liu *et al.*, "A system for road sign detection, recognition and tracking based on multi-cues hybrid," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2009, pp. 562–567.

[13] H. Fleyeh and E. Davami, "Eigen-based traffic sign recognition," *IET Intell. Transp. Syst.*, vol. 5, no. 3, pp. 190–196, Sep. 2011.

[14] J. F. Khan, S. M. A. Bhuiyan, and R. R. Adhami, "Image segmentation and shape analysis for road-sign detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 83–96, Mar. 2011.

[15] S. Maldonado-Bascón, S. Lafuente-Arroyo, P. Gil-Jiménez, H. Gómez-Moreno, and F. López-Ferreras, "Road-sign detection and recognition based on support vector machines," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 264–278, Jun. 2007.

[16] H. Gómez-Moreno, S. Maldonado-Bascón, P. Gil-Jiménez, and S. Lafuente-Arroyo, "Goal evaluation of segmentation algorithms for traffic sign recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 4, pp. 917–930, Dec. 2010.

[17] J. Greenhalgh and M. Mirmehdi, "Real-time detection and recognition of road traffic signs," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1498–1506, Dec. 2012.

[18] S. Salti, A. Petrelli, F. Tombari, N. Fioraio, and L. D. Stefano, "A traffic sign detection pipeline based on interest region extraction," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Aug. 2013, pp. 1–7.

[19] S. Salti, A. Petrelli, F. Tombari, N. Fioraio, and L. D. Stefano, "Traffic sign detection via interest region extraction," *Pattern Recognit.*, vol. 48, no. 4, pp. 1039–1049, Apr. 2015.

[20] N. Barnes and A. Zelinsky, "Real-time radial symmetry for speed sign detection," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2004, pp. 566–571.

[21] G. Loy and N. Barnes, "Fast shape-based road sign detection for a driver assistance system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2004, pp. 70–75.

[22] B. Höferlin and K. Zimmermann, "Towards reliable traffic sign recognition," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2009, pp. 324–329.

[23] M. A. García-Garrido, M. Ocaña, D. F. Llorca, M. A. Sotelo, E. Arroyo, and A. Llamazares, "Robust traffic signs detection by means of vision and V2I communications," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Oct. 2011, pp. 1003–1008.

[24] R. Belaroussi and J.-P. Tarel, "Angle vertex and bisector geometric model for triangular road sign detection," in *Proc. IEEE Workshop Appl. Comput. Vis.*, Dec. 2009, pp. 1–7.

[25] R. Belaroussi and J.-P. Tarel, "A real-time road sign detection using bilateral chinese transform," in *Proc. Adv. Vis. Comput.*, 2009, pp. 1161–1170.

[26] M. Boumediene, C. Cudel, M. Basset, and A. Ouamri, "Triangular traffic signs detection based on RSLD algorithm," *Mach. Vis. Appl.*, vol. 24, no. 8, pp. 1721–1732, Nov. 2013.

[27] F. Parada-Loira and J. L. Alba-Castro, "Local contour patterns for fast traffic sign detection," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2010, pp. 1–6.

[28] I. Landesa-Vázquez, F. Parada-Loira, and J. L. Alba-Castro, "Fast real-time multiclass traffic sign detection based on novel shape and texture descriptors," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Sep. 2010, pp. 1388–1395.

[29] R. Timofte, K. Zimmermann, and L. Van Gool, "Multi-view traffic sign detection, recognition, and 3D localisation," *Mach. Vis. Appl.*, vol. 25, no. 3, pp. 633–647, Apr. 2014.

[30] A. Ruta, F. Porikli, S. Watanabe, and Y. Li, "In-vehicle camera traffic sign detection and recognition," *Mach. Vis. Appl.*, vol. 22, no. 2, pp. 359–375, Mar. 2011.

[31] L. Chen, Q. Li, M. Li, and Q. Mao, "Traffic sign detection and recognition for intelligent vehicle," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2011, pp. 908–913.

[32] S. Šegvić, K. Brkić, Z. Kalafatić, and A. Pinz, "Exploiting temporal and spatial constraints in traffic sign detection from a moving vehicle," *Mach. Vis. Appl.*, vol. 25, no. 3, pp. 649–665, Apr. 2004.

[33] C. Bahlmann, Y. Zhu, V. Ramesh, M. Pellkofer, and T. Koehler, "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2005, pp. 255–260.

[34] B. Alefs, G. Eschemann, H. Ramoser, and C. Beleznai, "Road sign detection from edge orientation histograms," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2007, pp. 993–998.

[35] D. Deguchi, M. Shirasuna, K. Doman, I. Ide, and H. Murase, "Intelligent traffic sign detector: Adaptive learning based on online gathering of training samples," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2011, pp. 72–77.

[36] N. Pettersson, L. Petersson, and L. Andersson, "The histogram feature—A resource-efficient Weak Classifier," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2008, pp. 678–683.

[37] G. Overett, L. Petersson, L. Andersson, and N. Pettersson, "Boosting a heterogeneous pool of fast HOG features for pedestrian and sign detection," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2009, pp. 584–590.

[38] G. Overett, L. Tychsen-Smith, L. Petersson, N. Pettersson, and L. Andersson, "Creating robust high-throughput traffic sign detectors

using centre-surround HOG statistics," *Mach. Vis. Appl.*, vol. 25, no. 3, pp. 713–726, Apr. 2014.
[39] A. Møgelmose, D. Liu, and M. M. Trivedi, "Traffic sign detection for U.S. roads: Remaining challenges and a case for tracking," in *Proc. IEEE 17th Int. Conf. Intell. Transp. Syst.*, Oct. 2014, pp. 1394–1399.
[40] C. Liu, F. Chang, and Z. Chen, "Rapid multiclass traffic sign detection in high-resolution images," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 6, pp. 2394–2403, Dec. 2014.
[41] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.
[42] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[43] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed. (Dec. 2015). "SSD: Single shot multibox detector." [Online]. Available: https://arxiv.org/abs/1512.02325
[44] N. Dalal, "Finding people in images and videos," Ph.D. dissertation, Institut National Polytechnique Grenoble, Grenoble, France: 2006.
[45] Q. Zhu, M.-C. Yeh, K.-T. Cheng, and S. Avidan, "Fast human detection using a cascade of histograms of oriented gradients," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2. Jun. 2006, pp. 1491–1498.
[46] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, vol. 1, no. 1, pp. 886–893.
[47] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.
[48] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 8, pp. 1532–1545, Aug. 2014.
[49] G. Gualdi, A. Prati, and R. Cucchiara, "Multi-stage sampling with boosting cascades for pedestrian detection in images and videos," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 196–209.
[50] P. Dollár, R. Appel, and W. Kienzle, "Crosstalk cascades for frame-rate pedestrian detection," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 645–659.
[51] R. Achanta, F. Estrada, P. Wils, and S. Süsstrunk, "Salient region detection and segmentation," in *Proc. Int. Conf. Comput. Vis. Syst.*, 2008, pp. 66–75.

Dongdong Wang received the B.E. degree in information engineering from Jilin University, Changchun, China, in 2006. He is currently working toward the Ph.D. degree with the Pattern Analysis and Learning Group, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His main research interests include computer vision and pattern recognition and their applications in intelligent transportation systems.

Xinwen Hou received the B.S. degree from Zhengzhou University, Henan, China, in 1995; the M.S. degree from University of Science and Technology of China, Anhui, China, in 1998; the Ph.D. degree from the Department of Mathematics, Beijing University, Beijing, China, in 2001. From 2001 to 2003, he did Post-Doctoral Research with the Department of Mathematics, Nankai University, Tianjin, China. In 2003, he joined the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, where he is currently an Associate Professor. He has authored over ten papers on the international journals and conferences. His research interests include machine learning, video semantics understanding, and face recognition.

Jiawei Xu received the B.E. degree in automotive engineering from Shanghai University of Engineering Science and Technology, Shanghai, China, in 2007; the M.E. degree in electronic engineering from Hallym University, South Korea, in 2010; and the Ph.D. degree in computer science from University of Lincoln, U.K., in 2015. He is currently a Post-Doctoral Research Fellow with the School of Computer Science, University of Lincoln. His research interests include visual attention model, visual searching simulation, and bioinspired visual intelligence. He received the Best Paper in Korean Multimedia Society in 2010.

Shigang Yue (M'05) received the B.Eng. degree from Qingdao Technological University in 1988, and the M.Sc. and Ph.D. degrees from Beijing University of Technology (BJUT) in 1993 and 1996, respectively. He was with BJUT as a Lecturer from 1996 to 1998 and an Associate Professor from 1998 to 1999, and also with City University of Hong Kong (MEEM) as a Senior Research Assistant from 1998 to 1999. He was an Alexander von Humboldt Research Fellow with Prof. Henrich with the Faculty of Computer Science, University of Kaiserslautern, Germany, in 2000 and 2001, respectively. Before joining University of Lincoln as a Senior Lecturer in 2007 and promoted to a Reader in 2010, he held research positions with the University of Cambridge from 2006 to 2007, Newcastle University from 2003 to 2006, and University College London from 2002 to 2003, respectively. He has been a Professor with the School of Computer Science, University of Lincoln, U.K., since 2012. He is the Founding Director of the Computational Intelligence Laboratory, Lincoln. His research interests are mainly within the field of artificial intelligence, computer vision, robotics, information processing, brains, and neuroscience. He is particularly interested in biological visual neural systems, evolution of neuronal systems and applications, e.g., in collision detection for vehicles, interactive systems, and robots. He is the Coordinator for several EU FP7 projects. He is a member of INNS, ISAL, and ISBE.

Cheng-Lin Liu (F'15) received the B.S. degree in electronic engineering from Wuhan University, Wuhan, China; the M.E. degree in electronic engineering from Beijing University of Technology, Beijing, China; and the Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation of Chinese Academy of Sciences, Beijing, China, in 1989, 1992, and 1995, respectively. He was a Post-Doctoral Fellow with Korea Advanced Institute of Science and Technology and later with Tokyo University of Agriculture and Technology from 1996 to 1999. From 1999 to 2004, he was a Research Staff Member and later a Senior Researcher with the Central Research Laboratory, Hitachi, Ltd., Tokyo, Japan. Since 2005, he has been a Professor with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China, where he is currently the Director. His research interests include pattern recognition, image processing, neural networks, machine learning, and the applications to character recognition and document analysis. He has authored over 200 technical papers at prestigious international journals and conferences. He is a Fellow of the IAPR.