Real Time Traffic Light Detection with Adaptive Background Suppression Filter

Zhenwei Shi, *Member, IEEE*, Zhengxia Zou and Changshui Zhang, *Member, IEEE*

Abstract

Traffic light detection plays an important role in intelligent transportation system and many detection methods have been proposed in recent years. However, illumination variation effect is still of its major technical problem in real urban driving environments. In this paper, we propose a novel vision-based traffic light detection method for driving vehicles, which is fast and robust under different illumination conditions. The proposed method contains two stages: the candidate extraction stage and the recognition stage. On the candidate extraction stage, we propose an adaptive background suppression algorithm to highlight the traffic light candidate regions while suppress the undesired backgrounds. On the recognition stage, each candidate region is verified and is further classified into different traffic light semantic classes. We evaluate our method on video sequences (more than 5000 frames and labels) captured from urban streets and suburb roads in varying illumination and compared with other vision-based traffic detection approaches. The experiment shows the proposed method can achieve a desired detection result with high quality and robustness, simultaneously, the whole detection system can meet the real-time processing requirement of about 15*fps* on video sequences.

Index Terms

Traffic light detection, Adaptive background suppression filter, Support vector machine.

I. Introduction

Recently, researches on intelligent vehicles, especially which autonomously drive in urban environment have become more popular. Detecting the state of traffic lights and understanding their semantics at interactions is

This work was supported by the National Natural Science Foundation of China under Grant 61273245 and Grant 91120301, the Beijing Natural Science Foundation under Grant 4152031, the funding project of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University under Grant VR-2014-ZZ-02, the Fundamental Research Funds for the Central Universities under Grant YWF-14-YHXY-028 and Grant YWF-15-YHXY-003. *(Corresponding author: Zhenwei Shi.)*

Zhenwei Shi (Corresponding Author) is with State Key Laboratory of Virtual Reality Technology and Systems, School of Astronautics, Beihang University, Beijing 100191, China and with Beijing Key Laboratory of Digital Media, Beihang University, Beijing 100191, China and also with Image Processing Center, School of Astronautics, Beihang University, Beijing 100191, China, (e-mail: shizhenwei@buaa.edu.cn).

Zhengxia Zou is with State Key Laboratory of Virtual Reality Technology and Systems, School of Astronautics, Beihang University, Beijing 100191, China and with Beijing Key Laboratory of Digital Media, Beihang University, Beijing 100191, China and also with Image Processing Center, School of Astronautics, Beihang University, Beijing 100191, China, (e-mail: zhengxiazou@buaa.edu.cn).

Changshui Zhang is with the State Key Laboratory on Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: zcs@mail.tsinghua.edu.cn).

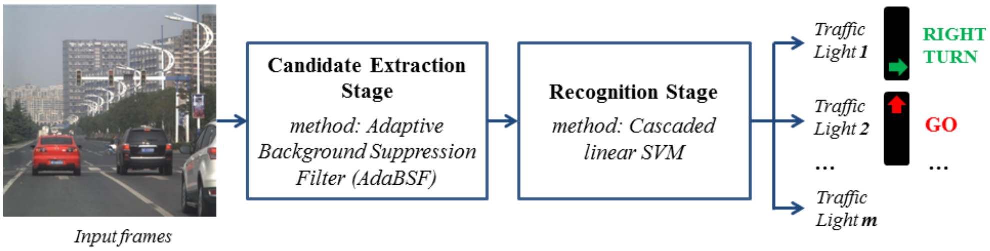

Input frames
Candidate Extraction Stage
method: Adaptive Background Suppression Filter (AdaBSF)
Recognition Stage
method: Cascaded linear SVM
Traffic Light 1
RIGHT TURN
Traffic Light 2
GO
...
...
Traffic Light m

Fig. 1. Procedure of the proposed traffic light detection method.

essential for autonomous driving in real-world situations [1]. Some researches have focused on the vision-based traffic light detection methods by analysing the traffic lights' states from the frames captured by the onboard camera automatically.

In early works, Yung *et* al. proposed a vision-based method to detect traffic lights in order to recognize the red light runners in 2001 [2]. Other early traffic light detection methods [3–7] are built based on various techniques such as Markov Random Field, Neural Networks, Affine moment invariant, fuzzy method and morphological techniques. However, some of these methods [2, 5–7] require that the camera equipped at a fixed place nearby the traffic lights, and some of them [3, 6] cannot meet the real-time processing requirement. Therefore, these traffic light detection methods are not very applicable for intelligent vehicles. Some researchers have done some works on estimating the distance from the vehicle to the traffic lights by image processing techniques [1]. The researchers of [8] and [9] use GPS data and digital maps to identify the traffic lights in urban environments. In [10] and [1], the traffic lights are first labeled from the recorded video data, and then localized by image mapping techniques based on high precision GPS. Apart from these GPS-based detection methods, some purely vision-based traffic light detection methods have been adopted recently [11–26].

Procedures of the purely vision-based methods can be grouped into two stages, 1) the candidate extraction stage and 2) the recognition stage. In the first stage, color thresholding techniques are used to get the candidate regions of the light emitting units [11, 13, 14, 17–20, 22]. Among these methods, the HSI space [11, 13, 19, 20], and normalized RGB space [17, 18, 22] are commonly used. The threshold value of the above methods are usually chosen by hand, but such skills may lack flexibility and physical meaning. In [14], the authors extract the light emitting units by modeling the H and S value in HSI space according to a 2D Gaussian distribution, and the parameters are learned by training images. However, the processing speed of their method is slow and can not be applied in real time. Apart from these methods, researchers have not always used color information. Shape information based methods are also frequently used, including the well-known Hough transform [17], [18], fast radial symmetry transform [23] and spot light detection [14–16]. In the second stage, decisions are made on each candidate region, and their states are further determined by analysing the color and shape of the light emitting units

[1] http://www.lara.prd.fr/benchmarks

and their surrounding area. Previous works of this stage mainly focus on template matching methods [11, 15, 20], symmetry information based methods [17, 18] and machine learning based methods [19, 21, 24].

General traffic light detection methods mainly focus on locating the traffic lights in each frame and understanding their control semantics. The former one aims to determine the location of traffic lights in each frame, and the later one aims to recognize them as different light types (stop, go, no-left-turn, etc.). Detecting traffic lights in real driving environments is not an easy task. Although many researchers make efforts to provide the reliability required by the intelligent vehicles to safely pass through intersections, most of these methods, more or less, are defective in real driving environments. The color of the emitting units will vary a lot under dynamic light conditions with most video cameras. When the vehicles are driving under low contrast environments such as in cloudy and nightfall, the detection systems usually suffer from the lost of shape and edge information of traffic lights. When vehicles are driving against the strong sunlight, the light emitting units may lost color information (even vanish) in the bright sky. Under such conditions, the performances of some above mentioned methods will be greatly affected. Among these methods, color thresholding methods could fail to segment the light emitting candidates, template matching methods and symmetry based methods are also instable under extreme conditions i.e. partial lost under strong sunlight and shape distortion under low contrast conditions. Besides, the hough transform based detection methods in [17, 18] and the fast radial symmetry transform method in [23] can not well distinguish traffic lights from tail lights. Apart from these, the time efficiency of the detection algorithm also plays an important role in real driving environment. Since the image sequences share a high correlation (spatiotemporal persistency) between the two adjacent frames, some tracking or state routine estimation methods [19, 22, 23, 27] are also integrated to improve the real time performance and the accuracy. Thus, the major obstacles for traffic light detection can be summarized as two factors, 1) illumination variations and 2) time efficiency.

Considering the above factors, we propose a novel purely vision-based traffic lights detection method which is robust to different illuminations in real driving environments. In other related fields, i.e., traffic sign detection and pedestrian detection, the most commonly used sliding-window detection approach usually suffers from the large computation of the exhaustive search and the multi-scale problems [28]. In order to reduce the number of detection windows, training an objectness measure to produce a small set of candidate windows has been shown to speed up some sliding window approaches [28–32]. In this paper, we first propose a new objectness detection method, Adaptive Background Suppression Filter (AdaBSF) to search for the traffic lights' candidate regions from a certain frame. AdaBSF is fast, accurate and robust to different illuminations. It reflects the traffic light's objectness, say, how likely an detection window covers a traffic light of any category. It assigns a score to each pixel of the input image, which represents the degree of pixel's belongness to a certain traffic light area. By using AdaBSF, the outputs of undesired backgrounds are suppressed while the outputs of possible traffic light area are highlighted. On recognition stage, of which the task can be seen as a basic classification problem, each candidate window is verified and further classified into different classes. The local color histograms [33] and Histograms of Oriented Gradients (HOG) [34] are used as two basic feature descriptors, and the cascaded linear Support Vector Machine (SVM) [35] is used as the classifier. Procedure of the proposed traffic light detection system is illustrated in Fig. 1,

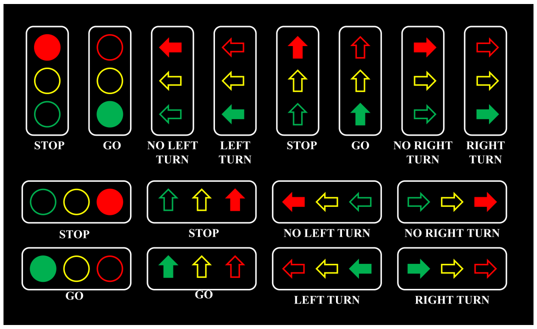


Fig. 2. 16 basic types of traffic lights that can be detected by our system.

where in the first stage, the traffic light candidate regions are extracted by the proposed AdaBSF method. Then, in the second stage, each candidate region is further verified and recognized into different semantic types, e.g. right turn, go and etc.

Traffic lights have various appearance and there are significant differences in how they mounted and positioned. Most existing systems only focus on either “circle” type or “arrow” type detection [24], which limits their application in the real world environment. Fig. 2 illustrates the basic types of traffic lights that can be detected in our system. Our system covers 16 common types of traffic lights in urban environment. To our best knowledge, there is only one public dataset for traffic light detection task [15], thus we also provide a new dataset which consists of more than 5000 frames and labels from 8 video sequences. This dataset is collected in urban streets and suburb roads by a moving vehicle in different light conditions, i.e, clear day, cloudy day and under strong sunlight, which is more challenging and closer to real driving environment.

The rest of this paper is organized as follows. In Section II, we introduce AdaBSF and some theoretical analyses. In section III we introduce our traffic light verification and classification method. Some experimental results are given in Section IV and conclusions are drawn in Section V.

II. Adaptive background suppression filter

For most traffic light detection methods, the sliding window detection approaches usually suffer from large computation and multi-scale problem. Thus, candidate extraction methods are commonly used. In this section, we will first introduce our candidate extraction method AdaBSF, and then we will discuss its parameter settings.

A. AdaBSF: Motivation

A good traffic light candidate extraction method should meet the following requirements:

1) *High recall rate*: candidate regions should cover all potential targets since any undetected targets at this stage cannot be recovered later.
2) *Robustness*: the algorithm should be stable under different illuminations and interferences.
3) *Generalization ability*: the algorithm should have good generalization ability to unseen types of targets.

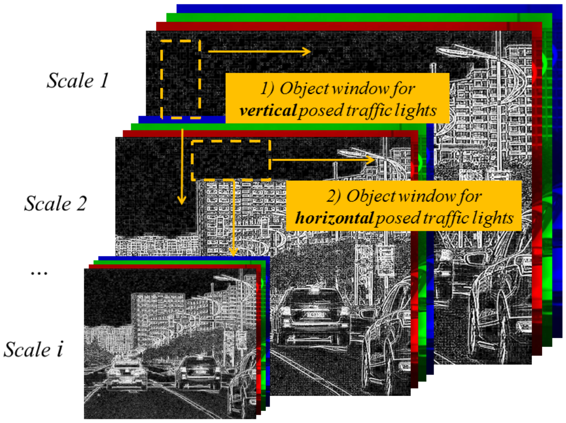

Scale 1
Scale 2
...
Scale i
1) Object window for **vertical** posed traffic lights
2) Object window for **horizontal** posed traffic lights

Fig. 3. Structure of AdaBSF. A 4-channel (R, G, B and normalized gradient) feature map is used. In each frame of detection, the feature map is down-sampled to different scales while the detection window is remained as fixe size.

4) *High computational efficiency*: this is required especially for onboard traffic light detection systems.
5) *Low false alarm rate*: on basis of meeting all above requirements, the algorithm should reduce the amount of computation of subsequent detectors.

Unfortunately, no prior methods satisfied all these goals simultaneously. On overall consideration of above requirements, we proposed AdaBSF algorithm. Researches from cognitive psychology suggest that human brains first perceive objects by simple edge features in perceive layer, then identify them in subsequent layers. In AdaBSF algorithm, the normalized gradients of an input image are first calculated as the simple edge features. Then, combined with their original R, G, B raw pixels values, we obtain a simple 4-channel feature map. Each detection window W_i in this feature map can be represented as a basic feature vector $\mathbf{x}_i \in \mathbb{R}^{D\times 1}$. To search for the vertical and horizontal posed traffic lights, the window size is fixed as 16×8 pixels and 8×16 pixels, respectively. Thus, the dimension of each detection window is $D = 16 \times 8 \times 4 = 256$. In order to solve the multi-scale problem, we down-sample the original image to different scales while keep the object window as fixed size. The aim of AdaBSF algorithm is to design an Finite Impulse Response (FIR) filter specified by the vector $\mathbf{w} = [w_1, w_2, \ldots, w_{256}]^T$ that

$$y_i = \mathbf{w}^T \mathbf{x}_i. \tag{1}$$

The output y_i assigns a score to each detection window, which represents how likely the detection window covers a traffic light. The structure of AdaBSF algorithm is shown in Fig. 3.

B. AdaBSF: Model

Suppose $\mathbf{x}^b$ and $\mathbf{x}^t$ represent the background window and target window of an input frame, respectively. The expectation of output energy from background windows can be represented as

$$\begin{aligned} E\{y_b^2\} &= E\{(\mathbf{w}^T\mathbf{x}^b)^2\} \\ &= \mathbf{w}^T E\{\mathbf{x}^b\mathbf{x}^{bT}\}\mathbf{w} \\ &= \mathbf{w}^T\mathbf{R}_b\mathbf{w}, \end{aligned} \tag{2}$$

where $\mathbf{R}_b = E\{\mathbf{x}_i^b\mathbf{x}_i^{bT}\}$ represents the correlation matrix, which is accumulated from background windows which do not contain any traffic lights. We hope to design an FIR filter $\mathbf{w}$ which suppresses the backgrounds $\mathbf{x}^b$ while preserves the traffic lights $\mathbf{x}^t$, which means to minimize the output energy of all background windows subject to a constraint that the filter's response to $\mathbf{x}^t$ is a fixed value (e.g. 1):

$$\begin{aligned} \min_{\mathbf{w}} \quad & \mathbf{w}^T\mathbf{R}_b\mathbf{w} \\ s.t. \quad & \mathbf{w}^T\mathbf{x}^t = 1. \end{aligned} \tag{3}$$

However, the appearance of traffic lights may vary a lot due to the complex light conditions. To enhance the robustness of the filter, more traffic light samples with different appearances are added to the constraint term. Considering that the feasible region may become empty when the number of constraints becomes very large, we relax the constraints by introducing nonnegative variables $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_{N_t})^T$, where N_t represents the number of constraints. The AdaBSF can be obtained by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi}} \quad & J(\mathbf{w},\boldsymbol{\xi}) = \mathbf{w}^T\mathbf{R}_b\mathbf{w} + \alpha\sum_{i=1}^{N_t}\xi_i + \beta\|\mathbf{w}\|_2^2 \\ s.t. \quad & \mathbf{w}^T\mathbf{x}_i^t \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, 2, \ldots, N_t \end{aligned} \tag{4}$$

where α and β are two positive weight coefficients. The objective function $J(\mathbf{w},\boldsymbol{\xi})$ consists of three parts. The first part $\mathbf{w}^T\mathbf{R}_b\mathbf{w}$ represents the expectation of output energy of background windows, which is used to suppress the undesired backgrounds. The second part $\alpha\sum_{i=1}^{N_t}\xi_i$ represents the penalty variables, which is used to highlight the traffic lights. It is a convex relaxation for the number of targets which do not satisfy the constraints $\mathbf{w}^T\mathbf{x}_i^t \geq 1$. The third part $\beta\|\mathbf{w}\|_2^2$ is a regularization term to reduce the model's complexity and also to increase its generalization ability.

It should be noticed that we do not add background samples to constraints, which means that any misclassification of background samples here will not contribute to the penalty variables $\alpha\sum_{i=1}^{N_t}\xi_i$. This is mainly because that objectness detection is different to object classification problems. Classification problems assign $\{-1, +1\}$ labels to the positive and negative classes. It treats two classes equally in objective function, except that their labels are different. In most existing classification algorithms, any misclassification of positive and negative samples both contributes to objective function at the same time. However, in candidate region detection problems, we prefer to concentrate more on targets than backgrounds. This is because that any of undetected targets cannot be recovered in the following process.

C. AdaBSF: Optimization

Define $\tilde{\mathbf{w}}$, $\mathbf{f}$, $\mathbf{H}$, $\mathbf{A}$ and $\mathbf{b}$ by the following transformation

$$\tilde{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\xi} \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} \mathbf{0}_D \\ \alpha \mathbf{1}_{N_t} \end{bmatrix}, \quad \mathbf{b} = -\begin{bmatrix} \mathbf{1}_{N_t} \\ \mathbf{0}_{N_t} \end{bmatrix}, \tag{5}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{R}_b + \beta \mathbf{I} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} \end{bmatrix}, \quad \mathbf{A} = -\begin{bmatrix} \mathbf{X}^t & \mathbf{I} \\ \mathbf{O} & \mathbf{I} \end{bmatrix}, \tag{6}$$

where $\mathbf{H} \in \mathbb{R}^{(D+N_t)\times(D+N_t)}$, $\mathbf{A} \in \mathbb{R}^{(2N_t)\times(D+N_t)}$, $\mathbf{X}^t = [\mathbf{x}_1^t, \mathbf{x}_2^t, \ldots, \mathbf{x}_{N_t}^t]^T \in \mathbb{R}^{N_t \times D}$. $\mathbf{1}_n = [1, 1, \ldots, 1]^T \in \mathbb{R}^{n\times 1}$ is a unit vector, $\mathbf{0}_n = [0, 0, \ldots, 0]^T \in \mathbb{R}^{n\times 1}$ is a zero vector. $\mathbf{I}$ and $\mathbf{O}$ are identical matrix and zero matrix of corresponding sizes. Then, we can rewrite (4) as the following quadratic programming problem

$$\begin{aligned} \min_{\tilde{\mathbf{w}}} \quad & J(\tilde{\mathbf{w}}) = \tilde{\mathbf{w}}^T \mathbf{H} \tilde{\mathbf{w}} + \mathbf{f}^T \tilde{\mathbf{w}} \\ s.t. \quad & \mathbf{A}\tilde{\mathbf{w}} \leq \mathbf{b}. \end{aligned} \tag{7}$$

In this paper, we introduce an interior method named barrier method [36] to solve (7). Define

$$c_i(\tilde{\mathbf{w}}) = log(\mathbf{a}_i^T \tilde{\mathbf{w}} - b_i) \tag{8}$$

as the log barrier function, and $\mathbf{a}_i^T \tilde{\mathbf{w}} - b_i$ corresponds to the ith constrain of (7). By using (8), (7) can be approximated by the following unconstrained optimization problem

$$\min_{\tilde{\mathbf{w}}} \quad B_L(\tilde{\mathbf{w}}) = J(\tilde{\mathbf{w}}) - \frac{1}{\mu} \sum_{i=1}^{D+N_t} log(c_i(\tilde{\mathbf{w}})) \tag{9}$$

where $\mu > 0$ represents the barrier factor which sets the accuracy of the approximation. Since $\mathbf{H}$ is a positive definite matrix, thus $B_L(\tilde{\mathbf{w}})$ is a convex function, and we can use Newton's method to solve (9). It can be proved that when $\mu \to \infty$, (7) and (9) become equivalent [36]. The basic idea of the barrier method is to increase μ and solve the corresponding problem (9) repeatedly. In this way, the solution of (9) approaches the optimal point of original problem (7) gradually. Algorithm 1 gives the solution of AdaBSF algorithm.

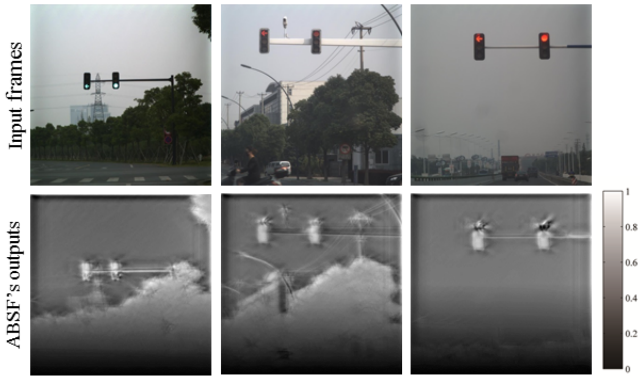


Fig. 4. AdaBSF's results of frame captured under different light conditions. The first row shows the original input frames. The second row shows the AdaBSF's filtering results. Noticed that traffic lights usually appears in the upper half part of the scene, we have manually reduced the score at the very bottom of each output map.

Algorithm 1 Barrier method for designing AdaBSF

```
Initialization:
    1: Select a strictly feasible $\tilde{\mathbf{w}}$ that satisfies $\mathbf{A}\tilde{\mathbf{w}} \leq \mathbf{b}$. Set $\mu > 0$, tolerance $\epsilon_1 > 0$ and $\epsilon_2 > 0$.
Main iteration:
    2: Use $\tilde{\mathbf{w}}$ as the start point.
    Subiteration(solve (9) by Newton's method):
    3: Compute the gradient and Hessian matrix of (9):
       $\nabla B_L(\tilde{\mathbf{w}}) = 2\mathbf{H}\tilde{\mathbf{w}} + \mathbf{f} - \frac{1}{\mu}\sum_{i=1}^{D+N_t} \frac{\mathbf{a}_i}{\mathbf{a}_i^T\tilde{\mathbf{w}} - b_i}$,
       $\nabla^2 B_L(\tilde{\mathbf{w}}) = 2\mathbf{H} + \frac{1}{\mu}\sum_{i=1}^{D+N_t} \frac{\mathbf{a}_i\mathbf{a}_i^T}{(\mathbf{a}_i^T\tilde{\mathbf{w}} - b_i)^2}$.
    4: Compute the descent direction $\mathbf{d}$:
       $\mathbf{d} = -\nabla^2 B_L(\tilde{\mathbf{w}})^{-1}\nabla B_L(\tilde{\mathbf{w}})$.
    5: Choose a step size $\rho$ by exact line search:
       $\rho = arg\min_{\rho>0} \ B_L(\tilde{\mathbf{w}} + \rho\mathbf{d})$.
    6: Update $\tilde{\mathbf{w}} \leftarrow \tilde{\mathbf{w}} + \rho\mathbf{d}$.
    Subiteration stop criterion: if $\|\nabla B_L(\tilde{\mathbf{w}})\|_2^2 < \epsilon_1$,
    else: go back to 3.
Main iteration stop criterion: if $\frac{1}{\mu} < \epsilon_2$,
else: increase $\mu \leftarrow 2\mu$ and go back to 2.
```

After obtaining the optimal solution $\tilde{\mathbf{w}}^\star = \begin{bmatrix} \mathbf{w}^\star \\ \boldsymbol{\xi}^\star \end{bmatrix}$, the AdaBSF's output of an detection window W_i is given by

$$y_i = \mathbf{w}^{\star T}\mathbf{x}_i. \tag{10}$$

Then, we compare y_i with a threshold T,

After this progress, there are two steps going from the filtered images to the candidate windows. In the first step, we compare the AdaBSF output value y_i with a threshold T, if $y_i \geq T$, the current window is identified as a target window, and need to be further processed. If $y_i < T$ the current window is identified as a background window. In the second step, to further reduce the number of false alarms, non-maximal suppression [28] is used at the each scale after getting the filtering results. Concretely, in each scale, each pixel of the result image is compared with its neighbors of the corresponding neighbor size. If the current pixel value equals the max value in its neighbors, it is then identified as the center of a target window with the corresponding window size. Finally, the results from different scales are merged together. In order to make the filter robust to illumination changes, traffic light samples taken in different time period of a day are added, e.g. during the morning time, the noon, and nightfall. Fig. 4 shows three typical scene images and their AdaBSF's corresponding filtering results. Although the light condition changes and the color of the light emitting units vary a lot in these frames, we can still see that AdaBSF suppresses the undesired backgrounds while highlights all the traffic lights successfully.

D. AdaBSF: Parameter Analysis

Two important parameters of AdaBSF algorithm are penalty coefficient α and regularization coefficient β.

The first parameter α is used to control the amount of penalty required in (4). An appropriate choice of α is important to AdaBSF algorithm. By using the hinge-loss function

$$H(t) = \begin{cases} 0 & t \geq 0 \\ -t & t < 0 \end{cases} \tag{11}$$

(4) can be transformed to the following unconstrained optimization problem

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi}} \; J(\mathbf{w},\boldsymbol{\xi}) &= \mathbf{w}^T\mathbf{R}_b\mathbf{w} + \alpha\sum_{i=1}^{N_t} H(\mathbf{w}^T\mathbf{x}_i^t - 1) + \beta\|\mathbf{w}\|_2^2 \\ &= E\{(\mathbf{w}^T\mathbf{x}^b - z^b)^2\} + \alpha N_t E\{H(\mathbf{w}^T\mathbf{x}_i^t - z^t)\} \\ &\quad + \beta\|\mathbf{w}\|_2^2, \end{aligned} \tag{12}$$

where $z^b = 0$ and $z^t = 1$ represents the labels of the background and target classes. We can see that, the first two terms of (12) have the similar formulation, except that their cost functions are different: the first one is least squares while the the second one is hinge-loss. Thus, the corresponding parameters in front of these two terms should have comparable values. Thus, α should be chosen as inversely proportional to the number of constrains, and α and $1/N_t$ should have the same order of magnitude $\alpha \sim 1/N_t$.

The second parameter β is also important. Suppose λ_i represents the ith eigenvalue of the matrix $\mathbf{R}_b$, $i = 1, 2, \ldots, N_t$. The condition numbers [36] of matrices $\mathbf{R}_b$ and $\mathbf{R}_b + \beta\mathbf{I}$ can be expressed as follows

$$\begin{aligned} \kappa(\mathbf{R}_b) &= \frac{\lambda_{max}}{\lambda_{min}}, \\ \kappa(\mathbf{R}_b + \beta\mathbf{I}) &= \frac{\lambda_{max} + \beta}{\lambda_{min} + \beta}. \end{aligned} \tag{13}$$

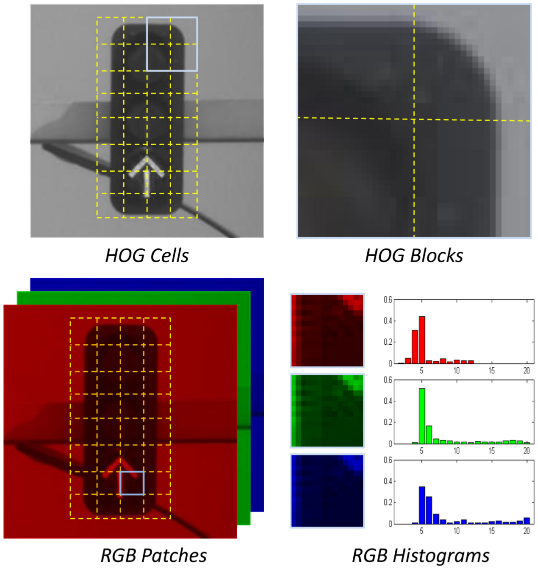


Fig. 5. The HOG descriptor and the RGB histograms are used as two basic feature descriptors in our cascaded SVM classifiers.

When $\lambda_{max} \gg \lambda_{min}$, the inverse of matrix $\mathbf{R}_b$ becomes unstable and the contour of $\mathbf{w}^T\mathbf{R}_b\mathbf{w}$ becomes very flat and ill-conditioned. In order to make AdaBSF algorithm stable, the condition number $\kappa(\mathbf{R}_b + \beta\mathbf{I})$ should not be too large, that is to say, we should make the following relationship established

$$\kappa(\mathbf{R}_b + \beta\mathbf{I}) \leq \eta, , \tag{14}$$

where $\eta > 0$ represents a positive number. In this paper ,we make an empirical choice, say $\eta = 100$. In this way, the parameter β can be determined from (13) and (14) by

$$\beta = max\{0, \frac{1}{\eta - 1}(\lambda_{max} - \eta\lambda_{min})\}. \tag{15}$$

III. Traffic light recognition

After getting a series of candidate windows by AdaBSF, each window is analysed by more discriminative features to determine whether it really contains a traffic light instance or not. If does, the window image is then further classified into different traffic light classes. This task can be seen as a basic multi-class classification problem.

A. Color and Shape Descriptors

The color and shape information are important to traffic lights detection. To be specific, the light emitting unit usually has specific colors and shapes. It is usually surrounded by particular backgrounds, e.g. the black rectangle shade, the supported bar, the trees and buildings. Among various shape features, the HOG feature proposed by Dalal and Triggs [34] is commonly used for object detection. The HOG feature computes the gradient histograms on densely overlapped cells and blocks. However, HOG feature does not include color information of objects. It is

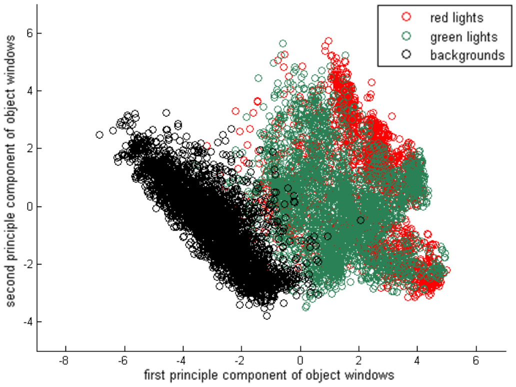


Fig. 6. The first two principle components of the traffic lights and background data. The distribution suggests linear separable of the data in their original feature space. In this paper, we choose linear SVM as a basic classifier.

suggested in [33] and [37] that the combination of local color feature and HOG feature will improve the performance of detectors compared with conventional HOG approaches. Thus we extend HOG with color descriptors.

Similarly, we first evenly divide the object window into different patches. For each patch, we compute the local color histograms on three channels [33], respectively. Then we use local RGB histogram combined with HOG as our basic feature descriptors, as is shown in Fig. 5. The final feature descriptor of a object window can be represented as follows

$$FEAT = [HOG, HIST_R, HIST_G, HIST_B], \tag{16}$$

where HOG represents HOG descriptor, $HIST_R$, $HIST_G$ and $HIST_B$ represent color histograms descriptors accumulated on R, G, B channels, respectively. The classification windows we used in this stage are directly cropped from the original sized frames according to the AdaBSF's detection results. We crop each object window to 60×30 pixels and 30×60 pixels to detect vertical posed traffic lights and horizontal posed traffic lights, respectively. For the HOG descriptor, the block size is set to 10 pixels, the block stride and cell size are set to 5 pixels. For the local color descriptor, the patch size is set to 5 pixels, the gray level of pixels at each channel is evenly quantified to 10 grades. Then for each detection window, a HOG descriptor with 2250 dimensions and a color descriptor with 2160 dimensions are calculated. The dimension of the final combined descriptor is 4410.

B. Classifier Design

On recognition stage, accuracy and processing speed are both important. The Suppot Vector Machine (SVM) is popular for its good generalization ability and its beautiful mathematical forms. SVM is a supervised learning method that constructs a hyperplane to separate feature points into two classes. The "support vectors" are those feature points that hold-up the maximum margin of the hyperplane. Linear SVM is fast and easy to train, which is helpful to our classification system for our large amount of training data and its high dimensionality. In this paper,

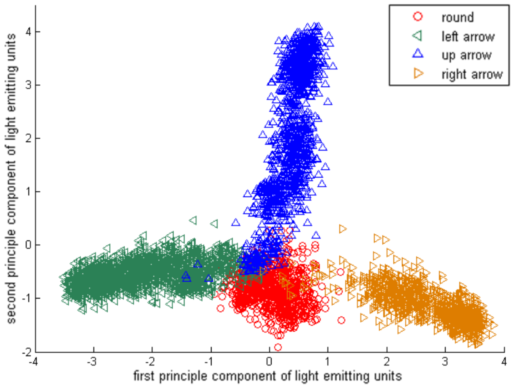


Fig. 7. The first two principle components of different kinds of traffic lights.

we design a cascade of SVM classifiers to classify each traffic light into different sematic classes, as illustrated in Fig. 8. Considering the computational efficiency, we choose linear SVM as a basic classifier of our system. First, for each traffic light candidate window, the combined feature vector and a linear SVM is used to verify whether it really contains a traffic light. If not, the candidate region is determined as a background region, else, the traffic light candidate windows are further classified into two basic types, "red type" and "green type", by checking the color of their light emitting regions. Then different classifiers are used for traffic lights with red or greed emitting units. Finally, the exact light type can be determined by "$1 - vs - 1$" voting method. By using the principle component analysis, we can make a simple visuallization of the traffic light data in 2D space. Fig. 6 illustrates the first two principle components of the candidate regions produced by AdaBSF, and Fig. 7 illustrates the first two principle components of different types of traffic lights. The combined descriptor provides an effective description of the different types of traffic lights. It can be seen that, we can nearly separate the arbitrary two types of data by a linear SVM classifier easily in 2D space, which suggests the data of highly linear separable in its original feature space.

IV. Experimental Results

In this section, we use several experiments to evaluate the performance of our traffic light detection system. We test on several video sequences which are captured by the moving vehicle on urban streets and suburb roads under different light conditions. To further evaluate the performance of our system, we also compare with the methods in [15] and [23] on a public traffic light detection dataset.

A. Dataset

We evaluate our system on a dataset of 8 video sequences (5654 frames with labels). The sequences are taken by a telephoto lens camera in urban streets and suburb roads. The camera was mounted in front of the interior rear-view mirror of a driving unmanned vehicle provided by the Beijing Union University and Military Traffic Institute of China. This experiment is supported by the Major Research Plan of National Natural Science Foundation of China

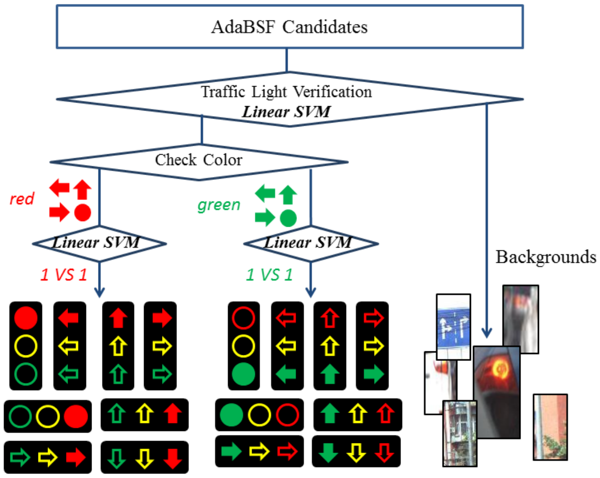


Fig. 8. Cascaded SVM classifier for traffic light recognition.

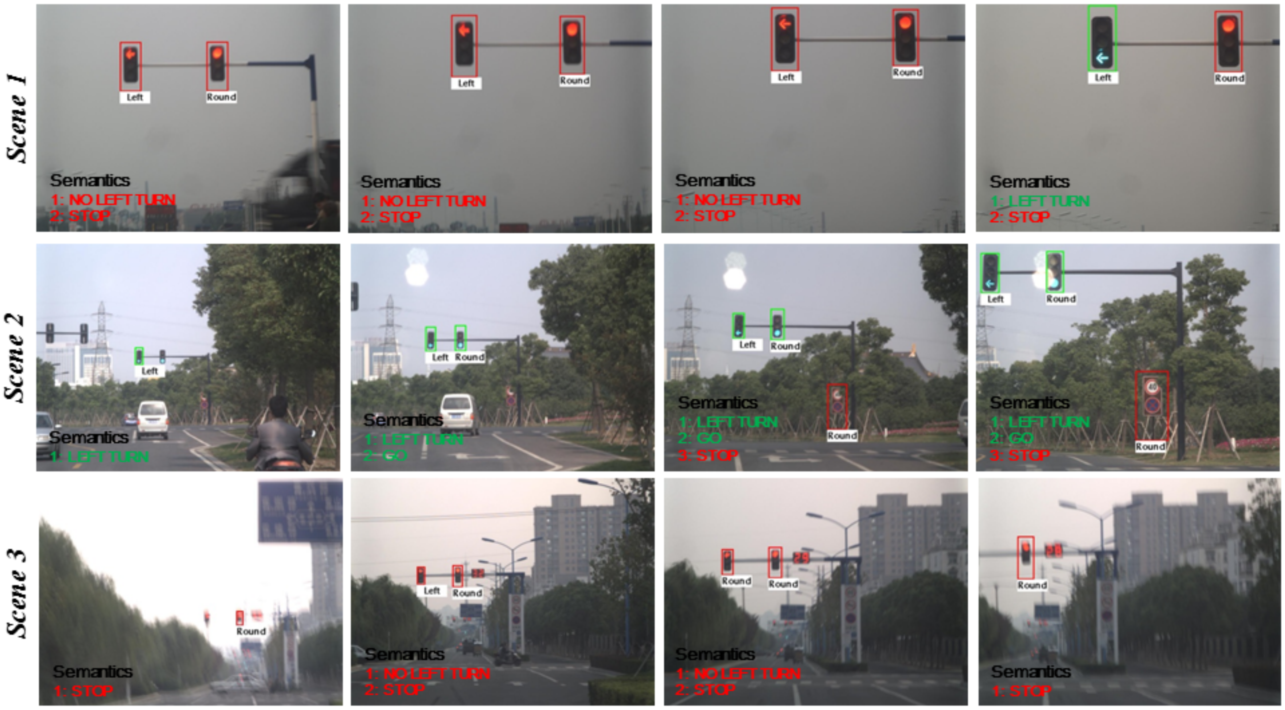


Fig. 9. Some traffic detection results of our system. In scene 2, a traffic sign is falsely detected as a red light, and a green light is missed since the target is too far. In scene 3, a red light is missed since the frames are blurred by the motion of the vehicle. Due to the limited space, each frame is cropped to 750×1000 pixels for illustration.

(NSFC), “Cognitive Computing of Visual and Auditory Information” [2]. All the images are captured in Changshu, Jiansu Province, China. The frame size is 1000×1000, and the frame rate is $5fps$. The 8 video sequences are taken during different time periods of a day and in different light conditions, i.e, clear day, cloudy day and driving against strong light. All traffic lights in these frames have been labeled manually. Notice that we do not collect data at night, since all distinguishing features, aside from light color, are lost at night with most video cameras [12], and purely vision-based traffic light detection becomes extremely difficult at night. In our dataset, not all the traffic lights are labeled. Some ambiguous traffic lights are excluded due to heavy motion and background clutter [3]. Apart from these, since it is nothing much significance of detecting the traffic lights with a far distance or not facing the camera, some traffic lights whose diagonal length are smaller than 20 pixels or partially outside the frame, are also excluded. It should be noticed that any detected yellow lights by our system will treated as red lights to ensure safety. The traffic light detected in these regions will not be taken into account neither as a “false positive” nor as a “true positive”. Eliminating all these traffic light instances, a total of 2713 green traffic lights and 5245 red traffic lights in 5654 frames are labeled for test.

We also select 20952 traffic lights from another 10000 frames (not from the above 8 sequences) as training samples. The negative training samples are a set of 15773 light-free windows randomly sampled from these frames. Table I lists the training-set details. Together with their left-right refections, 41904 traffic lights and 31546 background samples are finally used for training the cascaded SVM classifier. All the samples are resized to 60×30 pixels (for vertical posed lights) and 30×60 pixels (for horizontal posed lights) for classification. The classifiers are then re-trained for several times using this augmented dataset (initial data + hard examples) to produce the final detector.

TABLE I

DETAILS OF THE TRAINING DATA FOR CASCADED SVM CLASSIFIER

Type	circle	arrow up	arrow left	arrow right	Total
red	3494	3072	2535	1874	10975
green	3218	3457	2000	1302	9977
negative	-	-	-	-	15773

B. Detection on Video Sequences

At the beginning of candidate extraction stage, each frame is down-sampled at 14 scales: $1, 1.8, 2.1, \ldots, 15$. Since the initial frame size is 1000×1000 and the AdaBSF filter size is 16×8, we can detect all the traffic lights with the size of 8×16 to 120×240 pixels. 500 traffic lights samples are added to constraints in (4). Since the maximum and

[2] website URL: http://ccvai.xjtu.edu.cn/en/mes.do?method=list

[3] “Heavy motion and background clutter” refers to those lights whose light type cannot be easily identified human eyes or those clustered parts are larger than half of their sizes.

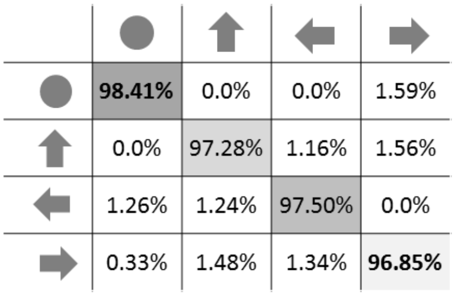
minimum eigenvalues of matric $\mathbf{R}_b$ are $\lambda_{max} = 2.98$ and $\lambda_{min} = 7.61 \times 10^{-10}$, we determine AdaBSF parameter $\alpha = 0.001$, $\beta = 0.03$. 8 video sequences are tested in our system. Some typical results are displayed in Fig. 9. The detection results of green traffic lights and red traffic lights are marked by the green rectangles and red rectangles, respectively. In the lower left corner of each frame lists the detailed information of each light. In Fig. 9, each row shows the detection results of 4 frames in the same scene sequence. Frames in scene 1 and scene 2 are captured in suburb environments. Frames in scene 3 are captured in urban environments. We saved the detection results on every frame, and after the whole processing, the *precision* and *recall* rate are counted. Detailed information of the dataset and the detection results are listed in table II. Results of these frames suggest a overall high accuracy and stability of our system. The *precision* and *recall* rate are computed as follows: $precision = \frac{N_{tp}}{N_{tp}+N_{fp}}$, $recall = \frac{N_{tp}}{N_{tp}+N_{fn}}$, where N_{tp} represents number of true-positives, N_{fp} represents number of false-positives and N_{fn} represents number of false-negatives. We do not combine our method with other tracking methods since our detection system has already achieved a satisfactory performance. The confusion matrix of different types of traffic lights is shown in Fig. 10. The light type with the highest classification rate is “round type” (98.42%) while which with the lowest rate is “arrow right type” (96.85%). This is mainly because the “arrow right” samples we used for training is slightly insufficient (see table I), which can be improved by adding more training samples to this type of set. It should be noticed that, the light types and their semantics are two different concepts. Although some traffic lights have totally different appearance, they could have the same semantics. For example, green up and green round traffic lights both mean going straight. Red up and red round traffic lights both mean stop.

TABLE II

DETAILS OF THE VIDEO SEQUENCES USED IN OUR EXPERIMENTS AND DETECTION RESULTS

Sequence	Length	nFrames	mTargets	Location	Capture Time	Precision	Recall
1	1m50s	550	green:210 / red:275	suburb	midday	0.911	0.971
2	1m2s	311	green:230 / red:149	suburb	midday	0.979	0.968
3	3m27s	1033	green:539 / red:1184	urban	midday	0.883	0.943
4	2m25s	725	green:335 / red:695	urban	afternoon	0.909	0.984
5	3m15s	975	green:405 / red:924	urban	afternoon	0.961	0.919
6	4m21s	1305	green:633 / red:1129	urban	afternoon	0.929	0.962
7	37s	185	green:75 / red:159	urban	afternoon	0.840	0.944
8	1m54s	570	green:286 / red:730	urban	nightfall	0.966	0.884
Total	18m51s	5654	green:2713 / red:5245	-	-	**0.922**	**0.9469**

One important parameter in our method is the threshold T. To choose an appropriate value of T, the recall rate and the corresponding candidate window number per frame are counted with different choices of T. Fig. 12 plots the recall curves of the red lights and the green lights on different T, respectively. We can see that, if we set smaller value of T, then we will get higher recall rate, but at the same time, we will get more candidate windows. When T is set smaller than 0.1, the recall rate decreases quickly, contrarily, when T is set larger than 0.1, the number of candidate windows will increase greatly, which will slow down the following processing pipeline. An appropriate



	●	⬆	⬅	➡
●	**98.41%**	0.0%	0.0%	1.59%
⬆	0.0%	97.28%	1.16%	1.56%
⬅	1.26%	1.24%	97.50%	0.0%
➡	0.33%	1.48%	1.34%	**96.85%**

Fig. 10. Confusion matrix of different types of traffic lights.

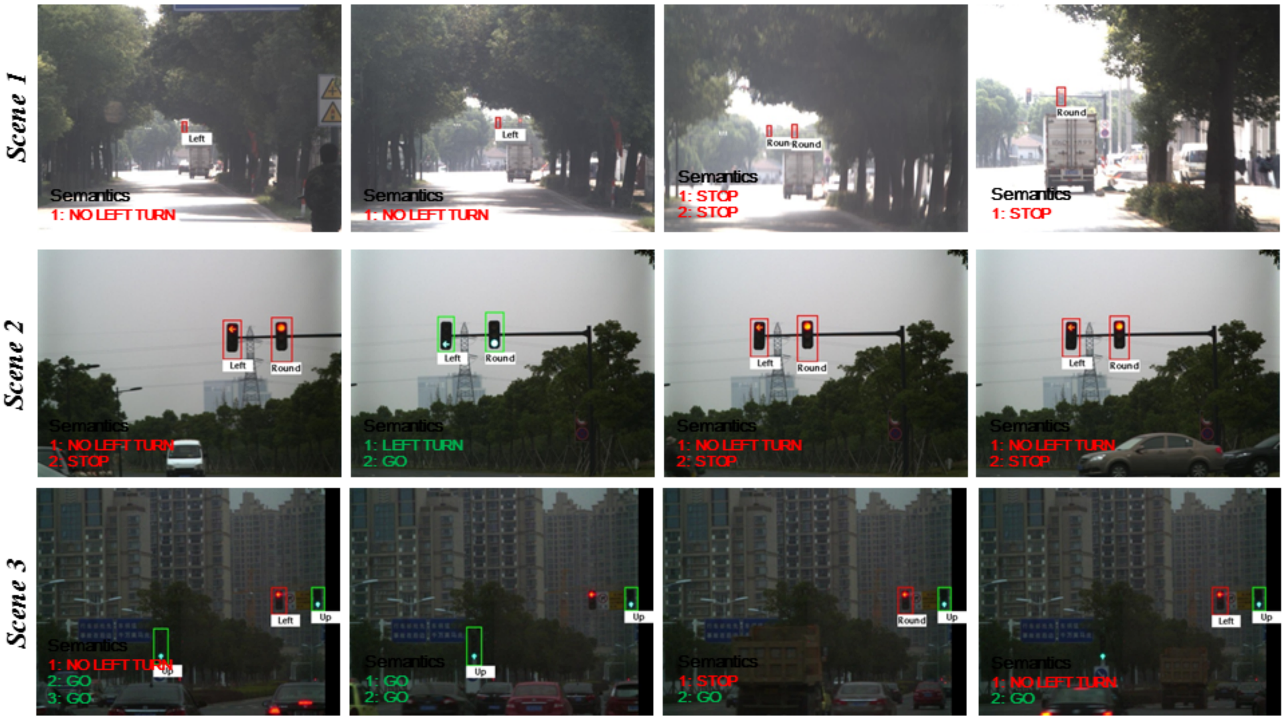


Fig. 11. Detection results in various light conditions. In scene 1, two red traffic lights are missed due to the effect of the strong sunlight. In scene 3, a red traffic light and a green light are missed since their shape information of the lampshades are lost. Due to the limited space, each frame is cropped to 750×1000 pixels for illustration.

choice of T should be a trade off between the recall rate and the time efficiency. Thus in our paper, we set $T = 0.1$.

C. Detection under Strong Sunlight and Low Contrast Conditions

Illumination changes are challenges for both candidate extraction stage and recognition stage. To improve the robustness of AdaBSF, more traffic light samples in different light conditions are added to constraints in (4). We select 728 frames from the dataset, where most of these images are taken against strong head-on sunlight or taken in

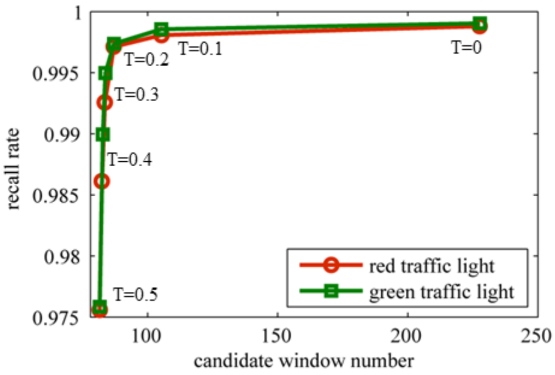


Fig. 12. Relationship between the recall rate, candidate window number and the choice of threshold T.

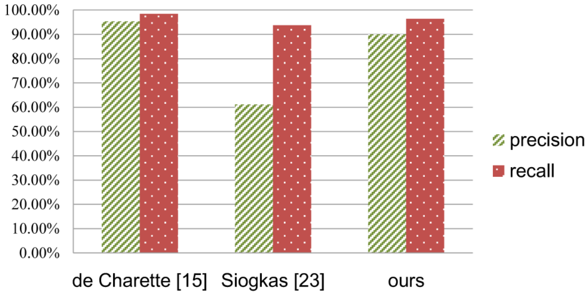


Fig. 13. *Precision* and *recall* rates on TLR dataset [15] of three traffic light detection methods: de Carette [15], Siogkas [23] and ours. Notice that in our method, although all the training samples are from our own data, and we do not have any prior knowledge of dataset [15], we still obtain the comparable results.

extreme low contrast conditions. Some typical results are displayed in Fig. 11, where each row shows the detection results of 4 frames in the same scene sequence. Frames in scene 1 are captured against the strong sun light in midday. We can see the lampshade is nearly engulfed by the bright sunlight. Frames in scene 2 are captured in nightfall. The color of the light emitting units vary a lot: the red lights becomes “orange” and the green lights become “white”. Frame in scene 3 are captured in extreme low contrast conditions. The necessary shape features for traffic lights detection are almost lost. The traffic light detection task becomes very difficult in these conditions. We can see two red lights and one green light are missed in scene 1 and scene 3 due to the illumination variations. Although some false negative exists, the performance of AdaBSF is still acceptable in these extreme conditions. Results in these frames suggest the high robustness of our AdaBSF and our detection system.

D. Test on Dataset [15]

To further illustrate the robustness of our detection system, we also test on the dataset provided by [15] (Traffic Lights Recognition (TLR) public benchmarks). We compare with the method in [15] and [23] on this dataset. TLR

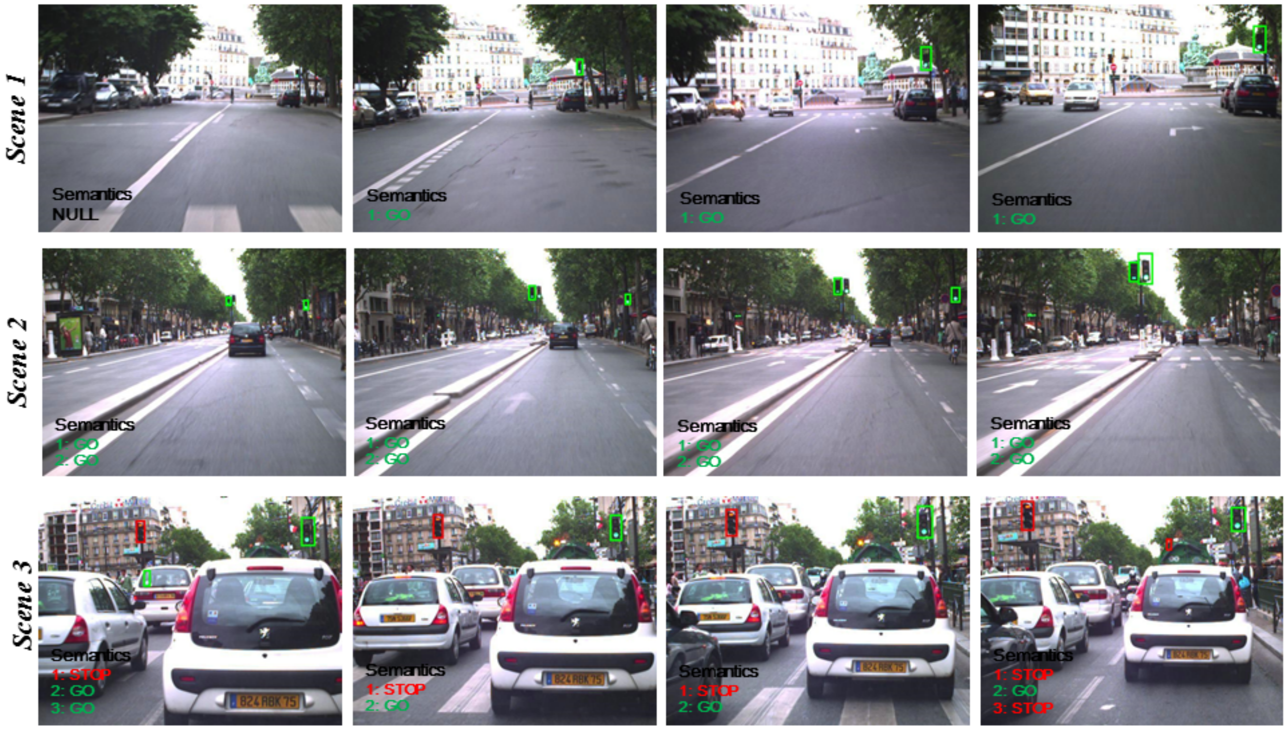


Fig. 14. Traffic light detection results on TLR dataset [15]. In scene 2, a green light is missed. In scene 3, a false detection is made since a spot reflected by the mirror resembles a green traffic light both in color and shape. These false results cannot be easily excluded, unless it is used in correlation to other computer vision modules like a vehicle detector or a road detector [23].

dataset is captured by a moving vehicle in a dense urban environment in Paris. The dataset includes 11179 frames, 9168 instances of traffic lights (3381 green lights, 58 yellow lights, 5280 red lights and 449 "ambiguous" [4]), and the frame size is 480×640. All the ambiguous lights are excluded for comparison. It should be noticed that this dataset only contains "round shape" traffic lights, and it does not contain any "arrow type" traffic lights. Thus, if the current object window is verified as a traffic light window, we do not need to further classify the it into different shape classes. The only thing we need to do is checking its color ("green light" or "red light"). The parameter α, β and T is set to 0.001, 0.03 and 0.1, respectively, which are the same as those used in last experiment. Fig. 14 shows some typical detection results of TLR dataset. Our system successfully recognises most of the traffic lights in these frames. Fig. 13 plots the *precision* and *recall* rates of the three traffic light detection methods, and results of [15] and [23] are reported by their authors. It should be noticed that in our method, all the training samples in this experiment come from our own dataset. Although we do not have any prior information of this dataset, our system still has comparable detection performance with methods of [15] and [23]. This experiment shows our detection method has a good transplantation ability.

[4] More details of this dataset can be found on http://www.lara.prd.fr/benchmarks/trafficlightsrecognition?redirect=1#dataset.

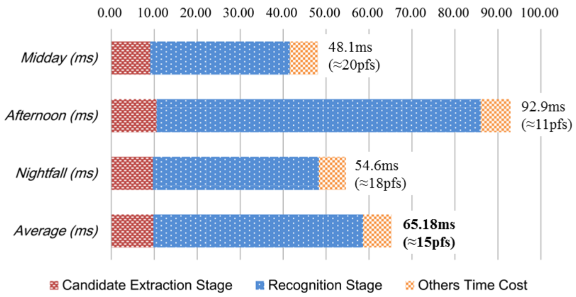


Fig. 15. Computation time of the two stages in different time periods of a day: midday, afternoon and nightfall. Other time cost refers to disk read / write time and result labeling time and etc.

E. Time Efficiency

In this subsection, the computational time of each stage of our method is counted. We test our algorithm on an Intel i7 PC with 16G RAM, and the programming language is C++. Some methods are used to speed up the whole detection pipeline. On candidate extraction stage, the inner product of AdaBSF at each window can be easily replaced by a 2D-convolution process, which can be accelerated by several parallel methods. This makes the AdaBSF takes about only 10ms to search out all the traffic lights per frame. On recognition stage, classifications for different traffic light windows can be paralleled in several individual threads. We use open-MP toolkits to implement the parallel operation, which accelerates the recognition stage 2-3 times. Fig. 15 shows the processing time of each stage in different time periods of a day. The frames we used are those introduced in subsection (A). Although the time for those frames taken in the afternoon is a little bit longer than other two time periods (this is probably because AdaBSF produces more candidates in the afternoon), it still reaches a satisfactory and stable time performance. The average processing speed of our detection system is about 15 *fps*, and this speed could meet the real time requirements of the driving vehicles for the most conditions.

V. Conclusion

We propose a purely vision-based traffic light detection method, which is robust to different illuminations. Most existing approaches concerning such problem use color thresholding solely or use simple template matching methods, thus tend to be affected by light changes. We propose a new traffic light candidate extraction algorithm, AdaBSF, which is suitable and effective in this task. Experimental results suggest the proposed method is fast and robust. By applying proposed method on collected video data and public dataset, improvement over conventional approaches is achieved. The whole detection procedure can be performed in real time.

Acknowledgment

The authors would like to thank R. de Charette and F. Nashashibi for providing the traffic light recognition dataset. The authors also would like to thank the Associate Editor and the four anonymous reviewers for their very useful comments and suggestions which greatly improve the quality of this paper.

References

[1] J. Levinson, J. Askeland, J. Dolson, and S. Thrun, “Traffic light mapping, localization, and state detection for autonomous vehicles,” in *Proc. IEEE Int. Conf. Rob. Autom.*, 2011, pp. 5784-5791.

[2] N. H. C. Yung and A. H. Lai, “An effective video analysis method for detecting red light runners,” *IEEE Trans. Veh. Technol.*, vol. 50, no. 4, pp. 1074-1084, Jul. 2001.

[3] Z. Tu and R. Li, “Automatic recognition of civil infrastructure objects in mobile mapping imagery using a markov random field model,” *International Archives of ISPRS*, vol. 33, no. 2, 2000.

[4] D. M. Gavrila, U. Franke, C. Wohler and S. Gorzig, “Real time vision for intelligent vehicles,” *IEEE Instrum. Meas. Mag.*, vol. 4, no. 2, pp. 22-27, Jun. 2001.

[5] G. K. Pang and H. H. Liu, “LED location beacon system based on processing of digital images,” *IEEE Trans. Intell. Transp. Syst.*, vol. 2, no. 3, pp. 135-150, Sep. 2001.

[6] T. Shioyama, H. Wu, N. Nakamura and S. Kitawaki, “Measurement of the length of pedestrian crossings and detection of traffic lights from image data,” *Meas. Sci. Technol.*, vol. 13, no. 9, pp. 1450, Sep. 2002.

[7] Y. C. Chung, J. M. Wang and S. W. Chen, “A vision-based traffic light detection system at intersections,” *Journal of Taiwan Normal University*, vol. 47, no. 1, pp. 67-86, 2002.

[8] F. Lindner, U. Kressel and S. Kaelberer, “Robust recognition of traffic signals,” in *Proc. IEEE Intell. Veh. Symp.*, 2004, pp. 49-53.

[9] V. John, K. Yoneda, B. Qi, Z. Liu and S. Mita, “Traffic light recognition in varying illumination using deep learning and saliency map,” in *Proc. IEEE Conf. Intell. Transport. Syst.*, 2014, pp. 2286-2291.

[10] N. Fairfield and C. Urmson, “Traffic light mapping and detection,” in *Proc. IEEE Int. Conf. Rob. Autom.*, 2011, pp. 5421-5426.

[11] H. Tae-Hyun, J. In-Hak and C. Seong-Ik, “Detection of traffic lights for vision-based car navigation system”, *Lect. Notes Comput. Sci.*, 2006, pp. 682-691.

[12] Y. K. Kim, K. W. Kim and X. Yang, “Real time traffic light recognition system for color vision deficiencies,” in *Proc. IEEE Int. Conf. Mechatronics. Autom.*, 2007, pp. 76-81.

[13] K. H. Lu, C. M. Wang and S. Y. Chen, “Traffic light recognition,” *J. Chin. Inst. Eng.*, vol. 31, no. 6, pp. 1069-1075, Sep. 2008.

[14] Y. Shen, U. Ozguner, K. Redmill and J. Liu, “A robust video based traffic light detection algorithm for intelligent vehicles,” in *Proc. IEEE Intell. Veh. Symp.*, 2009, pp. 521-526.

[15] R. de Charette and F. Nashashibi, “Real time visual traffic lights recognition based on spot light detection and adaptive traffic lights templates,” in *Proc. IEEE Intell. Veh. Symp.*, 2009, pp. 358-363.

[16] R. de Charette and F. Nashashibi, “Traffic light recognition using image processing compared to learning processes,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Rob. Syst.*, 2009, pp. 333-338.

[17] M. Omachi and S. Omachi, “Traffic light detection with color and edge information,” in *Proc. IEEE Int. Conf. Comput Sci. Inf. Technol.*, 2009, pp. 284-287.

[18] M. Omachi and S. Omachi, “Detection of traffic light using structural information,” in *Proc. IEEE Int. Conf. Signal. Process. Proc.*, 2010, pp. 809-812.

[19] J. Gong, Y. Jiang, G. Xiong, C. Guan, G. Tao and H. Chen, “The recognition and tracking of traffic lights based on color segmentation and camshift for intelligent vehicles,” in *Proc. IEEE Intell. Veh. Symp.*, 2010, pp. 431-435.

[20] C. Wang, T. Jin, M. Yang and B. Wang, “Robust and real-time traffic lights recognition in complex urban environments,” *Int. J. Comput. Intell. Syst.*, vol. 4 no. 6, pp. 1383-1390, Dec. 2011.

[21] Z. Cai, Y. Li, and M. Gu, “Real-time recognition system of traffic light in urban environment,” in *Proc. IEEE Symp.*

Comput. Intell. Secur. Defence Appl., 2012, pp. 1-6.

[22] M. Diaz-Cabrera, P. Cerri and J. Sanchez-Medina, “Suspended traffic lights detection and distance estimation using color features,” in *Proc. IEEE Conf. Intell. Transport. Syst.*, 2012, pp. 1315-1320.

[23] G. Siogkas, E. Skodras and E. Dermatas, “Traffic Lights Detection in Adverse Conditions using Color, Symmetry and Spatiotemporal Information,” in *Proc. Int. Conf. Comput. Vis. Theory Appl.*, 2012, pp. 620-627.

[24] Q. Chen, Z. Shi and Z. Zou, “Robust and real-time traffic light recognition based on hierarchical vision architecture,” in *Proc. Int. Congr. Image Signal Process.*, 2014, pp. 114-119.

[25] C. Jang, C. Kim, D. Kim, M. Lee and M. Sunwoo, “Multiple exposure images based traffic light recognition,” in *Proc. IEEE Intell. Veh. Symp.*, 2014, pp. 1313-1318.

[26] S. Sathiya, M. Balasubramanian and D. V. Priya, “Real time recognition of traffic light and their signal count-down timings,” in *Proc. IEEE Int. Conf. Inf. Commun. Embed. Syst.*, 2014, pp. 1-6.

[27] A. E. Gomez, F. A. Alencar, P. V. Prado, F. S. Osrio and D. F. Wolf, “Traffic lights detection and state estimation using Hidden Markov Models,” in *Proc. IEEE Intell. Veh. Symp.*, 2014, pp. 750-755.

[28] M. M. Cheng, Z. Zhang, W. Y. Lin and P. Torr, “BING: Binarized normed gradients for objectness estimation at 300fps,” in *Proc. IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2014, pp. 3286-3293.

[29] B. Alexe, T. Deselaers and V. Ferrari, “What is an object?” in *Proc. IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2010, pp. 73-80.

[30] B. Alexe, T. Deselaers and V. Ferrari, “Measuring the objectness of image windows,” *IEEE Trans. Pattern. Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2189-2202, Nov. 2012.

[31] I. Endres and D. Hoiem, “Category independent object proposals,” in *Proc. Lect. Notes Comput. Sci.*, 2010, pp. 575-588.

[32] J. R. Uijlings, K. E. van de Sande, T. Gevers and A. W. Smeulders, “Selective search for object recognition,” *Int. J. Comput. Vision.*, vol. 104, no. 2, pp. 154-171, Sep. 2013.

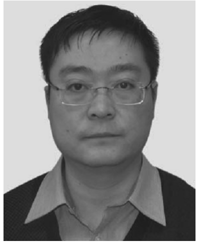
[33] K. E. Van De Sande, T. Gevers and C. G. Snoek, “Evaluating color descriptors for object and scene recognition,” *IEEE Trans. Pattern. Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1582-1596, Sep. 2010.

[34] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proc. IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2005, vol. 1, pp. 886-893.

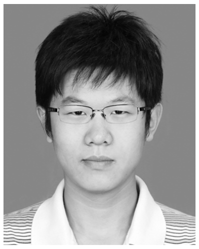
[35] V. N. Vapnik and V. Vapnik, “Statistical learning theory (Vol. 1),” 1998, New York: Wiley.

[36] S. Boyd and L. Vandenberghe, “Convex optimization,” 2004, Cambridge university press.

[37] F. Shahbaz Khan, R. M. Anwer, J. van de Weijer, A. D. Bagdanov, M. Vanrell and A. M. Lopez, “Color attributes for object detection,” in *Proc. IEEE Conf. Comput. Vision. Pattern. Recognit.*, 2012, pp. 3306-3313.



Zhenwei Shi Zhenwei Shi (M13) received his Ph.D. degree in mathematics from Dalian University of Technology, Dalian, China, in 2005. He was a Postdoctoral Researcher in the Department of Automation, Tsinghua University, Beijing, China, from 2005 to 2007. He was Visiting Scholar in the Department of Electrical Engineering and Computer Science, Northwestern University, U.S.A., from 2013 to 2014. He is currently a professor in the Image Processing Center, School of Astronautics, Beihang University. His current research interests include remote sensing image processing and analysis, computer vision, pattern recognition, and machine learning.



Zhengxia Zou Zhengxia Zou received the B.S. degree from the School of Astronautics, Beihang University, Beijing, China, in 2013. He is currently working toward the Ph.D. degree in the Image Processing Center, School of Astronautics, Beihang University. His research interests include target detection and pattern recognition.

Changshui Zhang Changshui Zhang (M02) received the B.S. degree in mathematics from Peking University, Beijing, China, in 1986 and the M.S. and Ph.D. degrees in control science and engineering from Tsinghua University, Beijing, in 1989 and 1992, respectively. In 1992 he joined the Department of Automation, Tsinghua University, where he is currently a Professor. He has authored more than 200 papers. His research interests include pattern recognition and machine learning.

Dr. Zhang is an Associate Editor of Pattern Recognition Journal. He is a member of the Standing Council of the Chinese Association of Artificial Intelligence.