Computing and Informatics, Vol. 39, 2020, 439–463, doi: 10.31577/cai_2020_3_439

TRAFFIC LIGHT RECOGNITION FOR REAL SCENES BASED ON IMAGE PROCESSING AND DEEP LEARNING

Mingliang Che

School of Geographic Science
Nantong University
226019 Nantong, China
e-mail: dawnche@163.com

Mingjun Che

Wuxianshenghuo (Hangzhou) Info Tech Ltd.
311100 Hangzhou, China
e-mail: chemingjun@126.com

Zhenhua Chao, Xinliang Cao

School of Geographic Science
Nantong University
226019 Nantong, China
e-mail: chaozhenhua@ntu.edu.cn, cxliang@mail.ustc.edu.cn

Abstract. Traffic light recognition in urban environments is crucial for vehicle control. Many studies have been devoted to recognizing traffic lights. However, existing recognition methods still face many challenges in terms of accuracy, runtime and size. This paper presents a novel robust traffic light recognition approach that takes into account these three aspects based on image processing and deep learning. The proposed approach adopts a two-stage architecture, first performing detection and then classification. In the detection, the perspective relationship and the fractal dimension are both considered to dramatically reduce the number of invalid candidate boxes, i.e. region proposals. In the classification, the candidate boxes are classified by SqueezeNet. Finally, the recognized traffic light boxes are reshaped by postprocessing. Compared with several reference models, this approach is significantly competitive in terms of accuracy and runtime. We show that our approach is lightweight, easy to implement, and applicable to smart terminals, mobile devices or embedded devices in practice.

Keywords: Traffic light recognition, color features, perspective relationship, fractal dimension, SqueezeNet

Mathematics Subject Classification 2010: 68U10

1 INTRODUCTION

Traffic light recognition in urban environments is crucial in many cases, such as in detecting traffic signs during semiautomatic or fully autonomous driving [1], helping pedestrians with some form of visual impairment [2], estimating the distance between vehicles and detected traffic lights [3] and assisting in the correction of the map coordinates of a floating car [4]. Thus, many studies address the problem of detecting traffic lights, and it remains an active challenge.

Traffic light recognition belongs to the field of object recognition, i.e. object detection in computer vision. The crucial task of object recognition requires solving two problems: locating objects and then classifying them. Current approaches to traffic light recognition can be divided into two categories: two-stage strategies and one-stage strategies. Regarding the former, the region proposals are generated first. Then, some classifiers are used to classify the region proposals. Many methods can be used to create region proposals. The sliding window method is the easiest, but it is time-consuming. Eventually, selective search was proposed [5, 6]. This method combines the strengths of both an exhaustive search and segmentation. In terms of efficiency, selective search is better than the sliding window method. However, selective search is not sensitive to small objects, such as traffic lights, and its runtime is longer. In contrast to the time-consuming selective search, a speed-up algorithm, i.e. edge detection, was proposed [7]. Edge detection achieves higher object recall and is faster to compute. However, the algorithm does not perform well when segmenting individuals. Another study used binary semantic segmentation to detect region proposals. However, both the precision and computing speed of the method on a CPU need to be improved [8]. Some studies generate region proposals from other perspectives, such as spot light detection [9], color and shape features [10, 11, 12, 13, 14], map information [15] and so on. These methods usually require making strong assumptions and may generate many redundant candidate regions. However, they are based primarily on image processing techniques; thus, their runtimes are relatively short.

In the two-stage strategy, the typical classifiers include template matching [9, 16], the support vector machine (SVM) [17], the hidden Markov model (HMM) [18], and deep learning [17, 19, 20]. The template matching technology is the earliest object recognition application and is easy to operate. However, its rate of correctly recognized traffic lights is closely related to the templates. A stiff template always limits the adaptability of the method to individual objects, especially for dynamic traffic lights in fisheye camera images. The SVM is a better classifier due to its self-learning. However, the SVM performs better on small sample training sets. When the training data becomes large, the SVM becomes time-consuming and has a very limited accuracy when addressing a multiclassification problem. The HMM can help in determining the current state of the traffic light detected based on the obtained state processing. The accuracy achieved by this method is not too high. Owing to the recent advances and performances of deep neural networks, deep learning has been used for image classification. The very representative and popular model is the R-CNN [21] and its accelerated versions [22, 23, 24]. The R-CNN uses the features in the image extracted by deep learning to train the classifier. This process makes the accuracy of object recognition optimal in effect. However, it is time-consuming and requires much storage space. Moreover, it still has problems in identifying small objects, such as traffic lights [25].

In contrast to the two-stage strategy, the one-stage architecture is quite different, as it does not use any prior knowledge of the object locations. It uses algorithms to directly output categories and the corresponding positioning. The more popular models include YOLO [26] and the SSD [27]. YOLO, that is, "You Only Look Once", is a unified and real-time object detector that uses only a single deep neural network to detect objects in images. The smaller version of YOLO, i.e. fast YOLO, performs well when conducting real-time object detection in a video [28]. However, YOLO makes more localization errors and lags behind state-of-the-art detection systems in terms of accuracy [26]. Thus, an improved YOLO approach, i.e. YOLO v2 (also called YOLO 9000), was proposed. Compared with YOLO, YOLO v2 uses a novel multiscale training method, runs at varying sizes, and offers an easy tradeoff between speed and accuracy [29]. Then, an advanced YOLO approach, i.e. YOLO v3, was presented. YOLO v3 is obviously faster than the reference approaches when realizing the same accuracy [30]. From YOLO to YOLO v3, the speed increases, while the accuracy does not, especially regarding the location precision. The SSD model appeared after the YOLO model. The SSD completely eliminates proposal generation and subsequent pixel or feature resampling stages and encapsulates all computations in a single network, which makes the SSD easy to train and optimize. The experimental results show that the SSD has competitive accuracy relative to that of the comparison models and is much faster. However, the SSD is not very good at or even unable to perform small object recognition. In addition, similar to the YOLO model, the size of the SSD is also very large.

Overall, the two-stage approaches have higher precision and a longer runtime, while the one-stage methods have lower precision but a shorter runtime. Deep learning has high precision, but its model is usually very large, which is extremely disadvantageous for devices with lower memory and weaker processors outside the laboratory. Moreover, deep learning does not seem to be compatible with the accuracy and runtime.

In this paper, our approach takes a two-stage architecture to ensure high recognition accuracy. We used image processing to generate the region proposals and then applied the deep convolution neural network (CNN) to classify them. To reduce the runtime, we further optimized the detector by considering the perspective relationship and the fractal dimension. To decrease the model size, we introduced the lightweight SqueezeNet model, which strongly compresses the dimensions of the feature map [31]. Finally, based on the ground-truth image dataset, the performance of our approach was tested.

2 MATERIALS AND METHODS

2.1 Dataset Description

To evaluate our traffic signal recognition approach, two datasets for real scenes were employed. The first dataset includes data that we collected, which were derived from the position-image synchronous data captured by the driving recorder on a floating car. The time interval of the image set is 5 s. The dataset was recorded in the urban areas of Nantong and contained red, green and blue (RGB) color images. The image quality of the dataset is relatively high and has a resolution of 800×600 pixels. The dataset contains more than 700 images and more than 1 000 traffic lights. The scenes in the dataset can be divided into simple scenes and complex scenes. In the former, the traffic signals are exposed to the sky and are easily identifiable. In the latter, the backgrounds of traffic signals are diverse and may include trees, buildings, piers, and billboards. Moreover, the taillights of cars and outdoor lights resemble traffic signals from a distance, which causes further interference. In this situation, traffic signals are not recognizable.

The latter dataset is the publicly available benchmark dataset of the La Route Automatise (LaRA) benchmark provided by de Charette et al. [9]. The camera applied in the LaRA dataset has a focal length of 12 mm and a resolution of 640×480 pixels. The image color mode of the dataset is RGB full color. This dataset has been recorded in French urban areas and has more than 11 000 frames, many of which are stationary. To strengthen the difference between two adjacent frames, we resampled the source dataset every 5 frames. In the resampled data, some very blurry frames will be removed manually. The final resampled data contain more than 1 500 frames and more than 2 000 traffic lights. Use of the dataset allows a comparison between our approach and the reference methods.

2.2 Method Schema

2.2.1 Detector

A traffic light has a few special and important features, such as its border shape, lamp holder color and light color, which help people to distinguish it in real scenes. However, in image processing, the traffic light border shape is difficult to extract because it is easily affected by the illumination intensity, the light size and disturbances from background pixels. At night or when the traffic light is either very small or embedded in foliage, the traffic light border shape is very difficult to estimate. The illumination intensity and the light size also significantly affect the lamp holder color in the image. By contrast, the traffic light color, which is noteworthy and unique whether at day or at night, is relatively stable and is applied to locate the candidate regions of traffic lights.

The images in RGB mode are first converted to the hue, saturation, and value (HSV) color space. Compared with the RGB color space, the HSV color space is more suitable for segmentation and is more robust against illumination variation [12]. In this section, the regions of red, green and yellow are extracted to create the candidate boxes, i.e. the region proposals, that contain the traffic lights. The desired color is extracted from an image based on the HSV values resulting from the statistics of the positive samplings (Figure 1).

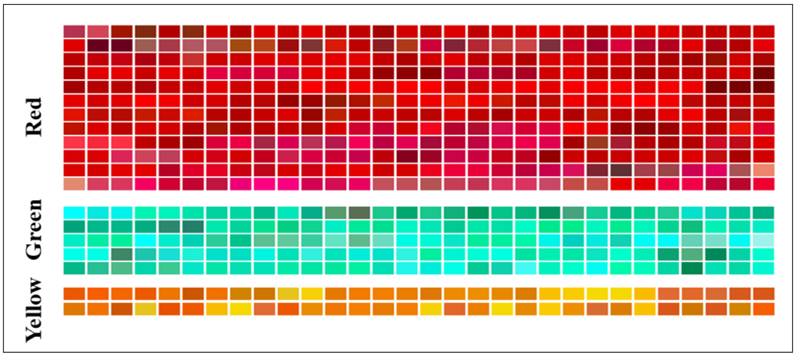


Figure 1. Color statistics of traffic lights in RGB mode

Suppose the triple (h, s, v) represents the HSV value of every pixel in an image; the expected color needs to satisfy the empirical relationships. In Figure 1, the range of each component in the color triple is calculated. Then, there is a slight change in the range. This process is performed to cover the traffic light color as much as possible and to find as many candidate regions as possible (Figure 2 b)).

To detect the red lights, the color thresholds were set as follows.

$$0 \leq h_{red1} < 15, \quad 115 \leq s_{red1} \leq 255, \quad 115 \leq v_{red1} \leq 255, \tag{1}$$

$$165 \leq h_{red2} \leq 180, \quad 120 \leq s_{red2} \leq 255, \quad 90 \leq v_{red2} \leq 255. \tag{2}$$

To extract the green and yellow lights, the color thresholds were set as shown below.

$$55 \leq h_{green} < 90, \quad 60 \leq s_{green} \leq 255, \quad 90 \leq v_{green} \leq 255, \tag{3}$$

$$15 \leq h_{yellow} \leq 25, \quad 195 \leq s_{yellow} \leq 255, \quad 205 \leq v_{yellow} \leq 255. \tag{4}$$

During the color extraction, the candidate regions, i.e. the approximate positions of the traffic lights, were detected (Figure 2 b)), and the corresponding binary images were created. To further calculate the profiles of the candidate regions, the *CHAIN_APPROX_SIMPLE* contour approximation algorithm in OpenCV was applied. In the algorithm, by compressing the horizontal, vertical, and diagonal segments and leaving only the end points, as little key pixels as possible are used to present the outlines. According to the contours of the candidate regions, the homologous bounding rectangle can be calculated. Then, the candidate box was estimated as follows.

$$bw = \text{int}(w' \cdot c \cdot k), \quad bh = \text{int}(h' \cdot c \cdot k), \tag{5}$$

$$w' = \min(w, h), \quad h' = \min(w, h) \tag{6}$$

where bw and bh are the width and height of the candidate box, respectively. w and h are the width and height of the bounding rectangle, respectively. The coefficient c is the ratio of the traffic light width to the lamp holder border width. In this paper, c equals 2.0. The zoom factor k ranges from 1.0 to 1.5. It can expand the background pixel information around the traffic light and can contribute to determining whether or not the candidate region contains the traffic light.

In the candidate boxes, some contain traffic lights, while some contain noise (Figure 2 c)). Apparently, some candidate boxes do not contain traffic lights because their size is inappropriate. To remove the obvious noise, the size estimation formula for the traffic lights was used. Depending on the perspective, everything looks small at a distance and large up close. Thus, a mathematical relationship exists between the light width and the corresponding y-coordinate value (Figure 3).

$$bw' = -0.101 \cdot y + 38.43 + t \tag{7}$$

where bw' denotes the estimated box width, y denotes the y-coordinate value, and t denotes the tolerance. When bw is less than bw', the candidate box is most likely noise. With this process, noisy boxes that are obviously smaller or larger than the normal size will be removed (Figure 2 d)). At the same time, some indistinguishable disturbance terms, e.g., some countdowns of traffic lights, are also removed (Figure 2 d)). Obviously, this process is carried out to reduce the number of false positives and shorten the runtime for recognizing traffic lights.

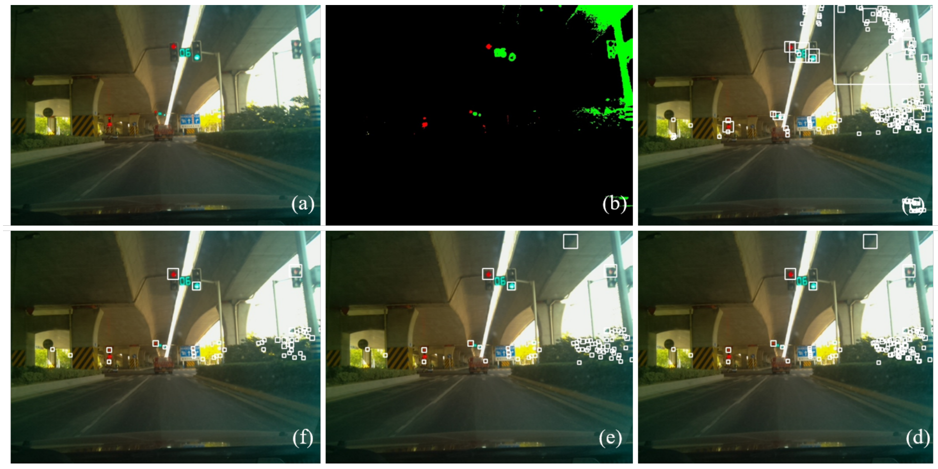


Figure 2. Sample detections of traffic lights in an image. a) is the original image, b) represents the candidate regions, c) represents the rough candidate boxes, d) represents the candidate boxes processed via size-noise reduction based on the perspective relationship, e) represents the candidate boxes processed via overlap removal with IOU, and f) represents the candidate boxes further processed via texture-noise reduction with the fractal dimension.

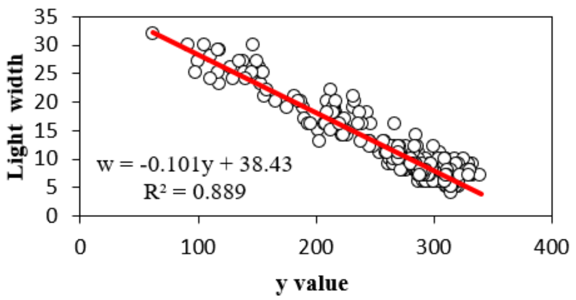


Figure 3. Relationship between the light width and the corresponding y-coordinate value

During the image processing above, multiple candidate boxes may be covering each other around the same traffic light. To remove the overlapping boxes, the intersection over union (IOU) method was applied, and the threshold was set to 0.6. As a result, the remaining candidate boxes became small in number but contained nearly all traffic lights (Figure 2 e)).

Now, the detector is quite qualified for the detection task. To further shorten the runtime of the proposed approach, we need to reduce the backgrounds of the candidate boxes. In Figure 2 e), some candidate boxes do not visibly contain traffic lights. We can make this judgment by the image texture. The fractal dimension is a useful feature for texture segmentation in image processing [32]. The box-counting approach is one of the most frequently used techniques for estimating the fractal dimension of an image and is defined below.

$$F = \lim_{r \to 0} \frac{\log(N_r)}{\log(1/r)} \tag{8}$$

where F is the fractal dimension and N_r is the least number of boxes that must completely cover the broken lines in the scale r. Before calculating the fractal dimension, the image first needs to be processed by the Sobel filter. Then, the image is transformed to binarization. Thus, the gray plane is converted to the broken lines in two dimensions (Figure 4). By calculating the numbers of white and black pixels, respectively, the ratio equation is as follows.

$$R = \frac{N(pix = 255)}{N(pix = 255) + N(pix = 0)} \tag{9}$$

where R is the ratio and $N(pix = 255)$ and $N(pix = 0)$ denote the number of pixels, with values equal to 255 and 0, respectively.

The conspicuous texture discrepancy between traffic lights and backgrounds can be quantified by calculating the fractal dimension and the R value (Figure 4). The fractal dimension of traffic lights principally ranges from 1.3 to 2.0. However, the fractal dimension range for backgrounds is from 0 to 2. Thus, a threshold line can be drawn to distinguish them. The R value is also helpful in distinguishing traffic lights and backgrounds. As shown in Figure 4 g), the fractal dimension of the image was high, indicating that the texture of the image was rough. The spatial distribution of the texture may be similar to that of traffic lights, which makes it difficult to distinguish the texture. However, its R value was evidently lower than that of traffic lights. Therefore, the texture noise can be further eliminated by the R value.

2.2.2 Classifier

Once the candidate boxes are generated, the classification task can begin. In this paper, the candidate boxes contain four states, i.e. the backgrounds, red lights, green lights and yellow lights. To distinguish them, SqueezeNet was employed. Among the SqueezeNet modules, the fire module, which is comprised of a squeeze convolution layer and an expand layer, is the building block. It can enable the CNN to preserve accuracy on a limited budget of parameters [31]. SqueezeNet begins with a standalone convolution layer (conv1), followed by 8 fire modules (fire2–9), and finally ending with a conv layer (conv10). The number of filters per fire module from the beginning to the end is gradually increased. Moreover, SqueezeNet performs max-pooling with a stride of 2 after layers conv1, fire4, fire8, and conv10. According to the SqueezeNet architecture, we reproduced it with the TensorFlow library. The size of the input image was adjusted to 64×64. The channel of each layer retained

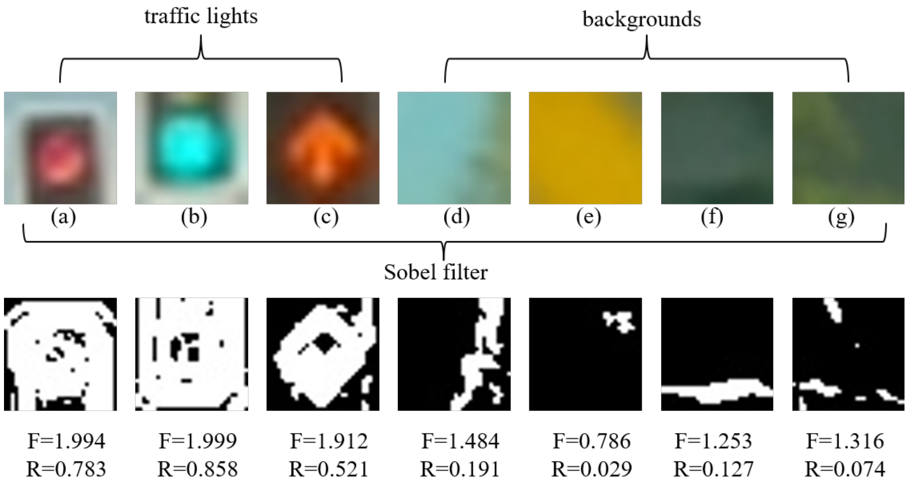


Figure 4. Texture discrepancy between traffic lights and backgrounds

its original value. The number of categories was set to 4. The learning rate was set to 0.0001. The input images were first processed by the equalization method before training SqueezeNet.

2.2.3 Postprocessing

The candidate boxes were transformed into classified boxes after processing them with the classifier. At this step, several overlaps may still exist among the boxes. To collapse them, the non-maximum suppression (NMS) method was needed. NMS is a key postprocessing step in many computer vision tasks. On the basis of the sorted scores, it uses an iterative procedure to retain only one box per group, corresponding to the precise local maximum of the response function, ideally obtaining only one detection per object [33]. Then, the perspective relationship (Equation (7)) was used again to slightly reshape the boundaries of the recognized traffic lights. The overall processes of the entire method are summarized in Figure 5.

2.3 Reference Methods

The performance of the proposed approach is compared with that of several popular approaches. One approach is YOLO v3, which was briefly introduced in the first section. YOLO v3 still imitates the mechanism of the human visual system. It strives to predict what objects are present and where they are by looking at an image only once. Similar to the previous versions, YOLO v3 regards object detection as a single regression problem, straight from image pixels to bounding box coordinates

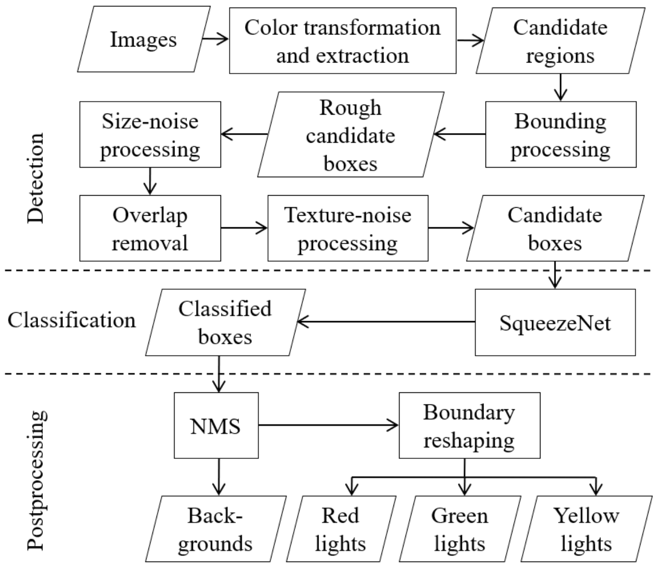


Figure 5. Flow chart of the proposed approach

and class probabilities. The codes of YOLO are open. According to the instructions, YOLO v3 is used on the Linux platform.

Other approaches include the de Charette approach [9], DeepTLR model [19], FusionTLR model [25], Bayesian algorithm [35], Haltakov approach [11], and Siogkas approach [36]. These approaches, which have been validated using a publicly available dataset, are applied to evaluate our method. Details about these approaches are provided in the corresponding literature.

2.4 Evaluation Metrics

The approaches used to evaluate the recognition performances are defined as follows:

$$Acc = (TP + TN)/(TP + TN + FP + FN), \tag{10}$$

$$Pre = TP/(TP + FP), \tag{11}$$

$$Rec = TP/(TP + FN), \tag{12}$$

$$F1 = 2 \cdot (Pre \times Rec)/(Pre + Rec) \tag{13}$$

where *Acc* is the accuracy, *TP* is the number of true positives, *TN* is the number of true negatives, *FP* is the number of false positives, and *FN* is the number of false negatives. *Pre* is the precision, and *Rec* is the recall. According to the *Pre* and *Rec* values, the *F1* score is calculated.

When compared with other models, the metric 'AUC', i.e. the area-under-the-curve, of the precision-recall curves was also used in addition to the above indicators.

2.5 Experimental Situation

The image dataset was divided into two parts: training data and verification data. In the training data, the training samples created by the detector contain the red lights, green lights, yellow lights and backgrounds. The backgrounds account for nearly 94 %. To balance the sample ratios, we used some strategies to process the positive sample data. These strategies include moving the four borders of the traffic light separately, graying the image, scaling the image, using a Gaussian filter, sharpening the image, equalizing the image, stretching the image to the left and right, and stretching the image upward and downward. Based on the image dataset, the model training and verification platform was a single-core CPU @ 2.5 GHz.

3 RESULTS ANALYSIS

3.1 Classifier Performance

The classifier, i.e. SqueezeNet, used in the proposed approach was trained 25 000 times by learning the training datasets (Figure 6). The corresponding weight files were produced per 2 500 times. The accuracy varied sharply before 3 000 iterations and gently after. The maximum value of the accuracy was 0.973, which appeared near the 22 000th iteration. The variation in the loss function curve was acute from beginning to end. However, the trend of the loss curve was clear. At the 12 500th iteration, the minimum value appeared, and the corresponding accuracy was 0.968. After that, the change was slightly dramatic. However, the trend of the loss curve was not affected.

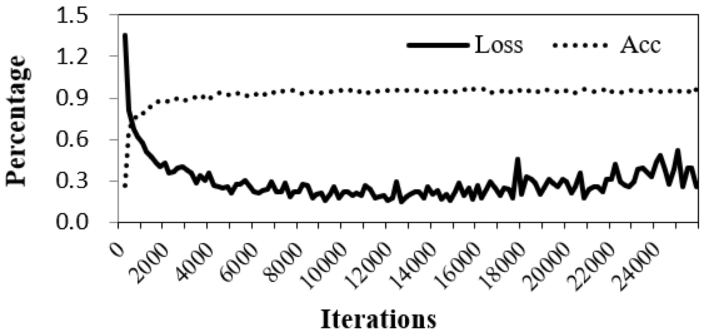


Figure 6. Training curve of the classifier

The validation dataset was used to evaluate the classification performance. Figure 7 shows the precision and recall of the classifier at different iterations. Before the

7 500th iteration, the manifestation, i.e. distinguishing the four types of candidate boxes, was unsatisfactory. In this stage, the classifier distinguished the red traffic lights and the green traffic lights comparatively easily. However, for the yellow traffic lights and the backgrounds the situation was terrible. The precision of distinguishing backgrounds was lower than 0.8, and the recall was higher than 0.9. The recall of distinguishing the yellow traffic lights was lower than 0.7, and the precision was higher than 0.87. These results indicate that more yellow traffic lights were falsely classified as backgrounds by the classifier. After that the 7 500th iteration, the classifier produced better classification results. Both the precision and the recall when distinguishing the four types of candidate boxes were high. Figure 8 also provides the overall classification accuracy and the corresponding *F1* score. Obviously, at the 12 500th iteration, the performance of the classifier was the best. The performance when distinguishing red traffic lights was inferior to that when distinguishing green traffic lights but better than that when distinguishing yellow traffic lights.

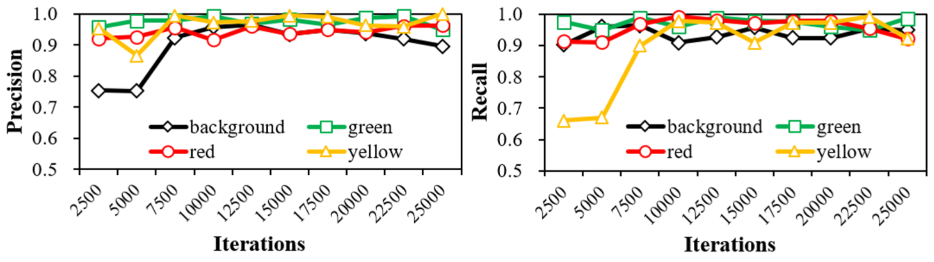


Figure 7. Precision a) and recall b) of the classifier

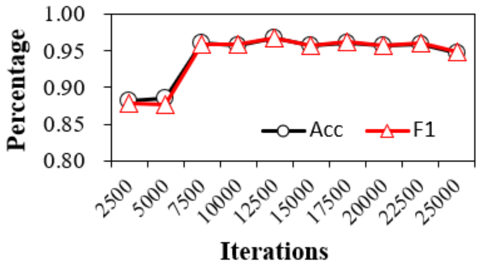


Figure 8. Overall classification accuracy and *F1* score of the classifier

3.2 Recognition Evaluation

The verification images were used to evaluate the recognition performance of the proposed approach. The *F1* score varied with the IOU threshold (Figure 9). Obviously, the overall recognition accuracy declined with increasing IOU. The situation

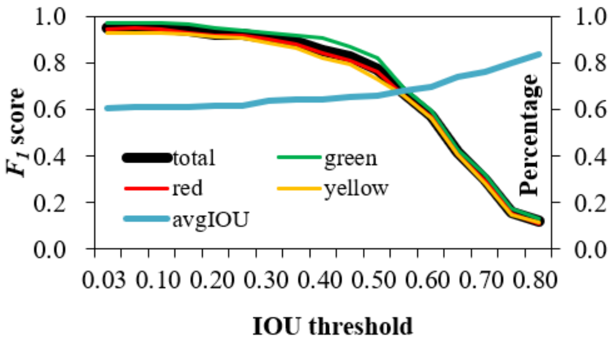


Figure 9. Relationships between the recognition accuracy and the IOU

remained the same for each type of traffic light. However, irrespective of the change in the IOU, the average IOU curve always exceeded 0.6. When the IOU was lower than 0.1, the total *F1* score maximum value was 0.951, and the corresponding average IOU was 0.611. In the figure, when the IOU falls between 0.1 and 0.3, the total *F1* score only slightly varied and was always higher than 0.9. When the IOU varied from 0.3 to 0.45, the total *F1* score was always greater than 0.8. When the IOU was between 0.45 and 0.6, the total *F1* score was always greater than 0.6.

These results suggest that the performance of the proposed approach in terms of recognizing traffic lights was satisfactory. The estimation of the light boundaries by the proposed approach was also approbatory. Moreover, the red traffic light recognition performance achieved by the proposed approach was inferior to the performance of green traffic light recognition but superior to that of yellow traffic light recognition.

The runtime of the proposed approach was analyzed (Figure 10). After the addition or removal of modules, the runtimes of the corresponding approaches were compared. In Figure 10 a), the total runtime of the 'full' approach was the shortest, followed by that of the 'perspective' approach, while the longest was that of the 'none' approach. In the runtime list of each approach, the elapsed time of the detector was the shortest. However, the detector of the 'fractal' approach had a longer elapsed time than that of the other approaches, which was associated with the fractal computation. Relative to the others, the detector of the 'fractal' approach required extra time to accomplish the fractal-computation task. Multiple values needed to be calculated in each candidate box to estimate the fractal dimension. Thus, the stage of the 'fractal' approach was somewhat time-consuming.

The elapsed time of the classifier was longest and directly affected the total runtime of each approach. In fact, in this stage, the elapsed time was related to the classifier-self and the number of candidate boxes. The former, i.e., the classifier distinguishing each candidate box, was approximately 5.5 ms. The latter was not constant. When the number of candidate boxes was large, the elapsed time of the classifier was necessarily long, and vice versa. When neither the perspective relation module nor the fractal dimension module was used, the number of candidate boxes was very large. Conversely, the number of candidate boxes was small. Thus, the elapsed time of the 'none' approach was the longest, followed by that of the 'fractal' approach, while the shortest was that of the 'all' approach. Therefore, the two modules played important roles in reducing the runtime (Figure 10 b)). The time compression ratio of the perspective relation module was as high as 0.57 and that of the fractal dimension module was 0.29. Using the two modules synchronously, the time compression ratio was as high as 0.62. Finally, the average runtime needed to process a single image for the 'full' approach was approximately 240 ms, indicating that the proposed approach can process four images per second. When detecting traffic lights under the offline condition, the proposed approach was very fast. Even so, the proposed approach can completely satisfy the practical requests for some online tasks requiring real-time image processing, such as processing the floating car data from the low frequency (> 30 s) to the high frequency (1–10 s).

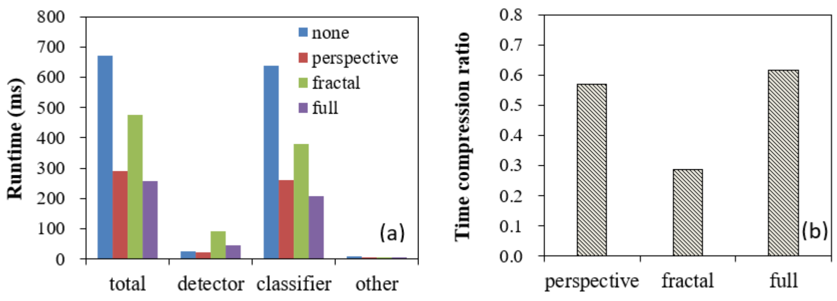


Figure 10. Analysis of runtime. In the figures, 'perspective' denotes the approach containing the perspective relation module; 'fractal' denotes the approach containing the fractal dimension module; 'all' denotes the approach containing the two modules; and 'none' represents the approach without the two modules.

Figure 11 shows the performance in recognizing the different sizes of traffic lights. The minimum resolution of the proposed approach for recognizing the traffic lights was set to five pixels. When the size of the traffic lights was lower than 8 pixels, the *F1* score was always higher than 0.92. When the size of the traffic lights was expanded to 11 pixels, the proposed approach yielded its best results. Then, the *F1* score of the proposed approach declined. This decline is related to the chromatism of traffic lights, which is frequently caused by light pollution. For example, when the backlight behind the traffic light is too bright, the traffic light color becomes lighter. In this case, the proposed approach can do nothing due to the existence of chromatism. The statistics showed that larger traffic lights were more likely to be impacted by light pollution than smaller lights, which explains the phenomenon above. From another perspective, the proposed approach has prominent advantages in recognizing small traffic lights.

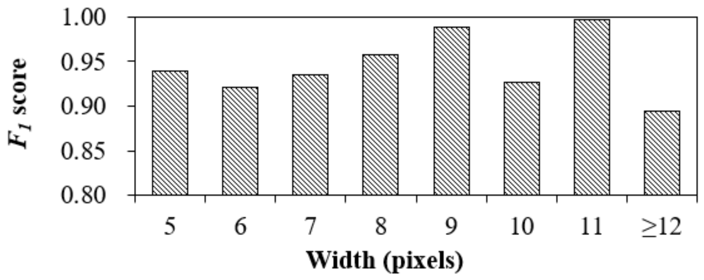


Figure 11. Traffic light size recognition performance achieved by using the proposed approach

4 DISCUSSION

4.1 Comparison with the Other Reference Models

A comparison of the recognition results of different approaches based on the LaRA dataset is shown in Table 1. Compared with state-of-the-art approaches, the *F1* score of the proposed approach was slightly lower than that of the Bayesian algorithm but higher than the scores of other approaches. The recognition performance of the proposed approach in the reference models is superior. Similar to the Bayesian algorithm and de Charette approach, the precision of the proposed approach was higher than that of the recall. This finding shows that the error rate for traffic signals recognized by the proposed approach was lower than the omission rate. However, opposite results were obtained for the Haltakov approach, Siogkas approach and DeepTLR model. During the verification, we discovered that some traffic signals appeared very small and some light textures were vague. In addition, the similarity between the traffic signals and the backgrounds due to the lower image quality hindered detection. These factors reduced the recall of the proposed approach and increased the omission rate. In addition to comparing the *F1* scores, the area under the curve (AUC) indicators were also compared. Specifically, a comparison between the proposed approach and the FusionTLR model was performed. The red light recognition performance achieved by the FusionTLR model was obviously better than the green light recognition performance. However, the case of the proposed approach was completely different. For the LaRA dataset, some red signals that were quite small and blurry were very similar to the backgrounds. Thus, their detection using the proposed approach was difficult.

The runtime costs were also analyzed. The runtime of the proposed approach was approximately 120 ms per image on the LaRA dataset, which is longer than that of most approaches and indicates that the proposed approach based on the current implementation was relatively disadvantaged in processing the video images.

Approach	**Precision**	**Recall**	***F1***	**AUC**$_{Red}$	**AUC**$_{Green}$	**Platform**	**Reference**
Bayesian algo.	0.987	0.947	0.966				[35]
FusionTLR				0.920	0.893	CPU	[25]
Haltakov approach	0.728	0.801	0.763			CPU	[11]
Siogkas approach	0.612	0.938	0.741			CPU	[36]
DeepTLR	0.856	0.907	0.881			GPU	[19]
de Charette approach	0.845	0.535	0.655			CPU	[9]
Ours	0.904	0.897	0.900	0.898	0.930	CPU	

Table 1. Comparison of the proposed approach with state-of-the-art methods using the LaRA dataset

Furthermore, we compared the proposed approach with the other more famous object detection model, i.e. the YOLO model. The validation results revealed that the performances of YOLO were unsatisfactory on both our datasets and the LaRA dataset. The statistical results showed that both the precision and recall of YOLO were lower than those of the proposed approach. A comparison of their vision results is shown in Figure 12. Subplot (a) shows that the proposed approach can distinguish the red and yellow lights better. However, YOLO falsely classified the yellow lights as red lights. In subplot (b2), mistakes and omissions in recognizing the traffic lights occurred when using YOLO. This observation indicates that YOLO is unsuitable for detecting small objects, which is in agreement with the viewpoints of some studies [20, 25].

4.2 Strategies for Further Model Optimization

4.2.1 Improving the Recognition Accuracy

The recognition accuracy of the proposed approach depends on the candidate box quality of the detector and the classification accuracy of the classifier. The candidate boxes produced by our detector did not ensure coverage of all traffic lights. The statistics showed that the probability of the candidate boxes covering all traffic lights was approximately 0.98. The reason for this was that our detector was sensitive to the image color, which was closely related to the image quality. Many factors affect the image quality of an on-vehicle camera. The major elements contain image blurring due to shaking of the camera, color distortion due to light pollution, and the lower resolution of the camera. Upon encountering poor image quality, omission may occur for the candidate boxes. Therefore, further optimization of our detector will be studied in future work.

In the subsequent classification, a deep CNN, i.e. SqueezeNet, was applied. SqueezeNet is small and has a high classification precision. However, in this paper,

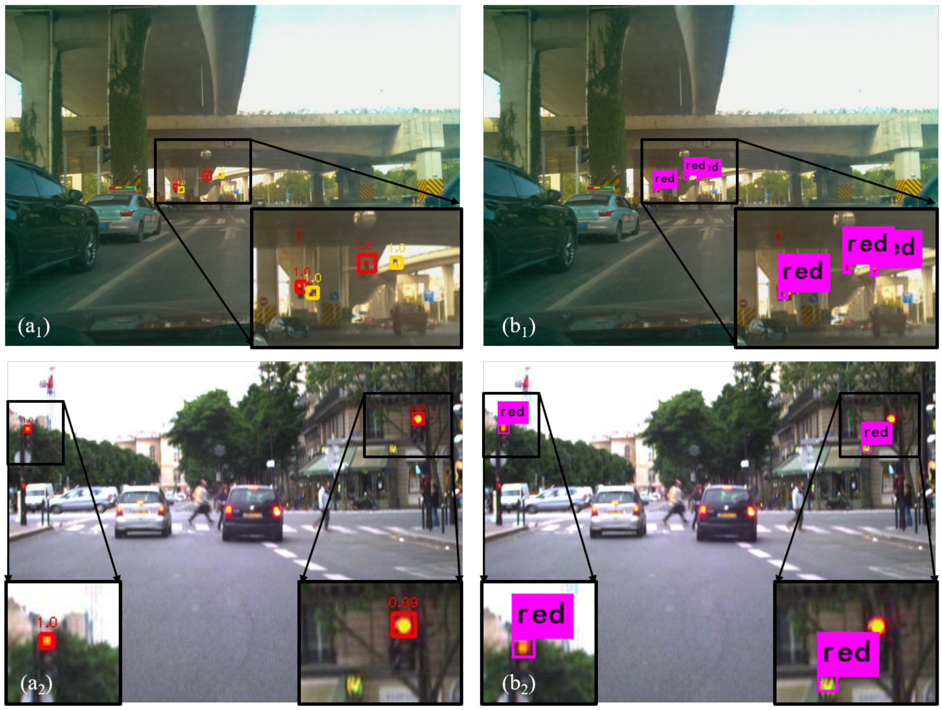


Figure 12. Vision results on 1) our validation dataset and 2) the LaRA validation dataset obtained by a) the proposed approach and b) the YOLO model

there is an upper limit to its classification precision. When training SqueezeNet, we found some difficulty in exceeding the *F1* score of 0.97 regardless of how the training samples were adjusted. Thus, we designed and developed a new and simple but highly efficient deep CNN classifier, denoted as SimpleNet. SimpleNet contains two convolution layers, two pooling layers and two fully connected layers. The loss function uses cross entropy. After tens of thousands of iterations, the *F1* score of SimpleNet can reach 0.985, which is then very difficult to exceed. Figure 13 shows the misclassifications between the classifiers. Obviously, both classifiers faced the same major challenges in distinguishing the backgrounds and traffic lights. However, the misclassification error between the interiors of the traffic lights was lower for the classifiers. Although SimpleNet has better classification performance, its file size is large, and the runtime is long. Therefore, some tradeoff among the model runtime, accuracy and size must occur.

By examining the classification data, we found that the contexts around some traffic lights were inadequate. This inadequacy caused the backgrounds to be similar to the traffic lights, especially for small traffic lights. The smallest resolution of the proposed approach was 5 pixels. When the size further decreased, the traffic

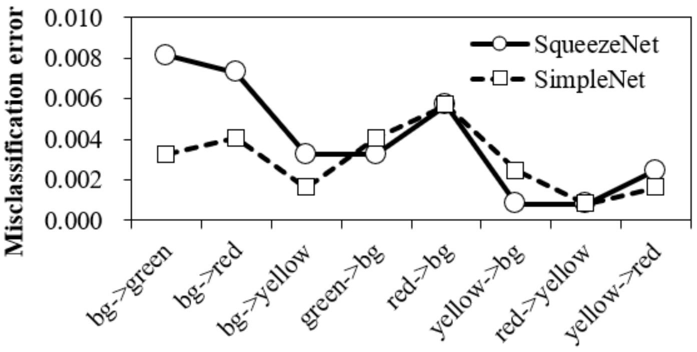


Figure 13. Analysis of the misclassification of classifiers

lights became very similar in appearance to the backgrounds. As a result, it was difficult to detect them by human eyes. This situation makes some traffic light detection systems meaningless because the on-vehicle camera was too far away from the traffic lights. Usually, human eyes require an increase in context information, i.e., expansion of the visual boundaries, to recognize small objects. However, this increase did not significantly improve the classifier performance because too many contextual data will overwrite the features of traffic lights. This deficiency stems from the fact that the classification by the CNN is based on the local image texture [34]. Thus, to resolve this problem, the use of contextual data to improve the classification performance of our classifier will be researched in future work.

4.2.2 Reducing the Runtime

Compared with that of the other traffic light recognition approaches, the runtime of the proposed method is slightly longer. However, the runtime of the proposed method can be considerably reduced in theory.

First, the proposed approach adopts serial computing technology, i.e. single-thread sequential processing, in the implementation. In the candidate box classification, the classifier sequentially classifies the candidate boxes. If more candidate boxes exist, the serial processing mechanism of the classifier will lead to accumulation of the runtime. Therefore, the overall runtime will be longer. In fact, no correlation among the candidate boxes exists, which is good for parallel computing. Under a single-core CPU, the processing of a single candidate box with our classifier requires approximately 5.5 ms. If multiprocess is used, according to Figure 10 a), the total time needed by the proposed approach to process a single image should not exceed 25 ms to ensure that video image processing and online real-time processing are possible. Therefore, in theory, it is feasible to use parallel computing to optimize the proposed approach, and the computational efficiency after the optimization will be significantly improved.

Second, for video images, the target tracker technology, which has a runtime advantage when processing frame images, can be coupled to our model. The multiple trackers were initialized at first by the traffic light boxes recognized by the proposed approach in the previous frame image and were then used to find the new positions of traffic lights in the next frame image and update the target position corresponding to each tracker, thereby improving the traffic light detection efficiency.

Finally, for on-vehicle camera images, the location of traffic lights has a certain regularity. In some areas, a high probability of traffic lights exists, while in other areas, the probability might be small or equal to zero. For example, the probability of a traffic light appearing above the horizon in an image is large. In contrast, traffic lights typically do not appear below the horizon. Therefore, an invalid scan of the algorithm can be prevented by setting the view boundary appropriately, thereby improving the detection efficiency and quality.

In addition, using a GPU for calculations may also significantly improve the efficiency of the proposed approach. Therefore, according to the above strategies, further optimization of the proposed approach to reduce the runtime will be performed in future studies.

4.3 Potential for Migrating to Embedded Devices

Currently, automatic recognition of traffic signs is very important for vehicle automatic driving or assisted driving. Traffic sign recognition systems have become an integral part of Advanced Driver Assistance Systems (ADAS). Fast, robust and real-time automatic traffic sign detection and recognition can significantly increase driving safety. Although many traffic signal recognition methods are available, they are not easily migrated to embedded devices because the hardware performance of embedded devices is generally lower than the computer conditions in the laboratory.

With the constraint, the proposed approach should meet the specified conditions. First, the proposed approach has a high accuracy rate and recall rate in terms of the recognition performance and has better performance in reference approaches. Second, the proposed approach has the potential to improve the time efficiency in terms of the cost calculation. According to the previous analysis, when multiple processes are employed for parallel classification and the target tracker technology has been adopted, this approach is fully applicable to online real-time recognition tasks. Last, the proposed approach is lightweight. The total size of the approach, including the trained weight file, does not exceed 10 MB. The approach only depends on the OpenCV and Tensorflow libraries during implementation; thus, it has minimal dependence on third-party development libraries and implementation is simple. These characteristics enable the proposed approach to have great potential for migration to embedded devices.

Similar to other methods, the proposed approach inevitably contains some parameters and coefficients, such as the hue, saturation and value (HSV) thresholds applied to extract the traffic signal candidate regions, the coefficients for establishing the perspective relationship equation, and the fractal and R thresholds applied for texture denoising. The optimal values of these parameters and coefficients may be related to the test environments. When migrating from one region to another region, these parameters and coefficients may need to be adjusted slightly to obtain better recognition results. However, if the training data are sufficient, these parameters and coefficients will yield global values or local optimal values classified by region in the form of a list. The same situation applies to the weight file of the classifier. The proposed approach at this time can address the variability of the scenarios as much as possible and no additional setting is required.

In future work, the model migration research for embedded devices, such as Raspberry Pi with ARM chip, will be investigated, and a comparative analysis will be performed with the existing ADAS products.

5 CONCLUSION

We proposed an approach to detect traffic lights by using a combination of image processing and deep learning. First, we designed a detector to create candidate boxes that can cover all traffic lights based on image processing. In the process, the perspective relationship and the fractal dimension were both considered to dramatically reduce the number of invalid candidate boxes. Then, the candidate boxes were transformed into classified boxes by using the classifier. Finally, the traffic light boxes were produced from the classified boxes via postprocessing. Overall, the approach occupies a small amount of storage space.

We trained and validated the proposed approach on our dataset and the LaRA dataset. Several state-of-the-art methods were employed as the reference approaches. The results showed that the performance of the proposed approach in terms of recognizing traffic lights was satisfactory. In addition, estimation of the light boundary by the proposed approach was approbatory.

The red traffic light recognition performance achieved by the proposed approach was inferior for green traffic lights but superior for yellow traffic lights. The proposed approach has prominent advantages in recognizing small traffic lights.

Compared with the reference models, the proposed approach has a significant competitive advantage in terms of the recognition accuracy. Under the current serial program architecture, the proposed approach was relatively disadvantaged in processing video images. However, the runtime of the proposed approach can be further greatly reduced in future work by using parallel computing to carry out optimization.

Furthermore, the proposed approach was very small regarding the model size. The file size of the proposed approach including the CNN weight file was less than 10 MB. Thus, this approach can be used on smart terminals, mobile devices or embedded devices in the future.

Acknowledgement

This project was supported by the Scientific Research Foundation for the introduction of talents in Nantong University (No. 17R27), Youth Project of Natural Science Foundation of Jiangsu Province (No. BK20170440) and National Natural Science Foundation of China (No. 41501422). We thank to Feitao Ma, an engineer from Alibaba Group, for his help in adjusting the training samples.

REFERENCES

[1] Diaz, M.—Cerri, P.—Pirlo, G.—Ferrer, M. A.—Impedovo, D.: A Survey on Traffic Light Detection. In: Murino, V., Puppo, E., Sona, D., Cristani, M., Sansone, C. (Eds.): New Trends in Image Analysis and Processing – ICIAP 2015 Workshops (ICIAP 2015). Springer, Cham, Lecture Notes in Computer Science, Vol. 9281, 2015, pp. 201–208, doi: 10.1007/978-3-319-23222-5_25.

[2] Ivanchenko, V.—Coughlan, J.—Shen, H.: Real-Time Walk Light Detection with a Mobile Phone. In: Miesenberger, K., Klaus, J., Zagler, W., Karshmer, A. (Eds.): Computers Helping People with Special Needs (ICCHP 2010). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6180, 2010, pp. 229–234, doi: 10.1007/978-3-642-14100-3_34.

[3] Vu, A.—Ramanandan, A.—Chen, A.—Farrell, J. A.—Barth, M.: Real-Time Computer Vision/DGPS-Aided Inertial Navigation System for Lane-Level Vehicle Navigation. IEEE Transactions on Intelligent Transportation Systems, Vol. 13, 2012, No. 2, pp. 899–913, doi: 10.1109/TITS.2012.2187641.

[4] Che, M.—Wang, Y.—Zhang, C.—Cao, X.: An Enhanced Hidden Markov Map Matching Model for Floating Car Data. Sensors (Basel), Vol. 18, 2018, Art. No. 1758, 19 pp., doi: 10.3390/s18061758.

[5] van de Sande, K. E. A.—Uijlings, J. R. R.—Gevers, T.—Smeulders, A. W. M.: Segmentation as Selective Search for Object Recognition. 2011 International Conference on Computer Vision (ICCV), 2011, pp. 1879–1886, doi: 10.1109/ICCV.2011.6126456.

[6] Uijlings, J. R. R.—van de Sande, K. E. A.—Gevers, T.—Smeulders, A. W. M.: Selective Search for Object Recognition. International Journal of Computer Vision, Vol. 104, 2013, No. 2, pp. 154–171, doi: 10.1007/s11263-013-0620-5.

[7] Zitnick, C. L.—Dollár, P.: Edge Boxes: Locating Object Proposals from Edges. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (Eds.): Computer Vision – ECCV 2014. Springer, Cham, Lecture Notes in Computer Science, Vol. 8693, 2014, pp. 391–405, doi: 10.1007/978-3-319-10602-1_26.

[8] Kim, H. K.—Yoo, K. Y.—Park, J. H.—Jung, H. Y.: Traffic Light Recognition Based on Binary Semantic Segmentation Network. Sensors (Basel), Vol. 19, 2019, No. 7, Art. No. 1700, 15 pp., doi: 10.3390/s19071700.

[9] de Charette, R.—Nashashibi, F.: Real Time Visual Traffic Lights Recognition Based on Spot Light Detection and Adaptive Traffic Lights Templates. IEEE Intelligent Vehicles Symposium, 2009, pp. 358–363, doi: 10.1109/IVS.2009.5164304.

[10] Diaz-Cabrera, M.—Cerri, P.—Medici, P.: Robust Real-Time Traffic Light Detection and Distance Estimation Using a Single Camera. Expert Systems with Applications, Vol. 42, 2015, No. 8, pp. 3911–3923, doi: 10.1016/j.eswa.2014.12.037.

[11] Haltakov, V.—Mayr, J.—Unger, C.—Ilic, S.: Semantic Segmentation Based Traffic Light Detection at Day and at Night. In: Gall, J., Gehler, P., Leibe, B. (Eds.): Pattern Recognition (DAGM 2015). Springer, Cham, Lecture Notes in Computer Science, Vol. 9358, 2015, pp. 446–457, doi: 10.1007/978-3-319-24947-6_37.

[12] Chen, Z.—Huang, X.: Accurate and Reliable Detection of Traffic Lights Using Multiclass Learning and Multiobject Tracking. IEEE Intelligent Transportation Systems Magazine, Vol. 8, 2016, No. 4, pp. 28–42, doi: 10.1109/MITS.2016.2605381.

[13] Islam, K. T.—Raj, R. G.: Real-Time (Vision-Based) Road Sign Recognition Using an Artificial Neural Network. Sensors (Basel), Vol. 17, 2017, No. 4, Art. No. 853, 32 pp., doi: 10.3390/s17040853.

[14] Zhou, X.—Yuan, J.—Liu, H.: Real-Time Traffic Light Recognition Based on C-HOG Features. Computing and Informatics, Vol. 36, 2017, No. 4, pp. 793–814, doi: 10.4149/cai_2017_4_793.

[15] John, V.—Yoneda, K.—Qi, B.—Liu, Z.—Mita, S.: Traffic Light Recognition in Varying Illumination Using Deep Learning and Saliency Map. 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2014, pp. 2286–2291, doi: 10.1109/ITSC.2014.6958056.

[16] Ren, F.—Huang, J.—Jiang, R.—Klette, R.: General Traffic Sign Recognition by Feature Matching. 2009 24th International Conference Image and Vision Computing, New Zealand, 2009, pp. 409–414, doi: 10.1109/IVCNZ.2009.5378370.

[17] Kim, H. K.—Shin, Y. N.—kuk, S. G.—Park, J. H.—Jung, H. Y.: Night-Time Traffic Light Detection Based on SVM with Geometric Moment Features. International Journal of Computer and Information Engineering, Vol. 7, 2013, No. 4, pp. 472–475.

[18] Gómez, A. E.—Alencar, F. A. R.—Prado, P. V.—Osório, F. S.—Wolf, D. F.: Traffic Lights Detection and State Estimation Using Hidden Markov Models. IEEE Intelligent Vehicles Symposium Proceedings, 2014, pp. 750–755, doi: 10.1109/IVS.2014.6856486.

[19] Weber, M.—Wolf, P.—Zöllner, J. M.: DeepTLR: A Single Deep Convolutional Network for Detection and Classification of Traffic Lights. IEEE Intelligent Vehicles Symposium (IV), 2016, pp. 342–348, doi: 10.1109/ivs.2016.7535408.

[20] Behrendt, K.—Novak, L.—Botros, R.: A Deep Learning Approach to Traffic Lights: Detection, Tracking, and Classification. IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 1370–1377, doi: 10.1109/ICRA.2017.7989163.

[21] Girshick, R.—Donahue, J.—Darrell, T.—Malik, J.: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.

[22] Girshick, R.: Fast R-CNN. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[23] Ren, S.—He, K.—Girshick, R.—Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems 28 (NIPS 2015), 2015, pp. 91–99.

[24] He, K.—Gkioxari, G.—Dollár, P.—Girshick, R.: Mask R-CNN. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988, doi: 10.1109/ICCV.2017.322.

[25] Li, X.—Ma, H.—Wang, X.—Zhang, X.: Traffic Light Recognition for Complex Scene with Fusion Detections. IEEE Transactions on Intelligent Transportation Systems, Vol. 19, 2018, No. 1, pp. 199–208, doi: 10.1109/TITS.2017.2749971.

[26] Redmon, J.—Divvala, S.—Girshick, R.—Farhadi, A.: You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788, doi: 10.1109/CVPR.2016.91.

[27] Liu, W.—Anguelov, D.—Erhan, D.—Szegedy, C.—Reed, S.—Fu, C. Y.—Berg, A. C.: SSD: Single Shot Multibox Detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.): Computer Vision – ECCV 2016. Springer, Cham, Lecture Notes in Computer Science, Vol. 9905, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0_2.

[28] Shafiee, M. J.—Chywl, B.—Li, F.—Wong, A.: Fast YOLO: A Fast You Only Look Once System for Real-Time Embedded Object Detection in Video. arXiv preprint arXiv:1709.05943, 2017.

[29] Redmon, J.—Farhadi, A.: YOLO9000: Better, Faster, Stronger. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.

[30] Redmon, J.—Farhadi, A.: YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767, 2018.

[31] Iandola, F. N.—Han, S.—Moskewicz, M. W.—Ashraf, K.—Dally, W. J.—Keutzer, K.: SqueezeNet: AlexNet-Level Accuracy with 50× Fewer Parameters and < 0.5 MB Model Size. arXiv preprint arXiv:1602.07360, 2016.

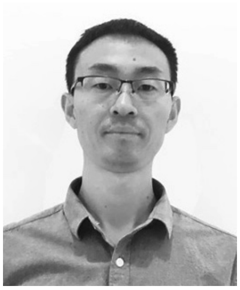
[32] Li, J.—Du, Q.—Sun. C.: An Improved Box-Counting Method for Image Fractal Dimension Estimation. Pattern Recognition, Vol. 42, 2009, No. 11, pp. 2460–2469, doi: 10.1016/j.patcog.2009.03.001.

[33] Rothe, R.—Guillaumin, M.—Van Gool, L.: Non-Maximum Suppression for Object Detection by Passing Messages Between Windows. In: Cremers, D., Reid, I., Saito, H., Yang, M. H. (Eds.): Computer Vision – ACCV 2014. Springer, Cham, Lecture Notes in Computer Science, Vol. 9003, 2014, pp. 290–306, doi: 10.1007/978-3-319-16865-4_19.

[34] Geirhos, R.—Rubisch, P.—Michaelis, C.—Bethge, M.—Wichmann, F. A.—Brendel, W.: ImageNet-Trained CNNs Are Biased Towards Texture; Increasing Shape Bias Improves Accuracy and Robustness. ICLR 2019, arXiv:1811.12231, 2019.

[35] Hosseinyalamdary, S.—Yilmaz, A.: A Bayesian Approach to Traffic Light Detection and Mapping. ISPRS Journal of Photogrammetry and Remote Sensing, Vol. 125, 2017, pp. 184–192, doi: 10.1016/j.isprsjprs.2017.01.008.

[36] Siogkas, G.—Skodras, E.—Dermatas, E.: Traffic Lights Detection in Adverse Conditions Using Color, Symmetry and Spatio-Temporal Information. International Conference on Computer Vision Theory and Applications – Volume 2: VISAPP (VISIGRAPP 2012), Rome, Italy, 2012, pp. 620–627, doi: 10.5220/0003855806200627.



Mingliang Che received his Ph.D. in cartography and geography information system from the University of Chinese Academy of Sciences in 2016. His current research interests include map matching, image processing and deep learning.

Mingjun Che received his B.Sc. Degree in computer science and technology from the Shandong Agricultural University in 2004. He currently works as an engineer in Wuxianshenghuo (Hangzhou) Info Tech Ltd. He does research in image processing, computer vision and artificial intelligence.

Zhenhua Chao is Associate Professor of the School of Geographic Sciences of Nantong University in China. He received his Ph.D. in remote sensing information extraction. His research interests are computer and information extraction. He has been involved in research areas in statistical downscaling and quantitative remote sensing for quite a time. He has published many papers in high quality publications.

Xinliang Cao received his M.Sc. Degree in software engineering from the University of Science and Technology of China in 2015. His current research interests include internet of things and artificial intelligence.