SqueezeDet-Based Nighttime Traffic Light Detection with Filtering Rules*

Yongbo Huo[1] Zhijing XU[1] Shitao Chen[1*] Yu Chen[1] Yuhao Huang[1] Nanning Zheng[1]

***Abstract*—Traffic light detection is an indispensable algorithm module in autonomous driving system. In general traffic scenarios, the current mainstream algorithms are able to detect and recognize traffic lights accurately. However, these algorithms may fail in the nighttime detection task due to the quality decrease of camera image, which is caused by the multiple light sources in this scene. Therefore, this paper proposed a SqueezeDet-based nighttime traffic light detection algorithm with false detection filtering rules. The remarkable contributions of this algorithm are: 1) Modifying the anchor size of the native SqueezeDet to fit the bounding box of the traffic lights, which improves the accuracy of the model. 2) Roughly determining the position of the traffic light in the image according to the prior knowledges based on the traffic lights, and the image is cropped to reduce the calculation time of the model 3) Formulating the filtering rules based on the position characteristics of the traffic lights, which improves the precision of the algorithm. In order to verify the performance of the algorithm, we performed experiments on our collected dataset and compared with the advanced target detection technology. The result demonstrates that our algorithm has a significant improvement in accuracy and speed.**

***Index Terms*—traffic light detection, convolutional neural network, filtering rules**

I. Introduction

In the past ten years, autonomous driving technology is rapidly developing as a promising game changer in autonomy and transportation. Driver assistance systems such as Adaptive Cruise Control (ACC) [1], Autonomous Parking System (APS) [2], and Autonomous Emergency Braking (AEB) [3] take important roles in guaranteeing both the safety and ride comfort while driving. Information gathered from traffic lights and signs detection are essential elements for a self-driving car to behave correctly. In the future, traffic flow control system can manage the traffic light signals through the Internet of Vehicle (IoV) [4] methods such as Vehicle-to-Infrastructure (V2I) [5] communication technology. And autonomous vehicles can also communicate with traffic lights system in this way. However, in both current and future periods, autonomous vehicles are more likely to run on the road together with manned vehicles in daily life, which requires a precise recognition of traffic light signals designed for human drivers at intersections.

Traffic Light Detection (TLD) [6]–[13] is a quite challenging task, since the detection algorithm is supposed to handle with traffic lights of different types, sizes and lighting conditions. Moreover, the algorithm should also be accurate and real-time. Nevertheless, the TLD at night is more difficult compared to that in daytime, which has the following challenges:

- In toward-light environments, the traffic lights seem to be larger than the actual size because of the halo phenomenon and the color of traffic lights is close to white due to the center color distortion, see Fig. 1(a) and Fig. 1(b).
- Street lights, car taillights and other light sources are interferences similar to the traffic lights, see Fig. 1(c).
- The reflection of traffic lights and other light sources on walls and roads will cause false detections, see Fig. 1(d).

TLD performs as an indispensable part in autonomous driving. It is supposed to be reliable, effective, stable and working in real-time. As the distance decreases between cars and intersections, the size of traffic lights will change from small to large in a drivers view. That is to say, autonomous vehicles must be able to detect different sizes of traffic lights.

SqueezeDet [14] was originally designed to detect objects on the KITTI and Pascal Voc dataset. The algorithm is supposed to face objects of different classes and sizes. Therefore, it is necessary to consider roughly covering as many objects as possible when designing anchors. For the TLD task, target objects are limited to a fixed range in both width and height. According to our research, the size of traffic lights ranges from 20 pixels to 100 pixels in width and height, and we adjust the anchor size to cover the smallest, medium, and maximum traffic lights. In this way, the prior anchor boxes can cover the

(a) (b) (c) (d)

Fig. 1. Challenges in TLD at night. Fig. 1(a) and Fig. 1(b) show that traffic lights seem larger and are white in the image due to halo disturbances and color distortion. Fig. 1(c) is the car taillight which is one of the interference elements in TLD. Fig. 1(d) illustrates the reflected light source on roads.

*This work was supported by the National Natural Science Foundation of China(NO.61773312, 61790563).

[1] Y.Huo, Z.Xu, S.Chen, Y.Chen, Y.Huang, N.Zheng are with the Department of Electrical and Information Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, P.R. China

*Correspondence: chenshitao@stu.xjtu.edu.cn

978-1-7281-4091-9/19/$31.00 ©2019 IEEE

corresponding object bounding boxes well which contributes to an improvement in accuracy of the detection.

As we all know, the processing time of neural network is highly related to the size of input images. The lower the resolution of input images, the better the processing speed. We also recognized that the position of traffic lights in image will go upward when the autonomous vehicle move from far to near. According to the prior knowledge such as the position of traffic lights in the image, we can crop the image and feed the remaining part that contains the traffic lights as the input of the network during the training and detection process, which can significantly reduce the processing time by 40%.

Finally, based on the original detection results of SqueezeDet, we designed a series of rules for filtering false positive detections using spatial position and color texture features of traffic lights, which improve the detection accuracy and achieve better detection results than the native SqueezeDet. We have found that most of traffic lights are arranged regularly, with the distance between them in one intersection is almost same. We can filter out some of false positive detections which are far from traffic lights, such as road lights and traffic lights, for non-motor vehicles. And all traffic lights including the traffic lights countdown timers are installed at the same height. Therefore, the higher road lights and lower taillights are abandoned. A much better detecting performance has been achieved in this manner.

II. Related Work

A. CNNs for Object Detection

Ross Girshick et al. proposed the Faster RCNN [15] which evolved from Selective Search [16], RCNN [17] and Fast RCNN [18]. The input of the network is a whole image. And the author divided the algorithm into two stages. One is object bounding box generation by a small CNN named Region Proposal Network (RPN). Another one is object classification. Several region proposals are generated through RPN for each image, then the network classifies each object in every region proposal and gets the corresponding bounding box of the target object after filtering. YOLO [19] is another object detection CNN proposed by Ross Girshick to speed up detection. The entire image is also the input of this algorithm, but it is a single stage detection pipeline. Different from Faster RCNN, YOLO regresses the position and class of the object bounding box directly, and this will reduce the processing time compared to Faster RCNN. SSD [20] is an one-stage object detection method proposed by Wei Liu et al., the authors used small convolutional kernels in the network. At the same time, there are different aspect radio of filters predicting objects from feature maps in various network layers, which can cope with objects of different scales. Through experiments, the results showed that SSD is faster and more accurate than YOLO.

B. Traffic Light Detection Algorithms

Julian Muller et al. proposed an improved traffic light detection algorithm TL-SSD based on SSD. Its a tough task for the native SSD algorithm to deal with small object detection, while TL-SSD is capable of handling this problem. TL-SSD uses Inception-V3 [21] instead of VGG_16 [22] as the basic network of SSD to improve processing speed and detection accuracy. The prior box generation is adapted to detect small objects. And they also use an improved Non-Maximum Suppression (NMS) algorithm implemented for getting a better detection result. Weber et al. proposed a real-time method of DeepTLR based on OverFeat [23]. However, they abandoned the traffic lights bounding boxes with width less than 8 pixels, since the network was not able to detect them. Behrendt et al. proposed an algorithm based on YOLO to detect traffic lights. The algorithm showed the ability to detect small traffic lights, but additional CNN was needed to filter out false detection and classify traffic lights. Pon et al. proposed a deep neural network that uses the smallest GPU storage resources to achieve real-time traffic light detection, but it was short of the detection performance. The accuracy of their method is lower than that of Faster RCNN.

III. Proposed Method

This section introduces our algorithm in detail, which is a SqueezeDet-based TLD algorithm at night. The Fig.2 shows the pipeline of our method.

A. Basic Principle

SqueezeDet is a single-stage detection algorithm inspired by YOLO and SqueezeNet [24], which generates and classify the candidate region proposals in one neural network. Firstly, an image is fed into the CNN to obtain a low-resolution, high-dimensional feature map. Then the feature map is taken to the ConvDet layer defined by the author to calculate the corresponding candidate bounding boxes. Each candidate bounding box has $C+1$ parameters, C is the number of classes to be distinguished, and the remaining parameter is confidence score. High confidence score means that a prediction region contains a high probability of interested objects, while the coverage between the prediction box and the real box is better as well. In addition, these C parameters are the conditional probability distribution of objects in each category in the candidate region proposals. Then the class with the highest conditional probability is considered as the class of prediction bounding box.

Finally, the top N prediction boxes with the highest confidence score are remained, and the NMS is using to filter out the redundant prediction boxes for getting the final detection results.

B. Modified Anchors and Input Images

Anchor is the original region proposal generated by the SqueezeDet at the beginning. For every anchor in each position of the input images, the neural network regresses the anchor to the candidate region proposals through calculating the margin between them. Since the anchor has a great influence on the detection performance of the entire CNN, we should modify the anchor size to fit the shape of objects well.

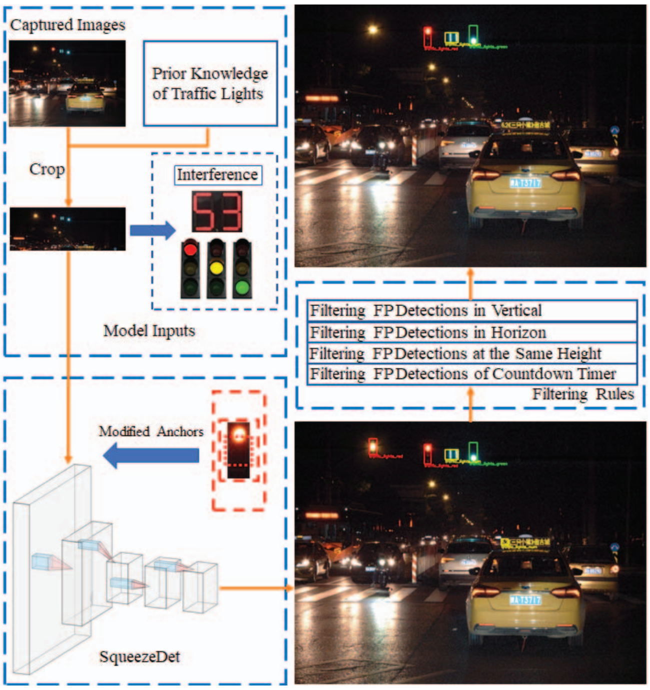


Fig. 2. Dection pipeline of our method. We utilize the prior knowledge of traffic lights to crop the images as the input of the network. What remain in the cropped images are mainly the traffic lights, traffic light countdown timers and less interference elements. Then we modify the anchors of the SqueezeDet to fit the size of traffic lights. Finally, filtering rules are used to filter out the false positive detections, thus obtaining more accurate detection results.

The anchors of native SqueezeDet is designed for the detection of pedestrians, vehicles and non-motor vehicles of different sizes in the KITTI dataset. Moreover, the size of target objects are different with various shape. In the contrast, the shape of traffic lights is regular in most instances. Traffic lights are mainly installed in vertical and horizontal directions, of which aspect ratio is generally around $1:2$ or $2:1$. Therefore, we can change the size of anchors and fit the traffic light bounding boxes better. The anchor size we set is $1/2, 1, 2$ times the width and height. After the combination, there are 9 kinds of anchors in the center of each grid on the input images. According to our experiments, we set the original anchor width to 27 pixels and height to 46 pixels.

On the other hand, since SqueezeDet uses the ConvDet layer for region proposal regression and classification, the entire neural network is a fully convolutional network. We can thus feed images of any resolution to the neural network because of the lack of fully connected layer. While for CNNs, the higher the resolution of input images, the greater the computational complexity of each convolutional layer, which will in turn lead to a prolonged processing time of the network. Real-time performance is an essential feature of TLD, and the processing speed must maintain a high level. In the TLD task, we have observed that the position of the traffic lights in the image changes according to the distance between the camera and the traffic lights. The closer the vehicle is to the intersection, the higher the position of traffic lights is in the image. Nowadays, we choose the upper part of the captured image as the input of the network, since we have found that almost all the traffic lights are appeared in the upper part of the images. If we crop the image captured by camera and get the proper part to feed into the SqueezeDet, the real-time performance of the entire algorithm can be improved. At the same time, we have found that removing the rest part of the image has less impact on the overall detection performance. Moreover, some of the vehicle taillights, ground reflection lights, pedestrian crossing lights and other interference light sources will appear in the lower half of the image. We can also filter out false positive detections and improve detection results in this way.

C. Filtering rules

As mentioned above, there are numerous interference elements in the TLD at night, such as road lights, taillights of cars, billboards and reflected lights from road surfaces. All these elements will cause the false detections of the CNN. On the one hand, a constant improvement in the detection performance of network is required. On the other hand, It is also needed to use the characteristics of traffic lights to design a series of rules to filter out false positive detection.

Generally speaking, the installation position of traffic lights at intersections is regular in Xi'an. For two or more traffic lights, they must be mounted horizontally at the same height and spaced about the same distance. When there is a traffic light countdown timer, the horizontal center line height of it will be approximately close to other traffic lights. Therefore, we develop the following rules of filtering out false positive detections in this manner.

- We use the coordinates of images defined in OpenCV and sort all the bounding boxes which are marked as traffic lights by the minimum value of y value in a queue. In this way, the false positive detection bounding boxes, which are not at the same height as the true traffic lights, can be arranged at the head or the end of the queue. We can use this queue to filter out several interference objects. The difference between the minimum values of the y value of each two adjacent prediction boxes is then calculated:

$$\Delta H_1 = y^i_{min} - y^{i+1}_{min}, i \in [1, bbox\_N - 1] \quad (1)$$

$$\Delta H_2 = y^{i-1}_{min} - y^i_{min}, i \in [1, bbox\_N - 1] \quad (2)$$

  The $bbox\_N$ in (1) and (2) is the sum of the total detection bounding boxes. We suppose that these two adjacent traffic lights are at the same height when ΔH_1 and ΔH_2 are both less than 10 pixels. When the height difference is more than 10 pixels, the i-th prediction box will be considered as a false positive detection and should be filtered out.
- On the basis of the first step, we calculate the difference between the minimum values of the x value of each two adjacent prediction boxes:

$$\Delta W_1 = x^j_{min} - x^{j+1}_{min}, i \in [1, bbox\_N - 1] \quad (3)$$

$$\Delta W_2 = x_{min}^{j-1} - x_{min}^{j}, i \in [1, bbox\_N - 1] \quad (4)$$

Since the traffic lights are arranged with the same distance in the horizontal direction, and the distance between two adjacent traffic lights is not too large, we compare the minimum value of x value with these two adjacent bounding boxes. If ΔW_1 and ΔW_2 is greater than 300 pixels, the i-th bounding box is a false positive detection.

- During the TLD at night, their will be disturbance from non-motor vehicles which are at the same height as traffic lights. Therefore We reorder the queue obtained from the first two steps through the minimum value of x of each prediction box. After sorting, the interference elements will always be arranged at the end of the queue. Considering that the difference between lights of non-motor vehicles and adjacent traffic lights will change from far to near, we use the ratio instead of the distance between two neighbouring lights to filter out the interference objects:

$$\varepsilon_{\Delta} = \frac{x_{min}^{k+1} - x_{min}^{k}}{x_{min}^{k} - x_{min}^{k-1}}, k \in [1, bbox\_N_{filtered} - 1] \quad (5)$$

where ε_{Δ} represents the ratio of the distance between each bounding box. In general, the distance between the light of non-motor vehicle and the nearby traffic light is usually larger than that of two adjacent traffic lights, so we set $\varepsilon_{\Delta} = 1.2$. The comparsion starts from the end of the queue, when $\varepsilon_{\Delta} > 1.2$, the light can be considered as a non-motor vehicle light.

- Finally, accurate predictions of traffic light bounding boxes can be obtained through the above steps. Next, the false positive detections which are classified in traffic light countdown timers should be handled. The horizontal center line of the traffic light countdown timer is approximately at the same height with traffic lights, while other interference objects such as billboards and bus number, are lower or higher. Thus this feature can be utilized to design the algorithm:

$$\Delta H_{horizon} = \left| \frac{y_{max}^{m} - y_{min}^{m} - H_{avg}}{2} + y_{min}^{m} - y_{min\_avg} \right| \quad m \in [0, bbox\_N_{TL\_time}] \quad (6)$$

$\Delta H_{horizon}$ represents the absolute value of the horizontal center line height difference between the traffic light countdown timer and the predicted box of the traffic light. $bbox\_N_{TL\_time}$ stands for the number of predicted bounding boxes for traffic lights and traffic light countdown timers. H_{avg} is the average height of all traffic lights, and y_{min_avg} is the average of the minimum value of y for each traffic light bounding box. It is clear that the current traffic light countdown timer is a false positive detection when $\Delta H_{horizon}$ is larger than 5 pixels.

Using the four steps mentioned above, we can filter out most of the false positive detections from the network. And the specific improvement of performance is shown in the next section.

IV. Experiments

In this section, we evaluate the proposed method in our dataset named Traffic Light at Night (TLN), and present a set of contrast experiments for the comparison between our method with two state-of-the-art object detection algorithms. Moreover, we have also further analyzed the influence of the processing step of our method.

A. TLN Dataset

According to our research, we found that there is none of public nighttime traffic lights dataset in China. The popular traffic light dataset such as the LISA and LaRA datasets are obtained in the foreign country, and LaRA only has the data in the daytime as well. Since our algorithm is focused on the TLD task in China, we use the images from the TLN dataset. The image of the dataset was captured by a gray point camera of GS3-U3-15S5C-C, using a CCD chip with the resolution of 1384*1032 and 20 FPS frame rate. We use a lens with a focal length of 12-36 mm and an aperture of 1-2.8. The scene of our dataset consists of urban roads in the Beilin District of Xi'an, Shaanxi Province, which contains dozens of traffic intersections and rich traffic light types. Our dataset has 4125 images in total, and we mark the traffic lights as three major classes, red, green and countdown timer, while the yellow traffic lights are also marked into the red class. In the red and green classes, there are four subcategories: straight, left, right, and circle. Therefore, there are a total of nine classes in the dataset. The ratio of each class is shown in table I.

B. Experimental Environment and Testing Standard

All Algorithms are programmed in Python with the same training set, validation set and test set as input. The experimental platforms are Ubuntu 16.04 LTS and NVIDIA GeForce RTX 2080Ti.

We compare the algorithms by Average Percision (AP), Mean Average Percision (mAP), speed and model size. As an important criteria to measure the object detection algorithm, mAP consists of precision, recall and average precision. Precision value is the ratio of the number of correctly predicted

TABLE I
TLN Dateset

Classes	Objects Number	Ratio(%)
traffic_lights_red_circle	2434	19.84
traffic_lights_red_left	1981	16,20
traffic_lights_red_right	356	2.89
traffic_lights_red_forward	0	0
traffic_lights_green_circle	3166	25.80
traffic_lights_green_left	571	4.65
traffic_lights_green_right	193	1.56
traffic_lights_green_forward	0	0
traffic_lights_red	4771	38.89
traffic_lights_green	3930	32.05
traffic_lights_num	3565	29.06
sum	12266	100

objects to the total number of objects in the ground truth, defined as follows:

$$Precision = \frac{TP}{TP+FP} \tag{7}$$

TP is the number of true positive objects. FP is the number of false positive objects. Recall value is the ratio of the number of correctly predicted objects to the total number of predicted objects, defined as follows:

$$Recall = \frac{TP}{TP+FN} \tag{8}$$

FN is the number of false negative objects.

The area under the Precision-Recall curve is the average precision value. In object detection experiments, each class corresponds to an specific AP value and the average of all AP values is the mAP value of the whole algorithm. We present the final results according to the original PASCAL overlap criteria [25], with IOU fixed to 50%.

C. Results

In comparison with one-stage detection algorithms such as YOLO_V3 and native SqueezeDet, our algorithm outperforms in terms of detection performance and processing speed in all classes. The detailed results are displayed in table II. It is clear that our proposed method outperforms in both YOLO_V3 and SqueezeDet algorithms, improving the detection performance. As for the detection results of different classes, the comparison of the algorithm with our method is illustrated in Fig. 3. The algorithm shows the best detection performance in green traffic lights, for the green color tends to have a more obvious color feature at night. The red traffic lights have a lower detection result due to many interference elements, and the worst result lies on the detection of traffic light countdown timers because of lack of training data and complex texture features. Anchors designed for KITTI dataset are still adopted by YOLO_V3 and SqueezeDet, which differ significantly in size with the traffic light bounding boxes, and it is not easy for the anchors to regress in the network. Therefore, the performance of the models trained by these two networks is not very good. While our method modifies the shape of anchors of SqueezeDet to suitable size, making the network to recognize the traffic light bounding boxes better. Moreover, since the original images are used directly as the network input for YOLO_V3 and SqueezeDet, it takes a longer time for feature map generation and loss function calculation. Table III shows that the processing speed will be improved through cropping the input images of the network. By contrast, we use the image properly cropped from the original image as the input, managing to not only filter out some interference elements, but also reduce the processing time of the network and improve the frame rate. Finally, after filtering out through our rules, we can get more accurate test results and improve the precision value of detection.

TABLE II
COMPARISON OF EXPERIMENTAL RESULTS FROM YOLO_V3, SQUEEZEDET AND OUR METHOD

Method	Classes	AP Per Class(%)	mAP(%)
YOLO_V3	traffic_lights_red	87.75	89.64
	traffic_lights_green	96.19	
	traffic_lights_num	84.97	
YOLO_V3 + Cropping	traffic_lights_red	89.56	91.30
	traffic_lights_green	97.23	
	traffic_lights_num	87.11	
YOLO_V3 + Filtering Rules	traffic_lights_red	96.66	95.16
	traffic_lights_green	98.87	
	traffic_lights_num	89.96	
SqueezeDet	traffic_lights_red	93.79	87.11
	traffic_lights_green	96.64	
	traffic_lights_num	70.91	
SqueezeDet + Cropping + Modified Anchors	traffic_lights_red	95.10	90.37
	traffic_lights_green	**99.73**	
	traffic_lights_num	76.27	
Proposed Method	traffic_lights_red	**99.57**	**97.62**
	traffic_lights_green	99.48	
	traffic_lights_num	**93.80**	

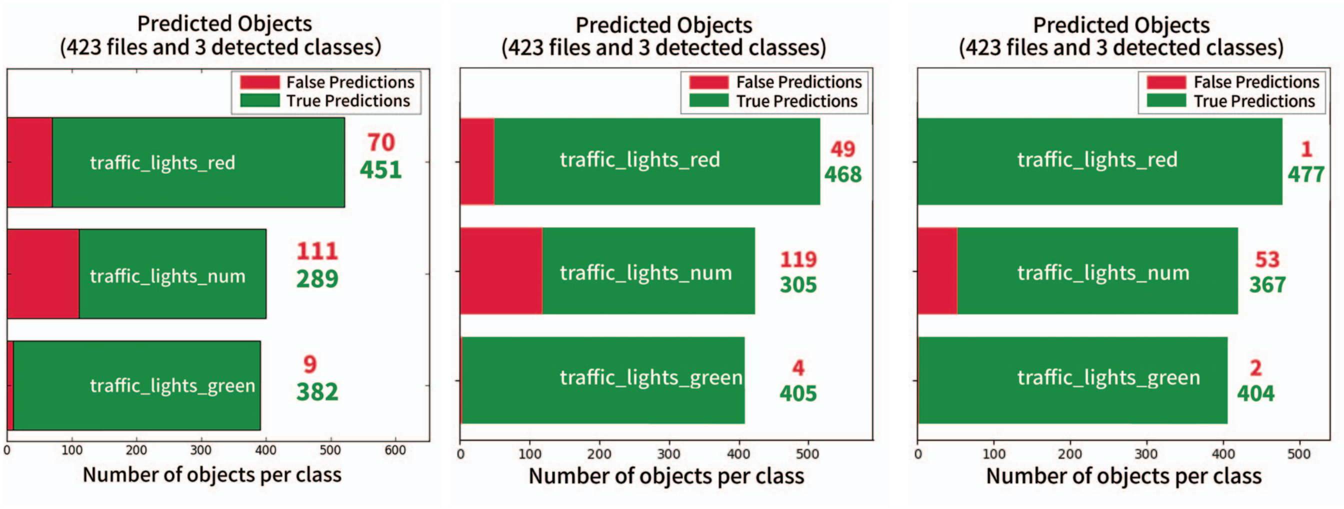
TABLE III
COMPARISON OF PROCESSING SPEED FROM DIFFERENT METHODS WITH CROPPING OR NOT

Methods	Frame Rate (FPS)
YOLO_V3	17.10
YOLO_V3 + Cropping	26.67
SqueezeDet	11.54
SqueezeDet + Cropping	23.45

Although YOLO_V3 is better at inference speed and performance than SqueezeDet, YOLO_V3 has a much larger parameter than SqueezeDet. The model size of YOLO_V3 is 246.3 MB, while the model size of SqueezeDet is only 18.3 MB, which is about $1/10$ of YOLO_V3. Therefore, SqueezeDet occupies less computing resources on the same computing platform than YOLO_V3. Because of autonomous vehicles have limited computing resources, the algorithms carried on them must meet the requirements of occupying less computing resources. Therefore, our proposed method uses SqueezeDet instead of YOLO_V3 as the detection algorithm, and also achieves outstanding TLD performance.

V. CONCLUSION

In this paper, we propose an improved algorithm for SqueezeDet-based TLD task at night. Efforts have been made on the native network through adjusting the anchor size to make it more suitable for the shape of the traffic light bounding box, so that the network can regress the bounding box better. In addition, we also crop the upper half of the image as the input of the network, which reduces the computation and improves the processing speed of the network. Finally, we design a series of filtering rules for false positive detections of traffic lights. A series of experiments have been performed on our dataset to confirm that our proposed method is capable of accelerating the speed of TLD and leads to an overall improvement in



(a) SqueezeDet

(b) SqueezeDet with Modified Anchor Size and Cropped Images

(c) Proposed Method

Fig. 3. Detection results of Proposed method on TLN dataset.

the detection performance. Meanwhile, the algorithm is also efficient for autonomous vehicles. As for the future work, we are supposed to utilize the High-Definition (HD) map, GPS information and parameters of camera to crop the captured image. Moreover, our algorithm can only detect the traffic lights with red, yellow and green. The detailed information such as direction and the number of countdown timer is not obtained. Therefore, we should improve our method to behave better in the future.

References

[1] W. Pananurak, S. Thanok and M. Parnichkun, ”Adaptive cruise control for an intelligent vehicle,” 2008 IEEE International Conference on Robotics and Biomimetics, Bangkok, 2009, pp. 1794-1799.

[2] D. M. Filatov and E. V. Serykh, ”Intellegence autonomous parking control system of four-wheeled vehicle,” 2016 XIX IEEE International Conference on Soft Computing and Measurements (SCM), St. Petersburg, 2016, pp. 154-156.

[3] M. Ariyanto, G. D. Haryadi, M. Munadi, R. Ismail and Z. Hendra, ”Development of Low-Cost Autonomous Emergency Braking System (AEBS) for an Electric Car,” 2018 5th International Conference on Electric Vehicular Technology (ICEVT), Surakarta, Indonesia, 2018, pp. 167-171.

[4] F. Yang, S. Wang, J. Li, Z. Liu and Q. Sun, ”An overview of Internet of Vehicles,” in China Communications, vol. 11, no. 10, pp. 1-15, Oct. 2014.

[5] J. Zhao, Y. Chen and Y. Gong, ”Study of Connectivity Probability of Vehicle-to-Vehicle and Vehicle-to-Infrastructure Communication Systems,” 2016 IEEE 83rd Vehicular Technology Conference (VTC Spring), Nanjing, 2016, pp. 1-4.

[6] J. Mller and K. Dietmayer, ”Detecting Traffic Lights by Single Shot Detection,” 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, 2018, pp. 266-273.

[7] M. Weber, P. Wolf and J. M. Zllner, ”DeepTLR: A single deep convolutional network for detection and classification of traffic lights,” 2016 IEEE Intelligent Vehicles Symposium (IV), Gothenburg, 2016, pp. 342-348.

[8] K. Behrendt, L. Novak and R. Botros, ”A deep learning approach to traffic lights: Detection, tracking, and classification,” 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 2017, pp. 1370-1377.

[9] A. Pon, O. Adrienko, A. Harakeh and S. L. Waslander, ”A Hierarchical Deep Architecture and Mini-batch Selection Method for Joint Traffic Sign and Light Detection,” 2018 15th Conference on Computer and Robot Vision (CRV), Toronto, ON, 2018, pp. 102-109.

[10] X. Du, Y. Li, Y. Guo and H. Xiong, ”Vision-Based Traffic Light Detection for Intelligent Vehicles,” 2017 4th International Conference on Information Science and Control Engineering (ICISCE), Changsha, 2017, pp. 1323-1326.

[11] R. Kulkarni, S. Dhavalikar and S. Bangar, ”Traffic Light Detection and Recognition for Self Driving Cars Using Deep Learning,” 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 2018, pp. 1-4.

[12] W. Zong and Q. Chen, ”Traffic Light Detection Based on Multi-feature Segmentation and Online Selecting Scheme,” 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), San Diego, CA, 2014, pp. 204-209.

[13] V. John, K. Yoneda, Z. Liu and S. Mita, ”Saliency Map Generation by the Convolutional Neural Network for Real-Time Traffic Light Detection Using Template Matching,” in IEEE Transactions on Computational Imaging, vol. 1, no. 3, pp. 159-173, Sept. 2015.

[14] B. Wu, F. Iandola, P. H. Jin and K. Keutzer, ”SqueezeDet: Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving,” 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, 2017, pp. 446-454.

[15] S. Ren, K. He, R. Girshick and J. Sun, ”Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017.

[16] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. IJCV, 2013.

[17] R. Girshick, J. Donahue, T. Darrell and J. Malik, ”Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,” 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580-587.

[18] R. Girshick, ”Fast R-CNN”, Proc. IEEE Int. Conf. Comput. Vis., pp. 1440-1448, 2015.

[19] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, ”You Only Look Once: Unified, Real-Time Object Detection,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 779-788.

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, A. C. Berg, ”SSD: single shot multibox detector”, CoRR, vol. abs/1512.02325, 2015.

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, ”Rethinking

the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 2818-2826.
[22] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2014, [online] Available:.
[23] P. Sermanet, D. Eigen, X. Zhange, M. Mathieu, R. Fergus, Y. LeCun, "OverFeat: Integrated Recognition Localization and Detection using Convolutional Networks", CoRR, 2013.
[24] Forrest N. Iandola et al., "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and 0.5 mb model size", 2016.
[25] M. Everingham, L.V. Gool, C.K.I. Williams, J. Winn, A. Zisserman, "The pascal visual object classes (voc) challenge", IJCV, 2010.