IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 21, 2020, accepted June 5, 2020, date of publication June 17, 2020, date of current version June 29, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3003229

An Approach Focusing on the Convolutional Layer Characteristics of the VGG Network for Vehicle Tracking

DANLU ZHANG, JINGGUO LV, AND ZHE CHENG
Beijing University of Civil Engineering and Architecture, Beijing 102616, China
Corresponding author: Jingguo Lv (lvjingguo@bucea.edu.cn)

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 41871367.

ABSTRACT Object tracking is a key technology in the field of intelligent transportation. To solve the partial occlusion problem in vehicle tracking, this paper analyzes the characteristics of a VGG convolutional neural network by experimental observation and describes its characteristics: (1) feature maps can be used for positioning, but they have redundancies, and (2) different layers of feature maps have different characteristics. After these characteristics are applied to vehicle tracking, a vehicle tracking algorithm is designed. For a given target vehicle, feature maps are generated on convolutional layers conv4_4 and conv5_4 of the VGG network, and the feature maps most relevant to the target vehicle are selected. These feature maps are used to capture target vehicle information and distinguish the target vehicle from backgrounds with similar appearances. The experiments use vehicle data from the LaSOT, VOT2017 and OTB2015 datasets to compare the vehicle tracking results of our proposed algorithm with those of other algorithms. The results show that the method proposed in this paper has certain advantages. According to algorithm implementation and vehicle tracking experiments, the proposed vehicle tracking method can solve the drift problem and is better than the traditional method at addressing drift problems.

INDEX TERMS Vehicle tracking, convolutional neural network, VGG, deep learning.

I. INTRODUCTION

At present, the ever-expanding scale of cities and rapid economic expansion have led to explosive growth in the number of cars. Therefore, the intelligent management of urban traffic must be carried out [1]–[4]. For a city's traffic problems, the current driverless technology and human-computer interaction technology continue to play an important role in the transportation field [5]. Object tracking technology is also evolving. Object tracking has always been a research focus and challenge in the field of computer vision and has been widely studied by researchers around the world. It is performed to estimate the position, shape and range of a tracked object in a continuous sequence of video images, thereby yielding relevant information such as the speed, direction and trajectory of the object to understand and analyze the moving object, resulting in better service. However, there are still a number of difficulties in tracking vehicles, such as moving vehicle feature variability, vehicle scale changes, illumination intensity inconsistencies, occlusion, and interference from complex backgrounds. There are still serious constraints on the performance and speed of vehicle tracking algorithms [6], [7]. Therefore, it is necessary to design a robust vehicle tracking algorithm during the process of building intelligent transportation.

The development of target tracking divided in two stages: in the first stage, the traditional object tracking algorithm is developed, and in the second stage, deep learning is used to solve the object tracking problem [8].

Early object tracking was performed using traditional algorithms such as the Kalman filtering [9], [10], particle filtering [11], [12], or mean-shift algorithm [13], [14], and optical flow methods [15]–[17]. Then, the rise of correlation filtering promoted the development of object tracking methods, and relevant filtering methods began to be applied to vehicle tracking, including the CSK (circulant structure kernel) filtering [18], KCF (kernelized correlation filtering) [19] and DSST (discriminative scale space tracking) algorithms [20].

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Yang.

The rise of deep learning methods has greatly improved the performance and efficiency of object tracking compared to the traditional algorithms. Traditional algorithms use artificial features to describe the object, while deep learning trains the network model on a large number of samples and obtains a CNN (convolutional neural network) for feature extraction. The expression of CNN features is stronger and its effect is better than the expression of artificial features. At present, many scholars use deep learning methods to propose different solutions to the problems in tracking. Aiming at the problem of foreground and background discrimination in tracking, [21], [22] used multi-area networks to distinguish targets from background, and [23] proposed a spatial attention map to suppress background interference during tracking. Aiming at the occlusion problem in the tracking process, [24], [25] used the convolution feature to locate the tracking target, and [26] locates the target by fusing different convolution features. Aiming at the problem of insufficient online training samples, [27] proposed the DLT (deep learning tracker) algorithm to reduce the need for training samples, and [28] proposed the SO-DLT (structured output DLT) algorithm to further solve the rotation problem. Aiming at the rotation problem in the tracking process, the tracking algorithms proposed in [29], [30] achieved better tracking results than previous methods. Aiming at the problem of slow tracking speed in the process of object tracking, [31] reduced the feature extraction part to reduce the number of dimensions, which improved the efficiency of the algorithm. Aiming at accounting for change in the appearance of the target, [32] proposed a dual model learning method combining multiple feature selection to solve the problem that deep convolutional features are not sensitive to changes in the appearance of the target.

Deep learning methods achieve better effects in dealing with occlusion, background clutter and real-time problems than traditional learning methods, but further improvements are needed in the process of deformation and in the positioning accuracy in the tracking process. This paper draws on the idea of the DLT algorithm, according to the characteristics of the vehicle, by analyzing the characteristics of the pretrained VGG-19 network, the characteristics of different layers of the convolutional layer are fully utilized, and the vehicle tracking algorithm is designed to deal with target deformation and occlusion. It achieves a better effect than the traditional methods, and effectively prevents tracking target drift and improves the tracking accuracy.

The vehicle tracking algorithm designed in this paper first inputs the first frame of video into a pretrained VGG-19 network and generates feature maps on the conv4_4 and conv5_4 layers. For the feature maps generated by the conv4_4 and conv5_4 layers, the constructed feature map selection sl-CNN is used to select the feature maps most relevant to the target vehicle and remove redundant feature maps. The constructed UnNet (universal network) is used to capture the target vehicle information from the feature maps generated at the conv5_4 layer; the constructed SpNet (specific network) is used to distinguish the target vehicle from vehicles with similar appearances from the feature maps generated at the conv4_4 layer. In the new frame of the video, the ROI (region of interest) centered on the target position of the previous frame is cut forward through the UnNet and SpNet networks, and then, foreground heat maps are generated. From the two foreground heat maps generated, interference detection is used to determine the position of the final target vehicle.

The contributions of this article are as follows:

(1) The feature maps of the pretrained VGG-19 network are analyzed, and their characteristics are obtained; the convolutional layer feature maps can be used for vehicle positioning, but redundancy exists. The high-level convolutional layer conv5_4 can acquire high-level semantic features to distinguish the target vehicle from the background, and the low-level convolutional layer conv4_4 can obtain local features that distinguish between a target vehicle and a vehicle with a similar appearance.

(2) According to the network characteristics of the VGG-19 network, the vehicle tracking algorithm is designed, which fully considers the characteristics of different convolutional layers and solves the problems of partial occlusion and target drift tracking.

(3) In the vehicle tracking algorithm, an sl-CNN model is designed for feature map selection, and the UnNet and SpNet are designed to generate foreground heat maps of the vehicle.

(4) The algorithm is tested with multiple datasets and more reliable conclusions are obtained than with previous algorithms. The vehicle tracking results are tested using multiple datasets, such as the LaSOT (Large-scale Single Object Tracking), VOT2017 (2017 Visual Object Tracking), and OTB2015 (2015 Object Tracking Benchmark) datasets, and various algorithms, such as the DLT, MOSSE (minimum output sum of squared error) filter, DCF (discriminative correlation filter), KCF, MEEM (multi-expert entropy minimization), Struck (structured output tracking with kernels), CSK, LSK (local steering kernel), Staple (sum of template and pixel-wise learner), DSST, and TLD (tracking-learning-detection) algorithms, are compared. The experiments show that the method in this paper is advanced.

II. VGG NETWORK FEATURE ANALYSIS

To deeply understand the characteristics of CNNs and apply them to the task of vehicle object tracking, the feature maps of the VGG network were analyzed in this study.

The feature analysis method of this paper is based on the application of the pretrained VGG-19 network on the ImageNet [33] image classification task. VGG-19 [34] is a 19-layer CNN. Its network structure is shown in Figure 1. The network structure consists of 16 convolutional layers and 3 fully connected layers. After several experiments, we find that the convolved layers conv4_4 and conv5_4 have certain characteristics. We believe that these two layers are the basis of vehicle tracking algorithm design, so our feature analysis method mainly focuses on these two layers

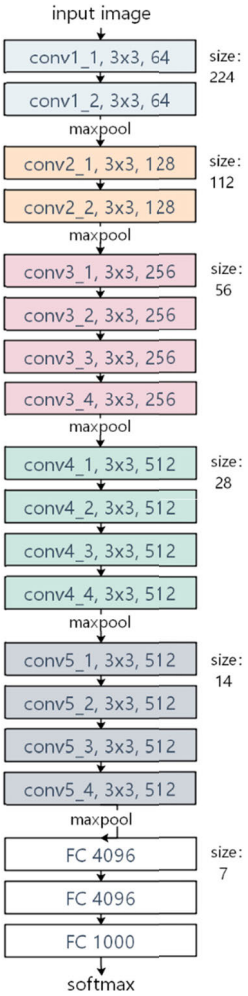

FIGURE 1. VGG-19 network structure.

(see 5.2 for a discussion of the choice of these two layers). The conv4_4 convolutional layer is the twelfth convolutional layer, the conv5_4 convolutional layer is the sixteenth convolutional layer, and both convolutional layers generate 512 feature maps.

A. FEATURE MAPS CAN BE USED FOR POSITIONING, BUT REDUNDANCY EXISTS

The feature maps of the VGG network can be used for the positioning of vehicle objects, but the feature maps are redundant. In the CNN, different convolutional layers can extract different levels of input image feature information, and low-level extracted features often contain details of the target in the image, which can be used to capture local features of the target, while the high-level feature further describes the semantic information of the image. An in-depth analysis of the feature maps shows that they can be used for the positioning of vehicle objects. Both the conv4_4 and conv5_4 convolutional layers are obtained by multiple convolutions, and their receptive fields are large. Figure 2 shows the feature maps of the conv4_4 and conv5_4 convolutional

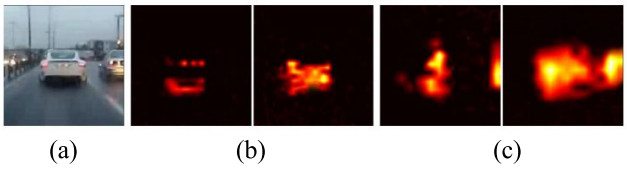

FIGURE 2. Feature maps of the VGG-19 network.

layers obtained by VGG-19, which have maximum activation values in the object region, where (a) is the original image of the input VGG network and (b) presents the feature maps of the conv4_4 layer, which are activated in the target area and are distinguished from the background; (c) are feature maps of the conv5_4 layer, which are activated in the target area and can capture target and interference information. In the figure, the activated portion of the feature maps is mainly located in the foreground object area and can be used to locate the target vehicle. Therefore, the VGG feature map can be used for the positioning of vehicle targets.

Both the conv4_4 and conv5_4 convolutional layers generate 512 feature maps, but not all feature maps can be used for the positioning of the target vehicle, as most contain noise and cannot be used for target positioning. In Figure 3, (a) is the input image, (b) is the ground-truth foreground mask, (c) present the feature maps of the conv4_4 layer, and (d) present the feature maps of the conv5_4 layer. As shown in Figure 3, the average feature map contains background noise. Therefore, feature maps responsive in both the vehicle area and the background area must be removed to prevent the tracker from identifying the background area of the image.

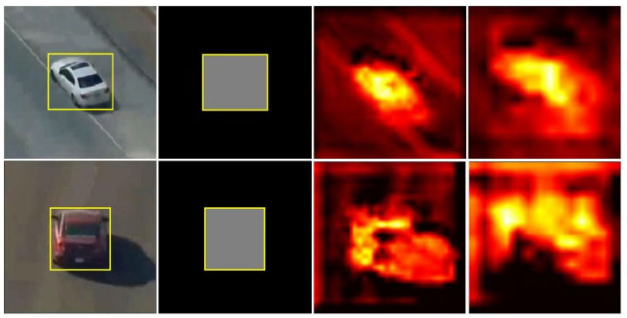

FIGURE 3. Feature maps of mixed noise.

The activation value of the feature map is the sum of all response values in the target vehicle area. Figure 4 shows pie charts of the proportion of the number of different activation value feature maps in the target vehicle area. The left side is for the conv4_4, convolutional layer, and the right side is for the conv5_4 convolutional area. As can be seen from the figure, the activation value of most feature maps in the object area is small. Therefore, the feature maps generated by VGG-19 are redundant and have little correlation with the target vehicle. Therefore, when implementing the vehicle object tracking task, a large number of redundant feature

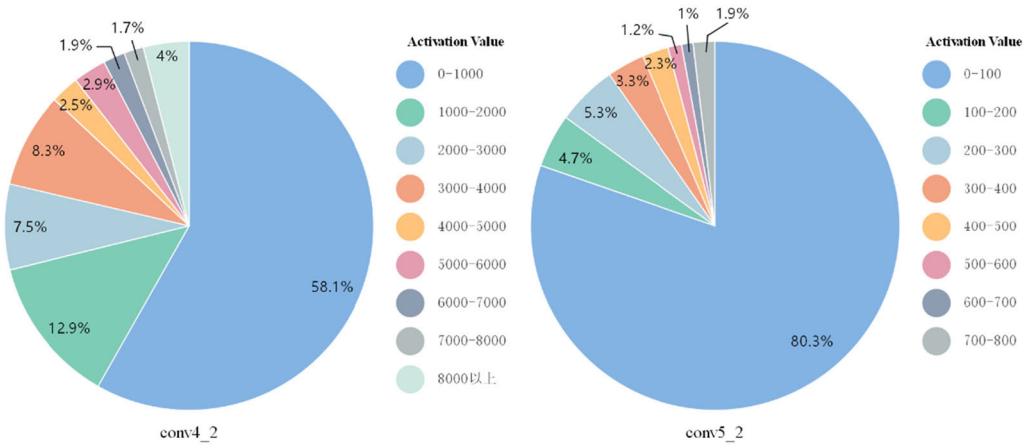

FIGURE 4. Pie chart of the quantity ratio of feature maps for different activation values.

maps should be removed, and a small number of effective feature maps should be selected to improve the performance of the tracker. The removal of redundant feature maps is described in detail in Section III-A.

B. DIFFERENT LAYER FEATURE MAPS HAVE DIFFERENT CHARACTERISTICS

Since the feature maps are affected by noise, they suffer from redundancy. In the experiment, the characteristics of the feature maps of the different layers are observed. The feature maps can be denoted by $\in \mathbb{R}^{d\times n}$, where d represents the dimension of each feature map vector, and n represents the number of feature maps. The image is associated with a foreground mask. The foreground mask can be represented as $\pi \in \mathbb{R}^{d\times 1}$, where, if the i-th neuron of each feature map is not located in the foreground object, $\pi_i = 0$; otherwise, $\pi_i = 1$. The foreground mask [35], [36] of the feature maps is reconstructed by solving equation (1):

$$\min_{c} \|\pi - Fc\|_2^2 + \lambda \|c\|_1, \quad s.t. \succcurlyeq 0 \tag{1}$$

where λ is a parameter used to balance the sparsity and the reconstruction error, and $c \in \mathbb{R}^{n\times 1}$ is the sparse coefficient.

The input vehicle image is forwardly propagated through the VGG-19 network, and the feature maps of the conv4_4 and conv5_4 convolutional layers can be obtained. For the conv5_4 convolutional layer, the feature maps obtained in the experiment are used to reconstruct the foreground mask by the above formula. Figure 5 shows the foreground masks of the feature maps of the conv5_4 convolutional layer, where the first two lines are the images of the buses and their foreground masks, and the last two lines are the images of the cars and their foreground masks. As can be seen from the first two rows of Figure 5, the selected feature maps can better locate the vehicle than the other feature maps. Even when the appearance and model of the vehicle change, the positioning effect is relatively stable, and positioning failures rarely occur. The latter two lines show

FIGURE 5. Foreground mask reconstructed by the feature maps of the conv5_4 layer.

that the selected feature maps separate the vehicle from the background and are not affected by vehicle rotation. The conv5_4 layer captures the high-level semantic features of the object but has insufficient discriminating power for different vehicles and cannot be directly used for vehicle positioning. In Figure 2, conv4_4 can separate the target vehicle from other vehicles, indicating that conv4_4 is more sensitive to different vehicle changes than conv5_4, but the robustness is relatively low.

Following the qualitative analysis above, a quantitative analysis is performed to illustrate this point. The experimental data are vehicle images selected for the vehicle dataset: 2,400 vehicle images belonging to eight different models and 2,500 images of non-vehicles. The first experiment uses the

feature maps of conv4_4 and conv5_4 to test the accuracy of classifying images into as either vehicles or non-vehicles. In the training phase, the positive training sample is four images belonging to four vehicles, and the sparse coefficient is calculated by equation (1). In the test phase, the reconstruction error of the foreground image is calculated according to the feature maps and the foreground mask of the input image, as in equation (2):

$$e = \min_i \|\pi - Fc_i\|_2^2 \tag{2}$$

If the error is less than a given threshold, the image is of a vehicle, and if the error greater than the given threshold, the image is of a non-vehicle. The second experiment divides all the vehicle images into different models. In the training phase, the training sample includes 20 images per vehicle, and the sparse coefficient is calculated by equation (1). In the test phase, the foreground mask reconstruction error of each model vehicle is calculated first, and then the image with the smallest error is classified into the corresponding vehicle model, as shown in equation (3):

$$id_{car} = \arg\min_i \|\pi - Fc_i\|_2^2 \tag{3}$$

The classification results of the two experiments are shown in Table 1. As can be seen from the table, in Experiment 1, for the results of classifying images as containing vehicles or non-vehicles, the overall classification accuracy obtained by the conv5_4 layer is higher than that of the conv4_4 layer, indicating that the feature map of the conv5_4 layer can better separate vehicles from non-vehicle objects. It can be concluded that the high-level semantic information obtained from the feature map of the conv5_4 layer can better distinguish vehicles from the background. In Experiment 2, from the results of classifying pictures into different models of vehicles, the overall classification accuracy obtained by the conv4_4 layer is higher than that of the conv5_4 layer, indicating that the feature map of the conv5_4 layer can better classify different types of vehicles. It can be concluded that the intermediate information obtained from the feature map of the conv4_4 layer can better distinguish between different types of vehicles. Therefore, when designing the vehicle tracking algorithm, the characteristics of the conv4_4 and the conv5_4 layers are fully considered. Then, the conv5_4 layer is selected to capture the target vehicle information, and the conv4_4 layer is selected to distinguish the target vehicle from a vehicle with a similar appearance. These rules are applied to improve the vehicle tracking accuracy.

TABLE 1. Vehicle classification accuracy using the different feature maps.

Feature Maps	conv4_4 Layer	conv5_4 Layer
Experiment 1	78.36%	87.89%
Experiment 2	85.68%	59.54%

III. VEHICLE TRACKING ALGORITHM BASED ON THE VGG NETWORK

The vehicle tracking algorithm is designed according to the characteristics of the VGG-19 network.

A. SELECTION OF THE FEATURE MAPS

According to the feature analysis from Section 2, it can be seen that the feature maps have redundancy, so the feature maps should first be filtered to remove the noisy feature maps. For the selection of the feature maps, an s1-CNN, a CNN model based on a target heat map, is used in the study. This method is performed separately on the conv4_4 and conv5_4 convolutional layers of the VGG-19 network. The sl-CNN consists of convolutional layers and a random dropout layer. It does not contain nonlinear transformations. The sl-CNN selects feature maps generated by the conv4_4 layer or the conv5_4 layer as an input image and then generates the target vehicle heat map M. The target vehicle heat map M is a two-dimensional Gaussian graph centered on the true position of the target vehicle, and its variance is proportional to the size of the target vehicle. After inputting the feature map to the sl-CNN, a dropout layer is first used to prevent overfitting, and then a convolutional layer with a 3×3 convolution kernel is used to obtain the output. The training of the sl-CNN is performed by minimizing the squared loss between the foreground vehicle heat map $\widehat{M}$ and the target vehicle heat map M. The loss function is:

$$L_{sl} = \left\|\widehat{M} - M\right\|^2 \tag{4}$$

The model parameters are obtained by back-propagation of the sl-CNN. The feature maps are then selected based on the effect of the model parameters on the loss function L_{sl}. By converting the input feature maps into $vec(F)$ and f_i as the i-th element of $vec(F)$, the change in the loss function can be calculated by the following second-order Taylor expansion:

$$\delta L_{sl} = \sum_i g_i \delta f_i + \frac{1}{2}\sum_i h_{ii}(\delta f_i)^2 + \frac{1}{2}\sum_{i \neq j} h_{ij}\delta f_i \delta f_j \tag{5}$$

where $g_i = \frac{\partial L_{sl}}{\partial f_i}$ is the first derivative, and $h_{ij} = \frac{\partial^2 L_{sl}}{\partial f_i \partial f_j}$ is the second derivative.

Since the number of elements in the feature maps is very large, the complexity of calculating all second derivatives h_{ij} is too large and time consuming. According to the Hessian matrix, the third term on the right side of equation (5) can be ignored. The first derivative g_i and the second derivative h_{ii} can be calculated by backpropagation. After setting f_i to zero, we can obtain:

$$\delta f_i = 0 - f_i \tag{6}$$

The importance of element f_i, which is denoted by s_i, is defined as the change in the objective function. According to equations (5) and (6), s_i can be calculated by:

$$s_i = -g_i f_i + \frac{1}{2} h_{ii} f_i^2 \tag{7}$$

The importance of the n-th feature map can be further defined as the sum of the importance of all its elements, namely:

$$S_n = \sum\nolimits_{x,y} s\,(x, y, n) \tag{8}$$

where $s(x, y, n)$ is the importance of the index of the element whose position is (x, y) in the n-th feature map.

All feature maps are sorted in descending order according to their importance, and then the top N feature maps are selected for vehicle target positioning. The N feature maps have the greatest influence on the objective function, so they are more associated with the positioning of the target vehicle than the other feature maps. In addition, in the experiment, feature map selection is performed only on the first image frame for object tracking.

B. CONSTRUCTION OF UNNET AND SPNET

After the redundant feature maps are removed, the feature maps most relevant to the target vehicle are obtained, that is, those feature maps on which the vehicle is positioned. Before positioning the target vehicle, one needs to build two networks: UnNet and SpNet. UnNet is a universal network built on the feature maps of the selected conv5_4 layer to capture information on the target vehicle; SpNet is a specific network built on the feature maps of the selected conv4_4 layer to distinguish the target vehicle from vehicles with similar appearances. Compared to the simple structure of the sl-CNN, the structures of SpNet and UnNet are more complicated, which helps make the tracking more accurate. SpNet and UnNet have the same architecture; both contain two additional convolutional layers and use rectified linear unit (ReLU) as the nonlinear activation function. The convolution kernel of the first convolutional layer has a size of 9×9, the padding is 4, the parameter initialization method is Gaussian filtering, the bias initialization method is continuous, and 36 feature maps are the output; the convolution kernel of the second convolutional layer has a size of 5×5, the padding is 2, the parameter initialization method is Gaussian filtering, the bias initialization method is continuous, and the foreground vehicle heat map is the output. Both networks have a learning rate of 0.1e-9 and a momentum of 0.6. The difference between the two networks is that the weight decay is set differently: the weight decay of UnNet is 0.1, while the weight decay of SpNet is 0.0005. These two networks need to be initialized in the first frame of the video sequence. They are initialized by minimizing the loss functions. The loss functions are as follows:

$$L = L_{Sp} + L_{Un} \tag{9}$$

$$L_T = \left\| \widehat{M}_T - M \right\|_F^2 + \beta \left\| W_T \right\|_F^2 \tag{10}$$

where $T \in \{Sp, Un\}$ represents the collection of SpNet and UnNet; $\widehat{M}_T$ represents the predicted foreground heat map; M is the target vehicle heat map; β is the tradeoff parameter of the weight attenuation; W_T is the weight parameter of the convolution layer.

C. VEHICLE POSITIONING

After the network is built, the two networks need to be used to predict the location of the target vehicle. First, in the new frame of the video sequence, a rectangular ROI centered at the target position in the previous frame is cropped. The foreground heat map of the target vehicle is then predicted by forwardly propagating the ROIs. First, the target vehicle is located on the heat map generated by UnNet. (a) The target vehicle position is expressed as:

$$\widehat{X} = (x, y, \sigma) \tag{11}$$

where x and y represent the center coordinates and σ represents the scale of the target bounding box. (b) The target position in the previous frame is represented by $\widehat{X}^{t-1}$. (c) The position of the candidate target vehicle in the current frame is assumed to be affected by the Gaussian distribution:

$$p\left(X^t \mid \widehat{X}^{t-1}\right) = N\left(X^t; \widehat{X}^{t-1}, \Sigma\right) \tag{12}$$

where Σ represents the variance in the positional parameters. (d) The confidence of the i-th candidate target vehicle is calculated as the sum of all the values of the heat map within the candidate region as follows:

$$conf_i = \sum\nolimits_{j \in R_i} \widehat{M}_{Un}(j) \tag{13}$$

where j represents a coordinate index; R_i is an area of the i-th target candidate according to its position parameter X_i^t; $\widehat{M}_{Un}$ represents the heat map generated by UnNet. (e) The candidate vehicle target with the highest confidence is chosen by UnNet.

Because UnNet is built on the conv5_4 layer, it captures the semantic features of vehicles and is not sensitive to changes in different vehicles. Therefore, the foreground heat map generated by UnNet will highlight the target vehicle and vehicles with similar appearances. In other words, the position of the target vehicle identified by UnNet may be the position of a vehicle with a similar appearance. To prevent trackers from tracking similar vehicles, interference detection methods are needed to determine the final target position. The target position predicted by UnNet is represented by $\widehat{X}_{Un}$, and the corresponding target area in the heat map is R_{Un}. Evaluate the probability of interference from the ratio of confidence values outside and inside the target area:

$$P_d = \frac{\sum_{j \in \widehat{M}_{Un} - R_{Un}} \widehat{M}_{Un}(j)}{\sum_{k \in R_{Un}} \widehat{M}_{Un}(k)} \tag{14}$$

where $\widehat{M}_{Un} - R_{Un}$ represents the background area on the heat map $\widehat{M}_{Un}$. When P_d is less than the set threshold, no common interference is considered to have occurred, and the target position predicted by UnNet is used as the final positioning result; otherwise, the heat map $\widehat{M}_{Sp}$ predicted by SpNet is used for the above target positioning step, and the result is the final target position.

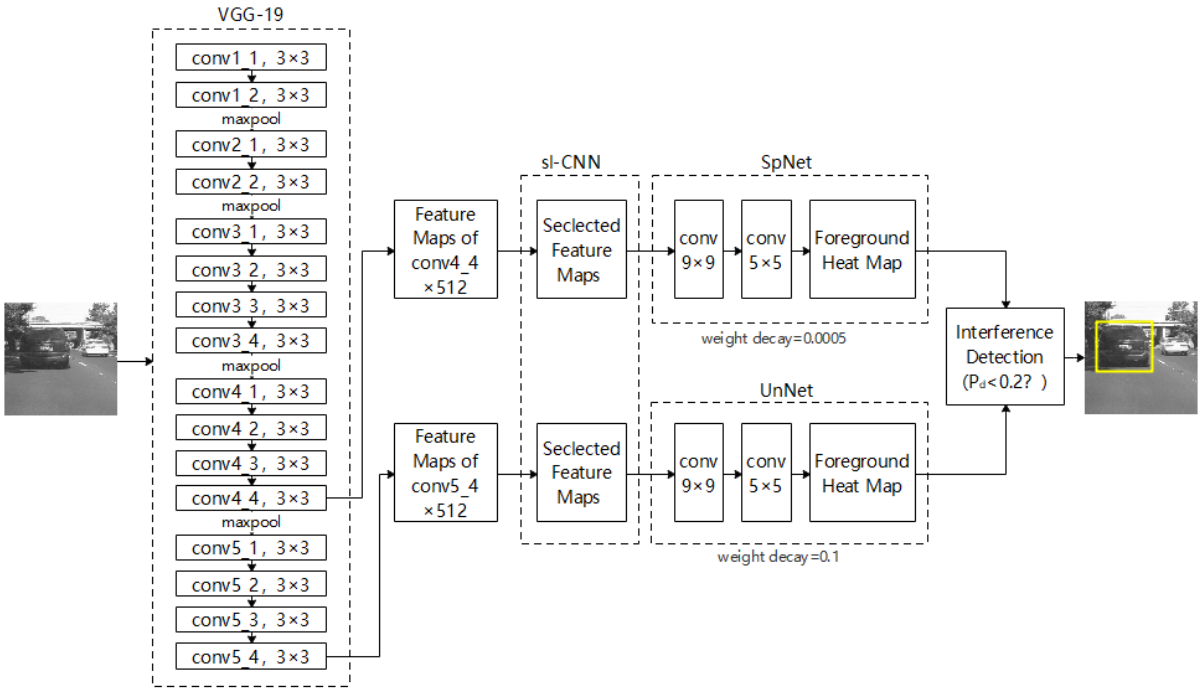

FIGURE 6. Algorithm flowchart.

D. ONLINE UPDATE OF THE VEHICLE TRACKING RESULTS

Online updates are primarily input into SpNet, allowing SpNet to adapt to changes in the target's appearance and to improve its identification of the foreground and background. In the experiment, SpNet is updated only after the first frame is initialized according to two different types of rules: (1) adaptive rules and (2) discriminative rules. According to the first type of rule, SpNet is fine-tuned every 15 frames using the tracking result with the highest confidence in the intermediate frames. According to the second type of rule, when the interference term is detected using equation (15), SpNet is further updated by minimizing the tracking results of the first frame and the current frame:

$$min\beta \left\| W_{Sp} \right\|_F^2 + \sum_{x,y} \left\{ \left[\widehat{M}_{Sp}^1 (x,y) - M^1 (x,y) \right]^2 + \left[1 - \Phi^t(x,y) \right] \left[\widehat{M}_{Sp}^t (x,y) - M^t (x,y) \right]^2 \right\} \quad (15)$$

where W_{Sp} represents the convolution weight of SpNet;$\widehat{M}_{Sp}^t$ represents the heat map of the t-th frame predicted by SpNet; (x, y) are the spatial coordinates. The foreground mask Φ^t represents the predicted target bounding box: if position (x, y) belongs to the target area, then $\Phi^t (x, y) = 1$; otherwise, $\Phi^t (x, y) = 0$. M^t represents the heat map generated according to the predicted target position.

The second term in the above equation represents the loss of the positioning target vehicle in the first frame. The estimated target vehicle area becomes unreliable when the appearance of the target vehicle is learned when an interfering vehicle appears or the target vehicle is severely occluded in the current frame. Therefore, the update is supervised by adding the first frame so that the deep learning model still learns the appearance of the target vehicle in the first frame. The third term in the above equation considers only the loss in the background region in the current frame and eliminates the loss in the unreliable target region so that this model will better identify the location where the interference occurs as the background. An advantage of the combination of the second and third terms in equation (14) is that it enables SpNet to well separate the target vehicle from the background and mitigate model degradation caused by interference factors or occlusion.

E. ALGORITHM FLOW

The flow chart of this algorithm is shown in Figure 6, which is divided into the following steps:

(1) The first frame image of the video sequence is input into the pretrained VGG-19 network, and feature maps are generated on the conv4_4 and conv5_4 convolutional layers.

(2) On the feature maps generated by the conv4_4 and conv5_ 4 layers, the feature maps most relevant to the target vehicle are selected using the sl-CNN to remove the redundant feature maps.

(3) On the feature maps generated by the conv5_4 layer, a UnNet is constructed to capture the target vehicle information; on the feature maps generated by the conv4_4 layer, an SpNet is constructed to distinguish target vehicles from

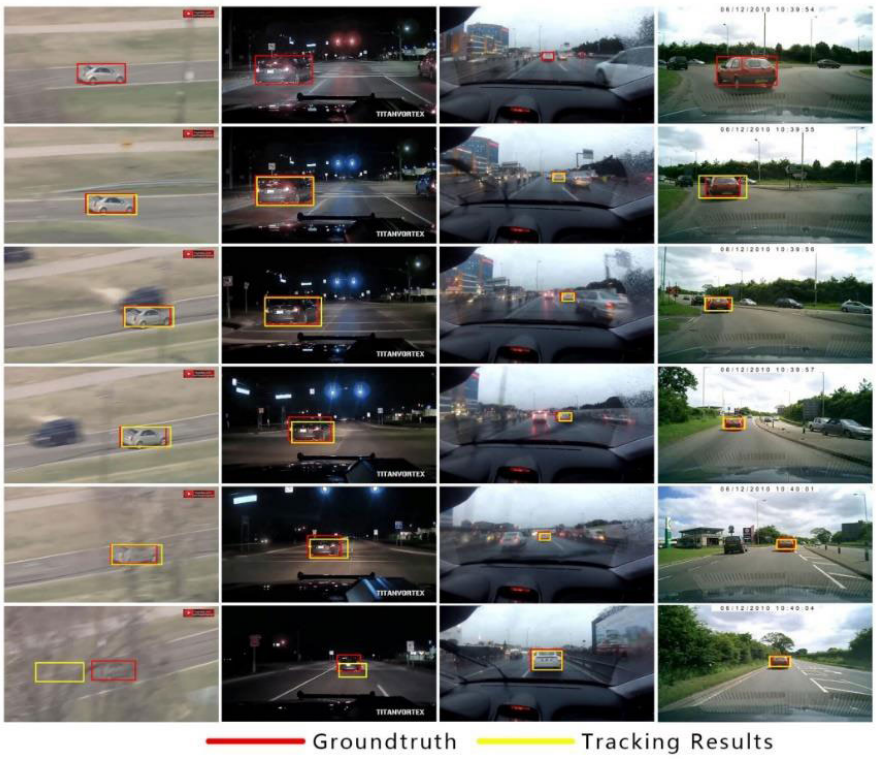

FIGURE 7. Vehicle tracking results.

vehicles with similar appearances. Both networks are initialized using the first image frame.

(4) In the new frame of the video sequence, the ROI centered on the target position of the previous frame is cropped and forward propagated through UnNet and SpNet. Then, each of these networks generates a foreground heat map.

(5) For the two foreground heat maps generated in step (4), interference detection is used to determine the position of the last target vehicle. When the ratio between the confidence values of the regions outside and inside the target area is less than the set threshold, it is considered that no common interference exists, and the target position predicted by UnNet is used as the final positioning result; otherwise, the target position predicted by SpNet is used as the final target location.

IV. EXPERIMENT

A. EXPERIMENTAL ENVIRONMENT AND DATASET

The experimental development environment is an Ubuntu 16.04 operating system, in which Python is mainly used for algorithm implementation. The program uses the deep learning framework TensorFlow and uses CUDA9.0 and cuDNN for GPU (Quadro M5000) acceleration to accelerate the implementation of the program. The training dataset is vehicle data from the Common Objects in Context (COCO) [37] dataset, with a total of 11,856 images of cars, bicycles, trucks, buses and motorcycles, as shown in Table 2. Each type of vehicle data includes many different scenarios, such as rain, night, rotation, lane change, acceleration, and occlusion. Vehicle training in these different scenarios has certain significance for training vehicle tracking models, making vehicle tracking suitable for stable tracking in different application scenarios. The test dataset is the vehicle dataset in LaSOT [38], a high-quality benchmark for single object tracking, the vehicle dataset in VOT2017 [39], and the vehicle dataset in OTB2015 [43].

TABLE 2. Number of images of each vehicle type in the training set.

Type	Car	Bicycle	Truck	Bus	Motorcycle	Total
Quantity	5324	957	1812	2875	870	11856

B. EXPERIMENTAL RESULTS AND ANALYSIS OF THE LASOT VEHICLE DATASET

This paper tests the proposed vehicle tracking algorithm based on the VGG network on the benchmark and then evaluates it. The experiment was trained a total of 50 times, with a minibatch size of 32. Each training step involved 371 iterations, with an average of 5 s per iteration, and the total training time was approximately 26 hours. When testing, the average time required for each test dataset was 3 minutes. Figure 7 shows four representative vehicle motion scenes and their tracking results, wherein the first image in a line is the first frame of each video sequence, and the remaining images in the line are the tracking results of the algorithm and the real position of the target vehicle.

The perspective of the first scene is the side view. The target car is a gray vehicle with low definition. The target vehicle travels fast on the road. During its journey, it passes a black car and is then obscured by roadside trees. As shown in the figure, the algorithm can track fast-moving vehicles and distinguish vehicles of different colors and models. When the target vehicle passes the black car, there is no offset for the tracking of the target vehicle. When the trees partially block the target vehicle, the vehicle can still be tracked, but when the trees almost completely block the vehicle, the tracking position completely disappears, and the tracked vehicle is completely lost.

The perspective of the second scene is the rear view. The scene is at night, the target car is a gray vehicle with medium definition. The target vehicle is waiting at a traffic light, there are vehicles flowing around it, and the lights changes during the waiting process. After the green light appears, the car moves forward along the road. It can be seen from the figure that for the algorithm of this paper, changes in the vehicle flow have little influence on the tracking accuracy. However, since the scene is at night when it is dark, the target vehicle appears almost black in the image, which causes the tracking frame to shift with the color change in the vehicle, indicating that the algorithm has a slightly poor tracking effect on vehicles at night.

The perspective of the third scene is the rear view. The scene is rainy. The target vehicle is a white vehicle with low definition. The target vehicle is driving on the road. During the driving process, the vehicle changes lanes. There are similar light-colored vehicles around it. After the lane change, part of the fast-moving wiper is occluded. Finally, the vehicle decelerates, the distance is decreased, and the size of the target vehicle becomes relatively large in the video sequence. As shown in the figure, for the algorithm proposed in this paper, the lane change of the target vehicle does not affect the tracking of the vehicle, and the partial occlusion of the wiper also has little effect on the tracking results. The scale change in the target vehicle caused by the change in distance is not large and does not affect the vehicle tracking results. However, the deformation of the target vehicle makes the tracking frame appear slightly smaller than the target vehicle.

The perspective of the fourth scene is the rear view. The target vehicle is a red vehicle with high definition. The target vehicle is driving on a road. First, the vehicle turns left. After the left turn, more vehicles can be seen on the road, and the target vehicle proceeds straight ahead and then turns right. It can be seen from this scenario that for the algorithm proposed in this paper, the turning of the target vehicle has little effect on the object tracking results, and the scale change in the target vehicle also has little effect on the tracking results, but the tracking frame is slightly offset from the position of the target vehicle.

To better evaluate the tracking effect of the algorithm proposed in this paper, the precision of the algorithm tracking results is evaluated by a quantitative performance evaluation index called the mean Intersection over Union (mIoU). In the accuracy evaluation, the proposed algorithm is compared with a mainstream tracking algorithm: the DLT algorithm. Table 3 shows the mIoU value of our algorithm and that of the DLT algorithm for different scenes.

TABLE 3. mIoU values of the proposed algorithm and the DLT algorithm in different scenarios.

Method	Scene1	Scene2	Scene3	Scene 4	Data Set
Ours	0.62	0.84	0.80	0.71	0.76
DLT	0.86	0.74	0.08	0.22	0.51

As shown in the above table, the mIoU value of our algorithm is generally higher than that of the DLT algorithm. Among them, in scene 1, the mIoU value of the DLT algorithm is higher than that of the algorithm proposed in this paper. After observing the tracking results, the reason is that the proposed algorithm lost the tracked target vehicle after being almost completely occluded, but the DLT algorithm continued to track the target. After the vehicle reappears, the position of the target vehicle can still be tracked, indicating that for the occlusion problem, the DLT algorithm has higher processing ability than the proposed algorithm. In scene 3, the mIoU value of the DLT algorithm is much lower than that of the algorithm proposed in this paper and is even less than 0.1. The main reason is that the target vehicle changed lanes. The DLT algorithm lost the target vehicle position when the target vehicle changed lanes and started tracking another nearby vehicle. Finally, the target position is completely lost, so the mIoU value is very low. This was not the case with the algorithm proposed in this paper. In scene 4, the mIoU value of the DLT algorithm is also much lower than that of the algorithm proposed in this paper. The main reason is that after the scale of the target vehicle in the video sequence decreased, the tracking frame of the algorithm did not decrease, indicating that the ability of this algorithm to deal with target vehicle scale change problems is not as good as that of the algorithm proposed in this paper.

Therefore, the proposed algorithm can obtain better vehicle tracking results than the DLT algorithm in most scenarios. The proposed algorithm can address vehicle tracking problems in complex and variable vehicle motion scenarios.

C. EXPERIMENTAL RESULTS ON THE VOT2017 VEHICLE DATASET

We also conducted experimental tests on the VOT2017 vehicle dataset. Five vehicle videos in the VOT2017 dataset were tested in the experiment. Following [39], we used two indicators, the *accuracy* and *robustness*, to evaluate the results. The *accuracy* is the average amount of overlap between the predicted and ground-truth bounding boxes during successful tracking periods. The accuracy of the t-th frame is defined as $\phi_t = \frac{A_t \cap A_{gt}}{A_t \cup A_{gt}}$; then, the calculation of the *accuracy* can be defined as the average accuracy of a video, namely, $\rho_A = \frac{1}{N_{valid}} \sum_{t=1}^{N_{valid}} \phi_t$, where N_{valid} is the number of valid frames. The *robustness* measures how many times the tracker loses

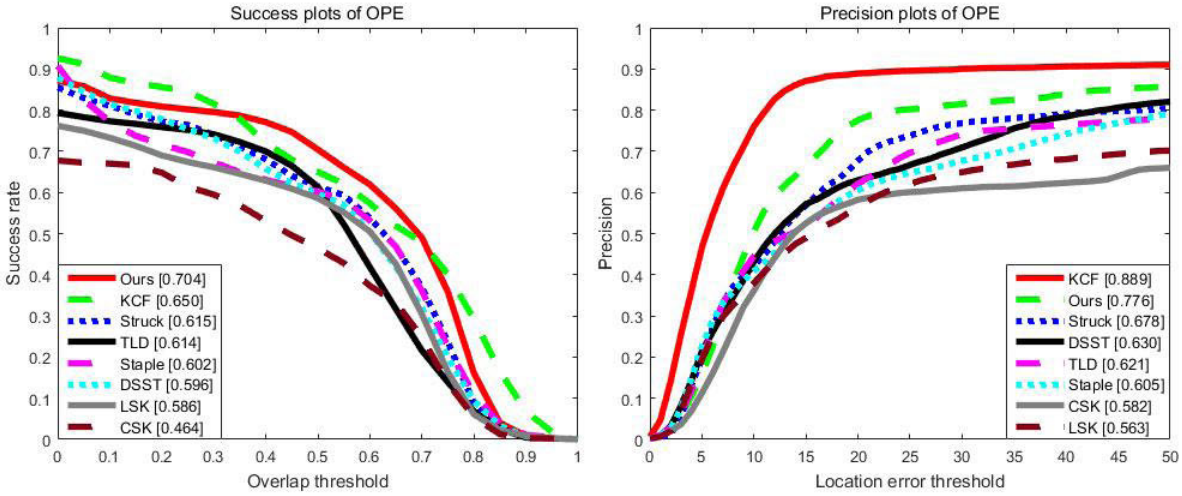

FIGURE 8. Precision and success plots of the different trackers on the OTB2015 vehicle dataset.

the target (fails) during tracking. Its calculation formula is $\rho_R = \frac{1}{N_{rep}} \sum_{k=1}^{N_{rep}} F(k)$, where N_{rep} is the number of repetitions and $F(k)$ is the number of failures in the k-th repetition. The higher the value of the *accuracy*, the better the tracking accuracy is; the lower the value of the *robustness*, the better the tracker robustness is. Table 4 shows the experimental results of the proposed method and the MOSSE [40], DCF [19], KCF [19], MEEM [41], and Struck [42] algorithms. The results show that the proposed algorithm has the highest *accuracy*, which 0.02 higher than that of the MEEM method; its *robustness* is also the best, which is 0.03 lower than that of the KCF method.

TABLE 4. Experimental results of different trackers.

Tracker	Ours	MOSSE	DCF	KCF	MEEM	Struck
Accuracy↑	0.56	0.43	0.48	0.53	0.54	0.50
Robustness↓	0.18	0.33	0.29	0.21	0.22	0.25

D. EXPERIMENTAL RESULTS ON THE OTB2015 VEHICLE DATASET

We also performed experimental tests on the OTB2015 vehicle dataset. In the experiment, fourteen vehicle videos in the OTB2015 dataset were tested. Following [43], we use the precision and success rate to perform a one-pass evaluation (OPE) of the results. Figure 8 shows the precision and success rate plots of the experimental results of our method and the CSK [18], LSK [44], Staple [45], DSST [20], TLD [46], KCF [19], and Struck [42] tracker algorithms. In the precision plot, our algorithm ranks second; in the success rate plot, our algorithm ranks first.

V. DISCUSSION

In this paper, the characteristics of the different convolutional layers in the VGG network are used, and a vehicle tracking algorithm is designed and compared with multiple tracking algorithms. The verification results show that our method has certain advantages. In this section, we discuss various factors that influence the tracking results of the proposed algorithm, including the choice of the CNN model, the choice of the convolutional layers, and the limitations of the algorithm.

A. SELECTION OF THE CNN MODEL

Currently, CNN models mainly include LeNet [47], AlexNet [48], GoogLeNet [49], VGG, and residual network (ResNet) [50]. Among them, LeNet is a representative early CNN models. It has a simple structure and only two convolutional layers. It has a shallow network structure, and its processing results for complex problems are not ideal. AlexNet is a classic CNN model and one of the models used by many current tracking algorithms. It consists of a 5 convolutional layers and a 3 fully connected layers. The VGG model simplifies the structure of the neural network. As the network deepens, the height and width of the image shrink according to certain rules. The training process of VGG converges faster than that of AlexNet. Moreover, studies have shown that when extracting CNN features from images, the VGG model is the preferred algorithm, but its shortcoming is that the number of parameters can be as high as 140 M, which requires more storage space than other models [31]. GoogLeNet is a new deep learning structure. Structures such as AlexNet and VGG all achieve better training effects when the depth of the network is increased. However, GoogLeNet uses an Inception structure to improve the training effect and make more efficient use of computing resources. However, studies have shown that VGG performs better than GoogLeNet in multiple migration learning tasks. The previous network model used layer stacking to deepen the number of network layers, but as the network deepened, the performance of the network declined. Therefore, ResNet ensures that the depth of the network can be deepened without decreasing the

performance. However, ResNet's structure is deep, and its characteristics are not easily to analyze directly. Therefore, considering the structure, advantages and disadvantages of each type of CNN model, we use the VGG in our proposed vehicle tracking algorithm and analyze the characteristics to design the vehicle tracking framework. Subsequent research may consider re-expanding the network based on the ResNet model. In addition, for the two common structures of the VGG model (VGG-16 and VGG-19), VGG-19 was chosen because its structure network is deeper than VGG-16's network structure and the characteristics are more obvious.

B. SELECTION OF THE CONVOLUTIONAL LAYERS

In Sections II-A and II-B, we analyzed the characteristics of the conv4_4 and conv5_4 convolutional layers. In fact, in the experiment, some features of each of the two layers (conv4 and conv5) can be similarly observed, but the characteristics of other small layers are relatively not significant. In addition, conv4_4 and conv5_4 are multiconvolved, and more obvious characteristics can be observed. Figure 9 shows the feature maps of each layer of conv4 and conv5. Each convolutional layer displays a representative feature map. As shown in the figure, except for the conv4_4 and conv5_4 layers, all the convolutional layers are greatly affected by noise. In addition, the feature maps of each layer in the conv4 layer with an activation value greater than 8000 inside the target region and an activation value less than 100 outside the target region and the feature maps of each layer in the conv5 layer with an activation value greater than 700 inside the target region and an activation value less than 20 outside the target region are defined as valid feature maps. Then, the number of valid feature maps of each layer in the conv4 layer and in the conv5 layer are counted, and the statistical results are plotted into a line chart, as shown in Figure 10. As shown in the figure, the conv4_4 and conv5_4 layers have the largest number of valid feature maps, so the characteristics are the most obvious. Therefore, the conv4_4 and conv5_4 convolutional layers are selected for analysis and algorithm implementation in specific experiments. After the algorithm is implemented, the combination of conv4_4 and conv5_4 is replaced with conv4_1 and conv5_1, conv4_2 and conv5_2, conv4_3 and conv5_3 in turn, and the tracking effects are not good.

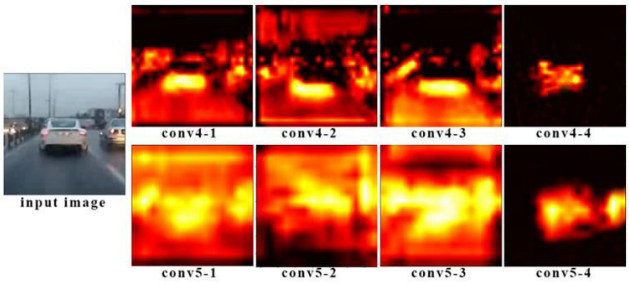

FIGURE 9. Feature maps of the different layers of conv4 and conv5.

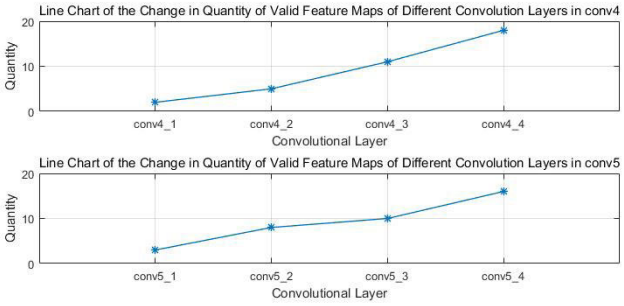

FIGURE 10. Line chart of the change in the number of valid feature maps of the different convolutional layers of conv4 and conv5.

C. LIMITATIONS

The results of the vehicle tracking experiments show that the proposed algorithm has certain advantages, but it still has certain limitations, mainly in the following aspects: (1) Although the algorithm can address the partial occlusion of the target vehicle to some extent, if the amount of occlusion is too large and the occlusion duration is too long, the target vehicle being tracked may be lost, causing a tracking failure. (2) Although the algorithm can address scale changes caused by the change in the distance relative to the target vehicle in the video sequence, the scale change will affect the tracking results. The next step should consider this problem. (3) At present, the algorithm has low efficiency, high time complexity and high spatial complexity. The VGG model needs to occupy a large amount of storage space and has high requirements on the operating environment. Therefore, the algorithm and the code must be adjusted to improve the tracking efficiency.

VI. CONCLUSION

Vehicle tracking is currently a challenging research topic in computer vision. It also has a wide range of application requirements and practical significance in the real world. In recent years, vehicle tracking algorithms have made great progress in terms of time and precision, but some of its problems still require further research and exploration.

This paper analyzes some important characteristics of VGG convolutional neural network features and proposes a vehicle tracking algorithm based on a VGG network with these characteristics. Experiments show that the proposed vehicle tracking method can effectively track vehicles and solve the drift problem and the partial occlusion problem in vehicle tracking.

Although this article solves some problems, there are still many shortcomings. For this reason, we will continue to study the following aspects:

(1) Optimizing the vehicle tracking algorithm to improve its operating efficiency.

(2) Further solving the complete occlusion problem in vehicle tracking so that when a completely occluded target vehicle reappears in the video sequence, it can be further tracked.

REFERENCES

[1] B. Karasulu and S. Korukoglu, "Moving object detection and tracking in videos," in *Performance Evaluation Software*. New York, NY, USA: Springer, 2013, pp. 7–30.

[2] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1442–1468, Jul. 2014.

[3] S. Ojha and S. Sakhare, "Image processing techniques for object tracking in video surveillance—A survey," in *Proc. Int. Conf. Pervas. Comput. (ICPC)*, Pune, India, Jan. 2015, pp. 1–6.

[4] J. Yang, R. Xu, J. Cui, and Z. Ding, "Robust visual tracking using adaptive local appearance model for smart transportation," *Multimedia Tools Appl.*, vol. 75, no. 24, pp. 17487–17500, Feb. 2016.

[5] S. Gnatzig, F. Schuller, and M. Lienkamp, "Human-machine interaction as key technology for driverless driving—A trajectory-based shared autonomy control approach," in *Proc. 21st IEEE Int. Symp. Robot Hum. Interact. Commun. (RO-MAN)*, Paris, France, Sep. 2012, pp. 913–918.

[6] A. Li, M. Lin, Y. Wu, M.-H. Yang, and S. Yan, "NUS-PRO: A new visual tracking challenge," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 335–349, Feb. 2016.

[7] P. Liang, E. Blasch, and H. Ling, "Encoding color information for visual tracking: Algorithms and benchmark," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5630–5644, Dec. 2015.

[8] S. V. Kothiya and K. B. Mistree, "A review on real time object tracking in video sequences," in *Proc. Int. Conf. Electr., Electron., Signals, Commun. Optim. (EESCO)*, Visakhapatnam, India, Jan. 2015, pp. 1–4.

[9] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–575, May 2003.

[10] T. Basar, *A New Approach to Linear Filtering and Prediction Problems*. New York, NY, USA: Wiley, 2009, pp. 167–179.

[11] P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez, "Particle filtering," *IEEE Signal Process. Mag.*, vol. 20, no. 5, pp. 19–38, Sep. 2003.

[12] X. Wang, S. Wang, and J. Ma, "An improved particle filter for target tracking in sensor systems," *Sensors*, vol. 7, no. 1, pp. 144–156, Jan. 2007.

[13] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 8, pp. 790–799, Aug. 1995.

[14] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2000, pp. 142–149.

[15] N. Dowson and R. Bowden, "Mutual information for Lucas–Kanade tracking (MILK): An inverse compositional formulation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 1, pp. 180–185, Jan. 2008.

[16] S. Alpert, M. Galun, A. Brandt, and R. Basri, "Image segmentation by probabilistic bottom-up aggregation and cue integration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 315–327, Feb. 2012.

[17] S. Yang, C.-X. Zhao, and W. Xu, "A novel salient object detection method using bag-of-features," *Acta Automat. Sinica*, vol. 42, no. 8, pp. 1259–1273, 2016.

[18] F. João, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. 12th Eur. Conf. Comput. Vis.*, Berlin, Germany, 2012, pp. 702–715.

[19] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[20] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proc. Brit. Mach. Vis. Conf.*, Nottingham, U.K., 2014, pp. 1–5.

[21] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," in *Proc. 32nd Int. Conf. Mach. Learn. (ICML)*, Lille, France, 2015, pp. 597–606.

[22] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 4293–4302.

[23] J. Zhang, Y. Wu, W. Feng, and J. Wang, "Spatially attentive visual tracking using multi-model adaptive response fusion," *IEEE Access*, vol. 7, pp. 83873–83887, 2019.

[24] H. Nam, M. Baek, and B. Han, "Modeling and propagating CNNs in a tree structure for visual tracking," 2016, *arXiv:1608.07242*. [Online]. Available: https://arxiv.org/abs/1608.07242

[25] S. Wang, Y. Liu, L. Qi, Z. Zhang, and Z. Lin, "An object tracking method based on background constraints and convolutional features," *Comput. Eng. Appl.*, vol. 56, no. 8, pp. 205–214, 2020.

[26] J. Zhang, X. Jin, J. Sun, J. Wang, and A. K. Sangaiah, "Spatial and semantic convolutional features for robust visual object tracking," *Multimed. Tools. Appl.*, vol. 79, pp. 1–21, Aug. 2018.

[27] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2013, pp. 809–817.

[28] N. Wang, S. Li, A. Gupta, and D.-Y. Yeung, "Transferring rich feature hierarchies for robust visual tracking," 2015, *arXiv:1501.04587*. [Online]. Available: https://arxiv.org/abs/1501.04587

[29] Y. Li, Q. Wu, S. Ding, and K. Hu, "A long-term tracking algorithm based on TLD and fDSST," *Electron. Opt. Control*, vol. 26, no. 4, pp. 44–48 and 70, Apr. 2019.

[30] F. Liu, G. Huang, L. Lu, H. Wang, and X. Wang, "Robust target tracking algorithm for adaptive template updating," *J. Frontiers Comput. Sci. Technol.*, vol. 13, no. 1, pp. 83–96, Jun. 2019.

[31] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ECO: Efficient convolution operators for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6638–6646.

[32] J. Zhang, X. Jin, J. Sun, J. Wang, and K. Li, "Dual model learning combined with multiple feature selection for accurate visual tracking," *IEEE Access*, vol. 7, pp. 43956–43969, 2019.

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Miami, FL, USA, Jun. 2009, pp. 248–255.

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Sep. 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[35] X. Yuan and S. Yan, "Visual classification with multitask joint sparse representation," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4349–4360, Jun. 2012.

[36] Y. Cong, J. Yuan, and J. Liu, "Abnormal event detection in crowded scenes using sparse representation," *Pattern Recognit.*, vol. 46, no. 7, pp. 1851–1864, Jul. 2013.

[37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[38] H. Fan, H. Ling, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, and C. Liao, "LaSOT: A high-quality benchmark for large-scale single object tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5374–5383.

[39] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. C. Zajc, T. Vojir, G. Hager, A. Lukezic, A. Eldesokey, and G. Fernandez, "The visual object tracking VOT2017 challenge results," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1949–1972.

[40] D. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Francisco, CA, USA, Jun. 2010, pp. 2544–2550.

[41] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: Robust tracking via multiple experts using entropy minimization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 188–203.

[42] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. S. Torr, "Struck: Structured output tracking with kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2096–2109, Oct. 2016.

[43] Y. Wu, J. Lim, and M. H. Yang, "Object tracking benchmark," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1834–1848, Sep. 2015.

[44] B. Liu, J. Huang, L. Yang, and C. Kulikowsk, "Robust tracking using local sparse appearance model and K-selection," in *Proc. 24th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Colorado Springs, CO, USA, Jun. 2011, pp. 20–25.

[45] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1401–1409.

[46] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.

[47] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.
[49] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.
[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

DANLU ZHANG received the B.E. degree in surveying and mapping engineering from the Beijing University of Civil Engineering and Architecture, Beijing, China, in 2017, where she is currently pursuing the M.Eng. degree with the Department of Remote Sensing.

Her current research interests include object recognition and object tracking.

JINGGUO LV was born in Shandong, China, in 1973. He received the B.E. degree in computer and applied majors from the Wuhan University of Surveying and Mapping, in 1997, the M.E. degree in photography and remote sensing from Wuhan University, in 2003, and the Ph.D. degree in charting and GIS from Beijing Normal University, in 2009.

He was an Assistant Researcher with the Chinese Academy of Surveying and Mapping, from 1997 to 2003. From 2003 to 2006, he served as the Deputy General Manager of Aerospace Titan Technology Company, Ltd. Since 2009, he has been teaching with the Beijing University of Civil Engineering and Architecture, where he has served as an Associate Professor. At present, he is mainly engaged in photogrammetry and remote sensing research. He has published more than 60 related articles, published four academic monographs, and authorized seven invention patents. There are nine software copyrights and four provincial and ministerial level scientific and technological progress awards. His research interests include remote sensing information extraction, digital image processing, and visual tracking.

ZHE CHENG received the B.E. degree in surveying and mapping engineering from Shandong Agricultural and Engineering University, in June 2018. She is currently pursuing the master's degree with the Beijing University of Civil Engineering and Architecture.

Her current research interests include photogrammetry, remote sensing, target detection, and target tracking.