EURASIP Journal on Image and Video Processing

RESEARCH Open Access

A vehicle tracking algorithm combining detector and tracker

Bo Yang[1], Mingyue Tang[2], Shaohui Chen[3], Gang Wang[4], Yan Tan[2] and Bijun Li[1*]

* Correspondence: lee@whu.edu.cn
[1]State Key Laboratory of Information Engineering in Surveying, Mapping, and Remote Sensing, Wuhan University, Wuhan 430079, China
Full list of author information is available at the end of the article

Abstract

Real-time multichannel video analysis is significant for intelligent transportation. Considering that deep learning and correlation filter (CF) tracking are time-consuming, a vehicle tracking method for traffic scenes is presented based on a detection-based tracking (DBT) framework. To design the model of vehicle detection, the You Only Look Once (YOLO) model is used, and then, two constraints including object attribute information and intersection over union (IOU), are combined to modify the vehicle detection box. This approach improves vehicle detection precision. In the design of tracking model, a lightweight feature extraction network model for vehicle tracking is constructed. An inception module is used in this model to reduce the computational load and increase the adaptivity of the network scale. And a squeeze-and-excitation channel attention mechanism is adopted to enhance feature learning. Regarding the object tracking strategy, the method of combining a spatial constraint and filter template matching is adopted. The observation value and prediction value are matched and corrected to achieve stable tracking of the target. Based on the interference of occlusion in target tracking, the spatial position, moving direction, and historical feature correlation of the target are comprehensively employed to achieve continuous tracking of the target.

Keywords: Vehicle detection, Correlation filter tracking, Lightweight convolutional learning network

1 Introduction

As a research hotspot in computer vision, vehicle tracking plays an important role in intelligent transportation and intelligent traffic event detection. There are currently two main object tracking frameworks: detection-based tracking (DBT) and detection-free tracking (DFT) [1]. DFT needs to manually initialize the tracking target, so it is only applicable to tracking a specified target and cannot automatically detect and track a new target that appears in the monitoring process. DBT integrates detection and tracking and can automatically detect the emergence of new targets or the disappearance of existing targets. Thus, DBT is capable of meet the actual requirements of the random disappearance of targets or the dynamic change of targets in the monitoring scene.

In most intelligent transportation applications, fixed cameras are used to monitor scenes that dynamically change in specific areas, so this study is conducted based on the DBT framework. DBT framework-based object tracking can be further categorized into background modeling-based [2–10] and foreground modeling-based [11, 12] object detection. The core of background modeling-based object detection algorithms lies in background initialization, i.e., how to obtain a background image frame without any moving objects. The foreground modeling-based method is complemented by establishing an appearance model for the objects or background, and then trains a classifier to establish a classifier model.

In real-world traffic monitoring, detection and tracking of vehicles in monitoring scenes are required in all-weather conditions at all times. Therefore, background modeling-based object detection will face challenges from a dynamically changing background, including illumination variation, disturbance from nonvehicle moving objects, diurnal variation, camera shaking, and so on. Thus, a foreground modeling-based vehicle detection model is used to detect and track vehicles in this study.

In terms of the number of tracking objects, vehicle tracking in traffic scenes can be divided into single vehicle tracking and multivehicle tracking. Many methods were developed to track a single vehicle, such as part-based particle filter method [13], a fusion method of multiple single-target trackers [2]. In paper [3], a group of mask templates generated by interframe differences were introduced, and the accelerated near-end gradient (APG) algorithm was adopted for template adaptive adjustment to improve the tracking accuracy and efficiency.

For multivehicle tracking in a traffic scene, many works were carried out by the DBT framework. In paper [4], target detection is realized via the VGG16 network, and then a sparse optical stream-based method is adopted for tracking. This method involves optical flow calculation; thus, its real-time performance is difficult in practice. Based on video of an unmanned aerial vehicle (UAV), the researchers adopted the fast region-based convolutional neural network (fast RCNN) to detect the vehicle, and then uses the Kalman filter to track the vehicle [5]. This method is more efficient but is not applicable to the situation of vehicle interleaving or disappearing. Focusing on the expressway scene, paper [6] adopted the Single Shot MultiBox Detector (SSD) based on deep learning to detect vehicles. The correlation of track timing information was combined with the kernelized correlation filter (KCF) to achieve vehicle tracking. This method cannot achieve the continuous and effective tracking of multiple vehicles for more complex urban scenes. Paper [7] used the fast-RCNN for vehicle detection and then followed by an additional branch based on a faster RCNN structure for vehicle tracking. The experimental results showed that the performance was satisfactory for situations with sparse vehicle distribution in a scene, but its effect needs to be verified for the more complex urban scenes.

The keys to automatic vehicle tracking based on the DBT framework is the object detector, the object tracker design and the strategy of integrating the detector and the tracker. Focusing primarily on the need of multichannel video monitoring in intelligent transportation, this study investigates a vehicle tracking method that satisfies the real-time multichannel video analysis requirements and improves vehicle tracking precision. Considering the complexity and application feasibility of the algorithm, in the object detection step, the detection result provided by You Only Look Once (YOLO) were

post-processed and then used as the input for the tracker. For the tracker, a lightweight convolutional neural network (CNN) was designed to extract image features and incorporate them into a correlation filter (CF) tracking framework. Finally, a tracker-detector integration strategy was designed to automatically track multiple vehicles in traffic scenes.

2 Related work

2.1 YOLO postprocessing-based vehicle detection

Based on the modeling technique, foreground modeling-based object detection algorithms can be further categorized into two types, namely, algorithms based on statistical learning and algorithms based on deep learning [14–19]. Statistical learning-based algorithms are carried out by training an object classifier based on hand-crafted features and use a multiscale sliding window to search within the object region. These algorithms are relatively highly resistant to environmental interference and unaffected by objects' shadows. However, these algorithms have relatively low processing speeds. Additionally, manually designed feature extraction operators are unable to address diverse objects. Conventional machine learning algorithms can only learn a small number of samples and have poor generalization ability.

With the emergence of large-scale datasets and the upgrade of computer software and hardware, deep learning methods are developed rapidly, and utilized in many applications. To study an explicit visual-semantic dictionary model for cross-modality understanding, a multi-task learning model was constructed, which models the co-occurrence and discriminative patterns within and between categories [20–22]. Actually, there are many researches have been studied for object detection based on deep learning recently. Comparing with the conventional methods, the precision of object detection based on deep learning networks has been rapidly improved. Among all the existed deep learning networks, regression models, which are represented by YOLO [19, 23], are advantageous in both detection precision and speed.

Considering its exceptional object detection capacity, the YOLO network model is used in this study to construct a vehicle detector that is suitable for monitoring scenes in intelligent transportation. First, the current YOLO network is transformed from generic object detection to vehicular object detection. Additionally, vehicles are classified into eight types based on the classification standard issued by transport agencies, namely, small, medium, and large coaches; small, medium, large, and superlarge trucks, container vehicles, and motorcycles. The YOLO network model is retrained on object classification based on traffic scene datasets.

Additionally, to learn universal and distinct features, the current YOLO model training algorithm trains the feature extraction network based on a combination of detection and classification datasets. The basic principle is described as follows. The detection dataset is used to learn the accurate locations of the objects, whereas the classification dataset is used to increase the number of classes to improve the robustness of the detection algorithm. Following this training model, the detector is prone to detecting one object in the semantic sense as multiple objects in real-world traffic monitoring scenes.

There are two main situations:

(1) Detection box nesting: part of an object is detected as an independent object.
(2) Detection box overlapping: one object is detected as multiple different objects, and the detected objects overlap one another.

To address these problems, the following strategy was proposed in this study to screen and combine the detection.

(1) First, for situations where a smaller rectangular box is nested inside a larger rectangular box, the smaller rectangular box is removed.
(2) Then, the nonmaximum suppression algorithm is employed to suppress rectangular object boxes with relatively low object confidences, and the rectangular object boxes with relatively high confidences are preserved. Additionally, the rectangular boxes are combined based on the object class information.

Specifically, in the detection results of rectangular object boxes with intersection over union (IOU, i.e., the ratio of the intersection of rectangular boxes to the union of rectangular boxes) values that are greater than the threshold th and of the same class, the rectangular box with a relatively low object confidence is directly removed. The rectangular object boxes with an IOU greater than th and of different classes are then combined. The smallest enclosing rectangle in the union region of the two rectangular boxes is selected.

Nonmaxima suppression in the proposed algorithm takes into account not only the IOU of the detection box but also the class attribute, i.e., it treats the class attribute as an additional restriction of the rectangular box constraint. This approach ensures that the merged detection box can satisfactorily preserve the semantic integrity of the object and provides the object tracker with a better initial value.

3 Method

3.1 CF object tracking based on deep features

An object tracker generally consists of three parts, namely, an appearance model, a motion model, and an update model. The general flow of an object tracking algorithm is described as follows: (1) each tracked object is represented by modeling, and an appearance model is established based on the initial information. (2) The appearance model is used to determine the location of the object in the current frame. (3) Based on the tracking results with respect to the current frame, an update strategy is used to update the appearance model to allow it to adapt to changes in the object and the environment. Based on whether the appearance model is established using the background information, object tracking algorithms can be categorized into two types: generative and discriminative models [24]. Because they use both the background and object information, discriminative models generally exhibit higher tracking performance than generative models. In recent years, discriminative models based on the CF tracking framework have garnered extensive attention due to their advantageous performance and efficiency [25]. The CF algorithm generates positive and negative samples by cyclically shifting feature maps (e.g., grayscale, color, and histogram of oriented gradients (HOG) feature maps) to learn the filter h and then perform a convolution operation on the image f_i:

$$g_i = f_i * h \tag{1}$$

Thus, a correlation information map is obtained. In the correlation map, the location with the maximum value is the location of the object.

As deep learning advances, deep features will replace conventional features, which will further improve the tracking performance of CF algorithms. The deep spatially regularized discriminative CF algorithm [26] learns a CF in the first layer of a single-resolution deep feature map. The hierarchical convolutional feature (HCF) algorithm [27] improves the tracking performance by training a CF based on the features of multiple convolutional layers.

Deep learning is advantageous because this technique, which is driven by data and tasks, is capable of automatically learning how to extract model features and avoids the incompleteness caused by hand-designed features.

The DCFNet algorithm proposed by Wang et al. [28] treats a CF as a layer of a deep network, thereby achieving end-to-end training for tracking tasks. While they have achieved relatively satisfactory tracking performance, these algorithms generally have relatively low processing speeds and are inadequate for practical applications. The time cost of CF tracking algorithms results from the feature extraction process as well as the online-learning, detection, and update processes of the filter.

3.1.1 Network structure design

Figure 1 shows the tracker network structure designed in this study. This network structure consists of three parts; namely, a feature extraction network, a CF layer, and a response loss layer. The feature extraction network is a vertically symmetrical twin structure. The upper branch of the feature extraction network is referred to as the historical branch, i.e., the branch where the location of the object is known. The lower branch of the feature extraction network is a branch where the location of the object is unknown, and its objective is to allow the network to learn how to search for the object in the subsequent frame when its location in the current frame is known.

Two key issues need to be addressed when using deep learning to extract features, namely, (1) how to design a suitable feature extraction network structure based on a specific task and (2) how to design a model training loss function to optimize the network parameters.

The feature extraction process for deep convolutional networks shows that shallow networks tend to obtain features of an object, such as physical outline, edges, color, and texture. The extracted features become increasingly abstract as the number of

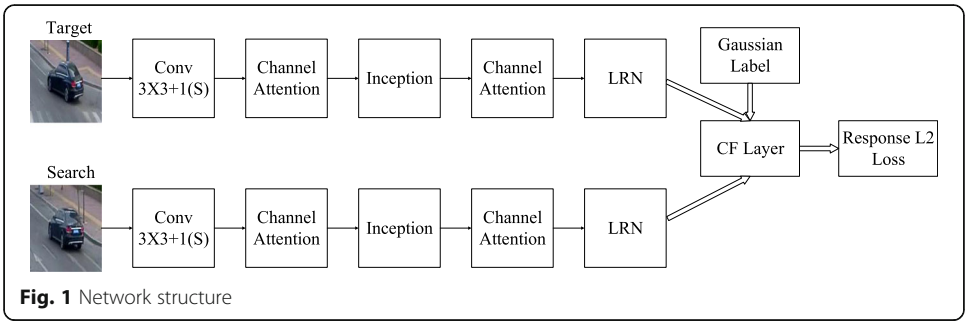


Fig. 1 Network structure

network layers increases. As the network deepens, the object positioning precision decreases. In a traffic scene, a significant change in the size of a vehicle occurs as the vehicle moves from far to near or from near to far. Therefore, an excessively large number of network layers will be unfavorable to small-scale detection and tracking. An increase in the number of network layers will cause an increase in the computational load and affect real-time application.

Inspired by this, a lightweight shallow feature extraction network was designed in this study, because shallow networks can more easily learn the features (e.g., physical outline, edges, color, and texture) of objects. This network consists of a convolutional layer, an inception module [29], two channel attention modules, and a local response normalization (LRN) layer.

(1) Convolutional layer: this layer contains 96 3 × 3 convolution kernels with a step size of 1.
(2) Channel attention mechanism module 1: this module recalibrates the feature maps generated by the convolutional layer, suppresses the invalid features, and improves the valuable features.
(3) Inception module: as demonstrated in Fig. 2, the inception module combines the features of the receptive fields of multiple scales (1 × 1, 3 × 3, and 5 × 5) and allows the network to determine the filter type in the convolutional layer on its own. This enriches the features learned by the network. Additionally, the 1 × 1 convolution kernels before the 3 × 3 and 5 × 5 convolution kernels reduce the feature channel thickness and the computational load of the network structure. The dimensions of the feature channels outputted by the receptive fields of the 1 × 1, 3 × 3, and 5 × 5 scales are 4, 8, and 4, respectively. This is because the receptive field of the 3 × 3 scale outperforms the receptive fields of the 5 × 5 and 1 × 1 scales in terms of local perceptibility.
(4) Channel attention mechanism module 2: this module recalibrates the feature maps generated by the inception module.
(5) LRN layer: this layer performs interchannel normalization on the outputted feature maps, limits the magnitude of the outputted eigenvalues, and renders the training process more stable.

The rectified linear unit (ReLU) activation function is used after each convolutional layer for response. This is mainly because ReLU is a piecewise linear function with a

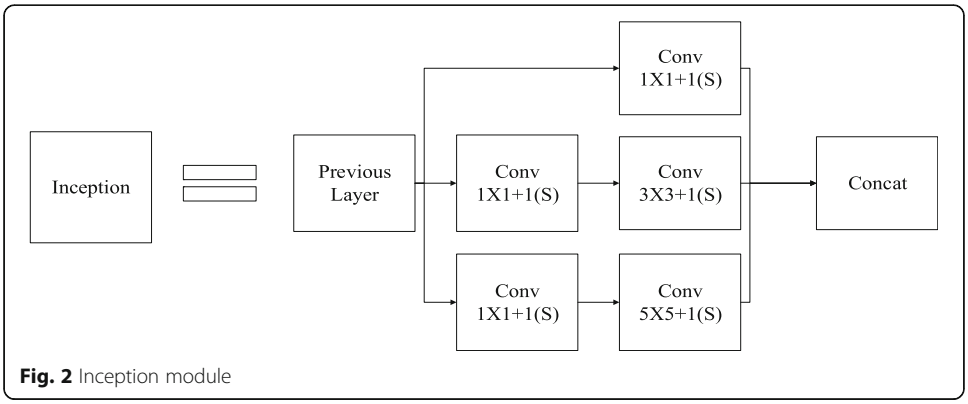


Fig. 2 Inception module

relatively high forward propagation and backward feedback speeds. Additionally, ReLU has a gradient of 1 in the > 0 region and is not associated with the vanishing or exploding gradient problem.

The conventional CF approach is adopted for the CF layer. The CF layer learns a filter based on all the cyclic shifts of the feature maps outputted by the historical branch and correlates the filter with the feature maps outputted by the current branch to generate a response map.

The response loss layer uses a two-dimensional (2D) Gaussian function with a peak at the center as the label and the L_2 norm to measure the loss between the response map and the label. The following section provides a detailed introduction to the key components of the network structure.

3.1.2 Channel attention mechanism

In the video-based vehicle tracking process, indiscriminately searching for a vehicle in all regions within the field of view is clearly time-consuming. The attention mechanism of biological vision can help an organism to quickly focus on an object of interest. Introducing the attention mechanism into computer vision to increase the search speed in regions where the vehicle is suspected to be located in the full field of view will undoubtedly be favorable to improving the vehicle detection and tracking performance [30]. In this study, a visual attention mechanism is introduced into the feature extraction network. This approach enables the feature extraction network to highlight the vehicle features in the scene, suppress the background features, and improve the effectiveness of the network in representing vehicle features, which improves vehicle detection and tracking precision and speeds. The channel attention module shown in Fig. 3 is introduced to the output of each layer [31].

This module consists of three components, namely, a squeeze operation, an excitation operation, and a scale operation.

For an input x with $c1$ feature channels, a feature map with a feature dimension of $c2$ is outputted after a convolution operation. By global pooling, the squeeze operation turns each 2D feature channel into a real number with a global receptive field and generates a one-dimensional vector consistent with the dimension of the feature map. This vector characterizes the global distribution of responses in the feature channel and allows the layers close to the input layer to have a global receptive field.

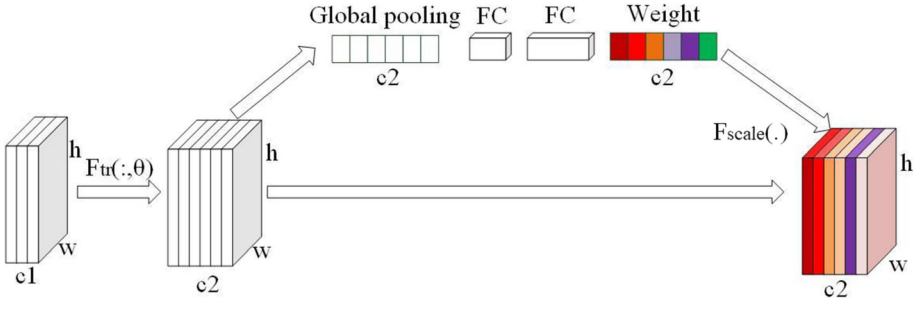


Fig. 3 Schematic of a squeeze-and-excitation channel attention module

The main goal of the excitation operation is to explicitly model the correlation between channels. This step is achieved through two fully connected layers. The first fully connected layer reduces the feature dimension to 1/16th of the feature dimension of the input. After obtaining a response from the ReLU activation function, the dimension is increased to the original dimension through another fully connected layer. Ultimately, the output and input have the same feature dimension. Finally, a sigmoid activation function is used to normalize the output value to the range of 0–1. This approach has the following advantages:

(1) It allows the structure to be nonlinear to better fit the complex correlation between channels.
(2) It reduces the number of parameters and computational load and ensures a lightweight network.

The weights outputted by the previous step represent the importance of the feature channels selected by the attention module.

3.1.3 CF and response loss layers

In the CF layer, let $M \times N$ be the dimensions of the input image block x in the historical branch. A feature map $\phi(x) \in R^{M \times N \times D}$ is obtained for the image block using the feature extraction network. Positive and negative samples are then generated by cyclically shifting the feature map and are then used to train a filter w, which can be obtained using a minimization equation (Eq. (2)):

$$\min_{w} \|Xw - y\|_2^2 + \lambda \|w\|_2^2 \tag{2}$$

where λ ($\lambda \geq 0$) is a regularization coefficient and $X = [x_1, x_2, \ldots, x_n]^T$ is a data matrix consisting of all the positive and negative samples generated by cyclically shifting the feature map. A closed-form solution to Eq. (3) can be calculated using the least-squares method [32]:

$$w = \left(X^T X + \lambda I\right)^{-1} X^T y \tag{3}$$

The above equation can be rewritten as the following equation in the complex number field:

$$w = \left(X^H X + \lambda I\right)^{-1} X^H y \tag{4}$$

X is a circulant matrix. Therefore, the filter in the lth ($1 \in \{1, \ldots, D\}$) feature channel can be written in the form of Eq. (5):

$$\hat{w}_l = \frac{\hat{y} \odot \hat{x}_l^*}{\sum_{i=1}^{D} \hat{x}_i^* \odot \hat{x}_i + \lambda} \tag{5}$$

where e signifies the Hadamard product, $\bullet^*$ signifies a complex conjugate operation, and $\hat{\bullet}$ signifies the discrete Fourier transform of the vector.

In the current branch, the to-be-searched-for image block z and the image block x have the same dimensions. Through the feature extraction network, a feature map $\phi(z)$

is obtained for the image block z. The response R of the feature map $\phi(z)$ and filter can be calculated using Eq. (6).

$$R = F^{-1}\left(\sum_{l=1}^{D} \hat{w}_l \odot \hat{\phi}(z)_l^*\right) \tag{6}$$

The loss function of the network is defined as the L_2 norm between the response R and the label $\tilde{R}$ of the 2D Gaussian function with the peak at the center. Equation (7) shows the loss function.

$$\begin{aligned} & L(\theta) = \left\|R - \tilde{R}\right\|^2 + \gamma\|\theta\|^2 \\ s.t. \quad & R = F^{-1}\left(\sum_{l=1}^{D} \hat{w}_l^* \odot \hat{\phi}_l(z,\theta)\right) \\ & \hat{w}_i = \frac{\hat{y}^* \odot \hat{\phi}_l(x,\theta)}{\sum_{i=1}^{D} \hat{\phi}_i(x,\theta) \odot (\hat{\phi}_i(x,\theta))^* + \lambda} \end{aligned} \tag{7}$$

The forward derivation process for the CF layer was previously provided. To achieve end-to-end training, it is also necessary to derive backpropagation forms. The backpropagation forms of the historical and current branches can be derived using the chain method, as shown in Eq. (8) (see elsewhere [26] for details).

$$\begin{aligned} \frac{\partial L}{\partial \phi_l(x)} &= F^{-1}\left(\frac{\partial L}{\partial (\hat{\phi}_l(x))^*} + \left(\frac{\partial L}{\partial (\hat{\phi}_l(x))}\right)^*\right), \\ \frac{\partial L}{\partial \phi_l(z)} &= F^{-1}\left(\frac{\partial L}{\partial (\hat{\phi}_l(z))^*}\right). \end{aligned} \tag{8}$$

3.1.4 Online update and scale adaptation

In the object tracking process, changes occur in the scale and angle of the object in the image sequence. Additionally, the background where the object is located also changes with time. To achieve accurate and stable object tracking, an online update strategy must adapt to the changes in the object and background.

Figure 4 shows the online tracking process of a deep network–CF-combined object tracker. First, the deep features of the object region in the first frame of the tracking video sequence is extracted and used to train an initial filter. Then, in the subsequent frame, a search region is set with the object's location in the previous frame as the center. Deep features within this search region are located and inputted into the filter for response. The maximum location in the response map is the location of the object in the current frame. Finally, the filter template is updated based on this new location.

In the tracking flow in Fig. 4, based on the response map, only the position of the object can be predicted, whereas changes in the scale of the object cannot be accurately perceived. If the object shrinks, the filter will learn a large amount of background information. Conversely, if the object expands, the filter will drift with the local texture of the object. To allow the tracker to adapt to scale variations, a multiscale search strategy is often adopted. The general flow of a multiscale CF tracking algorithm is described as follows:

(1) The deep features of the tracked object region in the first frame of the video sequence are extracted, and a CF is obtained by initialization.

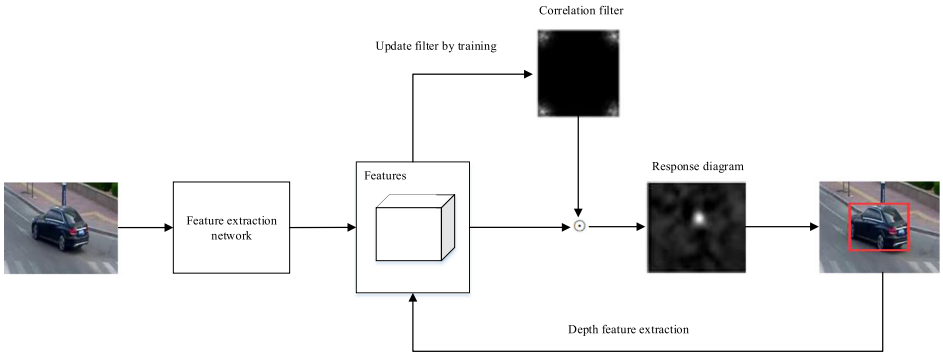


Fig. 4 Deep network–CF-combined tracker framework

(2) For each image frame in the subsequent input video sequences, an image pyramid is established based on the tracked object region predicted from the previous image frame. Equation (9) shows the pyramid scale factor.

$$\left\{a^s | s = \left\lfloor -\frac{S-1}{2} \right\rfloor, \left\lfloor -\frac{S-3}{2} \right\rfloor, \ldots, \left\lfloor \frac{S-1}{2} \right\rfloor \right\} \tag{9}$$

Because the filter template has a fixed size, it is necessary to uniformly normalize the multiscale images to the size of the filter template. Then, a multiscale feature map can be obtained by using the feature extraction network.

(3) The feature maps of each scale are first processed with a window function and then allowed to respond to the CF template. In each response map, the location of the maximum value is the predicted location of the object, and the corresponding scale is the scaling ratio for the object in the current frame.

(4) The image features of the new location of the object are extracted to update the CF template. Equation (10) shows the update strategy for the filter template. This update strategy is then implemented to sufficiently make use of the historical information provided by the video sequence and improve the robustness of the filter to allow the filter to cope with interference from external factors (e.g., illumination and blocking objects).

$$W = \eta W + (1-\eta) A_i \tag{10}$$

where η is the learning efficiency and A_i is the calculation results for the current frame.

3.2 Tracker–detector-integrated object tracking

The detector of the target is designed to provide the initial positioning for the target to be tracked and then provide objects for the tracker to achieve tracking. Based on the detector, automatic detection of the target can be achieved without manual determination of the initial tracking target and can address the dynamic change in the

appearance of the new target and the disappearance of the old target in the monitoring process, which is the necessary link of the automatic tracking of the target.

The target detector can be used to detect the target in the video image and extract the target information of each frame of the image. In the case of single target tracking, target tracking can be achieved by simply using the detector. In the case of multitarget application in the scene, however, multitarget tracking cannot be realized because the target detector cannot establish the corresponding relationship between two multitarget frames.

In this study, a YOLO detector–CF tracker-integrated object tracking algorithm was proposed for tracking moving objects in complex traffic scenes. First, a tracker is used to predict the locations of the objects in the subsequent frame. Observed values close to the predicted values are searched for close to the predicted values. Additionally, the CF template is used to select the matched observed values to correct the predicted values. Then, each object that fails to be tracked due to blocking is retracked based on the correlations between the spatial location, moving direction, and historical features. The experimental results have demonstrated that the proposed algorithm is relatively highly robust and able to retrack objects that were previously blocked for a short period of time. Figure 5 shows the flowchart of the object tracking algorithm designed in this study, which mainly consists of two parts: (1) matching between the observed and predicted values and correction of the predicted values; and (2) processing the blocked and new objects.

3.2.1 Peak-to-sidelobe ratio (PSR)-based tracking quality evaluation

In ideal conditions, a CF can predict the location of an object at the current moment if its location at the previous moment is known. When tracking objects in an actual traffic scene, a tracking drift or a loss of tracked objects can often occur due to mutual blocking between objects or other factors. Therefore, determining whether a tracking drift or a loss of tracked objects has occurred and, if so, addressing the problem are the key issues in achieving robust tracking. In this study, the PSR, which is extensively used

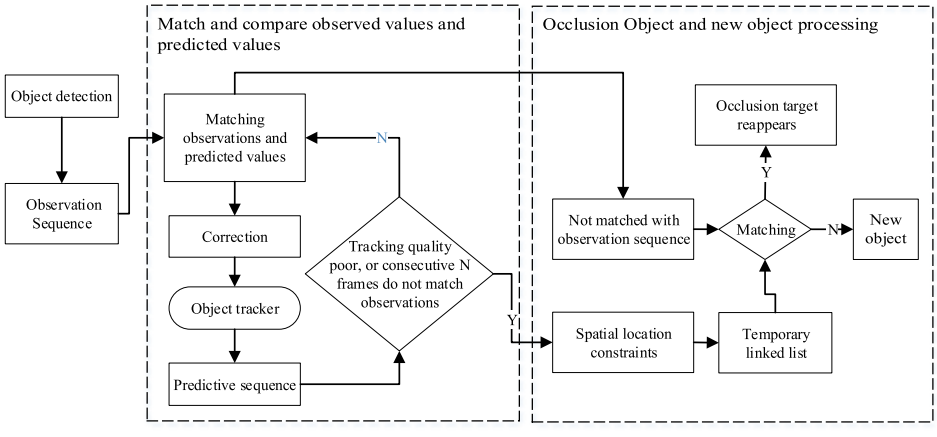


Fig. 5 Flowchart of the tracker–detector-integrated object tracking algorithm

when applying CFs, was used to evaluate the tracking quality. The PSR can be calculated using Eq. (11) [33].

$$PSR = \frac{g_{\max} - \mu_{s1}}{\sigma_{s1}} \tag{11}$$

where $g_{\max}$ is the peak value of the CF response map, and μ_{s1} and σ_{s1} are the average value and variance, respectively, within an $N \times N$ window, with the peak value of the response map as its center, respectively.

In this study, N was set to 12. Through testing and statistical analysis, the PSR was found to range from 5 to 10 in the normal tracking state. When PSR < 5, it can be determined that a tracking drift or loss of the tracked object has occurred. If a tracker exhibits relatively poor tracking quality or fails to match the observed values in multiple consecutive frames, it may be because the object has been blocked or has left the field of view. In this study, an effective detection region was established in the object tracking process. The distance d of an object from the boundary of the detection region along its moving direction can be calculated. When $d<D$ (D is the distance th between the object and the boundary), the object has left the field of view. Trackers whose relatively poor tracking quality has been determined to not be due to the departure of the object from the field or view, or that fail to match with the observed values in multiple consecutive frames, are added to a temporary linked list. Additionally, a survival period is also set. Trackers past the survival period will be removed.

3.2.2 Matching between the observed and predicted values and the correction of the predicted values

Continuous, steady object tracking can be achieved by correcting the tracker based on the observed values for the object. Therefore, how to establish the matching relationship between observed and predicted values is a key issue in achieving steady tracking. Generally, the predicted and observed values for an object are relatively close in terms of spatial distance. Matching between the predicted and observed values by spatial constraints is sufficient for scenes with relatively low vehicle densities. However, for scenes with relatively high vehicle densities and objects that relatively heavily block one another, as shown in Fig. 6, spatial constraints may easily lead to mismatching. In view of this problem, observed and predicted values were matched in this study through a combination of a spatial constraint and filter template. The spatial constraint is as follows:

$$IOU\left(r_d, r_p\right) = \frac{r_d \cap r_p}{r_d \cup r_p} > th \tag{12}$$

Observed values are selected as candidate matches if the IOU between them and the predicted values is greater than th. Extracting objects relatively close to the predicted values significantly reduces the search region and improves the processing efficiency. In this study, th was set to 0.2.

An image block is extracted with each candidate matched object selected in the previous step as the center. A deep network is then used to extract the image features. A response value is subsequently obtained by performing a correlation operation on the image features and the object filter template corresponding to the predicted value. The

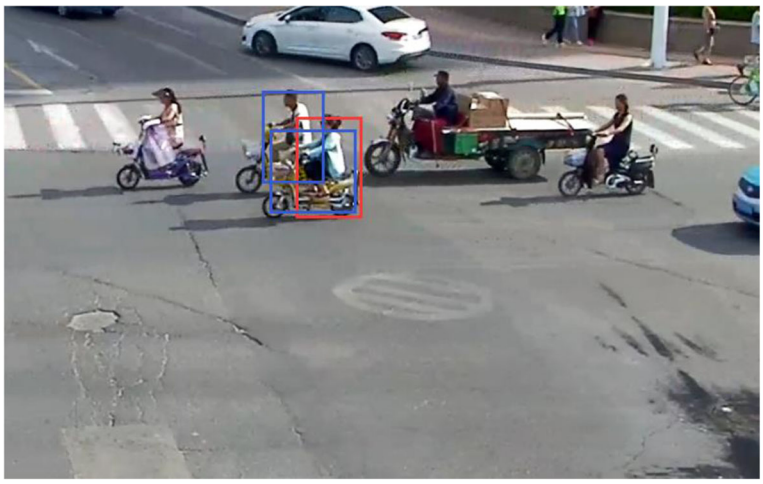


Fig. 6 Schematic that depict overlapping observed and predicted values of an object (red box signifies the predicted value; blue boxes signify the observed values)

candidate match with the highest response value is the ultimate match. To correct a predicted value, the higher predicted value and the highest response value corresponding to the observed value are selected because the peak response value represents the predicted object confidence, i.e.,

$$r = \begin{cases} r_d \ if \ g_{r_d} \geq g_{r_p} \\ r_p \ else \end{cases} \tag{13}$$

where r is the final result, r_d is the predicted value, and r_p is the observed value.

3.2.3 Tracking of blocked and new objects

An unmatched observed value may be a new object that has entered the field of view or an object that has been blocked again. To retrack blocked objects, objects that meet the following conditions are first searched for on the temporary linked list:

(1) An object on the temporary linked list that is located within a circular region with the observed value as the center and R as the radius.
(2) In a traffic scene, an object will not suddenly change its direction within a short period of time. The location where an observed value reappears should be in front of the moving direction of an object on the temporary linked list.

For each object that meets the above conditions, an image block is extracted with its observed value as the center, and the image features are extracted using a deep network. Then, the PSRs of the feature map and the response map outputted by the object tracker filter template are calculated. If the PSR of the response map is greater than th, then the object is the same object that previously disappeared, and the tracker is moved to the tracking linked list; otherwise, the object is a new object. A new object is located on the boundary of the monitoring region. As a result, its observed value only contains part of its information. Using this rectangular box to initialize the tracker will result in unsteady tracking results. To address the boundary issue, let $p(x_i, y_i)$ be the location of the center of an object at the current moment. The distance di (i = 0, 1, 2, and 3) of

the object from the boundary of the image can be easily calculated. When $\min(\{d_i, i = 0, 1, 2, 3\}) > dist$, the object has completely entered the monitoring region. Under this condition, the tracker for this object is initialized. In this study, th for the rematching blocked objects was set to 6.0.

4 Results and discussion

This section mainly shows a comparative analysis and discussion of the tracking algorithm proposed in this study and the available high-performance CF trackers in terms of speed and performance based on open datasets to examine the effectiveness of the proposed algorithm.

4.1 Experimental parameters and dataset

The experiment was conducted on a computer with a Windows 10 operating system, an Intel Core i7-7700 K quad-core processor, 8 GB of random-access memory, an NVIDIA GeForce GTX 1080 graphics card, and 8 GB of video memory. The video object detection dataset used in the ImageNet Large Scale Visual Recognition Challenge 2015 [34] was used as the experimental training dataset. This dataset contains 3862 training sequences, 555 validation sets, and 937 training sets. The length of the video sequences ranges from 56 to 458 frames. During the training process, samples of one object at two moments were selected from a video sequence to constitute a sample pair. A region 2.5 times the size of the object was clipped, and the areas beyond the boundary were filled with zeros. The sample input size was normalized to 75 × 75, and the batch size was set to 50. A stochastic gradient descent with a momentum of 0.9 was used to adjust the parameters. The weight decay coefficient and learning efficiency were set to 0.005 and 0.00001, respectively. A total of 20 epochs of iteration were performed. The number S of the CF scale pyramids was set to 3, and the scale factor a was set to 1.0275.

The test datasets OTB2013 and OTB2015 contain 100 video sequences and addresses the difficult problems in the visual tracking field, e.g., scale variation, rapid movement, and motion blur. The OTB standard test platform was used for testing.

4.2 Evaluation metrics dataset

Three indices, namely, one-pass evaluation (OPE), time robustness (TRE), and spatial robustness (SRE), were used to evaluate the performance of the tracking algorithm. The average precision and success plots were applied in the quantitative analysis of the robustness of these indices.

To evaluate the performance of the tracking algorithm, three indices—OPE, TRE, and SRE—were used for evaluation. In the quantitative analysis of the robustness of these indices, the average accuracy curve and average success rate curve were used to describe them.

(1) Average precision plot: precision refers to the percentage of video frames in which the Euclidean distance between the centers of the tracker-predicted and actual locations (unit: pixel) is shorter than the given th to all the video frames. A precision plot is produced by varying th.

(2) Average success plot: success refers to the percentage of video frames in which the overlap rate (OR) between the tracker-predicted and actual rectangular location boxes is greater than th for all the video frames. A success plot is produced by varying the overlapping ratio th. The area under the success curve (AUC) can more comprehensively reflect the performance of a tracker. OR is defined as the ratio of the area of the intersection between the predicted and actual rectangular boxes to the area of the union between the predicted and actual rectangular boxes.

$$OR = \frac{|\ r_p \cap r_g\ |}{|\ r_p \cup r_g\ |} \tag{14}$$

where r_p and r_g are the predicted rectangular box and actual rectangular boxes, respectively.

OPE was used to evaluate the tracking performance. Specifically, only the first frame was initialized, and then the average precision and success were calculated using the aforementioned methods. The 2013 and 2015 Online Tracking Benchmark (OTB) datasets were selected. These two datasets contain a total of 100 video sequences, covering 11 difficult problems in the visual tracking field, e.g., scale variation, rapid movement, and motion blur. The OTB standard test platform was used for testing.

4.3 Experimental results and discussion

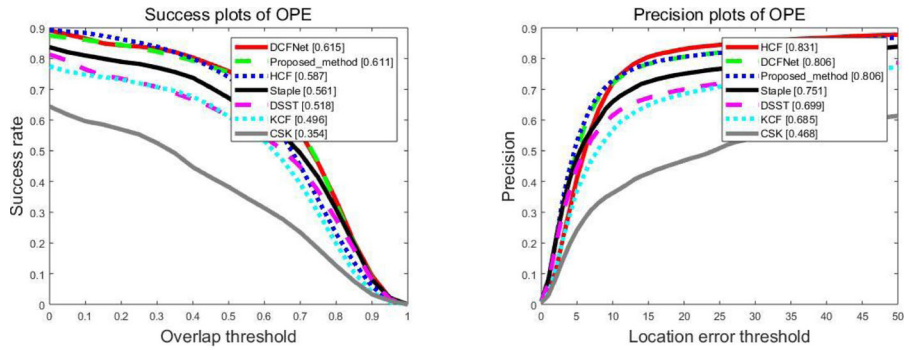
4.3.1 One pass success rate (OPE)

Figure 7 shows the average success and precision curves for various algorithms with respect to the OTB 2013 and 2015 datasets. In the average success plots, the number within each pair of square brackets is the AUC value. In the average precision plots, the number within each pair of square brackets is the average precision for 20 pixels. The experimental results show that the model designed in this study achieved a precision similar to that of the deep network–CF-combined trackers (e.g., HCF [27] and DCFNet [28]). Due to their deep network structure or high dimensionality, these algorithms have relatively low processing speeds but relatively high tracking precision and are advantageous over CF trackers (e.g., circulant structure kernel (CSK), KCF [6], discriminative scale space tracker (DSST), and Staple tracker) with hand-crafted features in precision.

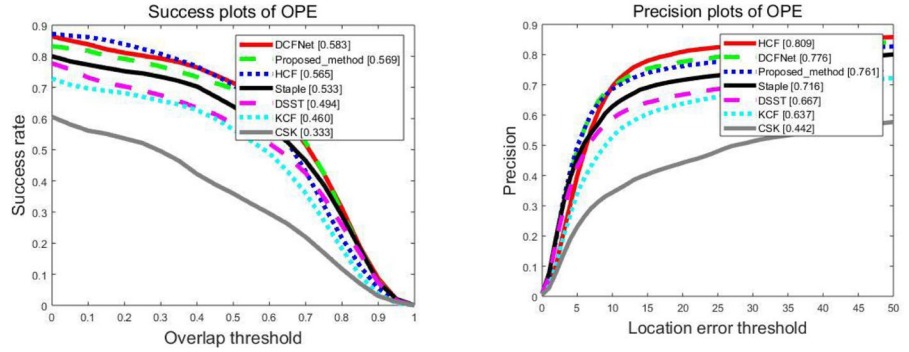
4.3.2 TRE

A total of 100 frames are randomly selected from each test sequence as the initial frame and the initial frame of tracking, which yields the average accuracy and average success rate, respectively, of the different tracking algorithms.

As shown in Fig. 8, for the OTB2013 dataset, the methods proposed in this paper rank first among the listed comparison methods in terms of average success rate curve and average accuracy. In the OTB2015 dataset, the average success rate curve and ranking of the average accuracy of the proposed method have both decreased but are comparable to Staple.



(a) Average success and precision curves of various algorithms with respect to the OTB 2013 database



(b) Average success and precision curves of various algorithms with respect to the OTB 2015 database

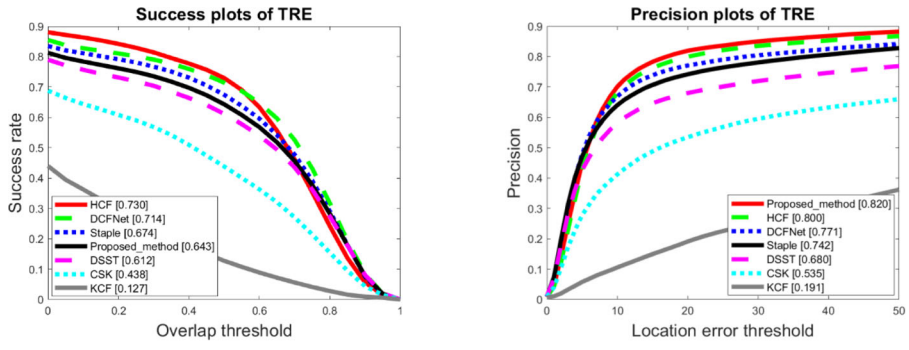
Fig. 7 Average success and precision curves of various algorithms

4.3.3 SRE

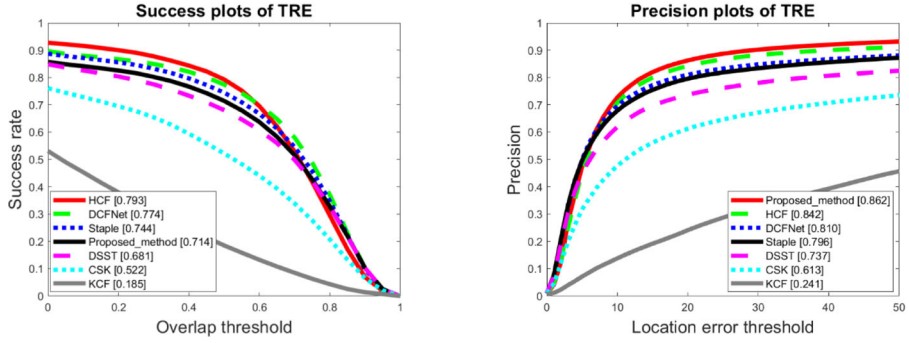
For each test sequence, the position and size of the initial target box were changed based on the ground truth. In the setting process of the position deviation, a rectangular area with a target size of 10% was constructed with the center of the ground truth as the center, and the spatial position was randomly selected in the area with a random frequency of 20% of the area size. In the process of setting the scale change, the scale change should be 0.8, 0.9, 1.1 and 1.2 times the accurate values.

As shown in Fig. 9, for the OTB2013 dataset, when the OR is less than 0.4, the average success rate of the algorithm in this paper is equal to DCF Net and ranks second only to the HCF. When the OR is greater than 0.4, the average success rate of the algorithm in this paper is higher than the rates of other algorithms. In terms of average accuracy, when the location error threshold is less than 19, the performance of the algorithm in this paper is in an absolute advantage. When the location error threshold is less than 19, the performance of the algorithm in this paper is similar to that of the HCF, which ranks first. In the OTB2015 dataset, the average success rate and average accuracy of the algorithm proposed in this paper also decreased but the difference was less than that for the algorithm that ranked first.

Integrating different datasets, average success rate and average accuracy evaluation indices ensures that the proposed method has a competitive performance.



(a) Average success and precision curves of various algorithms with respect to the OTB 2013 database



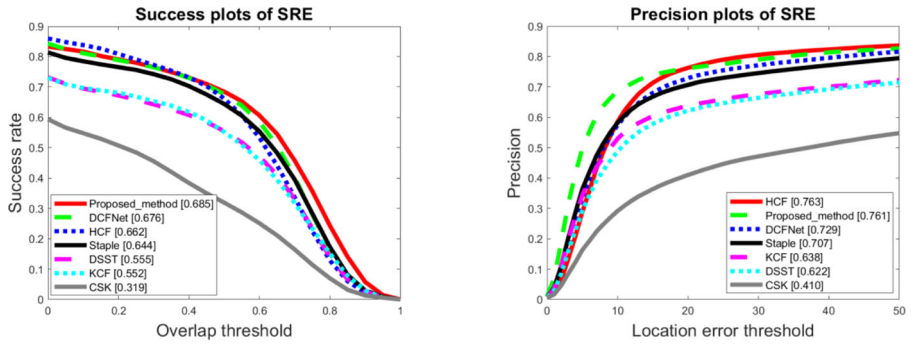
(b) Average success and precision curves of various algorithms with respect to the OTB 2015 database

Fig. 8 Average success and precision curves of various algorithms

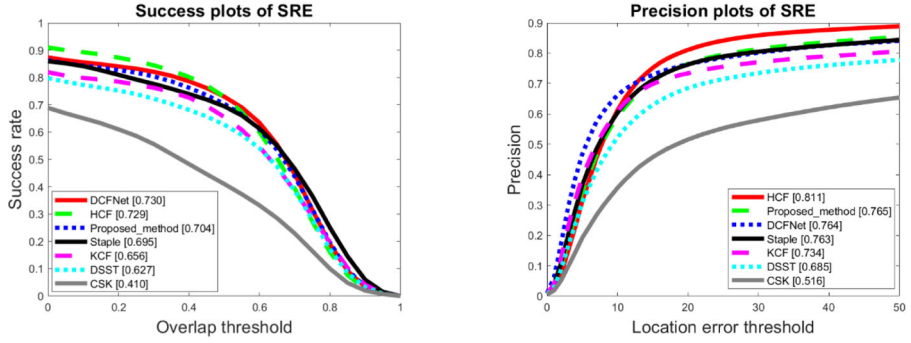
4.3.4 Real-time performance of the algorithm

The main objective of this study is to provide a suitable vehicle tracking algorithm for real-time multichannel video analysis in intelligent transportation. The real-time processing performance of the algorithm is important. The timeliness of object tracking, an important part of a real-time monitoring system, is very important. Table 1 summarizes the comparison of seven trackers in terms of processing speed. The comparison data show that the tracker designed in this study notably outperforms the deep network–CF-combined trackers in terms of processing speed. The relative acceleration rate of the proposed algorithm is 150% of that of the original DCFNet algorithm, and the speed of the proposed algorithm is near 14 times higher than that of the HCF algorithm. The advantage of the proposed algorithm can be mainly attributed to the capacity of the lightweight network designed in this study to learn compact, distinct features.

To comprehensively evaluate the timeliness and tracking precision indices of the tracking algorithms, a real-time multichannel video analysis application is applied as an example for comparison. If the real-time video processing requirement for each channel is 24 FPS, then an average processing speed that is higher than 96 FPS is required for real-time video analysis of four or more channels. The test results in Table 1 show that three algorithms satisfy this application, namely, CSK, KCF, and the proposed algorithm. The tracking precision indices shown in Figs. 7, 8, and 9 show that the proposed algorithm in this study is superior to the other two algorithms.



(a) Average success and precision curves of various algorithms with respect to the OTB 2013 database



(b) Average success and precision curves of various algorithms with respect to the OTB 2015 database

Fig. 9 Average success and precision curves of various algorithms

5 Conclusions

In this study, an object tracker–detector combined with an object tracking algorithm was proposed for tracking vehicles in traffic scenes. For object detection, a detection box merge strategy was used to prevent YOLO from detecting an object more than once or partially detecting an object. For the tracker design, a deep feature-based CF tracker was designed, and for tracker–detector integration, a tracker was first used to predict the location of an object in the subsequent frame. The tracking quality was evaluated based on the PSR. For trackers with relatively poor tracking quality or that have failed to match with the observed values in multiple consecutive frames, a spatial location constraint was applied to correct the predicted locations. Objects that failed to be tracked due to blocking were retracted based on the correlations between the spatial location, moving direction, and historical features. Through experiments, the proposed multiobject tracking algorithm was found to be capable of steadily and continuously tracking objects in traffic scenes and retracking blocked objects.

Table 1 Comparison of various trackers in terms of processing speed

Tracker	**Source**	**Average processing speed (frames per second)**
HCF	European Conference on Computer Vision 2012	269
KCF	arXiv 2014	172
DSST	British Machine Vision Conference 2015	24
Staple	Computer Vision and Pattern Recognition 2016	80
HCF	International Conference on Computer Vision 2015	11
DCFNet	arXiv 2017	60
Proposed method	This study	150

Abbreviations
YOLO: You Only Look Once; DBT: Detection-based tracking; DFT: Detection-free tracking; CNN: Convolutional neural network; CF: Correlation filter; IOU: Intersection over union; HOG: Histogram of oriented gradients; HCF: Hierarchical convolutional feature; KCF: Kernelized correlation filter; LRN: Local response normalization; ReLU: Rectified linear unit; 2D: Two-dimensional; CSK: Circulant structure kernel; OPE: One-pass evaluation; TRE: Time robustness; SRE: Spatial robustness

Acknowledgements
Thanks for the help of reviewers and editors.

Authors' contributions
All authors took part in the discussion of the work described in this paper. All authors read and approved the final manuscript.

Funding
This research was funded by the National Natural Science Foundation of China (41671441, 41531177, U1764262). This research was funded by the National Key Research and Development Project (2018YFB1600600).

Availability of data and materials
Please contact author for data requests.

Competing interests
The authors declare that they have no competing interests.

Author details
[1]State Key Laboratory of Information Engineering in Surveying, Mapping, and Remote Sensing, Wuhan University, Wuhan 430079, China. [2]School of Electronic Information and Communication, Huazhong University Of Science And Technology, Wuhan 430074, China. [3]Beijing Gaocheng Technology Development Co. LTD, Beijing 100043, China. [4]Shandong Highway Traffic Volume Survey and Management Institute, Shandong 255000, China.

Received: 26 September 2019 Accepted: 13 April 2020
Published online: 28 April 2020

References
1. Luo W, Xing J, Milan A, et al. Multiple object tracking: a literature review[J]. 2014.
2. I. Leang, S. Herbin, B. Girard, et al., On-line fusion of trackers for single-object tracking[J]. Pattern Recognition **74**, 459–473 (2018)
3. Z. Chen, X. You, B. Zhong, et al., Dynamically modulated mask sparse tracking[J]. IEEE transactions on cybernetics **47**(11), 3706–3718 (2016)
4. Hua S , Kapoor M, Anastasiu D C. Vehicle Tracking and Speed Estimation from Traffic Videos[C]// 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). IEEE, 2018: 153-160
5. Liu S , Wang S , Shi W , et al. Vehicle tracking by detection in UAV aerial video[J]. SCIENCE CHINA Information Sciences, 2019, 62(2): 24101
6. H.S. Song, Y. Li, J. Yang, Vehicle object tracking method based on highway scenario [J]. Computer Systems & Applications **28**(6), 82–88 (2019)
7. Xu Y , Wang J. A unified neural network for object detection, multiple object tracking and vehicle re-identification[J]. arXiv preprint arXiv:1907.03465, 2019.
8. Lipton A, Fujiyoshi H, Patil R. Moving target classification and tracking from real-time video [J]. In Pro of the 1998 DAPA image understanding workshop (IUW'98), 1998.
9. Stauffer C, Grimson W E L. Adaptive background mixture models for real-time tracking[C]// IEEE Computer Society Conference on Computer Vision & Pattern Recognition. IEEE Xplore, 1999.
10. Barnich O, Droogenbroeck M V. ViBE: A powerful random technique to estimate the background in video sequences[C]// 2009 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2009.
11. Viola P, Jones M. Rapid Object Detection using a Boosted Cascade of Simple Features[C]// null. IEEE Computer Society, 2001.
12. Dalal N, Triggs B. Histograms of Oriented Gradients for Human Detection[C]// null. IEEE Computer Society, 2005.
13. Fang Y , Wang C , Yao W , et al. On-road vehicle tracking using part-based particle Filter[J]. IEEE Transactions on Intelligent Transportation Systems, 2019:1-15.
14. Szegedy C, Toshev A, Erhan D. Deep neural networks for object detection[C]//Advances in neural information processing systems. 2013: 2553-2561.
15. Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2014: 580-587.
16. Girshick R. Fast r-cnn[C]//Proceedings of the IEEE international conference on computer vision. 2015: 1440-1448.
17. Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[C]// Advances in neural information processing systems. 2015: 91-99.
18. Dai J, Li Y, He K, et al. R-fcn: Object detection via region-based fully convolutional networks[C]//Advances in neural information processing systems. 2016: 379-387.
19. Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.
20. Yan, C., Li, L., Zhang, C., et al. Cross-modality bridging and knowledge transferring for image understanding. IEEE Transactions on Multimedia, 2019.

21. Yan, C., Gong, B., Wei, Y., et al. Deep multi-view enhancement hashing for image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020:370–383.
22. Yan, C., Shao, B., Zhao, H., et al. (). 3D Room layout estimation from a single RGB image. IEEE Transactions on Multimedia, 2020.
23. J. Redmon, A. Farhadi, *YOLOv3: An Incremental Improvement[J]* (2018)
24. Ge Baoyi, Zuo Xianzhang, Hu Yongjiang. Review of visual object tracking technology. Journal of image and Graphics. v. 23; No.268.08(2018):5-21.
25. Chen, Z., Hong, Z., Tao, D.: An experimental survey on correlation filter-based tracking. CoRR abs/1509.05520, http://arxiv.org/abs/1509.05520 (2015)
26. Danelljan M, Hager G, Shahbaz Khan F, et al. Convolutional features for correlation filter based visual tracking[C]// Proceedings of the IEEE International Conference on Computer Vision Workshops. 2015: 58-66.
27. Ma C, Huang J B, Yang X, et al. Hierarchical convolutional features for visual tracking[C]//Proceedings of the IEEE international conference on computer vision. 2015: 3074-3082.
28. Wang Q, Gao J, Xing J, et al. Dcfnet: Discriminant correlation filters network for visual tracking[J]. arXiv preprint arXiv: 1704.04057, 2017.
29. Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]. Cvpr, 2015.
30. Krizhevsky A, Sutskever I, Hinton G. ImageNet classification with deep convolutional neural networks[J]. Advances in neural information processing systems, 2012, 25(2).
31. Jie H, Li S, Albanie S, et al. Squeeze-and-excitation networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, PP(99):1-1.
32. J.F. Henriques, R. Caseiro, P. Martins, et al., High-speed tracking with kernelized correlation filters[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(3), 583–596 (2015)
33. D. S. Bolme, J. R. Beveridge, B. A. Draper and Y. M. Lui, Visual object tracking using adaptive correlation filters, 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, 2010, pp. 2544-2550.
34. O. Russakovsky, J. Deng, H. Su, et al., ImageNet large scale visual recognition challenge[J]. International Journal of Computer Vision **115**(3), 211–252 (2014)

Publisher's Note