TRR
JOURNAL OF THE TRANSPORTATION RESEARCH BOARD

Research Article

Transportation Research Record
1–15

DOI: 10.1177/0361198120912742
journals.sagepub.com/home/trr
SAGE

Vehicle Counting System using Deep Learning and Multi-Object Tracking Methods

Haoxiang Liang[1], Huansheng Song[1], Huaiyu Li[1], and Zhe Dai[1]

Abstract

Using deep learning technology and multi-object tracking method to count vehicles accurately in different traffic conditions is a hot research topic in the field of intelligent transportation. In this paper, first, a vehicle dataset from the perspective of highway surveillance cameras is constructed, and the vehicle detection model is obtained by training using the You Only Look Once (YOLO) version 3 network. Second, an improved multi-scale and multi-feature tracking algorithm based on a kernel correlation filter (KCF) algorithm is proposed to avoid the KCF extracting single features and single-scale defects. Combining the intersection over union (IoU) similarity measure and the row-column optimal association criterion proposed in this paper, matching strategy is used to process the vehicles that are not detected and wrongly detected, thereby obtaining complete vehicle trajectories. Finally, according to the trajectory of the vehicle, the traveling direction of the vehicle is automatically determined, and the setting position of the detecting line is automatically updated to obtain the vehicle count result accurately. Experiments were conducted in a variety of traffic scenes and compared with published data. The experimental results show that the proposed method achieves high accuracy of vehicle detection while maintaining accuracy and precision in tracking multiple objects, and obtains accurate vehicle counting results which can meet real-time processing requirements. The algorithm presented in this paper has practical application for vehicle counting in complex highway scenes.

Literature Review

Vehicle Object Detection

Traditional vehicle detection uses digital image processing methods, such as background subtraction (*1*), continuous video frame difference (*2*), and optical flow (*3*). The effectiveness of these methods depends on the background model and they are susceptible to circumstances such as lighting. In recent years, there have been excellent results in the use of deep learning networks for vehicle detection. For this purpose, there are many vehicle objects datasets, such as the Common Objects in Context (COCO) (*4*) dataset covering 80 common objects, including car, bus, truck, motorcycle, bicycle, and other vehicle objects, which is widely used in many fields. The Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago (KITTI) benchmark dataset (*5*), which includes images of a variety of traffic scenes, is used for self-driving cars and can solve problems such as three-dimensional object detection and tracking. The Traffic and Congestions (TRANCOS) dataset (*6*) used a surveillance camera to capture 1,244 pictures of traffic jams. However, most of the vehicles are occluded and no vehicle type is recorded, therefore TRANCOS cannot be widely used.

Deep learning networks involve a two-step method to generate candidate boxes for samples through multiple algorithms and then classify the samples by convolutional networks. Region-convolutional neural networks (R-CNN) (*7*), Fast R-CNN (*8*), and Faster R-CNN (*9*) use the two-step method to improve feature extraction and selection in convolutional networks. The two-step method has high detection accuracy, but the detection speed is slow. The one-step method is to convert the positioning problem of the object directly into a regression problem, which is fast but sacrifices detection accuracy. The single shot multibox detector (SSD) algorithm (*10*) uses a set of default anchors with different aspect ratios to locate objects accurately; the SSD algorithm combines multiple feature maps at different resolutions. The prediction

[1]School of Information Engineering, Chang'an University, Xi'an, Shaanxi Province, P.R. China

Corresponding Author:
Huansheng Song, hshsongtrb@163.com; hshsong@chd.edu.cn

results can handle objects of various scales. The You Only Look Once (YOLO) (*11*) series of algorithms divide the image into a fixed number of grids, and each grid is responsible for predicting objects whose center points fall into it. Based on YOLO and YOLO version 2 (*12*), YOLO version 3 (*13*) uses logistic regression to regress the box confidence to determine whether the intersection over union (IoU) of the a priori box and the actual box is greater than 0.5. If more than one a priori box satisfies the condition, only the largest a priori box of the IoU is taken. Meanwhile, YOLOv3 (*13*) uses the two-class cross-entropy loss to calculate the category loss so that the algorithm can handle multiple label problems for the same object. In the final object prediction, YOLOv3 uses three different scales to predict the object in the image and achieves fast detection speed. In summary, the vehicle dataset is used for training the deep learning network, and the vehicle detection model obtained performs well.

Object Tracking

Object tracking is an important task of any intelligent transportation system (ITS). Past research on single-object tracking has mainly used statistical methods and feature point matching, such as mean shift tracking algorithm (*14*) or Kalman filter algorithm (*15*). The Kalman filter algorithm (*15*) predicts the object's position by describing the state of the object to track the object. The minimum output sum of squared error filter (MOSSE) (*16*) algorithm uses the correlation filter for object tracking. Exploiting the circulant structure of tracking-by-detection with kernels (CSK) (*17*) the algorithm introduces the cyclic matrix based on MOSSE, which increases the training sample. After that, Henriques proposed the kernelized correlation filters (KCF) (*18*) algorithm, which can reach more than 100 Frames Per Second (FPS), and Danelljan proposed the color name (CN) (*19*) based on CSK, which uses color information to describe the object features and proves that color information has an important role in object tracking. The discriminative scale space tracker (DSST) (*20*) and scale adaptive multiple feature (SAMF) (*21*) algorithms were proposed to solve the problem of object scale change. The above correlation filtering based methods are all affected by the boundary effect, and the spatially regularized discriminative correlation filter (SRDCF) (*22*) adds spatial regularity to solve the boundary effect problem. The correlation filtering of KCF (*18*) and DSST (*20*) algorithms is to extract the histogram of oriented gradients (HOG) features, which can adapt to illumination change, and the characteristics of the color histogram are robust to object deformation. Finally, the Staple algorithm (*23*), combining the advantages of color histogram and HOG, was proposed.

The early research on multi-object tracking was based on object detection results, which were manually correlated to form the object trajectories. Currently, multi-object tracking based on data association methods is widely used. The multi-layer tracking framework proposed by Nevatia and colleagues (*24*) is a classic example. The Markov decision processes multi-object tracking algorithm (*25*) uses machine learning to strengthen the parameters of multi-object matching and construct the excitation function to analyze the multi-object tracking state. The object tracking algorithm based on deep learning has also become a hot topic in the field. The learning continuous convolution operators for visual tracking (C-COT) algorithm (*26*) uses the convolutional network to extract the object features. It combines relevant filtering ideas to win the Visual Object Tracking (VOT) championship in 2016. The efficient convolution operators for tracking (ECO) algorithm (*27*) accelerates the C-COT algorithm, but the tracking speed of the algorithm is only 8 FPS. Therefore, the object tracking method based on deep learning is still in the developing research stage, and the tracking method using correlation filtering is more suitable for practical use.

Methods

The methods used in this paper are shown in Figure 1. First, the video frames from highway surveillance cameras are read. Second, the deep learning network is used to train the vehicle dataset to obtain the vehicle detection model. Third, the multi-object tracking of vehicles is carried out. Using the improved KCF tracking method proposed in this paper, the vehicle trajectory data is correlated and matched. The processing strategy is given for the different matching of the vehicle and the trajectory. Fourth, the trajectory is analyzed. The direction of travel of the vehicle and the position of the detection line are automatically determined and updated. Finally, the number of trajectories of different types of vehicles passing through the detection line is counted.

Vehicle Detection

A total of 11,129 surveillance images were collected from 27 monitoring video cameras on China's G60 highway, including different scenes, different camera angles, different camera resolutions, and different weather conditions, with a resolution of 1920×1080. The vehicle dataset uses an annotation tool to mark the three types of vehicle objects for cars, buses, and trucks, respectively. Of the vehicle dataset, 80% is used as a training set and 20% is used as a test set, and the vehicle dataset is trained using the YOLOv3 object detection algorithm to obtain the vehicle detection model. The YOLOv3 network adopts

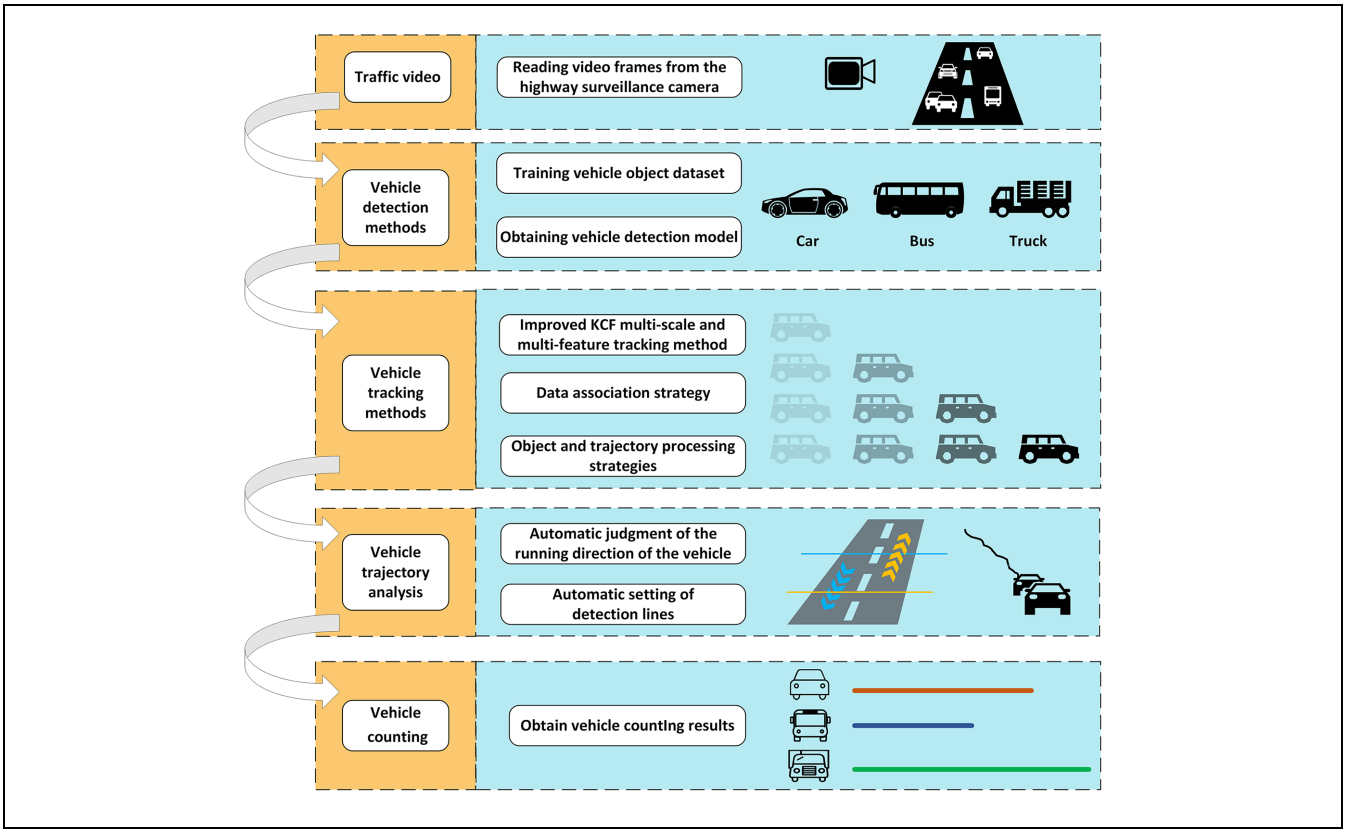

Figure 1. Schematic diagram of the methods used in this study. KCF = kernelized correlation filters.

three output scales: the shallow features detect small objects, the deep features detect large objects, and finally, the results of the three scale branches are combined to obtain the detection results. Therefore, the selection of the YOLOv3 network for vehicle detection can solve the problem of low detection accuracy caused by scale changes when vehicles are in motion from the perspective of the highway surveillance camera. Meanwhile, the network can meet the needs of real-time processing in highway settings.

Vehicle Multi-Object Tracking

Multi-object tracking is based on vehicle detection results. First, the object is associated with the trajectory, then different processing strategies are performed to generate continuous object trajectories.

The tracking module is used to obtain the vehicle trajectory. According to the vehicle trajectory, the same vehicle can be continuously identified to ensure that it is only counted once in the same scene, and the same vehicle is not considered as a newly generated object. Meanwhile, according to the trajectory, the moving area of the vehicle can be determined, so that the position of the detection line can be set dynamically to ensure that the detection line is in a suitable position within the moving area. This can avoid inaccurate counting due to the position of the detection line deviating from the moving area. Finally, the driving direction of each vehicle can be distinguished according to the trajectory so that the number of vehicles in different directions can be counted. This module refines the work of vehicle counting and guarantees the accuracy of vehicle counting results.

Improved KCF Multi-Scale and Multi-Feature Tracking Method. A serious drawback of the KCF algorithm (*18*) is that it can only use a single scale to track an object. After the object is determined, the scale of the tracking bounding box no longer changes. From the perspective of highway monitoring discussed in this paper, if the vehicle object becomes smaller, the tracker template will be affected by a large amount of background information. If the vehicle object becomes larger, the tracker template can only update the local information of the vehicle object. In the above two cases, the tracking will fail due to the inability to collect the vehicle object template accurately. Meanwhile, the HOG feature extracted by the KCF algorithm counts the object gradient direction information without comprehensively considering the change of the object during the movement.

For the single-scale problem of the KCF algorithm, the SAMF (*21*) and DSST (*20*) algorithms add multi-scale changes. The SAMF algorithm (*21*) introduces the scale pool, and uses different scales for sample collection. For the detection samples of different scales, the sample detection mechanism of KCF is used to calculate the maximum response value of the object. The DSST algorithm (*20*) adds a scale filter based on the position filter of the KCF algorithm. Due to the introduction of multiple scales, however, the sample detection speed is slow. For the single-feature problem of the KCF algorithm, SAMF (*21*) combines gray feature, HOG feature, and CN feature to enhance the ability to describe the object, and superimposes the above features to improve the accuracy of object tracking. The CN feature is a description of the local color feature information. For the RGB image, the color feature can be converted to the CN feature using the mapping table provided by Van De Weijer et al. (*28*). In addition, the Staple algorithm (*23*) calculates the response values of the HOG feature and the color histogram, respectively. The Staple algorithm then fuses the response values and selects the maximum response value as the predicted position of the object.

Based on the KCF algorithm (*18*) and combining the advantages of the DSST (*20*), SAMF (*21*), and Staple (*23*) algorithms, in this study an improved KCF multi-scale and multi-feature tracking method is designed to solve the vehicle tracking problem in complex highway scenes. First, the grayscale feature, the HOG feature, and the CN feature of the vehicle object bounding box were extracted. The response value calculation method of the KCF algorithm (*18*) was then used to calculate the response value of each feature. The color histogram response is calculated by using the same idea as the Staple algorithm (*23*). Second, the above features are vector added to calculate the final response value, and the object candidate box corresponding to the maximum response value is selected as the predicted position of the object. Finally, the maximum correlation filter of the DSST algorithm (*20*) is used to calculate the response value for scale estimation. Therefore, the prediction of the vehicle's object position and scale is completed.

Data Association Method. The proposed data association method uses the IoU similarity measure. When the vehicle object is moving, the vehicle detection rectangular box of consecutive frames will not change drastically. The IoU similarity measure uses this feature to find potential matching pairs in different frames correctly. Using the row-column optimal association criterion, the correct association pairs are then accurately found. Finally, according to different correlation results, different trajectory processing strategies are adopted to ensure continuous tracking of the same vehicle and avoid vehicle trajectory fragmentation. Therefore, when there are many vehicle objects in the highway scene, and the objects are similar in size and type, the proposed association method can correctly correlate the positions of the same vehicle in consecutive frames. This association method provides a better solution for vehicle object tracking in a wide range of traffic scenes.

(a) IoU similarity measure: For the same vehicle object, there will be a large overlap in the detection boxes of consecutive frames. The detection result $\mathrm{det}B_i^k$ indicates the vehicle box position obtained by the ith vehicle object in the kth frame. The existing frame vehicle trajectory t_{j}^{k-1} includes the unique trajectory ID, the vehicle box position in each frame before kth frame, the trajectory direction, and the image of the previous frames. Vehicle detection result detections $= \{\mathrm{det}B_1^k,\ \mathrm{det}B_2^k, \ldots, \mathrm{det}B_i^k, \ldots, \mathrm{det}B_{\mathrm{m}}^k\}$ of the kth frame and the vehicle detection results of the previous frames form the vehicle trajectory traces $= \{t_1^{k-1}, t_2^{k-1}, \ldots, t_{\mathrm{j}}^{k-1}, \ldots, t_{\mathrm{n}}^{k-1}\}$. The calculation for the IoU similarity measure is shown in Equation 1.

$$\mathrm{IoU}(\mathrm{det}B_i^k, \mathrm{tra}B_{\mathrm{j}}^{k-1}) = \frac{\mathrm{area}(\mathrm{det}B_i^k \cap \mathrm{tra}B_{\mathrm{j}}^{k-1})}{\mathrm{area}(\mathrm{det}B_i^k \cup \mathrm{tra}B_{\mathrm{j}}^{k-1})} \quad (1)$$

where $\mathrm{det}B_i^k$ indicates the ith object box of the kth frame vehicle detection result. $\mathrm{tra}B_{\mathrm{j}}^{k-1}$ represents the last frame object box in the existing jth trajectory t_{j}^{k-1}. When the overlap between two object boxes is large, the IoU similarity value is high. The IoU similarity measure is used to obtain the correlation matrix A_{mn} of the object and the trajectory. A_{mn} is shown in Equation 2. Each row in A_{mn} represents a similarity value of the current object detection box and the object box in each tracking trajectory, and each column represents a similarity value of the object box in each tracking trajectory and the current frame. For convenience, the IoU similarity value in the correlation matrix is denoted by D_{ij} in Equation 3.

$$A_{mn} = \begin{bmatrix} D_{11} & D_{12} & \ldots & D_{1(n-1)} & D_{1n} \\ D_{21} & D_{22} & \ldots & D_{2(n-1)} & D_{2n} \\ \vdots & \vdots & D_{ij} & \vdots & \vdots \\ D_{(m-1)1} & D_{(m-1)2} & \ldots & D_{(m-1)(n-1)} & D_{(m-1)n} \\ D_{m1} & D_{m2} & \ldots & D_{m(n-1)} & D_{mn} \end{bmatrix} \quad (2)$$

$$D_{ij} = (1 - \mathrm{IoU}_{(i,j)}) \quad (3)$$

where $\mathrm{IoU}_{(i,j)}$ represents the IoU similarity measure of the ith object detection box and the jth tracking trajectory object box.

(b) Row-column optimal association criterion: To associate the same vehicle object in different frames, it is necessary to search for the best association pair in the correlation matrix A_{mn}. First, the IoU similarity value D_{ij} is threshold-constrained to obtain the candidate association pair. Second, the row-column optimal association criterion is used to select the best correlation pair. That is, the IoU similarity value of the ith row belongs to the jth column, and the IoU similarity value of the jth column belongs to the ith row, as shown in Equation 4.

$$\begin{cases} D_{ij} \leqslant \text{Thresh} \\ I = \min(j_{\text{col}}) \\ J = \min(i_{\text{row}}) \end{cases} \tag{4}$$

where I represents the optimal line number and J represents the optimal column number in the correlation matrix A_{mn}. The same vehicle object has a large box overlap value between consecutive frames, and the optimal IoU similarity measure D_{ij} is small. Here, the minimum threshold Thresh of the IoU similarity measure is set to 0.5.

Object and Trajectory Processing Strategies. According to the data association result A_{mn} of the vehicle object and the vehicle trajectory, different methods are used for continuous object tracking, including the following three situations in Figure 2.

(i) If the kth frame objects and the k–1th frame trajectories are successfully matched, the k +1th frame is continuously read for vehicle detection, and the trajectories are updated.

(ii) If the trajectory matching fails, that is, the tracking trace does not match the detection box, then occlusion detection is performed. First, the improved KCF multi-scale and multi-feature tracking method proposed in this paper is used. In the unobstructed case, the maximum response value is kept above 0.7. When the object is occluded, the tracker template is continuously updated, the object features are continuously changed, and the maximum response value is significantly reduced. The response threshold is set to 0.5. If the value is higher than the response threshold of 0.5, there is no occlusion. The improved KCF algorithm is used to predict the vehicle position. It is considered that the vehicle miss detection causes the trajectory matching failure. If the value is less than the response threshold of 0.5, it is determined that occlusion is currently occurring, the position of the occlusion object is predicted by the Kalman

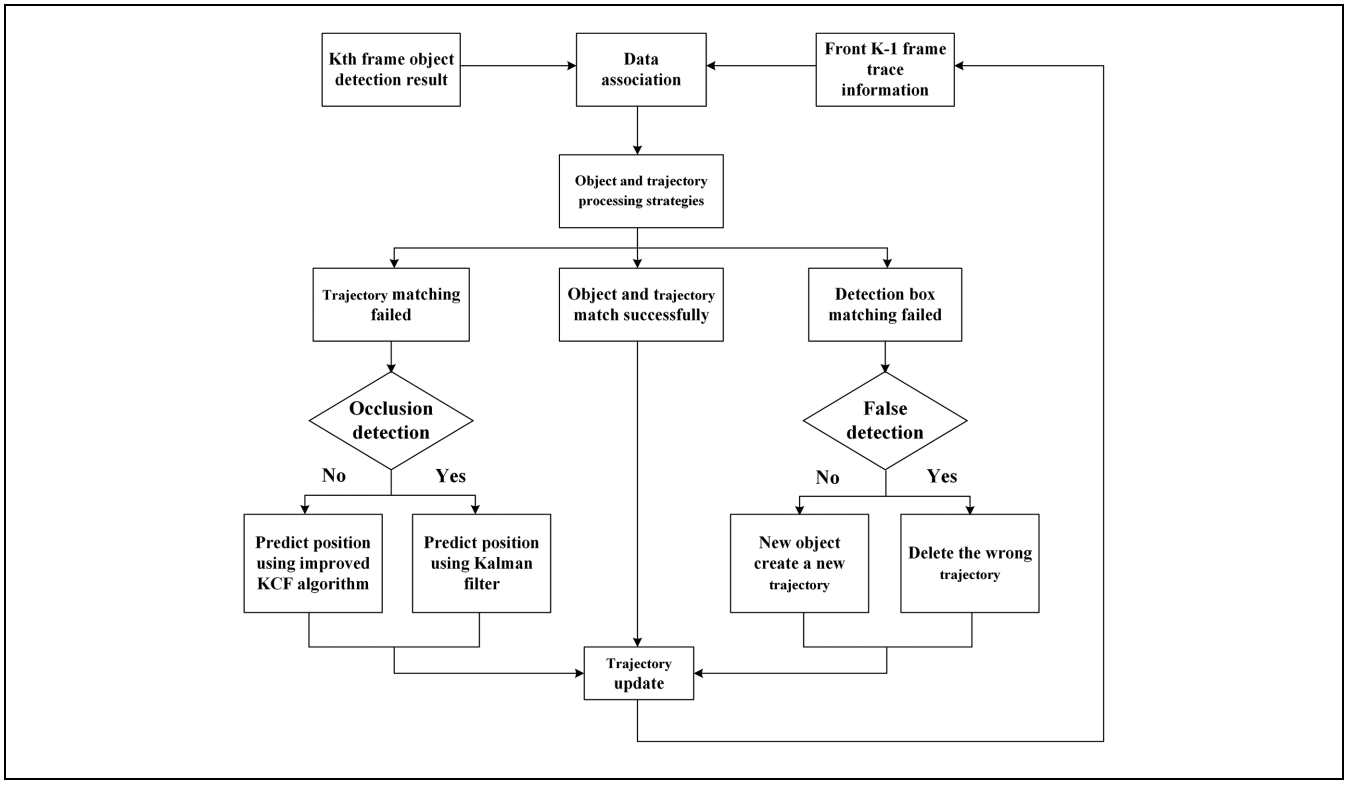

Figure 2. Vehicle tracking flow chart. KCF = kernelized correlation filters.

filter algorithm (*15*), and the state of the kth frame is predicted by the state value of the k–1th frame. When the vehicle leaves the obstruction, the vehicle is re-detected, and the data association method is used again to make the vehicle object and the trajectory match successfully, and the tracking is continued.

(iii) If the kth frame object detection box fails to match the k–1th frame trajectory, that is, the detection box does not match the corresponding trajectory, the false detection is performed. The object box is detected and tracked for three consecutive frames. If the object can be detected in three consecutive frames and successfully matched by the data association method, the detection box is a newly appearing object, and the data will be performed in the next frame. Otherwise the object is considered to be misdetected and the trajectory is deleted.

Vehicle Trajectory Analysis

Initial Setting of the Vehicle Counting System

(a) Automatic judgment of the direction of travel of the vehicle.

In most countries, highway lanes are divided into uplink and downlink. For example, in China, highways are marked with mileage stations, the station number increases in the upward direction, and the station number decreases in the downward direction. The highway surveillance camera is generally installed on a certain side of the road. For the PTZ surveillance camera in the highway scene studied in this paper, the camera angle is adjustable. The judgment of the vehicle's direction of travel has four cases, as shown in Figure 3.

For a certain highway section, the uplink and downlink directions remain unchanged. There are two different viewing angles of the surveillance video due to the camera rotation. To judge the vehicle's direction of travel, the vehicle trajectory is used to calculate the angle between the trajectory and the image coordinate. The calculation is as follows:

$$\Delta x = \text{last}X - \text{first}X \tag{5}$$

$$\Delta y = \text{last}Y - \text{first}Y \tag{6}$$

where last$X$ and last$Y$ indicate the end position of the trajectory, and first$X$ and first$Y$ indicate the start position of the trajectory. The angle θ is determined by Equation 7.

$$\theta = \text{atan2}\begin{cases} \arctan(\Delta y/\Delta x) & \Delta x > 0 \\ \arctan(\Delta y/\Delta x) & \Delta y \geq 0, \Delta x < 0 \\ \arctan(\Delta y/\Delta x) & \Delta y < 0, \Delta x < 0 \\ +\pi/2 & \Delta y > 0, \Delta x = 0 \\ -\pi/2 & \Delta y < 0, \Delta x = 0 \\ \text{undefined} & \Delta y = 0, \Delta x = 0 \end{cases} \tag{7}$$

$$\text{Trajectory direction}\begin{cases} 1 & |\theta| \geq 0°, |\theta| < 90° \\ 0 & |\theta| = 180°, \text{frameLen} < 15 \\ -1 & |\theta| \geq 90°, |\theta| < 180° \end{cases} \tag{8}$$

where 0 means that the current trajectory direction is undeterminable, 1 means uplink, and –1 means downlink. frameLen represents the length of the trajectory measured by the frames. When the trajectory is shorter than 15 frames, the direction judgment is not performed.

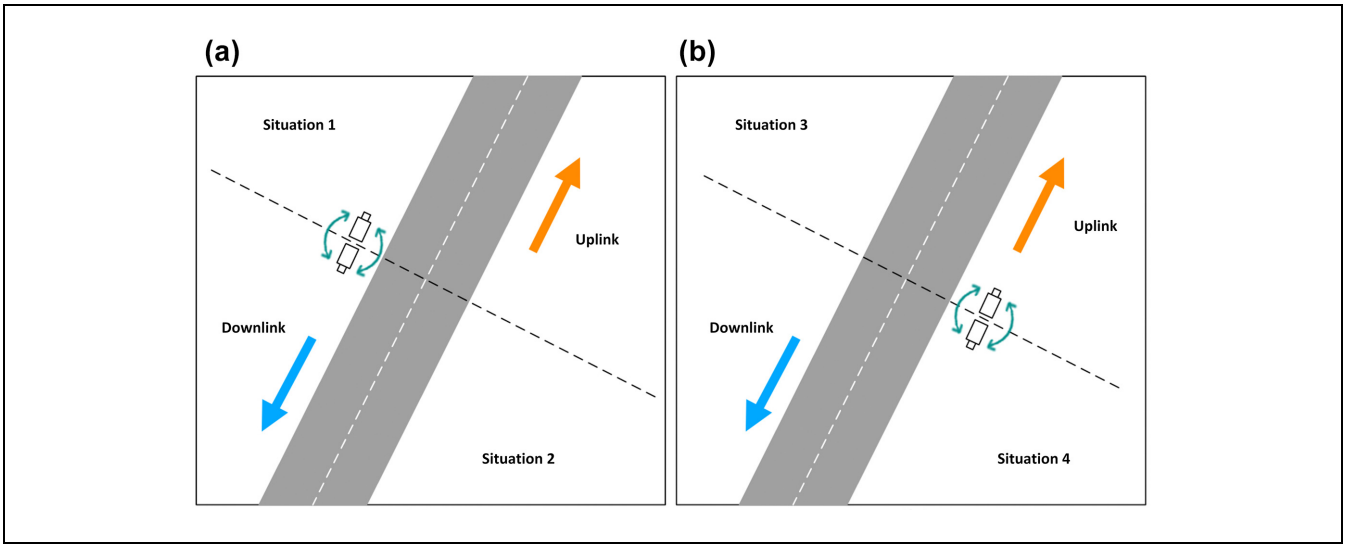

Figure 3. Vehicles' direction of travel at different camera angles; the camera is mounted on the (*a*) left and (*b*) right sides of the road.

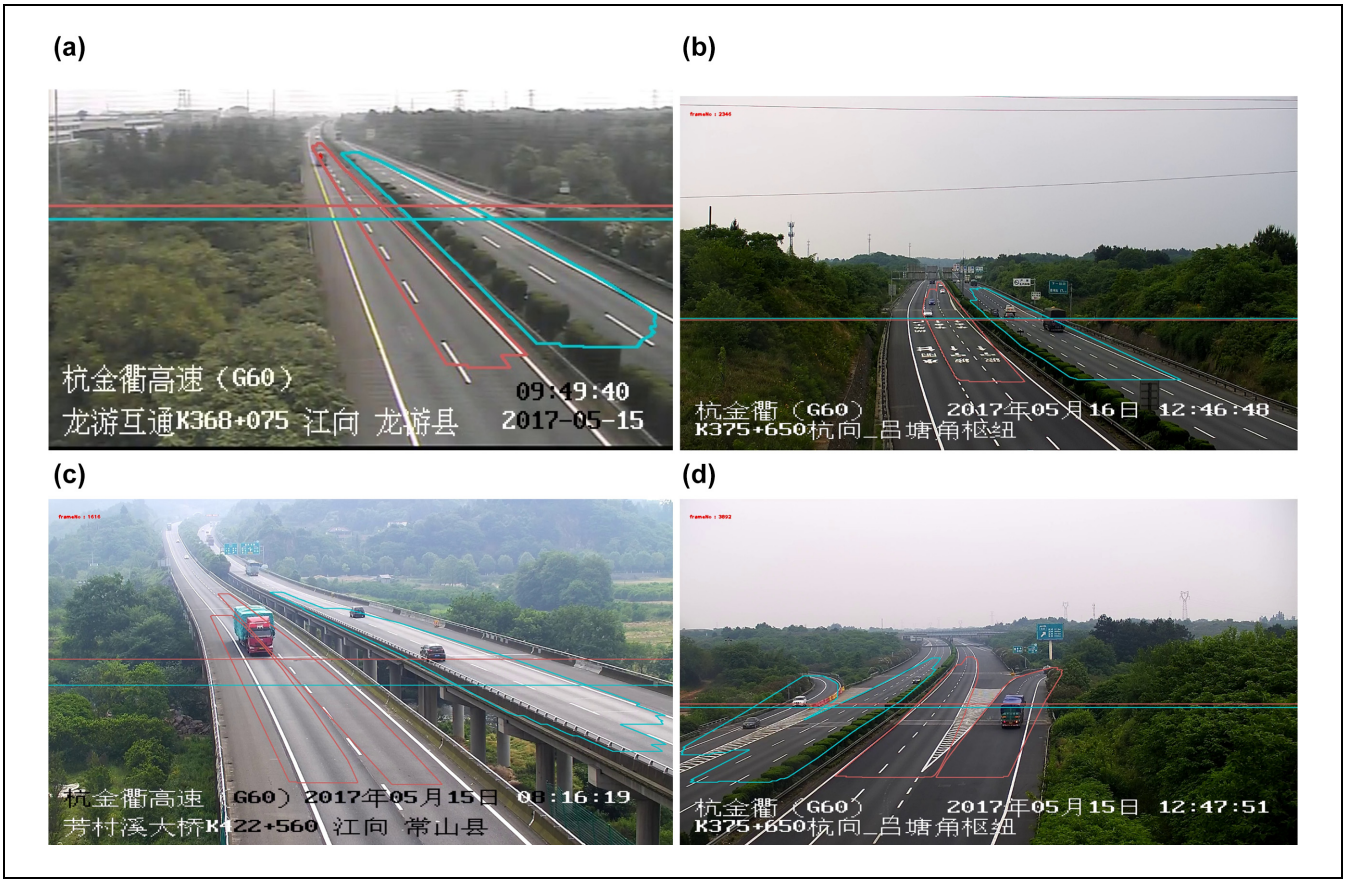

Figure 4. Automatic setting of the detection line: (*a–d*) show the detection line setting results in different scenes.

(b) Dynamic setting of the detection line.

For different monitoring angles, a different position of the detection line is required. The moving areas of the uplink and downlink directions are determined using the object trajectories on the image. The centroid coordinate values of the connected areas are taken as the detection line positions in each direction. The moving areas and detection lines are automatically updated according to the vehicle trajectories.

Figure 4 shows the results of the detection line setting in different scenes. The irregular areas defined by the red and blue coils indicate the vehicle motion areas in the uplink and downlink directions, respectively. The red horizontal line and the blue horizontal line indicate the uplink and downlink direction detection lines, respectively.

Vehicle Counting

The number of vehicles passing the detection line in a certain period is counted. The calculation for vehicle counting result p is Equation 9.

$$p = \frac{N}{t} \tag{9}$$

where t is the length of the monitoring period and N is the number of vehicles passing the detection line during the monitoring period.

In this study, the vehicles are divided into three categories: Car, Bus, and Truck. The number of type i vehicles in the unit time is n, and the total number of all types of vehicles is N. The counting of the type i vehicle in time t is shown in Equation 10.

$$p_i = \frac{n}{N} * p \tag{10}$$

Results

The experiments used an Intel i7-6800k CPU with NVIDIA GeForce GTX 1080Ti GPU graphics. The highway videos used in the experiments are divided into four different scenes, as shown in Table 1.

Vehicle Object Detection Results

First, set the training parameters of the YOLOv3 network: 80% of the vehicle dataset as the training set, 20% as the test set, the batch size is set to 16, the network

Table 1. Experimental Video Information

Scenes	Resolution	Number of frames
Scene 1	1280×720	21,439
Scene 2	1920×1080	22,481
Scene 3	1920×1080	22,450
Scene 4	1920×1080	22,473

input image size is 832×832, and the anchor size is calculated using the K-Meams++ (*29*) algorithm: [85.1911×90.7127, 7.7366×124.7856, 10.7746×139.1191, 27.8866×44.2784, 38.5279×62.4451, 50.2715×86.2852, 7.4877×11.8028, 11.9581×20.6095, 18.9967×30.4345], the learning rate is set to 0.0001 and the number of iterations is 45,000. Using the loss value calculation method of YOLOv3 (*13*), when the loss drops to 0.2, the training stops, and the vehicle detection model is obtained.

For the vehicle detection model, the test set samples were used to calculate the average precision (AP) and mean average precision (mAP) evaluation of the model. *Recall of detection* refers to the ratio of objects that are correctly detected to the total objects in the test set. *Precision of detection* refers to the ratio of positive samples correctly predicted to the total objects that have been detected.

$$\text{Recall of detection} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{11}$$

$$\text{Precision of detection} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{12}$$

where TP, FN, and FP are the numbers of true positives, false negatives, and false positives, respectively. According to Precision and Recall of detection, the P-R curve of each object category is drawn. AP is the integral value of the P-R curve, and mAP is the average value of APs of all categories.

The vehicle detection model based on the YOLOv3 network is obtained by using the COCO2014 dataset (*4*) and the vehicle dataset constructed in this paper. A comparison was made using the samples from the two datasets. In Table 2, the detection speed of the COCO model is basically the same as the model in this paper, and for large images the processing speed is slower than for small images. Meanwhile, because the dataset contains a large number of vehicle samples, including vehicle images of various scales and angles, the mAP value is higher than the COCO model.

Vehicle Multi-Object Tracking Results

Multi-object tracking was evaluated using distance precision (DP) and overlap precision (OP). In Equation 13, if the distance D between the tracking object center position C_t and the actual object center position C_g is less than ThreshDP, the object meets the DP threshold. DP_result is the ratio of the frames that meet the DP threshold to the video total frames. In Equation 14, the IoU of the tracking object box B_t and the actual object box B_g is calculated. When the IoU value is higher than the ThreshOP, the object meets the OP threshold. OP_result is the ratio of frames that meet the OP threshold to the video total frames.

$$D = \sqrt{(C_g(x) - C_t(x))^2 + (C_g(y) - C_t(y))^2} < \text{ThreshDP} \tag{13}$$

$$\text{IoU} = \frac{\text{area}(B_t \cap B_g)}{\text{area}(B_t \cup B_g)} > \text{ThreshOP} \tag{14}$$

All the tracking test data is shown in Table 3. Data for nine vehicle objects were selected from the VOT2016 dataset for this experiment (*30*), including the vehicle scale changes and deformation. Due to the complexity of the highway, that data cannot cover the deformation caused by vehicles turning and being occluded. Therefore, data for 20 vehicles from the highway scene were collected for experimental analysis. These trajectory data cover different vehicle types, frame lengths, and image resolutions, and consider different vehicle driving conditions, such as vehicle scale changes, deformations, and occlusions. Using these trajectory data the performance of the tracking algorithm in different complex traffic scenes was tested. The vehicle trajectory data has been published in: https://drive.google.com/file/d/

Table 2. Evaluation of Vehicle Detection Model

	Speed		AP			
Model name	1920×1080	1280×720	Car	Bus	Truck	mAP
COCO detection model	26FPS	34FPS	64.70%	84.25%	64.17%	71.04%
Vehicle detection model	24FPS	29FPS	89.35%	87.83%	90.48%	89.22%

Note: AP = average precision; mAP = mean average precision; COCO = Common Objects in Context; FPS = Frames Per Second..

Table 3. Trajectory Data Used in the Experiment

Data source	Data name	Vehicle category	Number of frames	Scale change	Shape change	Cover	Resolution
VOT 2016 (*30*)	BlurCar1	Car	742	Not obvious	Fuzzy	No	640×480
	BlurCar2	Car	585	Not obvious	Fuzzy	No	640×480
	BlurCar3	Car	357	Not obvious	Fuzzy	No	640×480
	BlurCar4	Car	380	Not obvious	Fuzzy	No	640×480
	Car1	Car	1020	Not obvious	No	No	320×240
	Car2	Car	913	Not obvious	No	No	320×240
	Car4	Car	659	Not obvious	No	No	320×240
	Car24	Car	3059	Not obvious	No	No	320×240
	CarScale	Car	252	Obvious	No	No	320×240
Data collected for this study	1	Truck	102	Obvious	Yes	No	704×576
	2	Truck	140	Not obvious	No	No	1920×1080
	3	Truck	200	Obvious	No	No	1920×1080
	4	Car	112	Not obvious	No	Part	1920×1080
	5	Truck	102	Not obvious	No	Part	1920×1080
	6	Car	100	Obvious	No	No	1920×1080
	7	Truck	90	Obvious	Rotational	No	1920×1080
	8	Truck	102	Not obvious	No	Part	1920×1080
	9	Car	60	Not obvious	No	Part	1920×1080
	10	Car	153	Obvious	No	No	1920×1080
	11	Bus	100	Obvious	No	No	1920×1080
	12	Truck	77	Not obvious	No	No	1920×1080
	13	Car	30	Obvious	No	Part	704×576
	14	Car	83	Not obvious	No	No	1920×1080
	15	Bus	80	Obvious	Rotational	No	1920×1080
	16	Car	100	Obvious	No	Part	1920×1080
	17	Truck	120	Obvious	Rotational	No	1920×1080
	18	Truck	90	Not obvious	No	No	1920×1080
	19	Truck	100	Obvious	Rotational	Part	1920×1080
	20	Car	60	Not obvious	No	Total	1920×1080

Note: VOT = Visual Object Tracking.

Table 4. Vehicle Trajectory Test Results on Different Data Sources

	DP_result		OP_result		FPS	
Algorithm name	VOT2016	Self-made data	VOT2016	Self-made data	VOT2016	Self-made data
KCF (*18*)	0.951	0.819	0.693	0.784	174	113
DSST (*20*)	0.954	0.827	0.816	0.847	57	43
SAMF (*21*)	0.952	0.854	0.844	0.850	12	7
Staple (*23*)	0.903	0.851	0.737	0.848	24	18
Proposed method	0.952	0.860	0.843	0.859	24	16

Note: DP = distance precision; DSST = discriminative scale space tracker; FPS = Frames Per Second; KCF = kernel correlation filter; OP = overlap precision; SAMF = scale adaptive multiple feature; VOT = Visual Object Tracking.

1WydfdThXD2s5DQjZQeu7uU9QojXST8m6/view?usp=sharing.

In this paper, four typical correlation filter tracking algorithms are selected as comparative experiments. They were analyzed from three aspects: *DP*_result, *OP*_result, and processing speed. In the experiment, set ThreshDP to 20 and ThreshOP to 0.5. For all object tracking algorithms in the experiment, the parameters provided by the original author are used.

As can be seen from Table 4, for the VOT2016 dataset, the KCF algorithm has a fast processing speed but a low OP. In the tracking of BlurCar1, the Staple tracking fails, so the DP is the lowest. For data collected for this study, the scene is complicated, so DP and OP are lower than for the VOT2016 data. In the case of occlusion and deformation, the comparison algorithms failed in tracking, but the proposed method tracked the object successfully due to its occlusion processing mechanism. For

Table 5. Results of Various Tracking Algorithms in KITTI Tracking Dataset

Algorithm name	MOTA	MOTP	Recall	Hz
KCF (*18*)	67.2%	82.1%	78.9%	74.2
DSST (*20*)	68.7%	84.4%	83.5%	36.7
SAMF (*21*)	67.9%	80.4%	82.2%	5.6
Staple (*23*)	69.4%	83.8%	82.7%	13.8
Proposed method	70.3%	83.9%	85.6%	18.4

Note: DSST = discriminative scale space tracker; KCF = kernel correlation filter; KITTI = Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago; MOTA = multiple object tracking accuracy; MOTP = multiple object tracking precision; SAMF = scale adaptive multiple feature.

processing speed, there is only one category of Car in the VOT2016 dataset, and the object box is small, so the speed is fast. The vehicle boxes in the Bus and Truck categories of the data are large, and it takes much time to calculate the maximum response during tracking. Overall, the improved KCF multi-scale and multi-feature tracking algorithm designed in this paper has good adaptability to complex highway traffic scenarios.

The KITTI (*5*) tracking dataset and evaluation metrics (*31*, *32*) of multiple object tracking accuracy (MOTA), multiple object tracking precision (MOTP), Recall, and Hz were used to compare the performance of multi-object tracking algorithms. MOTA combines false negatives, false positives, and mismatch rate and shows overall tracking accuracy. MOTA is defined as:

$$\mathrm{MOTA} = 1 - \frac{\sum_t (fn_t + fp_t + \mathrm{IDS}_t)}{\sum_t g_t,} \tag{15}$$

where *fn*, *fp*, and IDS are the false negatives, the false positives and the total number of times that a tracked trajectory changes its matched ground truth identity, respectively; *g* is ground truth detections, and *t* is the frame *t*. MOTP shows the percentage alignment of predicted bounding box and ground truth. MOTP is defined as:

$$\mathrm{MOTP} = 1 - \frac{\sum_{i,t} d_t^i}{\sum_t c_t,} \tag{16}$$

where d_t^i is the distance between predicted object detection and *i*th ground truth detection in frame *t*, c_t is the number of matches in frame *t*. Recall is the ratio of correctly matched detections to ground truth detections. Hz is the processing speed in fps. Higher values of these evaluation metrics are preferred.

Since the tracking method does not require any training on the tracking data, the 21 training sequences in the KITTI tracking dataset were used as a validation set for algorithm testing, as well as the bounding box detection results from the YOLOv3 detector.

Table 5 shows the results of different online tracking methods on the KITTI dataset. Compared with KCF (*18*), the proposed method improves MOTA by 3.1%. Compared with the DSST (*20*), SAMF (*21*), and Staple (*23*) methods, the proposed method also has improved MOTA. Due to the similarity between vehicles, the Recall of the proposed method reached 85.6%, which indicates that the data association method proposed in this paper can accurately match the vehicle object. For processing speed, although the proposed method is not the fastest, it can still reach 18.4 FPS with accurate tracking, which is close to real-time processing. Compared with the single-scale or single-feature tracking method, the above results show that the multi-scale and multi-feature tracking method is a better choice.

Therefore, experiments on the KITTI dataset with various lighting, traffic flow, and road conditions show that the proposed method is effective for multi-object tracking in complex traffic scenes. The vehicle trajectory generated by the tracking module provides high-quality results for vehicle counting.

Vehicle Counting Results

This experiment counts the number of different categories of vehicle trajectories through the dynamic detection line, and displays the results to the upper left of the video frame, as shown in Figure 5. The traffic videos of different scenes in Table 1 are used. These videos include different resolutions and traffic flow conditions so that they can demonstrate the performance of the proposed method under different traffic conditions. The method is compared with the actual number of total vehicles in the test video. The results are shown in Table 6.

In Table 6, "Extra number" refers to the number by which the proposed method exceeds the actual number of vehicles. "Missing number" refers to the number by which the proposed method is less than the actual number of vehicles. The Correct rate is calculated as in Equation 17. The Average correct rate is the average of the correct rates for each category in each scene. For Table 6, the resolution of scene 1 is 1280×720, which causes the object detection to be missed and affects the counting results. Missed detection occurs mostly on small cars with dark colors, so the average accuracy of the vehicle counting is less than 90%. Car objects, being smaller, may be blocked by trucks and buses. For extra number, when truck objects are running parallel to each other, mutual occlusion may cause false detection. When there is no occlusion, the vehicle has already passed the detection line before it starts to generate a trajectory, which makes it impossible to count this object. Overall, the average accuracy of the vehicle count results in this paper is

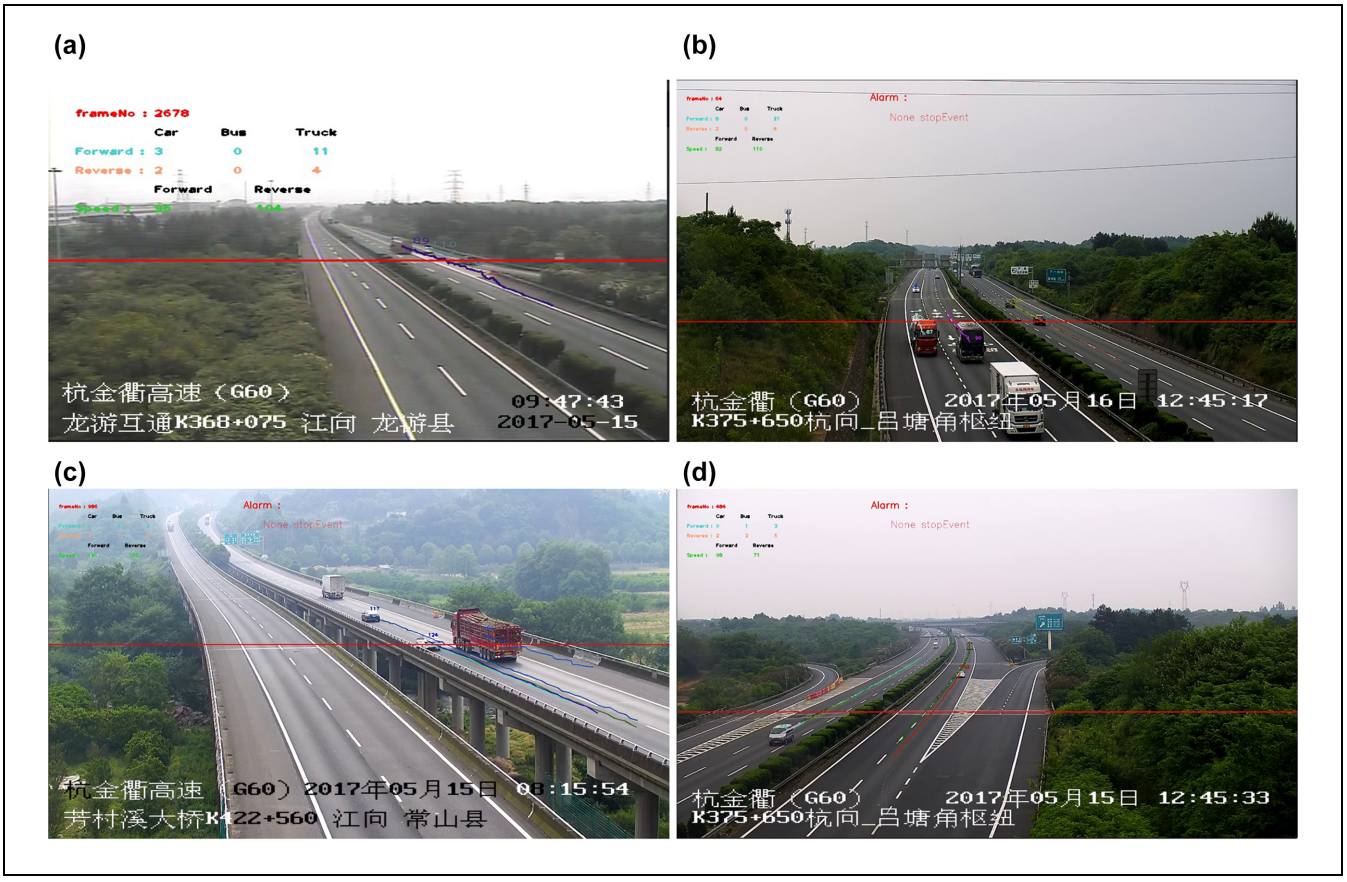

Figure 5. Visual display of vehicle counting: (*a–d*) show the vehicle count results for different scenes. In the upper left corner of each image, the counts for different types of vehicles in the uplink and downlink directions are displayed.

Table 6. Vehicle Counting Results

Scenes	Number of frames	Vehicle type	Actual number of vehicles	Proposed method	Extra number	Missing number	Correct rate	Average correct rate
Scene 1	21,439	Car	225	225	10	10	0.91	0.88
		Truck	202	222	22	2	0.88	
		Bus	20	21	2	1	0.85	
Scene 2	22,481	Car	304	306	5	3	0.97	0.94
		Truck	130	137	8	1	0.93	
		Bus	28	30	2	0	0.93	
Scene 3	22,450	Car	118	120	4	2	0.95	0.94
		Truck	103	101	5	7	0.88	
		Bus	5	5	0	0	1.00	
Scene 4	22,473	Car	306	325	24	5	0.91	0.91
		Truck	88	97	9	0	0.90	
		Bus	27	29	2	0	0.93	

maintained at around 90%, which is adequate for practical application.

$$\text{Correct rate} = \frac{\text{Actual number of vehicles} - (\text{Extra number} + \text{Missing number})}{\text{Actual number of vehicles}} \tag{17}$$

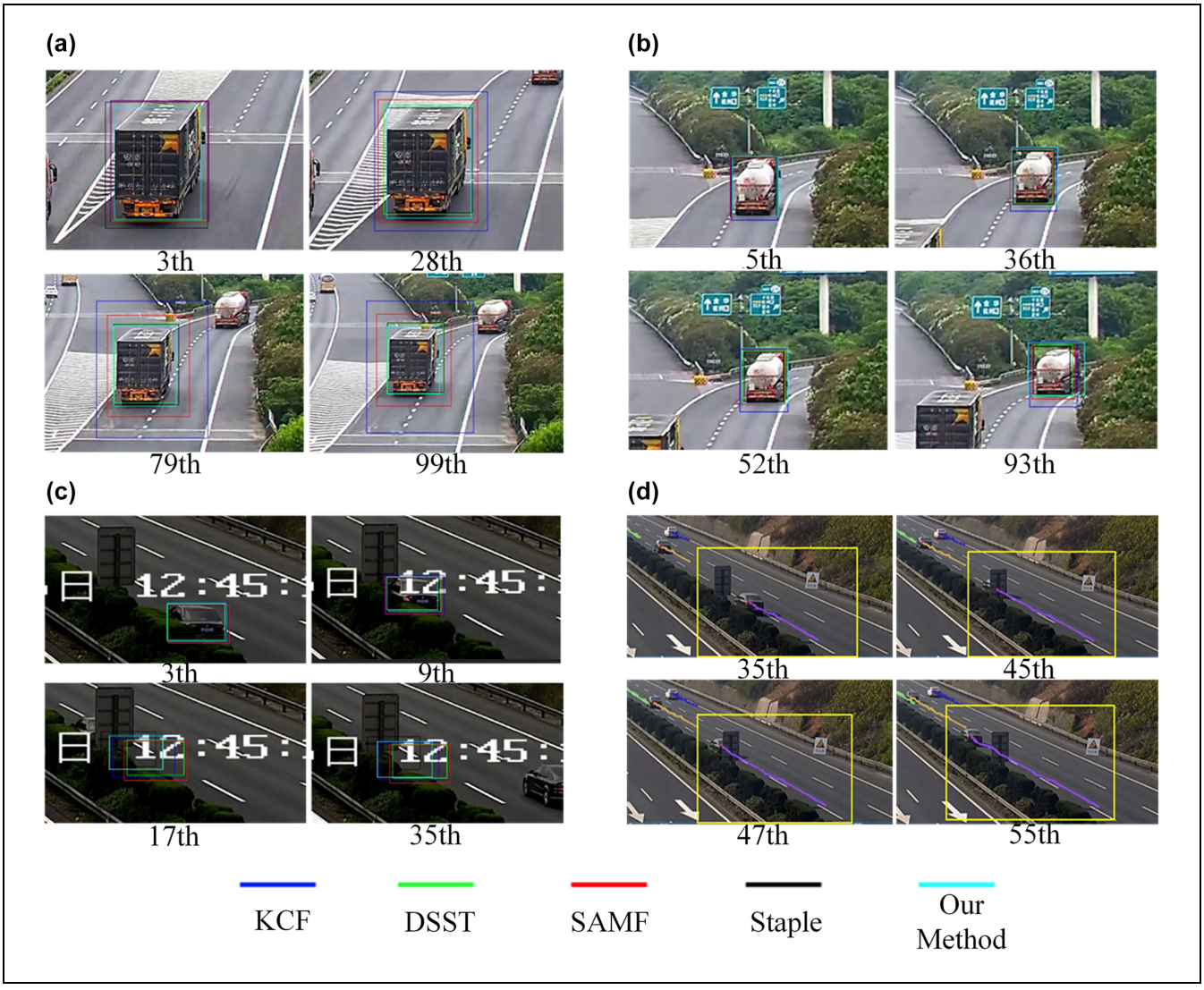

Figure 6. Results of different tracking algorithms: (*a*) different tracking algorithms used on a truck; (*b*) tracking results for deformed view; (*c*) tracking results for occlusion; and (*d*) result of proposed method when dealing with occlusion. DSST = discriminative scale space tracker; KCF = kernel correlation filter; SAMF = scale adaptive multiple feature.

Discussion

Comparison of Multi-Object Tracking

To visualize the comparison of each tracking algorithm with the proposed algorithm, some particular objects are selected to explain in depth, as shown in Figure 6.

Figure 6*a* shows the results of each tracking algorithm for a truck. The bounding box of the maximum response obtained by each tracking algorithm is plotted in the figure. The KCF and SAMF algorithms do not adapt well to the scale changes of the vehicle. DSST, Staple, and the proposed algorithm have good adaptability to scale changes because of the addition of scale filters.

Figure 6*b* shows the vehicle image deformed due to rotation and occlusion. The KCF and DSST algorithms perform poorly because they extract only the HOG features of the vehicle, and the feature describes the local information of the object. The tracker is polluted by information from the vehicle's surroundings (such as the road surface) during the tracker template updating. Therefore, the tracking result only has information on the rear of the vehicle. Since the object is small in the scene while the foreground and background are less different, the Staple algorithm using the global color histogram can easily confuse the foreground and the background, resulting in low tracking accuracy. The SAMF algorithm using CN local color features and the proposed algorithm can adapt to the deformation and maintain recognition of the object from local color information, so the tracking effect is better than the other algorithms.

As shown in Figure 6*c*, when the vehicle passes road signage, occlusion occurs. The object feature is lost, and the continuously updated vehicle template does not extract vehicle information, causing the tracker to fail. As shown in the 17th frame, when the object is completely occluded, all algorithms fail to track it. At this time, using the occlusion detection mechanism proposed in this paper, the Kalman filter is used to predict the position of the vehicle. As shown in Figure 6*d*, the vehicle position is successfully predicted in the 45th frame and the 47th frame, respectively, and the tracking trajectory is drawn.

Based on the above experimental results, the improved KCF tracking algorithm proposed in this paper is better than KCF, DSST, and Staple at vehicle tracking. Meanwhile, the processing speed is faster than the SAMF and Staple algorithms, and the vehicle occlusion problem is well solved.

Follow-Up Work

The method proposed in this paper obtains the complete two-dimensional trajectory of the vehicle and uses the trajectory to calculate the traffic flow. Compared with the traditional counting method used by hardware, the method of this paper is low in cost and high in stability of the system, and does not require massive construction or installation on existing surveillance equipment. The vehicle counting system can meet the needs of real-time processing and can be widely used in highways.

According to the work undertaken for this paper, surveillance cameras can be further calibrated to obtain the internal and external parameters. Thereby, the position information of the vehicle trajectory is converted from the image coordinate system to the world coordinate system. According to the calibration result of the camera, the vehicle speed can be calculated. Combined with the vehicle detection and tracking methods outlined in this paper, data on more abundant traffic events can be obtained and modeled, such as illegal parking, vehicle converse running, and traffic jams.

Conclusions

This study proposes a vehicle counting system for highway surveillance scenes using a self-annotated vehicle dataset trained by the YOLOv3 network to obtain the vehicle detection model. Due to the multi-branch output characteristics of the YOLOv3 network, it has good detection results on small vehicles. According to the vehicle detection result, the improved KCF multi-scale and multi-feature tracking method proposed in this paper is used to track the vehicle object. The tracking method improves the characteristics of the single scale and single feature of the KCF (*18*) algorithm for vehicle position prediction. The method combines the advantages of DSST (*20*), SAMF (*21*), and Staple (*23*) algorithms to extract the grayscale features, HOG features, CN features, and color histogram features of vehicle objects, and integrates these features to calculate the maximum response to predict the vehicle positions. The DSST (*20*) algorithm scale estimation method is then used to make appropriate scale predictions for different vehicle sizes. The vehicle trajectory is obtained using the above methods with the data association method based on the IoU similarity measure and the row-column optimal association criterion. For the problem of matching failure between object and trajectory, various solutions are given. The vehicle trajectory is used to determine the vehicle running directions and vehicle moving areas automatically. The detection line is placed by taking the centroid coordinate value of the moving area so that the position of the detection line can be dynamically updated. Finally, the methods of this paper can accurately count the number of different vehicle trajectories passing through the detection line in a certain period.

The experimental results show that the proposed method can obtain high-precision vehicle object detection results, and has suitable multiple objects tracking accuracy and precision, thus obtaining accurate vehicle counting results. Meanwhile, it also maintains a high tracking speed. Finally, the results of this study provide valuable data that are important for the management and control of highways and can be an important reference for transportation researches.

Acknowledgments

The authors thank Shuo Ding, Xu Yun, and Ying Li for their technical support.

Author Contributions

The authors confirm contribution to the paper as follows: Literature search and review: Huansheng Song, Zhe Dai; data collection: Huaiyu Li, Haoxiang Liang; analysis and interpretation of results: Haoxiang Liang, Zhe Dai; draft manuscript preparation: Haoxiang Liang, Huansheng Song. All authors reviewed the results and approved the final version of the manuscript.

Data Accessibility Statements

The vehicle trajectory data generated for this study is available in Google Drive, https://drive.google.com/file/d/1WydfdThXD2s5DQjZQeu7uU9QojXST8m6/view?usp=sharing. Other datasets analyzed during the current study are not publicly available due to privacy reasons.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work is supported by the National Natural Science Foundation of China (No.61572083), the Ministry of Education Joint Fund Project of China (No.6141A02022610) and the Key Research and Development Program of Shaanxi Province of China (No.2018ZDXM-GY-047).

References

1. Manikandan, R., and R. Ramakrishnan. Video Object Extraction by using Background Subtraction Techniques for Sports Applications. *Digital Image Processing*, Vol. 5, No. 9, 2013, pp. 435–440.
2. Li, Q. L., and J. F. He. Vehicles Detection Based on Three-Frame-Difference Method and Cross-Entropy Threshold Method. *Computer Engineering*, Vol. 37, No. 4, 2011, pp. 172–174.
3. Liu, Y., Y. Lu, Q. Shi, and J. Ding. Optical Flow Based Urban Road Vehicle Tracking. *Proc., Ninth International Conference on Computational Intelligence and Security*, Leshan, 2013, pp. 391–395.
4. Lin, T. Y., M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. *European Conference on Computer Vision*, Zurich, Switzerland, 2014, pp. 740–755.
5. Geiger, A., P. Lenz, and R. Urtasun. Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. *Proc., 2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, Rhode Island, 2012, pp. 3354–3361.
6. Guerrero-Gómez-Olmedo, R., B. Torre-Jiménez, R. López-Sastre, S. Maldonado-Bascón, and D. Oñoro-Rubio. Extremely Overlapping Vehicle Counting. *Proc., Iberian Conference on Pattern Recognition and Image Analysis*, Santiago de Compostela, Spain, 2015, pp. 423–431.
7. Girshick, R., J. Donahue, T. Darrell, and J. Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, Ohio, 2014, pp. 580–587.
8. Girshick, R. Fast R-CNN. *Proc., IEEE International Conference on Computer Vision*, Santiago, Chile, 2015, pp. 1440–1448.
9. Ren, S., K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis & Machine Intelligence, Vol. 39, No. 6, 2015, pp. 1137–1149.
10. Liu, W., D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C-Y. Fu, and A. C. Berg. SSD: Single Shot Multibox Detector. *Proc., European Conference on Computer Vision*, Amsterdam, The Netherlands, 2016, pp. 21–37.
11. Redmon, J., S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, 2016, 779–788.
12. Redmon, J., and A. Farhadi. YOLO9000: Better, Faster, Stronger. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, 2017, pp. 7263–7271.
13. Redmon, J., and A. Farhadi. YOLOv3: An Incremental Improvement. *Computer Science*, arXiv preprint arXiv:1804.02767, 2018.
14. Comaniciu, D., V. Ramesh, and P. Meer. Real-Time Tracking of Non-Rigid Objects using Mean Shift. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head Island, SC, Vol. 2, 2000, pp. 142–149.
15. Ali, N. H., and G. M. Hassan. Kalman Filter Tracking. *International Journal of Computer Applications*, Vol. 89, No. 9, 2014, pp. 15–18.
16. Kristan, M., R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, and T. Vojíř. The Visual Object Tracking VOT2014 Challenge Results. *European Conference on Computer Vision*, Zurich, Switzerland, 2014, pp. 191–217.
17. Henriques, J. F., R. Caseiro, P. Martins, and J. Batista. Exploiting the Circulant Structure of Tracking-by-Detection with Kernels. *European Conference on Computer Vision*, Florence, Italy, 2012, pp. 702–715.
18. Henriques, J. F., R. Caseiro, P. Martins, and J. Batista. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37, No. 3, 2014, pp. 583–596.
19. Danelljan, M., F. Shahbaz, M. Khan Felsberg, and J. van de Weijer. Adaptive Color Attributes for Real-Time Visual Tracking. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, Ohio, 2014, pp. 1090–1097.
20. Danelljan, M., G. Häger, F. Khan, and M. Felsberg. Accurate Scale Estimation for Robust Visual Tracking. *Proc., British Machine Vision Conference*, Nottingham, United Kingdom, 2014.
21. Li, Y., and J. Zhu. A Scale Adaptive Kernel Correlation Filter Tracker with Feature Integration. *European Conference on Computer Vision*, Zurich, Switzerland, 2014, pp. 254–265.
22. Danelljan, M., G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning Spatially Regularized Correlation Filters for Visual Tracking. *Proc., IEEE International Conference on Computer Vision*, Santiago, Chile, 2015, pp. 4310–4318.
23. Bertinetto, L., J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr. Staple: Complementary Learners for Real-Time Tracking. *Proc., of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, 2016, pp. 1401–1409.
24. Huang, C., B. Wu, and R. Nevatia. Robust Object Tracking by Hierarchical Association of Detection Responses. *European Conference on Computer Vision*, Marseille, France, 2008, pp. 788–801.
25. Xiang, Y, A. Alahi, and S. Savarese. Learning to Track: Online Multi-Object Tracking by Decision Making. *Proc.,*

IEEE International Conference on Computer Vision, Santiago, Chile, 2015, pp. 4705–4713.
26. Danelljan, M., A. Robinson, F. S. Khan, and M. Felsberg. Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking. *European Conference on Computer Vision*, Amsterdam, The Netherlands, 2016, pp. 472–488.
27. Danelljan, M., G. Bhat, F. S. Khan, and M. Felsberg. Eco: Efficient Convolution Operators for Tracking. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, 2017, pp. 6638–6646.
28. Van De Weijer, J., C. Schmid, J. Verbeek, and D. Larlus. Learning Color Names for Real-World Applications. *Proc., IEEE Transactions on Image Processing*, Vol. 18, No. 7, New York, 2009, pp. 1512–1523.
29. Arthur, D., and S. Vassilvitskii. k-means++ : The Advantages of Careful Seeding. *Proc., Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, Louisiana, 2007, pp. 1027–1035.
30. Kristan, M., A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin, T. Vojíř, et al. The Visual Object Tracking VOT2016 Challenge Results. *European Conference on Computer Vision,* Amsterdam, The Netherlands, 2016.
31. Bernardin, K., and R. Stiefelhagen. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. *EURASIP Journal on Image and Video Processing*, Vol. 1, London, United Kingdom, 2008, pp. 1–10.
32. Li, Y., C. Huang, and R. Nevatia. Learning to Associate: Hybrid Boosted Multi-Target Tracker for Crowded Scene. *Proc., IEEE Conference on Computer Vision and Pattern Recognition*, Miami Beach, Florida, 2009, pp. 2953–2960.