Hindawi
Journal of Advanced Transportation
Volume 2019, Article ID 9060797, 16 pages
https://doi.org/10.1155/2019/9060797

Research Article

Extracting Vehicle Trajectories Using Unmanned Aerial Vehicles in Congested Traffic Conditions

Eui-Jin Kim,[1] **Ho-Chul Park**,[1] **Seung-Woo Ham**,[1] **Seung-Young Kho**,[2] **and Dong-Kyu Kim**[2]

[1]*Department of Civil and Environmental Engineering, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, Republic of Korea*
[2]*Department of Civil and Environmental Engineering and Institute of Construction and Environmental Engineering, Seoul National University, 1 Gwanak-ro, Gwanak-gu, Seoul 08826, Republic of Korea*

Correspondence should be addressed to Dong-Kyu Kim; dongkyukim@snu.ac.kr

Received 5 October 2018; Revised 1 January 2019; Accepted 26 February 2019; Published 1 April 2019

Guest Editor: Yasser Hassan

Obtaining the trajectories of all vehicles in congested traffic is essential for analyzing traffic dynamics. To conduct an effective analysis using trajectory data, a framework is needed to efficiently and accurately extract the data. Unfortunately, obtaining accurate trajectories in congested traffic is challenging due to false detections and tracking errors caused by factors in the road environment, such as adjacent vehicles, shadows, road signs, and road facilities. Unmanned aerial vehicles (UAVs), with incorporating machine learning and image processing, can mitigate these difficulties by their ability to hover above the traffic. However, research is lacking regarding the extraction and evaluation of vehicle trajectories in congested traffic. In this study, we propose and compare two learning-based frameworks for detecting vehicles: the aggregated channel feature (ACF), which is based on human-made features, and the faster region-based convolutional neural network (Faster R-CNN), which is based on data-driven features. We extend the detection results to extract vehicle trajectories in congested traffic conditions from UAV images. To remove the errors associated with tracking vehicles, we also develop a postprocessing method based on motion constraints. Then, we conduct detailed performance analyses to confirm the feasibility of the proposed framework on a congested expressway in Korea. The results show that Faster R-CNN outperforms the ACF in images with large objects and in those with small objects if sufficient data are provided. This framework extracts the vehicle trajectories with high precision, making them available for analyzing traffic dynamics based on the training of just a small number of positive samples. The results of this study provide a practical guideline for building a framework to extract vehicles trajectories based on given conditions.

1. Introduction

The trajectories of all the vehicles on the road provide very useful empirical data. To reduce traffic congestion and collisions, these data can be used to analyze various traffic phenomena such as drivers' characteristics, lane-changing behavior, traffic oscillation, capacity drops, and crash potential [1–4]. To conduct a reliable analysis of trajectory data, a framework is needed to efficiently and accurately extract the data. For more than 40 years, researchers have gathered such valuable data by applying the vision-based detection techniques to track vehicles from aerial platforms [5–9] and surveillance cameras mounted at elevated locations [10–12]. Although these studies have improved the overall quality of the data-gathering process, the need for improvements remains. In particular, the accurate tracking of vehicles in congested areas is known to be a challenging problem due to false detections and tracking errors caused by factors in the road environment such as adjacent vehicles, shadows, road signs, and road facilities. Nevertheless, a few studies have implemented vehicle detection and tracking scheme in congested traffic conditions [12, 13] which most traffic-flow studies have implemented [1–3].

To the best of our knowledge, Next Generation SIMulation (NGSIM) data [12] are the only published vehicle trajectory data in congested traffic conditions that include

all of the vehicles on the road. These data were collected by automated vehicle tracking systems using surveillance cameras, which tracked approximately 75% of the vehicles. Subsequently, manual correction of false-positive and false-negative errors was required, which is inefficient [14]. Even with a fully automated system, camera-based detection has difficulty converting pixel coordinates into real-world coordinates (i.e., camera calibration). With the development of remote sensing technology, global positioning system-based (GPS-based) measurements, such as those acquired by probe cars, smartphones, and car navigation systems, have been used to extract vehicle trajectories. However, these devices can only provide trajectories for equipped vehicles, and the accuracy is insufficient for analyzing traffic dynamics [15, 16].

Unmanned aerial vehicles (UAVs), the use of which has increased in traffic-related research, can mitigate these difficulties [17–20]. The key advantages of UAVs as a tool for collecting vehicles' trajectories are that they can hover (stationary flight) above the traffic to obtain images without causing any disturbance or geometric image distortion. This approach can facilitate efficient supervised learning for vehicle detection with simplified preprocessing. However, the images provided by UAVs have the disadvantages such that vehicles appear as small objects in a large-scale image, and they may be partially occluded by shadows and road facilities. This problem can be further exacerbated in congested traffic. Although some studies have tried to extract vehicle trajectories from UAV images, they have limited success in improving accuracy due to the use of conventional vehicle detection methods, such as background subtraction, optical flow, and blob analysis. These approaches cannot robustly detect the exact location of vehicles in congested traffic, which results in false tracks in the tracking process [7–9, 18].

To accurately detect vehicles, some researchers have suggested a supervised learning-based method for UAV images. These methods use a trained vehicle classifier that shows superior performance based on effective feature representation and a strong learning algorithm [13, 17, 19, 20]. Some of these methods extend the vehicle detection to tracking by matching the detected vehicles in sequential frames [13]. Recently, deep learning approaches, especially the convolutional neural network (CNN), have been applied with great success to object detection tasks. Whereas the possibility of the performance improvements based on human-made features has reached its limits, the structure of CNN and its variants continue to make a new breakthrough based on the data-driven features of the images [21]. Some of the CNN-based methods have been suggested for application in detecting vehicles from UAV images and have shown better performance than feature-representation schemes [22, 23]. However, these studies were conducted with sufficient amounts of training data and high-resolution images and involved high computational cost, which limits their practical application. In actual research, it is not easy to obtain a sufficient number of samples due to the limited operation history of UAVs and the large labeling effort required. In addition, the image resolution tends to be lower to cover a wide spatial range. The most effective detection method can vary depending on the given conditions. Therefore, it is necessary to consider the frameworks available for extracting vehicles trajectories that are the most practical in the analysis of actual traffic flow.

In this paper, we present a framework for extracting vehicles trajectories from UAV images in congested traffic conditions. After performing simplified preprocessing, we trained two different vehicle classifiers using the human-made features and the data-driven features extracted from the CNN structure, respectively. We then compared these two classifiers regarding their detection performance according to the number of training samples and the image sizes. Next, we extended this detection to tracking by assigning vehicle identifications (IDs), and we identified some tracking errors by detailed performance analysis and corrected them using postprocessing based on motion constraint in congested traffic. Unlike previous studies which evaluated only detection accuracy, we evaluated tracking accuracy including mismatching and location errors, using manually labeled ground-truth locations of the trajectories. Finally, with the extracted vehicle trajectories from the proposed method, we calculated and evaluated the aggregated traffic data that are generally used for the traffic-flow studies.

The key contributions of this paper can be summarized as follows: (a) we propose and compare two different learning-based frameworks for detecting vehicles and extend these to vehicle tracking, to extract multivehicle trajectories in congested traffic from UAV images; (b) our framework shows promising performance using only a small number of training samples and involves a reasonable computation cost for constructing large-scale vehicle trajectories dataset; (c) we propose postprocessing for trajectories in congested traffic using motion constraints and conduct a detailed performance analysis to present the quality of the trajectories; and (d) evaluation with the aggregated traffic data confirms that the extracted trajectories are enough to be applicable for the traffic-flow analysis.

The remainder of this paper is organized as follows. First, we describe the existing frameworks used to extract vehicle trajectories and examine their advantages and shortcomings. In the next section, we discuss in detail the methodology of our framework. Then, we present the experimental process and describe the data we used. We then conduct a detailed performance analysis of the results, which confirm the excellence of the proposed framework. Lastly, we discuss our findings, make brief concluding remarks, and share future research plans.

2. Related Work

Many previous studies have conducted vision-based vehicle detection and tracking in the collection of traffic information. The vehicle tracking outputs include temporal information about the object in sequential frames, whereas vehicle detection outputs only localize the objects in images. For single-object tracking, many studies have mainly focused on designing a sophisticated appearance model [24, 25] or a motion model [26] to deal with challenging environments involving occlusion and illumination variations. For multiple-object

tracking, the key issue is to overcome the various interruptions such as interaction among crowded objects, the similar appearance among objects, and track initialization and termination as the objects appear in or disappear from each frame [27, 28]. Therefore, the density of the objects in images determines the tracking method. UAV images taken in the vertical direction from above congested traffic can be categorized as semicrowded images, which most researchers manage using the "tracking-by-detection" approach [28]. In this section, we mainly review existing methods for detecting vehicles that heavily affect the performance of "tracking-by-detection" approach, as well as some extensions for tracking vehicles in UAV images.

The use of surveillance cameras, which are deployed extensively around the world, has been studied for a long time. Coifman et al. [10] proposed a real-time system for vehicle tracking and tested this system on both free-flow and congested traffic. Their system exhibited good performance in obtaining the speeds of vehicles, but inferior performance in obtaining their flow, density, and trajectories. Subsequently, Wan et al. [11] suggested an improved framework that included a novel 3-D Hungarian algorithm for tracking vehicles. This algorithm detected the center of the mass of the vehicle and solved the assignment problem for sequential frames to identify and match each vehicle. This framework showed excellent performance for speed and volume, but no vehicle trajectories of the vehicles were extracted in this study. Since surveillance cameras gather tilted angles in their images, the camera must be calibrated to transform the pixels in the image into real-world coordinates. This procedure distorts the shapes of vehicles, which results in only the vehicles' specific proportions being detected and tracked rather than providing bounding boxes around each vehicle. The camera calibration process also generates bias due to the sensitive parameters of the cameras [29].

Because of these fundamental difficulties associated with using surveillance cameras, UAV-based studies have been proposed as an alternative. Azvedo et al. [6] used median-based background subtraction and blob analysis to detect vehicles. The authors collected partial trajectories by aircraft flying over an extensive area. Ke et al. [18] suggested unsupervised, real-time vehicle detection and tracking systems that used interest-point tracking and clustering. Although the authors extracted no vehicle trajectories, their framework performed well in estimating speed but showed slightly lower performance in counting vehicles. Khan et al. [8] proposed an automated framework based on background subtraction using optical flow and blob analysis. These authors tested their framework on a four-street intersection but performed no quantitative evaluation of vehicle trajectories. The automated frameworks mentioned above that use unsupervised detection methods can be applied rapidly and conveniently without any training classifier. However, these methods are sensitive to the complexity of the scene, such as adjacent vehicles, shadows, road signs, and road facilities. Therefore, their performance decreases significantly in congested traffic in which vehicles are crowded together and moving slowly [13].

To overcome these limitations, supervised learning-based vehicle detection has been applied in computer vision based on the development of feature representation and a powerful learning algorithm. Since these methods detect vehicles as bounding boxes rather than points or blobs, they can be applied robustly in congested traffic conditions. Grabner et al. [17] were the first to suggest the supervised learning-based vehicle detection method using aerial images. The authors utilized the AdaBoost classifier with three feature types conducted tests on an urban road and a parking lot. Liu and Mattyus [20] proposed a multiclass vehicle detection process based on AdaBoost and integrated channel features. Their methodology showed higher accuracy in counting vehicles on the road than other relevant approaches. Xu et al. [13] extended the detection of vehicles using the Viola and Jones (V-J) object detection scheme [30] for the tracking of vehicles, and they showed that the V-J scheme outperforms many other methods based on feature representation. The authors trained the classifier using a vast number of training sets and conducted experiments at various sites, including those with congested traffic. Their framework performed well even in congested flow. However, they only evaluated the performance of this scheme using detection-based measures, such as precision and recall, and not tracking errors, such as mismatching and location errors. To confirm the usability of the vehicle trajectories for analyzing traffic dynamics, the measures are needed to consider the spatiotemporal context of the vehicles to account for tracking errors.

More recently, deep learning approaches based on CNN have shown impressive performance in object detection. Specifically, region-proposal-based detectors such as R-CNN [31], Fast R-CNN [32], and Faster R-CNN [33] have been proposed to precisely locate objects in an image. Xu et al. [22] applied the Faster R-CNN for vehicle detection from UAV images, and Faster R-CNN showed better performance than conventional methods with a sufficient amount of computation cost and training samples. However, the authors did not extend the detection to tracking, and a detailed comparison with a more advanced method based on human-made features is needed to support the conclusion of the performance excellence of the Faster R-CNN. Tang et al. [23] proposed a modified Faster R-CNN for small-sized vehicles from UAV images. They employed hierarchical feature maps to deal with features in small pixels. Their method showed better performance than that of the aggregated channel feature (ACF) detector. However, since the test was only conducted on urban streets and a parking lot, its performance in congested traffic should be determined.

Several implications can be drawn from the above review of previous studies. First, the learning-based vehicle detection method has shown superior performance compared with the conventional unsupervised method. CNN-based approaches, in particular, have shown promising performance in the detection of vehicles in UAV images. For practical application, however, evaluation in congested traffic conditions is necessary with respect to efficiency and accuracy. Second, although a few studies have extended vehicle detection to tracking, there is room for improvement by taking advantage of UAV images in congested traffic conditions. Therefore, an

overall framework for extracting vehicle trajectories should be specified. Third, more detailed analyses of tracking errors are needed to confirm the usability of vehicle trajectories.

3. Materials and Methods

3.1. Overview. Our framework for extracting vehicles trajectories comprises four sequential processes, as shown in Figure 1. First, we manually extract the road coordinates in the first frame and use these for all the frames, since the UAV can remain stable and balanced in flight. Then, we detect vehicles on the road using a supervised classifier trained by ACFs [34] and Faster R-CNN [33]. The locations of vehicles in each frame are collected during this process. To create the trajectories of vehicles over time, the detected locations of the same vehicles in sequential frames are matched by solving the assignment problem. Lastly, we perform postprocessing to remove three types of false positives that occurred in the tracking process, using motion constraints in congested traffic. Detailed descriptions of these processes are presented in the following sections.

3.2. Preprocessing: Road Extraction and Camera Calibration. Road extraction and camera calibration are the important preprocessing steps in the vision-based detection and tracking of vehicles. Road extraction can reduce the computation time required for the detection process and eliminate false-positive errors by limiting the search space. Camera calibration involves converting image coordinates into real-world coordinates. For this process, exact camera parameters are required, including the focal length, camera height, and tilt angle. These sensitive parameters influence the accuracy of the trajectory [29]. In this study, we simplified the two preprocessing steps described above by using UAV images. Since all images were obtained from a vertical direction above the traffic, this eliminated the need to calibrate the camera. In some cameras, the fish-eye effect must be removed [8], which was not necessary in our case. Also, since UAVs can hover and produce videos that maintain almost constant coordinates among frames, we were able to use the manually extracted road coordinates from the first frames for all the frames. Although, we focused on the automated process excluding preprocessing, real-time approaches require automated road extraction and camera calibration. Further details about these processes are presented in other work [8, 35].

3.3. Vehicle Detection

3.3.1. Aggregated Channel Features. ACFs [34], which were improved from integral channel features [36], have been shown to exhibit efficient and accurate performance in the detection of pedestrians [37]. Liu and Mattyus [20] recently showed that these features also enable the very fast and accurate detection of vehicles from UAV images. The integral image firstly suggested by Viola and Jones [38] is an intermediate representation of images for computing features, and, using this concept, Haar-like features, which are approximate local derivatives, can be computed at any scale or locations. ACFs are calculated for several channels by linear and nonlinear transformations of the input images. We used normalized gradient magnitude, six channels of the histogram of oriented gradients, and LUV color as channels [36]. Single pixel lookups aggregated from each channel in a smoothed image can be computed extremely fast using the integral image, and we used these as first-order features for detection. Higher-order channel features (e.g., Haar-like features) can also be computed using multiple first-order features. Since these features naturally integrate data from different sources (e.g., lighting and weather conditions), they are suitable for collecting traffic data. We computed the ACFs using fast feature pyramid, which extracts sufficiently approximated features on finely sampled pyramids for rapid calculation [34].

To extract only the richer ACFs, only critical feature should be selected in the learning algorithm. We used AdaBoost as the training classifier, which combines weak classifiers, including a small number of features, to form a stronger classifier. In each iteration, t, a new weak classifier, $h_t(\mathbf{x})$, is trained using adaptively weighted samples that had been previously misclassified by a weak classifier. This combined classifier has the form $H(\mathbf{x}) = \sum_t \alpha_t h_t(\mathbf{x})$, where α_t is calculated by greedily minimizing a loss function in each iteration, t [39]. As a weak classifier, we used a decision tree with a maximum depth of five. We trained a boosted classifier with 1,024 weak classifiers. Since using all of the weak classifiers on every pixel in an image is inefficient, to improve efficiency, a cascade of classifiers is required, which removes negative subwindows using a smaller boosted classifier. We used a "soft cascade structure" as described in [38], to relax the distinct stages of the cascade using the rejection threshold. When performing the detections, multiple detections near one vehicle occurred because our trained classifier used a sliding window to detect vehicles in the image. To suppress these overlapping detections, we used nonmaximal suppression with an overlap threshold, whereby if the overlapped area of two detected vehicles was greater than 40% of their total area, only the vehicle with the higher detection score (i.e., higher probability of being detected as a vehicle) was retained.

3.3.2. Faster R-CNN. R-CNNs extract region proposals, which indicate candidate locations for objects from the raw image and then compute the CNN features using these locations as inputs [31]. Fast R-CNN features a major change in the method used to compute the CNN feature [32]. After extracting region proposals from the data, this method uses the coordinates of region proposals to select regions of interest (ROI). R-CNN and Fast R-CNN both use selective searches to generate region proposals, whereas Faster R-CNN uses a region proposal network to improve efficiency [33]. Faster R-CNN comprises two parts as shown in Figure 2(a): a region proposal network (RPN) and an object detection network. The object detection network has the same framework as the Fast R-CNN. The RPN uses a feature map from a convolutional neural network and creates up to 300 region proposals. It uses an anchor to identify the area where objects

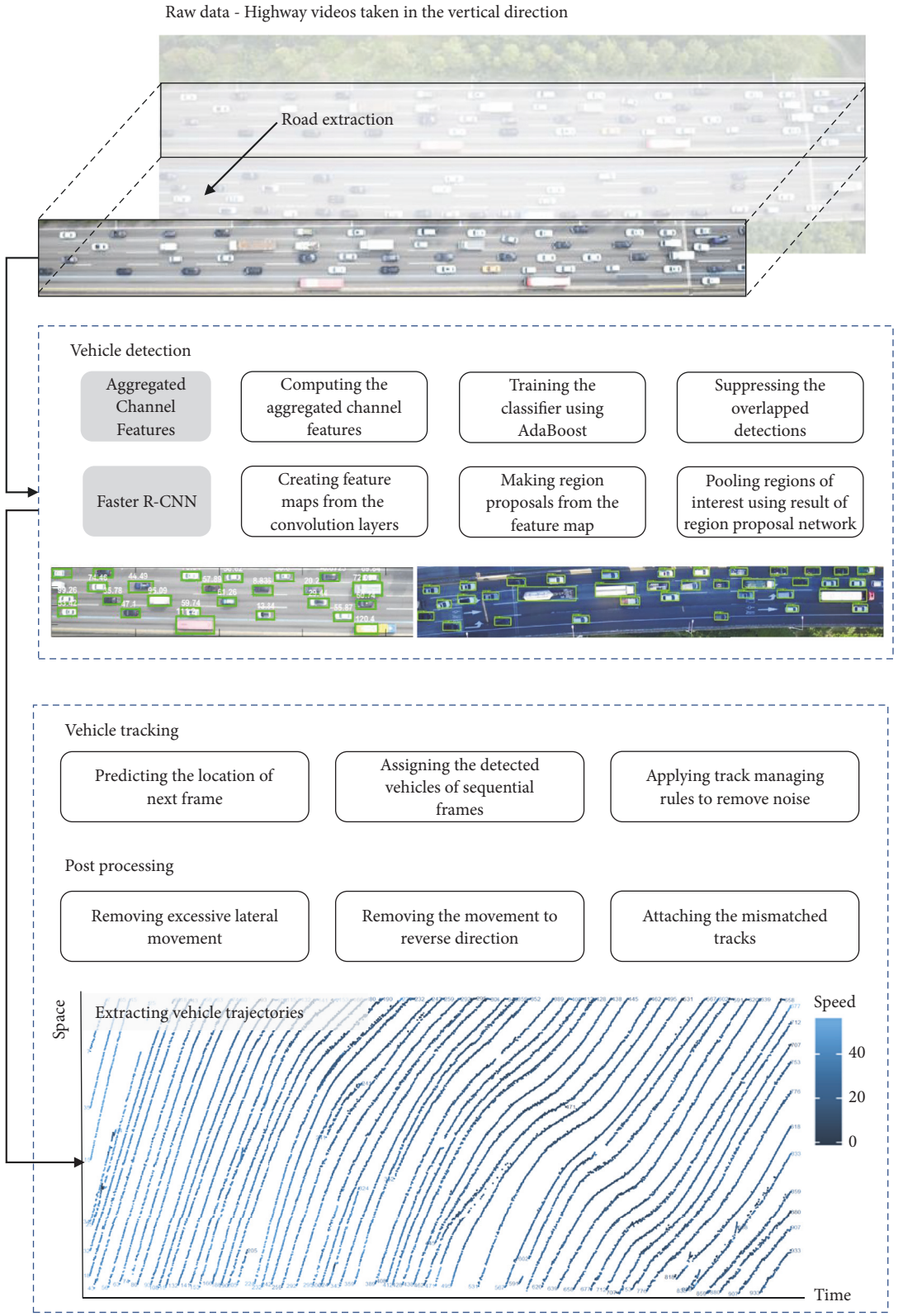

Figure 1: Flowchart of four sequential processes in the proposed framework for extracting vehicle trajectories, i.e., road extraction, vehicle detection, vehicle tracking, and postprocessing.

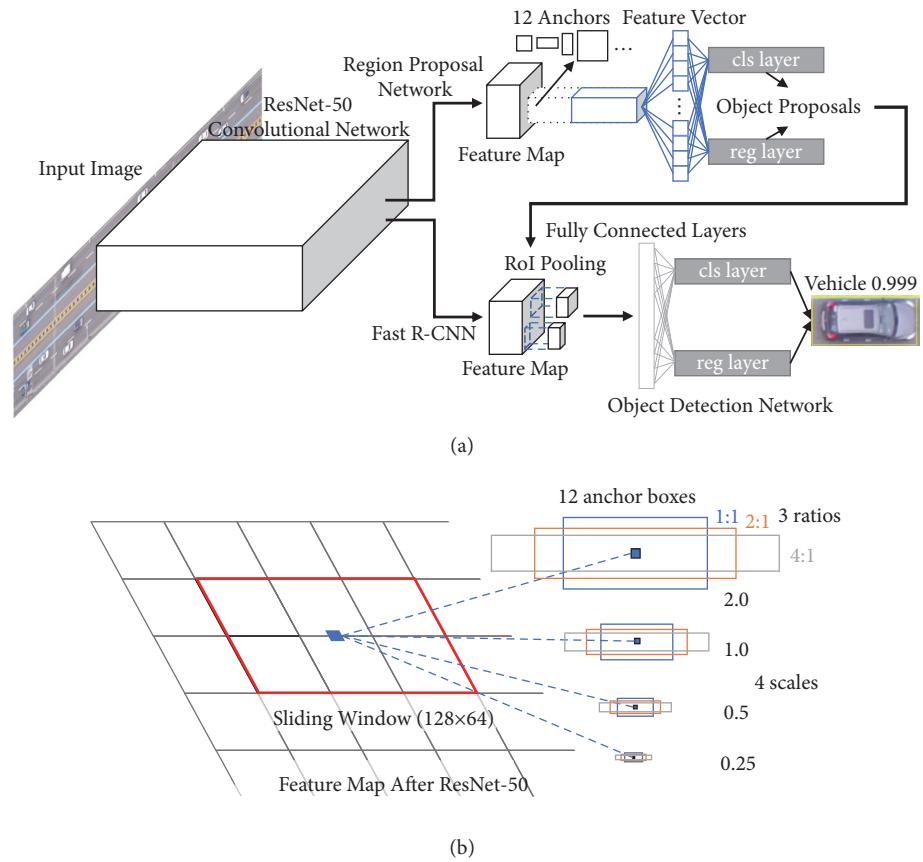

Figure 2: Architecture of Faster R-CNN: (a) overall architecture and (b) design of each anchor.

are likely to exist. The anchor is a type of sliding window with different sizes and ratios. We used an anchor with base size of 128×64 pixels, a scale from 0.25 to 4.0, and a ratio ranging from 1.0 to 4.0 as in Figure 2(b). We chose the hyperparameters based on background knowledge about the image which contain rectangular objects of relatively uniform size. The loss of each anchor is calculated by the cross-entropy loss, which has two classes: object and background. The result generated by the region proposal network is then given to the object detection network. ROI are pooled and each ROI is trained by a classifier and a bounding box regressor. The classifier loss and the bounding box regressor are calculated in integrated loss form, as follows:

$$L\left(\{p_i\},\{t_i\}\right) = \frac{1}{N_{cls}}\sum_i L_{cls}\left(p_i, p_i^*\right) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}\left(t_i, t_i^*\right) \quad (1)$$

where N_{cls} and N_{reg} are the minibatch size and number of anchor locations, respectively. p_i is the predicted probability of anchor i being an target object, and p_i^* is the ground truth, which has a value of 0 or 1, depending on whether anchor i is an object or not. t_i and t_i^* are modified forms of bounding-box and ground-truth-box coordinates. L_{cls} uses cross-entropy loss and L_{reg} uses the smooth L_1 loss that has $t_i - t_i^*$ as an input.

3.4. Vehicle Tracking. Based on the detection results, we implemented a simple motion-based tracker. The objective of tracking vehicles is to extract their trajectories over time by locating their positions. Of the various object-tracking methods [27, 28], we used simple point tracking based on individually detected vehicles. This method associates the vehicles in frame t to those in frame $t + 1$ and then assigns identifications (IDs) for extracting the trajectory of each vehicle. The vehicle tracking procedure has three steps. In the first, assuming constant velocity, we used previously tracked vehicles to predict their locations in frame $t + 1$ using a Kalman filter [40]. Although we could instead use the vehicle's location in frame t, the prediction process reduces the influence of noise associated with the detection error of the t frame and of some partially missed vehicles in one trajectory. In the second step, we associated the detected vehicles across frames and solved the optimization problem (i.e., assignment problem) to minimize the correspondence cost of predicted and detected vehicles in frame $t + 1$. We defined this cost as the Euclidean distance between the predicted and detected locations in the frame $t + 1$. To bound the search region in congested traffic conditions, we added the constraint that the instantaneous speed of the

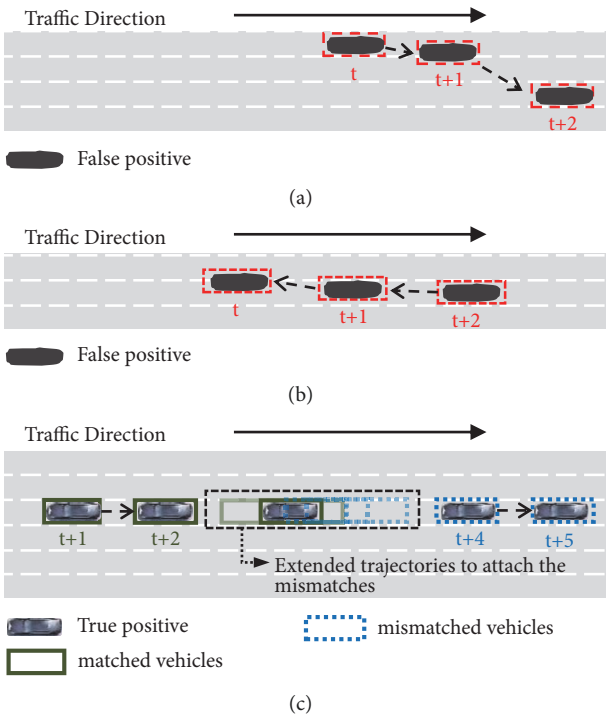

FIGURE 3: False tracks generated in congested traffic: (a) excessive lateral movement, (b) reverse direction, and (c) mismatching.

vehicles should be less than 85 km/h. We used the Hungarian algorithm to solve this optimization problem [41]. In the last step, we created and discarded tracks based on the assigned results and management rules. We created tracks when the visible count (i.e., the number of frames in which the vehicle was detected) was more than 15 frames, and we deleted them if they were not visible in at least 15 frames or if the visibility, which is the total number of visible tracks counted divided by the total number of frames, was less than 0.6.

3.5. Postprocessing. Vehicle tracking in congested traffic is significantly affected by false positives (i.e., detection of vehicles when none is present and detection of vehicles with incorrect size) and false negatives (i.e., detecting no vehicles when some are present). Duplicate detection, road facilities, shadows, and partially detected trucks are typical false positives in UAV images. Figure 3 shows three types of frequently occurring false tracks in the tracking process. Because false positives did not occur uniformly in sequential frames, their tracks appear to differ from the motion of a typical vehicle in a traffic jam. Excessive lateral movement and reverse direction are examples of cases in which tracks were generated by false positives, as shown in Figures 3(a) and 3(b). The mismatching in Figure 3(c) is an example of a track becoming disconnected in the middle and being converted to a new track. This error greatly influences the microscopic trajectory analysis, although it does not affect the collection of aggregated speeds or densities. Both false negatives and false positives can cause this disconnection.

Since vehicles in congested traffic have strong motion constraints, they can be used to remove false tracks. During postprocessing, we identify three types of false tracks and remove the errors associated with each. Three cases of post-processing were proposed to remove each case of errors. Initially, we eliminated the trajectories of vehicles with excessive lateral movement and opposite-direction movements based on their speeds and directions. Subsequently, we extended the trajectories of each vehicle by three seconds before and after the vehicle, while assuming constant speed, and attached mismatches in cases where two extended trajectories were overlapped (i.e., closer than half of the vehicle's length).

3.6. Quantitative Evaluation. We evaluated the false positives (FP) and false negatives (FN) identified in the vehicle detection procedure based on the manually labeled ground-truth locations of the vehicles. The precision and recall are well-known measures for object detection [42] to quantify these errors. Precision is defined as the accuracy of predicting true positives from among all of the detected samples as in (2), and recall is the number of true positives detected among all the ground-truth locations of vehicles, as in (3). The F-measure is the trade-off between recall and precision, which have equal importance as in (4). When the detected area and the ground-truth area along the road were overlapped by more than 50% of their combined area, we considered this to be a true positive (TP). We computed the F-measure having same precision and recall to compare the detection methods in the following sections.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F - measure = \frac{2 \bullet Recall \bullet Precision}{Recall + Precision} \tag{4}$$

We evaluated the vehicle tracking performance using the multiobject-tracking accuracy (MOTA) and multiobject-tracking precision (MOTP) metrics [43]. MOTA generates three error ratios with respect to ground truth (g_t) over time t, i.e., the ratio of misses (m_t), false positives (fp_t), and mismatches (mme_t), as shown in (5). Summing these three errors produces the total error. MOTP is the total position error for the overall frame ($\sum_{i,t} d_{i,t}$) of matched object–hypothesis pairs (i.e., detected true positives and ground-truth locations of vehicles) averaged by the total number of matches made ($\sum_t c_t$), as shown in (6).

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t}$$

$$the\ ratio\ of\ misses = \frac{\sum_t m_t}{\sum_t g_t}$$

Table 1: Description of video datasets.

Video	Site Locations	Number of Lanes	Spatial Range	Size of Vehicles	Traffic Density	Mean Speed	Lighting Condition
Test-Video 1	Korea Expressway No. 1	Four Lanes + Bus Exclusive Lane	188 m	40×91 pixels	50.2 veh/km/lane	22.3 km/h	Cloudy After Sunrise
Test-Video 2	Korea Expressway No. 120	Four Lanes + Entry and Exit Lane	137 m	60×132 pixels	56.1 veh/km/lane	20.8 km/h	Clear Before Sunrise

Table 2: Training and test sets for detection and tracking.

Video	Training Set	Test Set for Detected Vehicles	Test Set for Vehicle Trajectories
Test-video 1	4,000 labeled vehicles (one image every 6 sec)	1,000 labeled vehicles (one image every 6 sec)	32,800 labeled trajectories of 61 vehicles for two consecutive minutes (25 images every 1 sec)
Test-video 2	4,000 labeled vehicles (one image every 6 sec)	1,000 labeled vehicles (one image every 6 sec)	-

$$\text{the ratio of false positives} = \frac{\sum_t fp_t}{\sum_t g_t}$$
$$\text{the ratio of mismatches} = \frac{\sum_t mme_t}{\sum_t g_t} \tag{5}$$

$$MOTP = \frac{\sum_{i,t} d_{i,t}}{\sum_t c_t} \tag{6}$$

To evaluate the applicability of traffic-flow analysis, the five kinds of traffic data such as space-mean speed, density, flow, spacing, and the number of lane changes are calculated using the extracted trajectories from vehicle tracking and postprocessing. The space-mean speed, density, and flow are calculated by Edie's definition [44] as in (7).

$$\text{space mean speed} = \frac{\sum_i x_i}{\sum_i t_i}$$
$$density = \frac{\sum_i x_i}{|A|} \tag{7}$$
$$flow = \frac{\sum_i t_i}{|A|}$$

where x_i is the distance traveled by the i^{th} vehicle in the time-space region, t_i is the time spent by the i^{th} vehicle in the time-space region, and $|A|$ is the area of the time-space region. We also calculate the spacing of adjacent vehicles and the number of lane changes. All of these data except the number of lane change are aggregated by space-time resolution of 30m × 10s. A mean absolute percent error (MAPE) is used to evaluate these aggregated data. The number of lane changes is evaluated using a percent error (PE) after counting in total time and space.

4. Experiment and Results

4.1. Datasets. To verify the proposed frameworks, we recorded highway traffic videos in the vertical direction using a DJI Inspire Pro 1 equipped with a Zenmuse X5 camera, which is a quadcopter drone with 4K video and 3-axis gimbal stabilizers. The resolution of the video was 3,840 × 2,160 (25 fps). Table 1 shows details regarding the video data sets. We conducted the experiments in congested traffic with respect to two different conditions: lighting and the size of the vehicles. Those conditions have a significant impact on the vision-based approach even when using the learning-based method, which generally has the merit of generalized performance. We took the first test-video over a 188-m range of the four-lane Korea Expressway No. 1, which includes one exclusive bus lane, from 8:12 A.M. to 8:24 A.M on July 15, 2016. This was a cloudy morning at peak traffic after sunrise. The average traffic density was 50.2 vehicles/km/lane, and the average speed was 22.3 km/h. The average vehicle size was 40 × 91 image pixels. We took the second test-video on a 137-m section of Korea's four-lane Expressway No. 120, which includes entry and exit lanes, from 7:09 A.M. to 7:20 A.M. on September 20, 2016. This was a clear morning at peak traffic before sunrise, with a shadow covering the entire road. The average traffic density was 56.1 vehicles/km/lane, and the average speed was 20.8 km/h. The average vehicle size was 60 × 132 image pixels, which is approximately 1.5 times larger than that in the first video.

To train the detector, we used manually labeled training sets. We set the positive samples to include only vehicles, and the negative samples to include the background as well. As presented in Table 2, we constructed training sets with 4,000 positive samples in each test-video. We labeled these vehicles in more than 100 images at 6-second intervals to prevent one vehicle from being labeled more than twice. We tested the vehicle detection performance using images with 6-second intervals containing 1,000 labeled vehicles. In

vehicle tracking performance, we focused on evaluation of the proposed postprocessing with simple tracking algorithm, and, in particular, analyzing how tracking performance is improved by the postprocessing according to variation in detection performance. To reduce the large labeling effort, a detailed performance of tracking was evaluated using 2,880 images at 0.04-second intervals (two consecutive minutes) containing 32,800 carefully labeled trajectories of 61 vehicles only in test-video 1. We used a user-friendly labeling tool that utilizes cubic interpolation to perform this large-scale labeling task [37]. To implement the ACF, vehicle tracking, and postprocessing, we used the single-threaded MATLAB toolbox for computer vision [45] and an Intel i7-6700HQ CPU@2.6 GHz processor with 16 GB of memory, and Python programming language to implement only the Faster R-CNN, based on the tensor flow object detection API [46] with an Nvidia Geforce GTX 1050 TI, 4GB GDDR5.

4.2. Results of Vehicle Detection. After the simplified preprocessing of the UAV images, the vehicle classifier was trained using the ACF and Faster R-CNN on the labeled training data. To determine the efficiency of the training, we conducted a sensitivity analysis to identify the variation in the detection accuracy with the number of positive samples. To do so, we divided the training sets into sample subsets of 500, 1,000, 2,000, 3,000, and 4,000. Then, we randomly resampled each of these subsets ten times to reflect the performance variations. Figure 5 shows the F-measures of the ACF and Faster R-CNN with error bars for the two test videos. We calculated the F-measures of the error bars by adjusting the threshold for true positive to obtain the same precision and recall values. The error bar indicates the standard deviation of the measures.

In test-video 2, the performance of the Faster R-CNN was overwhelmingly superior to that of the ACF. The mean F-measure for the ACF was 84.7% for 500 samples and it peaked at 91.0% for 4,000 samples, with a standard deviation (SD) of 0.3%. The mean F-measure of the Faster R-CNN was 95.8% for 1,000 samples and it peaked at 97.1% for 4,000 samples, with an SD of 1.3%. These results indicate that the Faster R-CNN can capture richer vehicle features in images during clear weather. On the other hand, in test-video 1, the mean F-measure of the ACF was 80.9% for 500 training samples and it peaked at 93.0% for only 3,000 training samples, with an SD of 1.9%. It subsequently decreased slightly to 91.5% for 4,000 samples. The mean F-measure of the Faster R-CNN was 50.2% for 500 training samples and it gradually increased to 91.6% for 4,000 training samples, which was slightly lower than that of the ACF. Although using more training samples improves the performance of the Faster R-CNN in test-video 1, it shows comparable performance with the ACF for a small sample size. Performance degradation of the Faster R-CNN in lower-resolution images has been reported in previous studies [23, 47], and vehicles in UAV images can also suffer from these problems. This is because only the plain features that cannot be used to distinguish between vehicles are extracted from the ROI pooling layers due to the low-resolution feature map [47]. Therefore, if the image resolution is not high enough to identify sophisticated features of vehicle, the use of ACFs, i.e., intuitive human-made features, may be more effective in detecting vehicles than the Faster R-CNN. Also, the results from two videos show the relation between model complexity and saturation. The complexity of the model determines whether the detector continues to improve as the number of training samples increases. When the model complexity is close to saturation, their accuracy does not increase with additional training data and can even lead to overfitting [48]. ACFs might be saturated with about 3,000 samples in test-video 1, whereas the Faster R-CNN, which is more structurally complex, was not yet saturated with 4,000 samples. In test-video 2, which contains more information in the vehicle images, neither model was saturated with 4,000 samples, and their performances could be improved with more training data.

Figure 6 shows the false positives of the best-performing detectors trained by the ACF and Faster R-CNN for each test-video. In test-video 1, the false positives of both of the detectors were caused by duplicated detections of vehicles. These errors can generate the false track and mismatch in tracking process, which is main factor reducing accuracy of vehicle trajectory. On the other hand, in test-video 2, the ACF suffered from the false positives caused by truck as well as other road environments whereas the Faster R-CNN successfully coped with those false positives. This explains the performance difference between two detectors in test-video 2.

4.3. Results of Vehicle Tracking and Postprocessing. In our framework, since we performed vehicle tracking based on the detection results, the tracking accuracy increased with increases in the detection accuracy. To present the sensitivity of the tracking accuracy based on the detection results, we performed vehicle tracking using four different vehicle classifiers with F-measures of 86.0%, 90.1%, 94.5%, and 97.4% in test-set for vehicle tracking. We also used postprocessing with motion constraints to remove the expected errors in the tracking process. Then, we conducted a detailed performance analysis using the large-scale labeled trajectories in test-video 1 where the duplicated detections were frequently observed. Those false positives significantly decrease accuracy of vehicle trajectory by generating the false track and mismatch in tracking process. Also, the detection results of ACF, which showed better performance than Faster R-CNN in test-video 1, was used for vehicle tracking and postprocessing because both of the detectors shared a similar cause of the false positives. We selected lane 3 (Figure 4(a)) as the target, where lane changes occur frequently. Table 3 presents the variation in tracking performance based on the detection accuracy and postprocessing. Each row shows the four different classifiers and whether or not postprocessing was performed, and each column shows the corresponding detection and tracking performances. Figures 7(a) and 7(b) show the changes in the trajectories before and after postprocessing. The numbers in Figures 7(c) and 7(d) are the start and end points of each identified vehicle, respectively. The shaded regions of Figure 7(c) are the false positives that showed a different pattern from the typical tracks and the mismatches divided

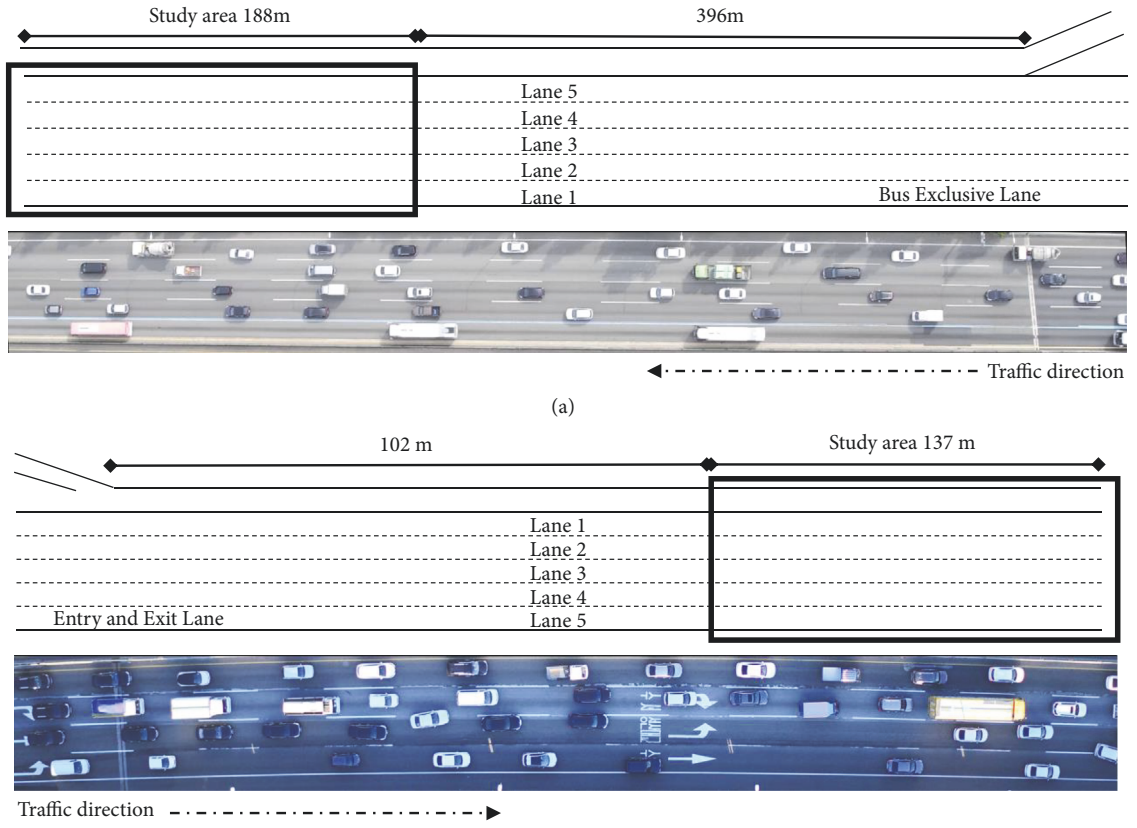

Figure 4: Images of the studied congested freeway from the UAV: (a) Korea Expressway No. 1 with cloudy after sunrise and (b) Korea Expressway No. 120 with clear before sunrise.

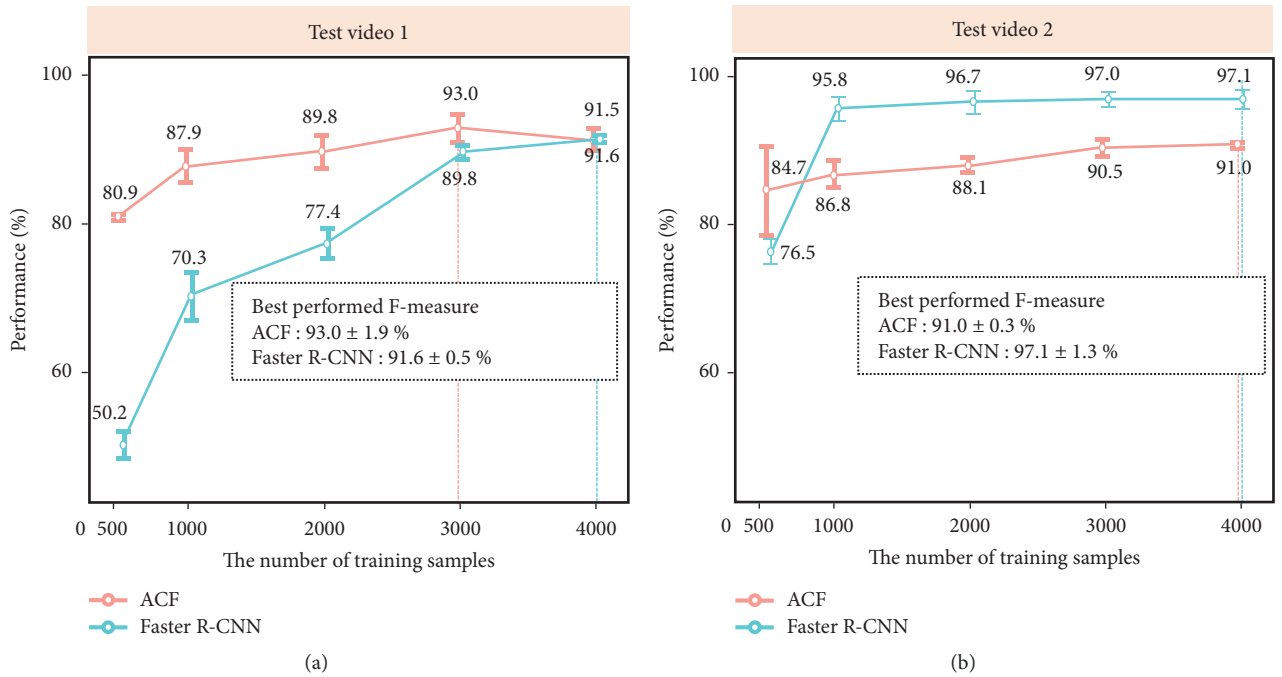

Figure 5: Variation in detection performance with the number of training samples. (a) Test-video 1 shows the best F-measure of 93.0% with a standard deviation of 1.9% using ACF with 3,000 samples, and (b) test-video 2 shows the best F-measure of 97.1% with a 1.3% standard deviation using Faster R-CNN with 4,000 samples.

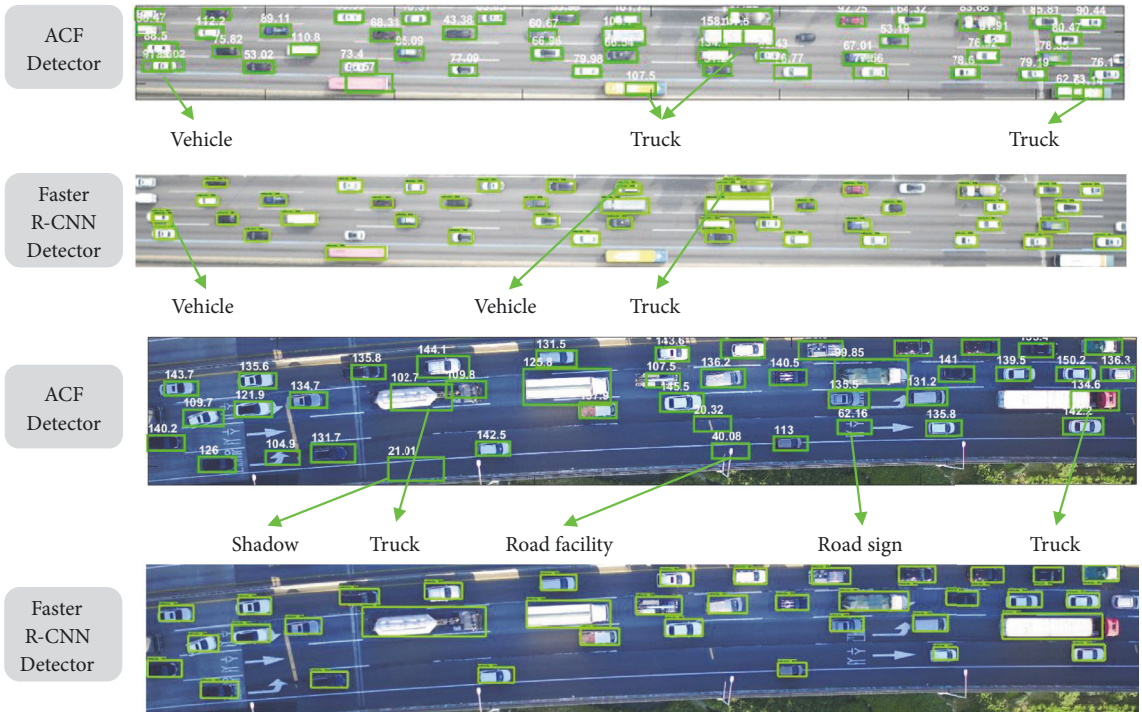

Figure 6: False positives from the two different detectors for each test-video.

into two different IDs. While postprocessing effectively removes the mismatched trajectories and false positives, it slightly increases the ratio of misses by removing some true positives. The postprocessing worked well for F-measures of 90.1%, 94.5%, and 97.4%, but it was less able to remove mismatches for an F-measure of 86.0%. This reduction was caused by difficulties in attaching mismatching trajectories when missed detection occurred to a greater extent in several consecutive frames. The MOTP was about 0.5–0.6 m regardless of the F-measure, which means that true positives provide reliable trajectories regardless of overall accuracy. In the best performance, the vehicle trajectories achieved location errors as low as 0.6 m with a MOTA of 89.9%. This suggests that the tracking process has a large impact on the overall quality of the vehicle trajectories as well as their detection.

4.4. Computation Time. The training and processing time is an important factor for extracting trajectories if a massive video dataset can be collected by UAVs. Table 4 shows the training times for detectors and the processing times for each step. In the table, we can see that the ACF and Faster R-CNN took about 31 minutes and 45 minutes, respectively, for 4,000 training samples using both the CPU and GPU. The training time of the Faster R-CNN is influenced by the hyperparameters, image sizes, and number of training samples. Therefore, the much shorter training time than that of a previous study [22] using a similar image size indicates the efficiency of the proposed method.

The total processing times of the ACF were 1.02 seconds and 1.34 seconds per image in the respective test videos, whereas those of the Faster R-CNN were 0.61 seconds and 0.71 seconds per image, respectively. These results show that our model can be conducted with reasonable computation times to construct a dataset, although these times cannot be applied to a real-time approach. When we extracted one hour of trajectories (25 fps) using the Faster R-CNN from test-video 1, it took about 15 hours, and more powerful GPUs could significantly accelerate these overall processes.

4.5. Comparison of Methods to Obtain Aggregated Traffic Data. For application to the traffic-flow analysis, it is necessary to evaluate the reliability of aggregated traffic data obtained from vehicle trajectories such as space-mean speed, density, flow, spacing, and the number of lane changes. We compare the eight combinations of detection and tracking method including the four detectors and the two trackers. The original V-J detector [30] and Single Shot Multibox Detector (SSD) [49], which are two well-known object detectors using a human-made feature and a data-driven feature each, are used for comparison with the proposed methods, i.e., ACF and Faster R-CNN. The Kanade-Lucas-Tomasi (KLT) feature tracker [50] is also used for comparison with the proposed Kalman filter-based tracking method (KF).

The evaluation is performed on the same dataset which was used to evaluate the vehicle tracking and postprocessing. Excluding the number of lane changes, each value of the traffic data is calculated by space-time resolution of 30m × 10s in lane 3. On the other hand, the number of lane changes is calculated in all the lanes over the entire period of time. Table 5 shows the testing results of eight combinations

Table 3: Variation in tracking performance based on the detection performance and postprocessing.

Detection, Tracking, and Postprocessing Performance							
Vehicle Classifier	F-measure	Postprocessing	Ratio of Miss	Ratio of False Positive	Ratio of Mismatch	MOTA	MOTP
Classifier 1	86.0%		13.4%	6.8%	10.3%	69.5%	0.54m
		√	16.6%	4.8%	8.3%	70.3%	0.52m
Classifier 2	90.1%		9.0%	7.7%	8.5%	74.8%	0.51m
		√	11.2%	3.9%	3.0%	81.9%	0.52m
Classifier 3	94.5%		4.7%	8.6%	6.8%	79.9%	0.54m
		√	6.7%	7.4%	0.1%	85.8%	0.60m
Classifier 4	97.4%		5.5%	3.7%	3.3%	87.5%	0.59m
		√	5.8%	3.2%	0.1%	89.9%	0.60m

Table 4: Computation time of detection, tracking, and postprocessing.

		Test-Video 1		Test-Video 2	
		ACF	Faster R-CNN	ACF	Faster R-CNN
Training Time for 4,000 samples		31 minutes	45 minutes	34 minutes	40 minutes
Processing Time per Frames	Vehicle Detection	0.83 seconds	0.41 seconds	1.15 seconds	0.52 seconds
	Vehicle Tracking	0.19 seconds		0.18 seconds	
	Postprocessing	0.01 seconds		0.01 seconds	
	Total	1.03 seconds	0.61 seconds	1.34 seconds	0.71 seconds

Table 5: Performance evaluation for obtaining aggregated traffic data.

Metric	Proposed Kalman Filter-Based Method (KF)				The Kanade-Lucas-Tomasi Tracker (KLT)			
	VJ	SSD	ACF	Faster R-CNN	VJ	SSD	ACF	Faster R-CNN
Processing Time (fps)	2.0	3.3	1.0	1.7	1.3	1.8	0.8	1.2
The Number of Lane Changes (PE, %)	+8.3%	+4.2%	+4.2%	+8.3%	+12.5%	+4.2%	+8.3%	+4.2%
Space-Mean Speed (MAPE, %)	3.0%	1.1%	*2.3%*	3.2%	3.5%	2.3%	3.4%	*1.9%*
Density (MAPE, %)	5.0%	6.7%	*3.1%*	5.8%	7.5%	2.7%	8.2%	*2.3%*
Flow (MAPE, %)	6.1%	6.8%	*3.2%*	7.1%	7.1%	2.8%	8.2%	*2.2%*
Spacing (MAPE, %)	7.4%	4.8%	*5.1%*	6.1%	10.1%	6.0%	11.1%	*5.7%*
Average (MAPE, %)	5.4%	4.9%	*3.4%*	5.6%	7.1%	3.5%	7.7%	*3.0%*

of detection and tracking method. With the proposed KF-based method, the ACF shows the best overall performance (3.4%) and the Faster R-CNN shows the lowest overall performance (5.6%). This is because the Faster R-CNN showed lower detection accuracy than ACF in test-video 1 although the detection accuracy is the critical factor for the KF-based method as shown in the prior section. The average performance of the SSD (4.9%) and the VJ (5.4%) is better than those of the Faster R-CNN, and the SSD achieved the fastest processing time (3.3 f/s). With the KLT feature tracker, however, the Faster R-CNN shows the best average performance (3.0%) with slightly slow speed (1.2 f/s) among all combinations. This is because the tracking precision is more important for collecting the aggregated data than the tracking accuracy, and the Faster R-CNN has the advantage of precision through the bounding box regression [33]. Also, the KLT tracker, which extracts features from the detected bounding box, can effectively supplement the low tracking accuracy. The averaged performance of SSD, which uses the bounding box regression, is also improved from 4.9% to 3.5% with efficient speed (1.8 f/s) while the average performance of ACF and VJ decreases with KLT. These results indicate that it is important to find an optimal combination of detector and tracker depending on the data to be collected.

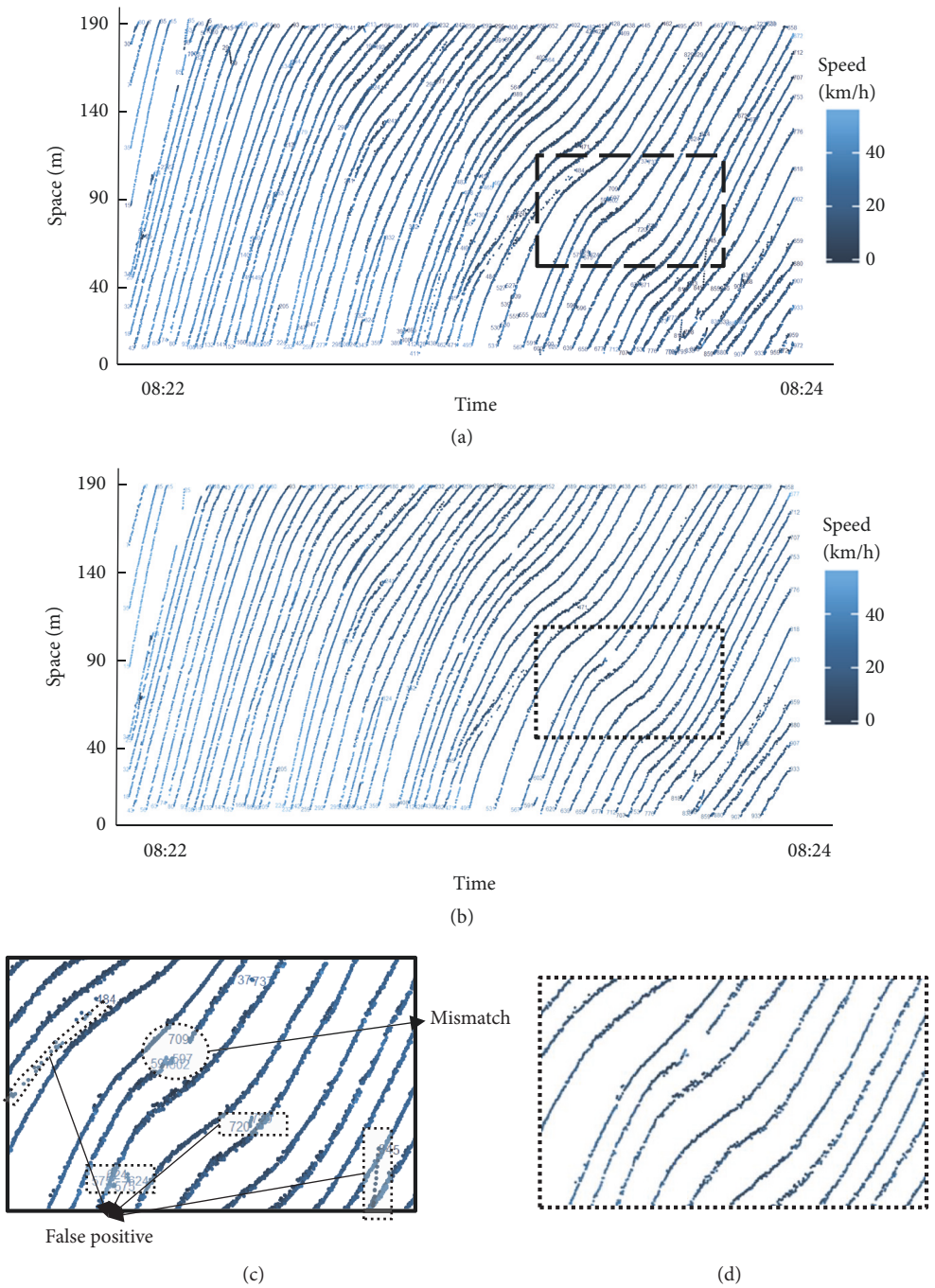

Figure 7: Extracted vehicle trajectories (a) before postprocessing by detection with 94.5% F-measure and (b) after postprocessing by detection with 94.5% F-measure. (c) Tracking errors including false positives and mismatches in trajectory data. (d) Results of postprocessing to remove tracking errors. The numbers in figures indicate the start and end points of each identified vehicle.

Within the test data, vehicle lane change occurred 24 times in total. The best-performed combinations of detectors and trackers, which are the ACF with KF, the SSD with KF, the SSD with KLT, and the Faster R-CNN with KLT, have 4.2% percent error (i.e., only one false positive). These results show that the lane changes can be well detected by the extracted trajectories.

5. Discussion and Conclusions

In this paper, we proposed an efficient and accurate framework for extracting vehicle trajectories recorded by UAVs to overcome the limitations reported in previous studies, i.e., low accuracy in congested traffic, lack of guideline for selecting proper detection method, and lack of detailed

performance analysis. We extended learning-based vehicle detection to tracking by combining postprocessing for vehicle trajectories in congested traffic. By evaluating the performance of the proposed framework on data from a highly congested highway, we found the Faster R-CNN to outperform the ACF in images with large objects and in images with small objects if sufficient data is provided. We note that the ACF showed comparable performance with the Faster R-CNN for small objects with a small sample size. Furthermore, we calculated and evaluated the various aggregated traffic data obtained from extracted vehicle trajectories to examine the applicability of traffic-flow analysis. The results of this study provide a practical guideline for building a framework to extract vehicle trajectories according to the given conditions.

The detection results indicate that the vehicle trajectories achieved as low as a 0.6-m location error with a MOTA of 89.9%, when using 97.4% of the detection results. False positives regarding vehicle detections, caused by road facilities, road signs, shadows, and trucks, were the main reason for errors in the extracted trajectories. In the tracking process, these not only became the false positives of trajectories but also caused mismatches with other trajectories. Their impacts are much greater in congested traffic, because there are so many interactions with other trajectories. In our postprocessing, we removed many of the false positives and mismatches, but some remained. Nevertheless, compared to the results of other studies, our study showed promising performance for extracting trajectories even in a traffic jam [13]. If we have a public dataset on congested traffic, we can readily compare the performance of our methods with those of other studies in future work.

Based on our detailed performance analysis, we discussed the usability and acceptability of the proposed framework for traffic-flow analysis. The sensitivity analysis, in which we varied the number of positive samples used to train a vehicle classifier, showed that it required only 1,000–4,000 positive samples to obtain a detector that performed well. This is a relatively small number of training samples compared with the number used in a previous study [13, 22]. The trajectories extracted based on the vehicle detections had acceptable accuracy. This indicates that the proposed framework may be used easily and efficiently in field studies without needing a vast amount of training data. Also, we showed that the MOTP, which indicates the position error of true positives, was about 0.5–0.6 m, regardless of the detection accuracy. This shows that our framework provided reliable results for true positives, even though it generated false positives, false negatives, and mismatches. In particular, the MOTP values in this study were acceptable for traffic-flow analysis. Considering that the average distance headway (i.e., the reciprocal of density) in this study was about 20 m, the location error was only about 2–3% of the headway. This value is lower than that of the GPS. When a vehicle equipped with a GPS travels, the location error ranges between 3–5 m if the vehicle had a differential GPS (DGPS), and the error is less than 1 m if the vehicle is equipped with a real-time kinetic GPS (RTK-GPS) [51]. Furthermore, we obtained more accurate vehicle trajectories by performing postprocessing to remove both the false positives and mismatches. The high precision of macro- and microscopic traffic data obtained by extracted trajectories also supports that this framework is efficient for practical use and can also provide accurate trajectories for traffic-flow analysis. Also, our comparison results show that finding the appropriate combination of detector and tracker is a more efficient way to improve performance than finding the best detector and tracker, respectively. Therefore, it is necessary to study the detector and tracker with various characteristics to find the optimal combination for tracking vehicles.

Since the overall process of extracting vehicle trajectories is very extensive, we have not addressed all aspects of it in detail, so there is scope for improvement. As shown by the tracking accuracy based on the detection results, the tracking process is also an important factor for improving the quality of vehicle trajectories. We used simple motion constraints in congested traffic to remove the expected error of the tracking process, and this could be improved further by the use of a more sophisticated appearance or motion model, which we plan to do in future research. Details about recently developed models are described in other studies [28, 52]. As mentioned earlier, automated road extraction and camera calibration should replace our simplified preprocessing to realize a fully automated approach when traffic surveillance based on UAVs becomes practical. Also, although we presented a binary classification that distinguishes only whether or not a vehicle is present, multiclass detection and tracking could improve the tracking performance by reducing the number of false positives caused by trucks. This would also extend the relevance of this framework for future traffic dynamics research. A vision-based framework can be affected by weather and the geometric road features. Although we tested our method for two different lighting conditions, in future research, the generality of a UAV-based framework must be verified for use in harsh conditions, such as night-time, fog, and roads that are partially occluded by shadows.

Data Availability

The data used in this research are provided by the Trlab research programme conducted at the Seoul National University, Seoul, Korea. The data will be available when readers ask the authors for academic purposes.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2016R1C1B1008492).

References

[1] D. Chen, J. Laval, Z. Zheng, and S. Ahn, "A behavioral car-following model that captures traffic oscillations," *Transportation Research Part B: Methodological*, vol. 46, no. 6, pp. 744–761, 2012.

[2] D. Chen, S. Ahn, J. Laval, and Z. Zheng, "On the periodicity of traffic oscillations and capacity drop: The role of driver characteristics," *Transportation Research Part B: Methodological*, vol. 59, pp. 117–136, 2014.

[3] S. Oh and H. Yeo, "Impact of stop-and-go waves and lane changes on discharge rate in recovery flow," *Transportation Research Part B: Methodological*, vol. 77, pp. 88–102, 2015.

[4] C. Oh and T. Kim, "Estimation of rear-end crash potential using vehicle trajectory data," *Accident Analysis & Prevention*, vol. 42, no. 6, pp. 1888–1893, 2010.

[5] J. Treiterer and J. A. Myers, "The hysteresis phenomenon in traffic flow," in *Proceedings of the 6th Symposium of Transportation and Traffic Theory*, pp. 13–38, 1974.

[6] C. L. Azevedo, J. L. Cardoso, M. Ben-Akiva, J. P. Costeira, and M. Marques, "Automatic Vehicle Trajectory Extraction by Aerial Remote Sensing," *Procedia - Social and Behavioral Sciences*, vol. 111, pp. 849–858, 2014.

[7] E. N. Barmpounakis, E. I. Vlahogianni, and J. C. Golias, "Extracting kinematic characteristics from unmanned aerial vehicles," *Transpotation Research Board 95th Annual Meeting Compendium of Papers*, vol. 16, no. 3429, 2016.

[8] M. A. Khan, W. Ectors, T. Bellemans, D. Janssens, and G. Wets, "Unmanned aerial vehicle-based traffic analysis: Methodological framework for automated multivehicle trajectory extraction," *Transportation Research Record*, vol. 2626, pp. 25–33, 2017.

[9] X. Cao, C. Gao, J. Lan, Y. Yuan, and P. Yan, "Ego motion guided particle filter for vehicle tracking in airborne videos," *Neurocomputing*, vol. 124, pp. 168–177, 2014.

[10] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research Part C: Emerging Technologies*, vol. 6, no. 4, pp. 271–288, 1998.

[11] Y. Wan, Y. Huang, and B. Buckles, "Camera calibration and vehicle tracking: Highway traffic video analytics," *Transportation Research Part C: Emerging Technologies*, vol. 44, pp. 202–213, 2014.

[12] "NGSIM—Next generation simulation," 2006, http://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm.

[13] Y. Xu, G. Yu, X. Wu, Y. Wang, and Y. Ma, "An Enhanced Viola-Jones Vehicle Detection Method from Unmanned Aerial Vehicles Imagery," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1845–1856, 2017.

[14] V. Alexiadis, J. Colyar, J. Halkias, R. Hranac, and G. McHale, "The next generation simulation program," *Institute of Transportation Engineers Journal*, vol. 74, no. 8, pp. 22–26, 2004.

[15] J. Wang, X. Rui, X. Song, X. Tan, C. Wang, and V. Raghavan, "A novel approach for generating routable road maps from vehicle GPS traces," *International Journal of Geographical Information Science*, vol. 29, no. 1, pp. 69–91, 2015.

[16] A. D. Patire, M. Wright, B. Prodhomme, and A. M. Bayen, "How much GPS data do we need?" *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 325–342, 2015.

[17] H. Grabner, T. T. Nguyen, B. Gruber, and H. Bischof, "On-line boosting-based car detection from aerial images," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 63, no. 3, pp. 382–396, 2008.

[18] R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui, and Y. Wang, "Real-Time Bidirectional Traffic Flow Parameter Estimation from Aerial Videos," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 4, pp. 890–901, 2017.

[19] S. Tuermer, F. Kurz, P. Reinartz, and U. Stilla, "Airborne vehicle detection in dense urban areas using HoG features and disparity maps," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 6, pp. 2327–2337, 2013.

[20] K. Liu and G. Mattyus, "Fast Multiclass Vehicle Detection on Aerial Images," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 9, pp. 1938–1942, 2015.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, 2012.

[22] Y. Xu, G. Yu, Y. Wang, X. Wu, and Y. Ma, "Car detection from low-altitude UAV imagery with the faster R-CNN," *Journal of Advanced Transportation*, vol. 2017, p. 10, 2017.

[23] T. Tang, S. Zhou, Z. Deng, H. Zou, and L. Lei, "Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining," *Sensors*, vol. 17, no. 336, 2017.

[24] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. van den Hengel, "A survey of appearance model in visual object tracking," *ACM Transactions on Intelligent Systems and Technology*, vol. 4, no. 58, 2013.

[25] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '16)*, pp. 4293–4302, 2016.

[26] A. Andriyenko and K. Schindler, "Multi-target tracking by continuous energy minimization," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011*, pp. 1265–1272, USA, June 2011.

[27] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: a survey," *ACM Computing Surveys*, vol. 38, no. 4, pp. 1–45, 2006.

[28] W. Luo, X. Zhao, and T.-K. Kim, "Multiple object tracking: A review," *Computer Vision and Pattern Recognition*, vol. 1, Article ID 1409.7618, 2014.

[29] N. K. Kanhere and S. T. Birchfield, "A taxonomy and analysis of camera calibration methods for traffic monitoring applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 2, pp. 441–452, 2010.

[30] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. I511–I518, December 2001.

[31] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 580–587, Columbus, Ohio, USA, June 2014.

[32] R. Girshick, "Fast R-CNN," in *Proceedings of the 15th IEEE International Conference on Computer Vision (ICCV '15)*, pp. 1440–1448, December 2015.

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

[34] P. Dollar, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.

[35] H. Zhou, H. Kong, L. Wei, D. Creighton, and S. Nahavandi, "Efficient road detection and tracking for unmanned aerial vehicle," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 297–309, 2015.

[36] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features," in *Proceedings of the British Machine Vision Conference (BMVC '09)*, pp. 56–68, London, UK, September 2009.

[37] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: an evaluation of the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2012.

[38] C. Zhang and P. Viola, "Multiple-instance pruning for learning efficient cascade detectors," in *Proceedings of theAdvances in Neural Information Processing Systems (NIPS)*, pp. 1681–1688, 2008.

[39] R. Appel, T. Fuchs, and P. Dollar, "Quickly boosting decision trees - pruning underachieving features early," in *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, pp. 594–602, 2013.

[40] G. Welch and G. Bishop, "An introduction to the Kalman filter," Tech. Rep., Department of Computer Science, University of North Carolina, Chapel Hill, NC, USA, 2004.

[41] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society For Industrial & Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[42] S. Agarwal, A. Awan, and D. Roth, "Learning to detect objects in images via a sparse, part-based representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1475–1490, 2004.

[43] K. Bernardin, A. Elbs, and R. Stiefelhagen, "Multiple object tracking performance metrics and evaluation in a smart room environment," in *Proceedings of the in Sixth IEEE International Workshop on Visual Surveillance, in conjunction with ECCV*, vol. 90, 2006.

[44] L. C. Edie, "Discussion of traffic stream measurements and definitions," in *Proceedings of the 2nd International Symposium on the Theory of Traffic Flow*, pp. 139–154, 1965.

[45] P. Dollár, "Piotr's computer vision matlab toolbox (PMT)," 2014, http://vision.ucsd.edu/pdollar/toolbox/doc/index.html.

[46] J. Huang, V. Rathod, C. Sun et al., "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '17)*, pp. 7310–7319, 2017.

[47] L. Zhang, L. Lin, X. Liang, and K. He, "Is faster R-CNN doing well for pedestrian detection?" in *Proceedings of the European Conference on Computer Vision (ECCV '16)*, pp. 443–457, 2016.

[48] X. Zhu, C. Vondrick, D. Ramanan, and C. C. Fowlkes, "Do we need more training data or better models for object detection?" in *Proceedings of the 2012 23rd British Machine Vision Conference, BMVC 2012*, UK, September 2012.

[49] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot multibox detector," in *Proceedings of the European Conference on Computer Vision (ECCV)*, vol. 9905, pp. 21–37, 2016.

[50] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 593–600, June 1994.

[51] N. M. Thamrin, N. H. M. Arshad, R. Adnan et al., "Simultaneous localization and mapping based real-Time inter-row tree tracking technique for unmanned aerial vehicle," in *Proceedings of the 2012 IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2012*, pp. 322–327, Malaysia, November 2012.

[52] A. Milan, L. Leal-Taixe, K. Schindler, and I. Reid, "MOT16: A benchmark for multi-object tracking," *Computer Vision and Pattern Recognition*, vol. 1, Article ID 1603.00831, pp. 5397–5406, 2016.