Monocular Visual Traffic Surveillance: A Review

Xingchen Zhang, *Member, IEEE*, Yuxiang Feng, *Member, IEEE*, Panagiotis Angeloudis, and Yiannis Demiris, *Senior Member, IEEE*

***Abstract*— To facilitate the monitoring and management of modern transportation systems, monocular visual traffic surveillance systems have been widely adopted for speed measurement, accident detection, and accident prediction. Thanks to the recent innovations in computer vision and deep learning research, the performance of visual traffic surveillance systems has been significantly improved. However, despite this success, there is a lack of survey papers that systematically review these new methods. Therefore, we conduct a systematic review of relevant studies to fill this gap and provide guidance to future studies. This paper is structured along the visual information processing pipeline that includes object detection, object tracking, and camera calibration. Moreover, we also include important applications of visual traffic surveillance systems, such as speed measurement, behavior learning, accident detection and prediction. Finally, future research directions of visual traffic surveillance systems are outlined.**

***Index Terms*— Visual traffic surveillance, camera calibration, speed measurement, accident detection, accident prediction.**

I. Introduction

VISUAL traffic surveillance systems have attracted widespread interest [1]–[6]. They have been widely adopted in vehicle type recognition [7], [8], traffic flow estimation [9], headway [10] and speed [1], [11] measurement, accident detection [12], [13] and prediction [2], and training data generation for behavior learning [14], [15]. However, the performance of traditional systems has been restricted by traditional object detection and tracking methods that used hand-crafted features. Thanks to the innovations in deep learning algorithms, new breakthroughs have emerged in object detection [16], [17], tracking [18], and classification [19], leading to substantial performance improvements. These breakthroughs have been increasingly incorporated in recently developed visual traffic surveillance systems [20]–[23].

Although there are a few earlier review papers on visual traffic surveillance, they have some key limitations. For example, Datondji *et al.* [6] carried out a detailed survey of visual traffic monitoring of road intersections. It was, however, published in 2016, and thus did not incorporate many of the seminal studies that followed, which are based on deep learning methods. For instance, the object detection and tracking methods covered in that paper were traditional methods using hand-crafted features. Furthermore, key considerations such as camera calibration, speed measurement, and accident detection, were not covered. Yang *et al.* [24] provided a comprehensive review of vehicle detection methods under various environments and the applications of vehicle detection in intelligent transportation systems. However, it focused on vehicle detection methods and thus did not cover traffic camera calibration, vehicle tracking, and speed estimation, etc. A more recent review by Yuan *et al.* [25] focused on machine learning techniques for next-generation intelligent transportation systems, but deviated from the area of visual traffic surveillance.

With the increasing research interest in deep learning-based visual traffic surveillance, many studies have been published in recent years. Therefore, this paper aims to provide a systematic review of these new methods. In summary, the main contributions of this paper are identified as follows:

- An up-to-date and comprehensive review of visual traffic surveillance systems is presented, starting with a description of common system architectures, and followed by an in-depth description of individual components.
- Three primary applications of visual traffic surveillance systems are reviewed, namely speed measurement, behavior learning, accident detection and prediction. A brief summary of popular traffic surveillance video datasets is also provided.
- The main performance evaluation methods are reviewed and future research directions on visual traffic surveillance systems are outlined.

The remainder of this paper is organized as follows. Section II introduces the overall architecture of visual traffic surveillance systems. Then, object detection and tracking methods are discussed in Section III and IV, respectively. Since camera calibration is essential in most visual traffic surveillance systems, relevant methods are reviewed in Section V. Common applications are summarized in Sections VI, VII, VIII, focusing on speed measurement, behavior learning, and accident detection and prediction, respectively. Section IX summarizes the main traffic datasets used by researchers and Section X discusses performance evaluation methods, followed by the discussion of future prospects in Section XI. Finally, Section XII concludes this paper.

Manuscript received 24 August 2021; revised 3 January 2022; accepted 19 January 2022. Date of publication 15 February 2022; date of current version 12 September 2022. The research in this paper was supported in part by the InnovateUK DRISK project, and in part by a Royal Academy of Engineering Chair in Emerging Technologies. The Associate Editor for this article was Z. Duric. *(Corresponding author: Xingchen Zhang.)*

Xingchen Zhang and Yiannis Demiris are with the Personal Robotics Laboratory, Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: xingchen.zhang@imperial.ac.uk; y.demiris@imperial.ac.uk).

Yuxiang Feng and Panagiotis Angeloudis are with the Transport Systems and Logistics Laboratory, Department of Civil and Environmental Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: y.feng19@imperial.ac.uk; p.angeloudis@imperial.ac.uk).

Digital Object Identifier 10.1109/TITS.2022.3147770

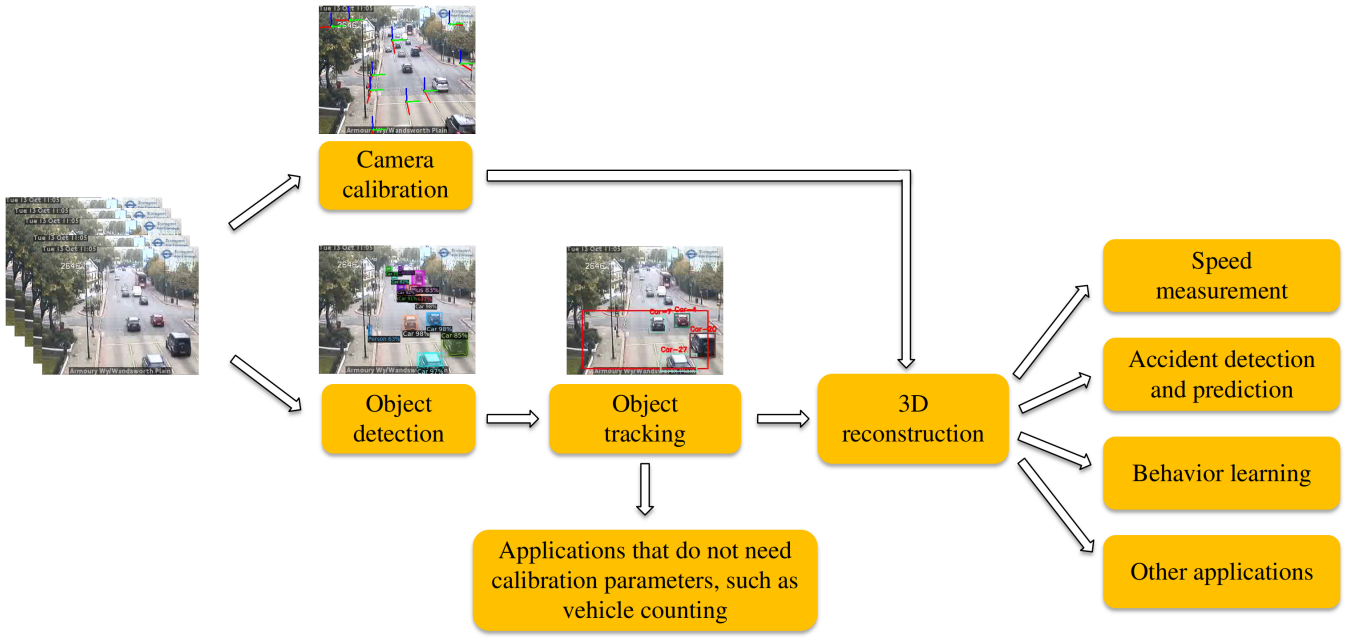


Fig. 1. The overall architecture of a visual traffic surveillance system consists of object detection, object tracking, camera calibration, 3D reconstruction, and applications. Camera calibration is not needed in some applications that only require 2D information. By contrast, in applications that require 3D information or real-world units, such as speed estimation and distance measurement, camera calibration is a necessary step. Video inputs are obtained from Transport for London's (TfL) traffic cameras (https://www.tfljamcams.net, Powered by TfL Open Data).

II. The Overall Architecture of Visual Traffic Surveillance Systems

Figure 1 shows the overall architecture of visual traffic surveillance systems, which usually consists of the following components:

- **Object detection**: Identifies vehicles, and other road users if necessary, in each frame alongside their type, position, bounding box, or mask.
- **Object tracking**: Links detection results in consecutive frames to derive vehicle trajectories.
- **Camera calibration**: Obtains camera parameters, i.e., intrinsic parameters, extrinsic parameters, and scene scale. These parameters are needed to perform 3D reconstruction and obtain real-world measurements. Calibration is not necessary if only 2D information is required.
- **3D reconstruction**: Projects a 2D image to 3D space, which is necessary for many downstream applications.
- **Applications**: Main applications include vehicle counting, speed measurement, behavior learning, accident detection and prediction, etc.

Some recently proposed visual traffic surveillance systems are summarised in Table I, from which we can make several observations. First, some methods focus on 2D detection and tracking, thus camera calibration is not needed. Second, most studies focus on vehicle detection and tracking. Only a few methods are capable of detecting other road users, such as pedestrians and cyclists. Third, most studies adopt existing general object detection and tracking methods directly. Finally, regarding accident detection and prediction, many studies have used learning-free methods due to the lack of appropriate training data. Meanwhile, distance, overlapping, kinematic patterns, speed, and trajectory deviation are commonly used criteria in these studies.

III. Object Detection

The purpose of object detection is to identify vehicles, and other road users if necessary, in each image frame. Typical outputs include object classes, positions, bounding boxes, or masks. The performance of object detection methods significantly impacts tracking and downstream applications. Before deep learning was applied to object detection, traditional detection methods were utilized, such as background subtraction [36] and image difference [1]. In recent years, deep learning has significantly improved the performance of object detection [17], [37], [38]. Consequently, most visual traffic surveillance systems have adopted deep learning-based detectors in the past several years, as revealed in Table I, while very few studies still used background subtraction [33].

In this section, we review the object detection methods employed in visual traffic surveillance systems recently. We also discuss some measures that can improve detection performance in the context of traffic surveillance.

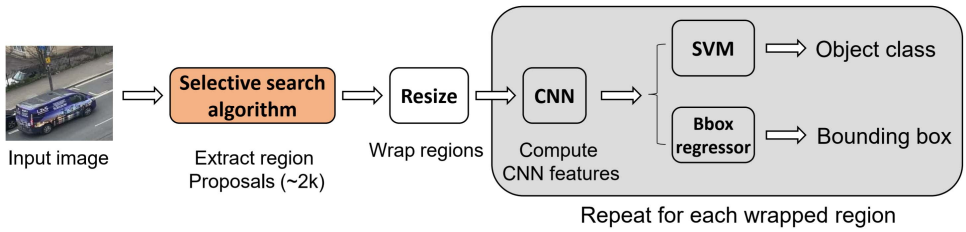
A. Deep Learning-Based Object Detection Methods

As can be seen from Table I, most recent visual traffic surveillance systems are based on deep learning methods such as SDD [39], YOLO [17], Faster R-CNN [37] and Mask R-CNN [38]. These methods can be classified into two-stage methods and one-stage methods. Two-stage methods use selective search algorithms [40] or region proposal networks (RPN) to generate proposals for region of interests (ROI). Afterwards,

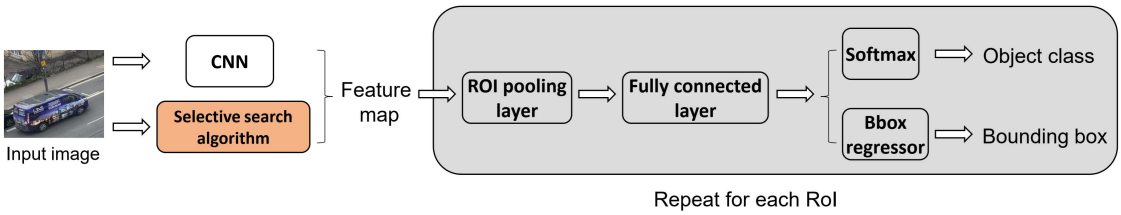
TABLE I

SOME EXISTING VISUAL TRAFFIC SURVEILLANCE SYSTEMS. THE MAIN COMPONENTS OF EACH VISUAL TRAFFIC SURVEILLANCE SYSTEM ARE SUMMARIZED FOR COMPARISON. 'N/A' INDICATES THAT THIS COMPONENT IS NOT CONTAINED IN THE VISUAL TRAFFIC SURVEILLANCE SYSTEM

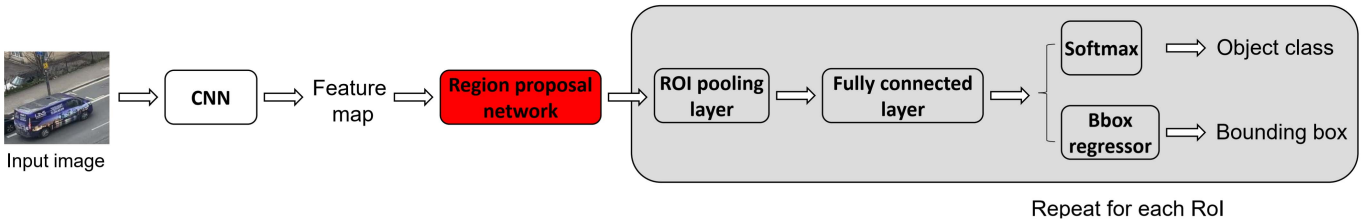
Ref.	Year	2D/3D	Object	Calibration	Detector	Tracker	Trajectory post-processing	Speed measurement	Accident detection	Accident prediction	Behavior learning
Dubska et al. [26]	2014	3D	Vehicles	Automatic methods (vanishing points)	N/A	N/A	N/A	Distance divided by time	N/A	N/A	N/A
Sochor et al. [27]	2017	3D	Vehicles	Automatic methods (vanishing points)	Faster R-CNN	Background subtraction Kalman filter	N/A	Distance divided by time	N/A	N/A	N/A
TCP [14]	2018	2D	Vehicles	N/A	SSD	Re^{-3}	N/A	N/A	N/A	N/A	Multivariate Gaussian distribution
Chang et al. [28]	2018	3D	Vehicles	Manual methods (Reference image-based) (Google Maps)	Faster R-CNN	A method similar to SORT	N/A	Median filtering Gaussian smoothing	N/A	N/A	N/A
Yu et al. [13]	2018	3D	Vehicles	Manual method (vanishing points)	Mask R-CNN	Deep SORT	N/A	FPS exponential smoothing	Distance or overlap	Distance or overlap	N/A
Tang et al. [20]	2018	3D	Vehicles	Manual method (hybrid method)	YOLOV2	Bottom-up clustering	N/A	Distance of two cameras	N/A	N/A	N/A
Clausse et al. [21]	2019	3D	Vehicles	Manual method (reference image-based) (satellite image)	Mask R-CNN	IOU	Rauch-Tung-Striebel smoother	FPS	N/A	N/A	N/A
Yu et al. [29]	2019	3D	Vehicles	Manual method (vanishing points)	Mask R-CNN	Deep SORT	N/A	Kalman filter	Kinematic patterns	N/A	N/A
Ijjina et al. [30]	2019	2D	Vehicles	N/A	Mask R-CNN	Centroid Tracking	N/A	FPS & distance between 5 frames	Three criteria	N/A	N/A
Wang et al. [22]	2019	2D	Vehicles	N/A	YOLO	Kalman filter	N/A	Number of frames it appears in	Speed	N/A	N/A
Seong et al. [31]	2019	2D	Vehicles	N/A	YOLOV2	Kalman filter and IOU tracker	Trajectory correction using optimal bounding box	N/A	N/A	N/A	N/A
Yu et al. [32]	2019	2D	Vehicles	N/A	N/A	N/A	N/A	N/A	Weighted extreme learning machine based detector	N/A	N/A
Singh et al. [33]	2019	2D	Road users	N/A	Background subtraction	Background subtraction	N/A	N/A	Denosing autoencoder	N/A	N/A
ViBe [15]	2019	3D	Road users	Manual method (reference image-based) (satellite image)	Mask R-CNN	A method similar to Deep SORT	N/A	N/A	N/A	N/A	Horizon GAIL
Huang et al. [34]	2020	2D	Vehicles	N/A	YOLO	Deep SORT	N/A	N/A	Distance measure	N/A	N/A
Zhang et al. [35]	2020	3D	Vehicles	Manual method (vanishing points)	Mask R-CNN	SORT	N/A	Travel distance in ROI divided by time (FPS)	N/A	N/A	N/A



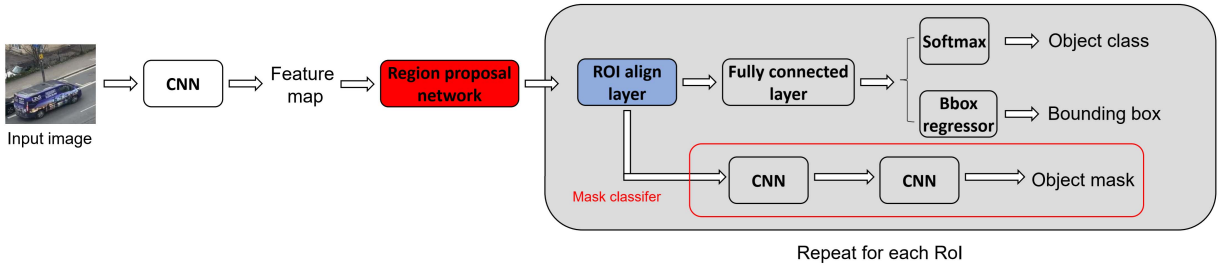
(a) The architecture of R-CNN [16]. An additional selective search algorithm is required to provide region proposals. Each region proposal is resized and sent to a CNN for feature extraction.



(b) The architecture of Fast R-CNN [42]. An additional selective search algorithm is required to provide region proposals. In contrast to R-CNN, the whole input image is sent to a CNN for feature extraction.



(c) The architecture of Faster R-CNN [37]. In contrast to R-CNN and Fast R-CNN, the selective search algorithm is not needed. A region proposal network is contained in the framework, and the whole pipeline can be trained in an end-to-end manner.



(d) The architecture of Mask R-CNN [38]. An additional branch is added to generate object masks. An ROI align layer is also designed to solve the alignment issue between feature maps and original images in Faster R-CNN. Mask R-CNN can also be trained in an end-to-end manner.

Fig. 2. Examples of two-stage object detection methods. From R-CNN to Mask R-CNN, two-stage object detection methods have been improved significantly to provide better object detection results and richer information.

a classifier is employed to process these proposals and generate detection results. By contrast, one-stage methods run detection directly using a dense sampling of possible locations. In the aforementioned detection methods, Faster R-CNN and Mask R-CNN are two-stage methods, while SDD, YOLO, and YOLOV2 are one-stage methods. In this section, we briefly introduce these detection methods.

1) Two-Stage Methods:

a) R-CNN: R-CNN [16] is a milestone in object detection. It is the first method showing that a CNN can lead to significantly better object detection performance than methods based on histogram of oriented gradients (HOG)-like features. As shown in Fig. 2(a), the main steps of R-CNN are as follows. First, around 2000 region proposals are extracted from the input image using the selective search algorithm [40]. Second, each region proposal is resized as 227×227 to facilitate the feature extraction with CNN. Third, these features are classified using class-specific Support Vector Machines (SVMs). R-CNN achieved 30% relative improvements on PASCAL VOC 2012 [41]. However, it suffers from three primary limitations. First, the CNN-based feature extraction of all 2000 region proposals is very time-consuming. Second, the adopted CNN requires fixed-size inputs. Third, R-CNN can not be trained in an end-to-end manner. Therefore, extensive efforts have been devoted to improve R-CNN.

b) Fast R-CNN: To mitigate the aforementioned deficiencies of R-CNN, Fast R-CNN [42], as shown in Fig. 2(b), was proposed with three major improvements. First, the whole

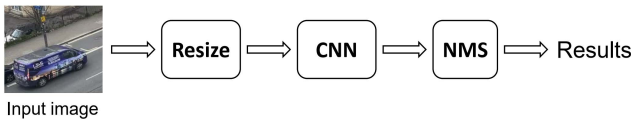
image is directly sent to a CNN for feature extraction. Therefore, feature extraction is only performed once. Second, ROIs are generated using the selective search algorithm and identified on the feature map. Afterwards, an ROI pooling layer and a sequence of fully connected layers are used sequentially to map each ROI to a feature vector. Third, Fast R-CNN has a multitask architecture and can be trained using a multitask loss function. Fast R-CNN has achieved a 4% improvement in mean average precision (mAP) on PASCAL VOC 2012 over R-CNN and is nine times faster than R-CNN.

c) Faster R-CNN: Although Fast R-CNN has achieved significant improvements in terms of accuracy and speed, it is still not an end-to-end solution. This is because it requires an additional stage for the generation of region proposals. To solve this issue, Ren *et al.* [37] proposed Faster R-CNN, whose architecture is shown in Fig. 2(c). It contains a region proposal network (RPN) that can generate region proposals. With this architecture, Faster R-CNN can be trained in an end-to-end manner, resulting in a significant speed improvement. It should be noted that Faster R-CNN is the first end-to-end deep learning-based detection method. Owing to its advantages, Faster R-CNN has been frequently used in recent visual traffic surveillance systems. For example, Chang *et al.* [28] adopted Faster R-CNN in their system to detect vehicles.

d) Mask R-CNN: Object masks are also desirable outputs of object detection methods. For example, a mask and a 3D bounding box projection are utilized by Clausse *et al.* [21] to compute the vehicle's center and orientation. Therefore, He *et al.* [38] proposed the Mask R-CNN, which outputs both bounding boxes and object masks. The architecture of Mask R-CNN is shown in Fig. 2(d). In addition to the existing branch for bounding box regression in Faster R-CNN, it adds a parallel branch for object mask prediction. Moreover, a ROI align layer is proposed to fix the misalignment between feature maps and original images in Faster R-CNN. It can reach 5 FPS on an Nvidia Tesla M40 GPU [38]. Because of these properties, Mask R-CNN has become the most frequently used detection method in recent visual traffic surveillance systems, as can be noted from Table I.

2) One-Stage Methods:

a) YOLO: Unlike those R-CNN series detection methods, YOLO [17] is a one-stage method, i.e., it does not need to generate region proposals first and then classify them. YOLO formulates object detection as a regression problem and uses a single CNN to predict bounding boxes and the associated class probabilities in one single evaluation. As shown in Fig. 3(a), YOLO consists of three main stages. First, the input image is resized to 448×448. Second, a CNN is run on the whole image. Third, non-maximal suppression (NMS) is performed to remove multiple detections and obtain the final results. In addition, YOLO divides the input image to $S \times S$ grids. For each grid, it predicts B bounding boxes and a confidence value. Therefore, each bounding box has five values (coordinates of the center, height, weight, and confidence) and C class probabilities. Consequently, the final prediction of YOLO is a $S \times S \times (5B + C)$ tensor.



(a) The pipeline of YOLO [17]. Region proposal networks are not needed in YOLO, and object detection is formulated as a regression problem.

(b) The architecture of SSD [39]. Multi-scale features are utilized in SSD to help handle objects of different sizes.

Fig. 3. Two typical examples of one-stage object detection methods. In one-stage methods, region proposal networks are not needed. Instead, these methods run detection directly using a dense sampling of possible locations.

YOLO is a real-time detection method, and can reach 45 FPS on a Titan X GPU. A smaller version of YOLO, i.e., Fast YOLO, can reach 155 FPS at the cost of slightly worse mAP than YOLO. Therefore, several studies [22], [34] have adopted YOLO for vehicle detection. However, YOLO generates more localization errors compared to other state-of-the-art methods. In addition, the detection of small objects belonging to the same group can be less accurate due to its strict spatial constraints on bounding box prediction.

b) YOLOV2: Based on YOLO, Redmon *et al.* [43] proposed YOLOV2 that has several improvements. First, the fully connected layers in YOLO are removed, and anchor boxes are introduced to predict the bounding boxes. In this way, over 1000 bounding boxes can be predicted for an image, which increase significantly from the 98 bounding boxes predicted in YOLO. Second, YOLOV2 resizes the input image to 416×416 instead of 448×448 in YOLO, thus an odd number of locations in the feature map can be obtained. Correspondingly, the image is divided to 13×13 grids instead of 7×7 in YOLO. Third, YOLOV2 can be jointly trained on the COCO detection dataset [44] and the ImageNet classification dataset [19] by using WordTree [43] to combine data and a joint optimization technique. It can hence detect objects that do not have labeled detection data. For example, YOLOV2 achieves 16.0 mAP on the 156 classes that it does not have labeled detection data for on the ImageNet detection validation set.

Owing to these advantages, it has also been increasingly adopted in visual traffic surveillance systems in recent years. For example, Tang *et al.* [20] used YOLOV2 to detect vehicles from traffic images. To improve the performance of YOLOV2 in the context of traffic surveillance, they selected 4,500 frames from the NVIDIA AI City Challenge and manually labeled eight categories, i.e., sedan, hatchback, bus, pickup, minibus, van, truck, and motorcycle, for training data. The pre-trained weights [43] were utilized to initialize the network. In addition, Seong *et al.* [31] also adopted YOLOV2 to detect vehicles in their system.

c) SSD: SSD [39] is another one-stage detection method that has been applied in visual traffic surveillance system [14]. SSD consists of a feature extractor (base network)

and additional layers. The feature extractor can be different networks, such as VGG [45], InceptionNet [46] or MobileNet [47]. The additional layers generate feature maps of different sizes. A key contribution of SSD is that it performs detection on these various feature maps. The SSD with a 300×300 input size (SSD300) outperforms YOLO in both frame rate (59FPS versus 45 FPS) and detection performance on VOC 2007 test set. Another variant of SSD (SSD512) obtains better detection performance at the cost of frame rate (22 FPS).

3) Summary of Object Detection Methods:

Two-stage methods are generally more accurate than one-stage methods. However, their computational costs are also higher and thus difficult for real-time applications. By contrast, one-stage methods are faster and can meet the real-time requirement. Considering these trade-offs, the 4 best-performing teams that participated in track 1 of the 5th AI City Challenge adopted YOLO-derived models to balance out effectiveness versus efficiency [48].

Along with the aforementioned methods, some other deep learning-based detection methods have also been employed in visual traffic surveillance systems. For example, Kocur *et al.* [49] adopted CenterNet [50], which models an object as a single point, for vehicle detection. Moreover, many other advanced detection methods have been proposed, such as YOLOV3 [51] and YOLOV4 [52]. However, to the best of our knowledge, the application of these detection methods in visual traffic surveillance systems has not been formally published at the time of this review.

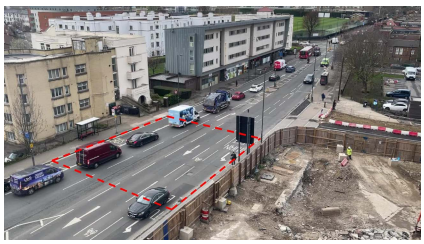
It should be mentioned that although most recent visual traffic surveillance studies used deep learning-based object detection methods, some studies [33] still employed background subtraction when the camera is fixed. Also, there is an opportunity to combine background subtraction and deep learning methods. For example, Messoussi *et al.* [53] chose the SpotNet [54] as the detection method, in which the segmentation branch was trained with semi-supervised segmentations obtained using background subtraction when the camera is fixed.

B. Methods to Improve Detection Performance

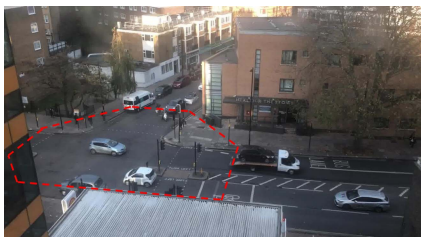
Along with the adopted method, some other factors can also affect the performance of object detection. Examples of these factors are tiny vehicles, vehicles that are not in the road area, and occlusion. To mitigate the influence of these factors, some measures have been proposed in existing research.

1) Region of Interest: A typical solution is to set a region of interest (ROI) in traffic videos. ROI is very useful for excluding noise, for example, tiny vehicles. In addition, if the dimension of a ROI is predetermined, it can also be utilized for vehicle speed estimation. Various ROIs have been designed in existing studies, including regular and irregular ROIs.

a) Regular ROIs: A regular ROI is usually a rectangular area set on the road plane. It is suitable for structured roads, for example, the straight road shown in Fig. 4(a). It has been used by Zhang *et al.* [35] for vehicle detection and tracking. In addition, they utilized the travel distance inside the ROI to compute vehicles' speed.



(a) An example of regular ROIs. Only vehicles inside this rectangle ROI are detected and tracked.



(b) An example of irregular ROIs. Only vehicles inside this ROI are detected and tracked.

Fig. 4. Examples of ROIs. These ROIs are widely adopted in visual traffic surveillance systems to improve object detection performance. Irregular ROIs are more flexible compared with regular ROIs.

b) Irregular ROIs: An example of irregular ROIs is shown in Fig. 4(b). The main benefit of irregular ROIs is that they are more flexible and can be applied to unstructured scenes [13], [21], [22].

2) Filter: Another measure to improve the detection performance is through post-processing filters. For example, Yu *et al.* [13] applied a filter to the detection results provided by Mask R-CNN. Their filter contains three sets of rules, each aims to exclude small vehicles, vehicles outside the road area, and occluded vehicles, respectively.

3) Retraining Detection Methods: The detection methods mentioned in Section III-A are initially designed for general purpose object detection. Therefore, they were trained using datasets that contain many kinds of objects. For instance, Mask R-CNN was trained using the COCO dataset [44], which contains more than 80 classes of objects. It should be noted that these pre-trained models may not perform well in the traffic surveillance context. Therefore, the third measure to improve the detection performance is to retrain these models with traffic datasets. For example, Sochor *et al.* [27] retrained the Faster R-CNN using the COD20K dataset [55]. Chang *et al.* [28] retrained the Faster R-CNN using the UA-DETRAC dataset and used a COCO pre-trained model for initialization. Seong *et al.* [31] retrained the YOLOV2 model using a self-collected vehicle dataset. In addition, retraining a model is necessary for some specific applications when the general-purpose detection methods are not applicable. For example, Zhang *et al.* [35] proposed to detect wheels of vehicles. However, this is not feasible using the Mask R-CNN model trained with the COCO. To solve

this issue, they collected a dataset consisting of cars, coaches, trucks, and wheels to retrain the model.

4) Position Correction Methods: In many studies [14], [31], the detected 2D bounding boxes are used to denote the position of vehicles. In particular, a point inside the 2D bounding box, for example, the center, is treated as the vehicle's position. However, several factors, such as camera view angle and the loss of depth information, can cause a deviation between this point and the vehicle's actual position. Researchers have proposed different methods to solve this issue. For example, Ren *et al.* [14] manually labeled a dataset with actual vehicle locations. Afterwards, a linear mapping between detection and labeled results was acquired for position correction. Seong *et al.* [31] used the determination of optimal bounding boxes for correction. Along with these 2D-based methods, some studies [27], [35], [56] have tried to adopt 3D bounding boxes and then derive the position of vehicles from them.

5) Methods for Dealing With Vehicle Occlusion: Vehicle detection with occlusion is a challenging task. Many approaches have been proposed to deal with vehicle occlusion. For example, Pham *et al.* [57] developed a vehicle detection method based on car windshield appearance because they thought the windshield is the most suitable cue in handling occlusion situations. For deep learning-based detection methods, soft non-maximum suppression (NMS) [58] was adopted to deal with heavy vehicle occlusion [59]. In addition, occlusion-focus region proposal algorithm [60] based on the Faster R-CNN detector was recently proposed to handle vechile occlusion. Moreover, occluded vehicles with accurate annotations can be added to the training data to improve the robustness of deep learning-based detectors to occlusion. Note that vehicle occlusion can also be handled during the tracking process. For example, Fang *et al.* [61] proposed a part-based method, which combines a part-based strategy with a particle filter, to deal with vehicle occlusion. Finally, 3D object detection method [62] may be an efficient way to deal with vehicle occlusion.

IV. Object Tracking

Object tracking in visual traffic surveillance aims to extract trajectories of multiple road users simultaneously. In recent years, most tracking methods used the tracking-by-detection architecture. Specifically, object detection is first performed on each frame to obtain corresponding detection results, which will then be associated to form the trajectories. Afterwards, post-processing is usually needed to smooth and improve the obtained trajectories. In this section, we first introduce the commonly used trackers in recent visual traffic surveillance systems and then briefly discuss relevant post-processing steps.

A. Multi-Object Tracking Methods

Multi-object tracking methods have been employed in visual traffic surveillance systems for many years. Before deep learning was introduced to the tracking field, many traditional methods have been proposed [63]. Some traditional tracking methods were still used nowadays, such as centroid tracking [30], background subtraction [27], [33], and Kalman filter [27]. For more details about traditional tracking methods, please refer to this excellent survey [6].

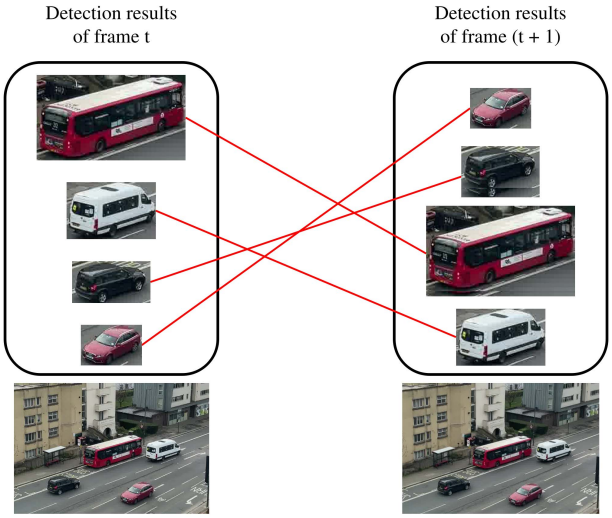


Fig. 5. Detection results of frame t (left) need to be matched with the detection results of frame $(t+1)$ (right) to form tracklets (short trajectories).

1) Matching Algorithms: In a tracking-by-detection architecture, it is crucial to match detection results in adjacent frames, as shown in Fig. 5. The Hungarian algorithm [64] and the Kuhn-Munkres algorithm [64] are commonly used in existing studies. The Hungarian algorithm [64] can match detection results in two different frames. It has been utilized in many multi-object tracking methods. For example, Chang *et al.* [28] used the Hungarian algorithm to associate vehicle detection results with existing tracks. However, the Hungarian algorithm treats each candidate equally. Therefore, the matching results may not be optimal. To overcome its deficiency, the Kuhn-Munkres Algorithm was proposed [64]. To differentiate the candidates, the Kuhn-Munkres algorithm assigns weights to detection results and the relationship between them. However, it should be noted that while the Kuhn-Munkres algorithm can achieve better matching results, it is more time-consuming than the Hungarian algorithm. Therefore, users need to balance the matching results and the computational costs when choosing matching algorithms.

2) IOU Method: The Intersection Over Union (IOU) [65] is another tracking method, which associates the detection results using their spatial overlaps between different frames, as shown in Fig. 6. For example, Clausse *et al.* [21] adopted the IOU method to group detection results into tracks. Specifically, the masks that have high overlap in successive frames are formed into tracks. Although the IOU method is simple and can work in many situations, it has some limitations. First, it may fail when vehicles are heavily occluded, e.g., when the density of vehicles on the road is high. Second, it has a requirement on video frame rate, which should be higher than the vehicle dynamics [21].

3) SORT: Simple online and real-time tracking (SORT) [66] is a representative online multi-object tracking method, which combines the Kalman Filter [67] and the Hungarian

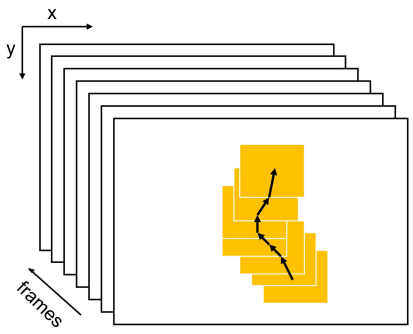


Fig. 6. The principle of the IOU tracker [65]. The detection results in different frames are associated using the spatial overlap between the bounding boxes in different frames. If the IOU is larger than a chosen threshold, the two bounding boxes are considered as the same object.

algorithm [64]. In SORT, only the size and position of bounding boxes are used for motion estimation and data association. The Kalman filter is used to handle motion prediction, and the Hungarian algorithm is employed for data association.

The detection result in the next frame is associated with the predicted result using the Hungarian algorithm [64], whose assignment cost matrix is computed using the IOU metric. If the association is successful, the detection result will be used to update the target state. In the update process, the velocity components are solved using a Kalman filter. However, if the association fails, i.e., no detection result is associated with the predicted result, the target state is simply updated using a linear constant velocity model. In this model, the state of each target, i.e., the bounding box, is represented as:

$$x = [u, v, s, r, \dot{u}, \dot{v}, \dot{s}]^T, \tag{1}$$

where u and v are the horizontal and vertical position of the center of the bounding box, s and r are the scale and aspect ratio of the bounding box, $\dot{u}, \dot{v}, \dot{s}$ are the changing velocities of these variables. Note that the aspect ratio is constant in SORT.

In SORT, if the IOU of a new detection with existing detection results is smaller than IOU_{min}, this new detection is considered as a new target. If a track is not detected for one frame, the track is terminated. Due to its simplicity, SORT is very fast and achieves 260 FPS on single core of an Intel i7 250GHz computer, thus can be applied to real-time applications. However, due to the small track termination threshold, SORT has many ID switches. In addition, SORT only uses the size and positions of bounding boxes to perform matching, thus the tracking is less robust.

4) Deep SORT: Deep SORT [68] is an improved version of SORT with several improvements. First, a CNN is trained to extract appearance information, which is integrated with motion information to act as the deep association metric. This metric allows Deep SORT to track objects over longer periods of occlusions, thus reducing the number of ID switches significantly. Second, if a track has not been associated with a detection result for 30 frames, it is considered to have left the scene. If a detection cannot be associated with any existing tracks, a tentative new track is created. If this new track can associate with a detection successfully for the first three frames, it is confirmed as a new track. Third, a matching cascade is introduced to give priority to more frequently seen objects when handling a longer period of occlusion. Finally, the aspect ratio is no longer constant, giving Deep SORT a better ability to handle size changes of objects. Therefore, the state of each target, i.e., the bounding box, is represented using eight parameters, i.e.,

$$x = [u, v, \gamma, h, \dot{u}, \dot{v}, \dot{\gamma}, \dot{h}]^T, \tag{2}$$

where u and v are the horizontal and vertical position of the center of the bounding box, γ is the aspect ratio and h is the bounding box height, $\dot{u}, \dot{v}, \dot{\gamma}, \dot{h}$ are the changing velocities of these variables.

Because of these improvements, Deep SORT has better performance and much fewer ID switches than SORT. Moreover, it still can operate at real time (20 FPS). Therefore, Deep SORT is one of the most frequently employed tracking methods in recent visual traffic surveillance systems [13], [29], [34].

5) Bottom-Up Clustering: Tang *et al.* [20] adopted a bottom-up clustering method based on visual and semantic features to generate trajectories. The bottom-up clustering method consists of two steps. First, detection results are grouped into tracklets using spatio-temporal consistency. Second, a clustering method is utilized to associate tracklets into longer trajectories. The loss used in the clustering operation is a combination of trajectory smoothness loss, velocity change loss, time interval loss between adjacent tracklets, and appearance change loss. For each tracklet, the loss change is computed for five separate operations (assign, merge, split, switch, and break), and the operation with the minimum loss change is selected. In this way, all tracklets are iteratively clustered into longer trajectories.

6) Summary of multi-object tracking methods:

Many multi-object tracking methods have been adopted in visual traffic surveillance systems. Deep learning is mainly utilized in the design of data association metrics to extract better features. In addition, because most tracking methods use the tracking-by-detection framework, their performance is highly dependent on the detection performance. Finally, many advanced trackers developed recently, such as RetinaTrack [69] and FairMOT [70], have not been utilized in visual traffic surveillance systems. We expect that further advanced trackers will be adopted to improve visual traffic surveillance systems in the coming years.

B. Trajectory Post-Processing

While the tracking accuracy has achieved significant improvements with modern deep learning-based methods, it still suffers from false detections and positional errors, especially for vehicle tracking in congested traffic. As indicated by Kim *et al.* [71], false detections consist of false positives (i.e., detection of vehicles when none is present and detection of vehicles with incorrect size) and false negatives (i.e., not detecting vehicles when some are present). They developed a post-processing method to remove those false detected tracks based on Kalman filter and motion constraints. Along

with false detections, positional errors are also a common issue in vision-based trajectories. It should be noted that the deviation in the detected positions can be magnified in speed measurement and even more in acceleration due to the derivation. Thus, post-processing is often needed to improve the consistency and hence obtain more smooth vehicle trajectories.

Many post-processing methods have been proposed to optimise obtained trajectories. For example, Xin *et al.* [72] treated post-processing as a bi-level optimization that minimizes the measurement errors and the internal inconsistency errors in position, speed, and acceleration. Moreover, Seong *et al.* [31] developed an algorithm using the basic angle and the length of the front of the vehicle to mitigate the positional error caused by the geometric displacement of the camera. Furthermore, Clausse *et al.* [21] used a Rauch-Tung-Striebel (RTS) smoother [73] with a point-mass constant velocity model to process the noisy measurements, while Tang *et al.* [20] utilized temporal median filtering to smooth out the noisy trajectory estimations.

Post-processing has also been applied in several public vehicle trajectory datasets. For instance, both Hamdar and Mahmassani [74] and Thiemann *et al.* [75] applied filtering approaches to solve discontinuities in the NGSIM dataset. Hamdar and Mahmassani [74] utilized a mono-dimensional Gaussian Kernel to smooth the vehicles' lateral position and speed information, while Thiemann *et al.* [75] applied a symmetric exponential moving average filter to smooth vehicles' position, speed, and acceleration simultaneously. Meanwhile, before releasing the HighD dataset, Krajewski *et al.* [76] also used an RTS smoother and a constant acceleration model to refine the position, speed, and acceleration in both longitudinal and lateral directions, which reduces the positioning error to a pixel size level.

V. Traffic Camera Calibration

Camera calibration is crucial in visual traffic surveillance systems, most prominently, to perform 3D reconstruction and speed measurement in the real world. If we denote a point $\mathbf{x}$ in the 3D space using homogeneous coordinates as $\mathbf{x} = (x, y, z, 1)^T$ and the corresponding point in 2D image plane using homogeneous coordinates as $\mathbf{x}' = (u, v, 1)^T$, we have:

$$\lambda \mathbf{x}' = \mathbf{P}\mathbf{x} = \mathbf{K}[\mathbf{R}|\mathbf{t}]\mathbf{x}, \tag{3}$$

where λ is a scene scale factor, $\mathbf{P}$ is a 3×4 projection matrix that can be further divided into $\mathbf{K}[\mathbf{Rt}]$, where $\mathbf{K}$ is the intrinsic camera parameter, $\mathbf{R}$ is the camera rotation matrix, and $\mathbf{t}$ is the translator vector. $\mathbf{R}$ and $\mathbf{t}$ are extrinsic parameters. As shown in Fig. 7, the goal of camera calibration is to formulate the transformation matrix that links the image and road coordinate frame. Sometimes these camera parameters are provided by the creator of datasets [29], while camera calibration should be performed for each traffic camera in most cases [28]. However, as traffic cameras are usually placed in hardly accessible locations and are focused at longer distances [27], the traditional pattern board-based calibration method is impractical. Therefore, various calibration methods have been proposed for traffic cameras, including conventional methods and deep learning-based methods. In this section, we will discuss these methods.

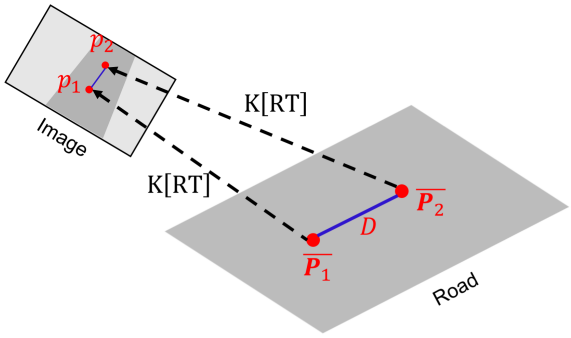


Fig. 7. The essential goal of traffic camera calibration is to obtain the camera parameters to compute the real distance $\mathbf{D}$ between two points $(\overline{\mathbf{P}}_1, \overline{\mathbf{P}}_2)$ on the road plane, given their positions $(\mathbf{p}_1, \mathbf{p}_2)$ on the image plane. This image is reproduced based on [80].

A. Conventional Methods

Generally speaking, conventional calibration methods can be divided into automatic and manual methods. Automatic methods do not require manual intervention, while manual methods require additional inputs, such as average vehicle size [1], average vehicle speed [77], camera position [78], annotation of the lane marking with known lane width [78] and marking dimensions [79], in the calibration process.

1) Manual Method: Manual calibration methods can be grouped into vanishing point-based methods and reference image-based methods.

a) Vanishing Point-Based Methods: With two vanishing points on the ground plane and some constraints on the intrinsic camera parameters, it is possible to compute the projection matrix $\mathbf{P}$ [81]. Therefore, vanishing points have been widely employed in traffic camera calibration methods. In practice, vanishing points may be provided by the creator of datasets. However, in most cases, vanishing points should be derived from vanishing lines, which can usually be manually labeled [13], [20], [29] or automatically detected [26], [35].

To manually label vanishing lines, Yu *et al.* [13], [29] first labeled two groups of parallel lines for each camera view and then used a least square error method proposed in [82] to derive two vanishing points. Based on this information, image points can be projected to the derived road plane. In addition, Yu *et al.* [10] used the vanishing point-based method proposed by He *et al.* [83], which requires a rectangular road mark as the manual input, to perform calibration.

To automatically detect vanishing lines, buildings [84], walking humans [85], [86], lane lines [35] and moving vehicles [87], [88] have been adopted. Among these methods, vehicles are more suitable for traffic camera calibration as other objects may not present in certain scenes.

b) Reference Image-Based Methods: For some scenes where the location is known, it is possible to use a reference image to assist the camera calibration. For example, Chang *et al.* [28] first utilized Google Maps to specify sufficient landmarks manually and estimated their 3D coordinates.

Subsequently, they followed a landmark-based camera calibration approach [89] to compute the camera projection matrix by minimizing the landmark projection square errors. Moreover, Behbahani *et al.* [15] obtained a top-down satellite image of a scene and then added landmarks to both the camera-captured image and the satellite image. The camera matrix was computed by using these landmarks. Similarly, Clausse *et al.* [21] identified four or more matching points between the camera image (2D points) and the satellite view (3D points). An optimization was then formulated to minimize the reprojection error of these matching points and find the rotation, translation and focal length of the traffic camera. This method can also be extended to estimate distortion coefficients by adding a distortion model.

The main advantage of reference image-based methods is that they can provide better calibration results. However, the landmarks and dimensions need to be set manually. In addition, it is infeasible to obtain its corresponding satellite image or Google Map image if the location of a camera is unknown. Therefore, this kind of method cannot be applied to arbitrary traffic videos.

c) Hybrid Methods: Some hybrid methods use vanishing points and reference images together. For example, Tang *et al.* [20] first labeled two pairs of vanishing lines, from which two vanishing points on the ground plane can be obtained. Afterwards, the camera projection matrix $\mathbf{P}$ was computed using these two vanishing points. To improve the performance, they selected a set of line segments from the image, whose endpoints are back-projected to the 3D ground plane and can be used to compute the estimated length. The ground-truth 3D lengths of these selected line segments were measured using Google Maps. The difference between these ground-truth lengths and the estimated lengths was used as the objective to optimize camera parameters, which was then solved iteratively using an evolutionary algorithm.

2) Automatic Methods: Automatic calibration methods are beneficial because they can be easily applied to many camera scenes. However, it is challenging to develop automatic traffic calibration methods and thus only few automatic camera calibration methods have been proposed [26], [27].

Dubska *et al.* [26] proposed an automatic method that allows calibration of traffic cameras, including scale, based on a 2-minute video. This method consists of three steps. First, three vanishing points that define the stream of vehicles are determined. Second, 3D bounding boxes are created for vehicles based on calibration results. Third, the scene scale is computed using the dimensions of 3D bounding boxes. A key innovation in [26] is the automatic estimation of the scene scale using histograms of sold and measured cars. Therefore, no user input is needed in the entire process for scene scale estimation. Another automatic method is proposed by Sochor *et al.* [27]. This method is an improved version of [26]. The main improvements lie in two aspects. First, more precise detection of vanishing points is proposed. Second, a novel scene scale method is developed. It should be mentioned that this work only considered fully visible cars.

Both methods can automatically detect vanishing points and derive the scene scale with some prior information. Moreover, they can be used from arbitrary viewpoints without any requirements on camera placement. However, they require straight road segments to detect vanishing points. Therefore, these methods do not apply to some situations, such as curved roads and roundabouts. To solve this issue, Kocur *et al.* [49] proposed a method that uses a CNN to detect vanishing points. It learns the ability to detect orthogonal vanishing points for individual vehicles in the scene from a large-scale dataset, which relies on the straight movements of vehicles. Consequently, this method can be applied to more cases, such as curved roads and parking spots. Furthermore, Bartl *et al.* [90] proposed to classify vehicles using a CNN and detect keypoints on these vehicles. Then, the dimensions of these keypoints are obtained from 3D models of vehicles. These information are then used for automatic calibration.

B. Deep Learning-Based Methods

In recent years, researchers have started exploring deep learning-based camera calibration methods, which can be roughly grouped into vanishing point detection methods and calibration methods.

DeepVP is proposed as a deep learing-based vanishing point detection method, which formulates the vanishing point detection as a classification problem [91]. Specifically, they designed a CNN that has 225 output values, denoting the 225 discrete possible vanishing point locations. To train the model, the authors collected a dataset consisting of 1,053,425 images and ground-truth vanishing points from the Google Street view. In addition, Li *et al.* [92] proposed a multi-task learning method to perform vanishing point detection and rail segmentation together. In the network, a feature extraction network and three sub-task networks were designed. These sub-task networks were used for vanishing point detection, vanishing region segmentation, and rail segmentation, respectively. The main benefit of deep learning-based vanishing point detection methods is that they do not have many requirements on the scene, such as straight road segments and buildings. However, these methods are still in their early stage. More related methods are expected to be proposed in the coming years.

For deep learning-based calibration methods, Bogdan *et al.* [93] proposed a CNN-based method to estimate the intrinsic parameters, i.e., focal length and distortion parameters, of wide field-of-view (FOV) cameras. This method only requires an image of the scene. Hold-Geoffroy *et al.* [94] also proposed a CNN-based method to estimate camera calibration parameters, including roll, pitch, and FOV, from a single image. Lopez *et al.* [95] proposed the first method that can jointly estimate extrinsic (tilt and roll) and intrinsic (focal length and radial distortion) parameters. In addition, Gordon *et al.* [96] proposed the first unsupervised method for intrinsic camera parameters estimation. Depth, egomotion, object motion, and intrinsic camera parameters can be learned simultaneously. Two adjacent video frames were used as the input to this network. In addition to these methods, Bashetty *et al.* [97] applied deep learning methods to estimate the focal length given an RGB

image. Lee *et al.* [98] proposed a neural network-based method to predict camera calibration parameters, using a single image of artificial scenes. Zhang *et al.* [99] proposed a dual-Siamese network to estimate the focal length and distortion parameters of a pan-tilt-zoom (PTZ) camera using an image pair as input. However, although several deep learning-based calibration methods have been proposed, none of these methods are explicitly designed for visual traffic surveillance systems. Therefore, it is still ambiguous how these methods perform in the context of visual traffic surveillance.

C. Scene Scale

In addition to the projection matrix, it is also essential to estimate the scene scale, as shown in Equation (3), to obtain measurements in real-world units. However, it is not possible to estimate the scene scale from vanishing points. Therefore, vanishing point-based methods need to compute two vanishing points and the scene scale, leading to 5 degrees of freedom.

Various methods have been proposed to estimate the scene scale. For example, Dubska *et al.* [26] utilized the 3D bounding box built around vehicles and the average dimensions of vehicles to compute the scale. Sochor *et al.* [27] proposed to use the alignment of a 3D model and a bounding box for scale inference. Another method worth trying is to use deep learning-based depth estimation to estimate the depth and then compute the scale accordingly. Once the vanishing points and scene scale are obtained, some methods [100]–[102] can be adopted to formulate the transformation matrix **K[RT]**.

D. Summary of Traffic Camera Calibration

Although many traffic camera calibration methods have been proposed, most of them still require manual input that may not be applicable in practice. Moreover, it should be noted that traditional calibration methods still dominate in this research area. The principles of calibration methods employed in most existing visual traffic systems, i.e., vanishing point-based and reference image-based methods, have been proposed for many years. More information about traffic camera calibration can be found in [80].

VI. Speed Measurement

Speed measurement is a prominent application of visual traffic surveillance systems. Measuring speed with a monocular camera has several advantages. First, a single camera can measure the speed of vehicles on multiple lanes. Second, images can be utilized for other detection and recognition tasks simultaneously. Third, many traffic surveillance cameras have already been installed and can be employed for speed measurement and other tasks. Consequently, vision-based speed measurement has attracted widespread interest [1], [26], [77], [103].

It is essential to measure distance on the road plane before performing speed measurement. Therefore, the detection and tracking results should be projected to the road plane. Therefore, we discuss both ground location estimation and speed measurement in this section.

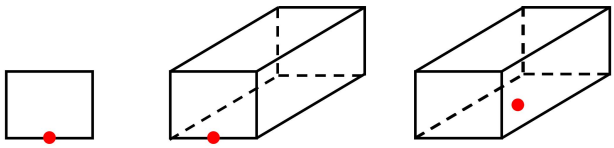


Fig. 8. Examples of reference point adopted in speed estimation. Left: the center of 2D bounding box bottom. Middle: the center of bottom-front edge of 3D bounding box. Right: the center of the bottom of 3D bounding box.

A. Ground Location Estimation

Ground location estimation consists of 2D and 3D methods.

1) 2D Methods: A straightforward method is to consider the center of the 2D bounding box as the vehicle's position and then project this center to the ground. This method works well for top view [76]. However, most traffic cameras have different angles of view. Consequently, this simple method will cause non-negligible errors in most traffic scenes.

2) 3D Methods: To facilitate 3D projection, the aforementioned camera calibration process is needed as a preliminary condition. Afterwards, several methods can be utilized to obtain the ground locations of vehicles. For example, Ren *et al.* [14] used the middle of the bottom edge of the 2D bounding box as the vehicle's ground position and then project this point to 3D space. In addition, Clausse *et al.* [21] first built the 3D bounding box for vehicles and then used the bottom of the 3D bounding box as the ground position of the vehicle.

B. Speed Estimation

As shown in Fig. 8, various methods can be used to determine the reference point for vehicle speed estimation. For example, Sochor *et al.* [27] chose the center of the bottom-front edge of the 3D bounding box. Tang *et al.* [20] selected the bottom center of the 2D bounding box and projected it to 3D space. Clausse *et al.* [21] used the bottom center of the 3D bounding box. The principle is using the same reference point across different image frames.

Travel distance and duration are needed to compute speed. There are several ways in existing research. A common method is to use the travel distance between 2 consecutive frames and the frame rate [13]. However, many works have argued that this method is unstable for calculating speed [26], [27] and instantaneous speed [28]. To solve these issues, some researchers used filtering to smooth the speed estimation. Both exponential smoothing [13] and Gaussian smoothing [28] were adopted for speed estimation. In addition, some researchers suggested to use time positions that are several frames apart [80]. Another method is to use the entire straight-line trajectory [26] as the travel distance. Moreover, ROI or virtual detection region has also been used for speed estimation. They first set ROI on the image and then used the travel distance within this region to compute speed [35]. Some other methods have also been proposed. For example, Yu *et al.* [29] obtained the speed vector of the bottom middle point of a vehicle's 2D bounding box from the internal state of the Kalman filter. It should be mentioned that the estimated speed

may oscillate. One possible cause is the variation of vehicle's bounding box in different frames [14].

VII. Behavior Learning

Vehicle trajectories can be employed to train their behavior models. These models can be used for vehicle prediction to improve traffic safety. Moreover, these models can be transferred to simulation environments to increase the realism of these traffic simulators.

A few behavior learning studies have used trajectories extracted from traffic cameras. For example, Ren *et al.* [14] proposed a method to convert the traffic camera perspective to the simulator view using a homography, which was estimated using four pairs of corresponding points between the camera and the simulator view. In this way, vehicle trajectories in the simulator view were obtained. Afterwards, they modeled the vehicle behavior using multivariate Gaussian distributions and trained the model with those trajectories. Another example is the ViBe proposed by Behbahani *et al.* [15], which first obtained trajectories of road users (vehicles, cyclists, pedestrians) from an initially uncalibrated camera and then adopted a learning from demonstration technique to learn naturalistic behavior models. These models were then utilized in a simulation of the scene using the Unity game engine [104] to increase the realism of autonomous vehicle simulation.

As a promising research direction, behavior learning using trajectories extracted from traffic surveillance cameras is still in its very early stage. We expect that more work will be conducted following this direction in the coming years.

VIII. Accident Detection and Prediction

Traffic accident detection and prediction are critical to traffic safety, thus have attracted tremendous research interest in recent years. An accident can be divided into three stages: pre-collision, collision, and post-collision [33]. Accident detection corresponds to the collision stage, while accident prediction corresponds to the pre-collision stage. It should be noted that some accident detection methods are offline because they need some additional information from the frames after the accident. For example, Singh *et al.* [33] use the trajectory information after the accident time to determine whether the intersection of trajectories is caused by occlusion or collision. In this section, we review the recent studies on accident detection and prediction in visual traffic surveillance systems.

A. Accident Detection

Accident detection methods can be generally divided into learning-free methods and learning-based methods.

1) Learning-Free Methods: Learning-free methods mainly utilize the kinematic information of vehicles to detect accidents. For example, Ki *et al.* [105] used the variation rate of velocity, position, area, and direction of moving vehicles. Meanwhile, Maalou *et al.* [106] utilized the sudden and sharp change in vehicles' velocity. The distance between vehicles and the overlap in their trajectories were also used to detect traffic danger [13]. Moreover, Ijjina *et al.* [30] proposed a computer vision-based accident detection method for traffic surveillance videos, which generates an accident score based on acceleration anomaly, trajectory anomaly, and change in angle anomaly. It should be noted that hand-crafted features were employed in these methods. More importantly, rules were designed manually to determine the occurrence of an accident. Consequently, learning-free methods are less robust.

2) Learning-Based Methods: With the development of deep learning techniques, some learning-based accident detection methods have been proposed. For example, Singh *et al.* [33] proposed a deep spatio-temporal method to detect road accidents. This method consists of two parts. The first part is collision detection based on trajectories of road users, which will give a collision score. The second part is a spatio-temporal network using denoising autoencoders. Specifically, three denoising autoencoders are employed to process appearance, motion, and the joint of both, respectively. A reconstruction error and an anomaly score are computed for each denoising autoencoder, resulting in three reconstruction errors and three anomaly scores in total. These reconstruction errors and anomaly scores are then fused with the obtained collision score to generate a final decision. Moreover, Yu *et al.* [32] developed a self-tuning iterative hard thresholding (ST-IHT) algorithm to learn sparse spatio-temporal features and a weighted extreme learning machine (W-ELM) for detection from imbalanced datasets. Furthermore, Bortnikov *et al.* [107] proposed a 3D CNN-based model for accident detection, which can extract both spatial and temporal features. In addition, Tian *et al.* [108] developed YOLO-CA [17]. They collected an accident dataset CAD-CVIS that contains 633 accident videos and annotated the locations of accidents in image frames for model training.

There are some difficulties in developing learning-based accident detection methods. First, there is a lack of sufficient accident training data. Second, the ratio between normal and accident videos in most existing datasets is imbalanced. To solve these issues, efforts have been devoted to collecting accident videos. For example, Shah *et al.* [12] collected the CADP dataset, which consists of 1416 accident videos. Spatio-temporal annotations are also provided for this dataset. Another solution is using synthetic videos. For example, Bortnikov *et al.* [107] trained a 3D CNN-based accident detection system with traffic videos generated from a view game Grand Theft Auto V, which is then tested on real traffic videos. This method can facilitate the generation of training data under various weather and scene conditions. However, the performance of this method degrades significantly on real videos due to the domain gap between synthetic and real videos. To solve this problem, Batanina *et al.* [109] adapted the trained model to real videos using domain adaptation techniques, which has significantly improved its performance on real videos.

B. Accident Prediction

Compared with accident detection, accident prediction is more beneficial for traffic safety. Existing methods can also be roughly grouped into learning-free methods and learning-based methods.

TABLE II

EXAMPLES OF EXISTING TRAFFIC SURVEILLANCE VIDEO DATASETS AND THEIR DETAILS. THE DATASETS RELEASED IN RECENT YEARS ARE MUCH LARGER AND DIVERSE COMPARED WITH DATASETS RELEASED BEFORE 2010. 'N/A' MEANS THE INFORMATION IS NOT AVAILABLE

Name/Reference	Year	Size	Resolution	Ground truth	Publicly available
Roundabout [110]	2008	1 video (93,500 frames)	360×288	N/A	Yes
QMUL Junction [111]	2009	1 video (90,000 frames)	360×288	anomaly description	Yes
QMUL Junction 2 [112]	2009	1 video (78,000 frames)	360×288	N/A	Yes
CADP [12]	2018	1,416 videos (518,256 frames)	various	temporal, spatial, accident	Yes
AAU traffic dataset [113]	2018	130,800 RGB-thermal image pairs	640×480	object mask (2,200 frames)	Yes
TCP [14]	2018	234 videos	1280 × 720	trajectory (2618 vehicles)	Yes
IITH Accident Dataset [33]	2019	127,138 frames	Not provided	accident (863 frames)	Yes
Fedorov et al. [9]	2019	986 frames	1920×1080	object mask	Yes
CAD-CVIS [108]	2019	633 videos	Various	accident location	Yes
BrnoCompSpeed [80]	2019	18 videos	1920×1080	speed, distance measuerment	Yes
BrnoCarPark [90]	2021	11 videos	1920×1080	distance measurements in the ground plane	Yes

1) Learning-Free Methods: Learning-free accident prediction methods mainly utilize the kinematic information of vehicles. For example, Yu *et al.* [13] first predicted the speed and trajectories of vehicles and then predicted accidents using this information. However, owing to the employed linear prediction of speed and position, this method can only handle vehicles moving on straight roads. Yu *et al.* [29] proposed an improved method, which also detects accidents based on kinematic patterns. The main improvements are as follows. First, some vehicle manoeuvres can be detected, such as vehicle turning to either direction and starting. Second, both speed and driving direction of vehicles were estimated. It is worth mentioning that both methods employed a manual camera calibration method, thus they cannot be applied to arbitrary traffic surveillance cameras automatically.

2) Learning-Based Methods: Only a few learning-based accident prediction methods for traffic surveillance videos have been proposed. For example, Shah *et al.* [12] proposed an accident prediction method based on Faster R-CNN and the Dynamic-Spatial-Attention LSTM [114]. The Faster R-CNN was employed to extract features, which were then fed into the DSA-LSTM to generate accident scores. Huang *et al.* [34] proposed a two-stream CNN consisting of a spatial stream for object detection and a temporal stream for leveraging motion features to predict accidents at real time.

IX. TRAFFIC SURVEILLANCE DATASETS

Traffic surveillance video datasets are crucial for deep learning-based visual traffic surveillance systems. Table II lists some existing surveillance video datasets. It can be noted that these datasets have distinct characteristics. Generally speaking, the number of traffic surveillance images (with bounding box or mask annotations) that can be used to train vehicle detection methods is still insufficient. In addition, there is only one dataset that contains the ground-truth speed measurements. It should be mentioned that some self-collected datasets have been used in different studies, such as the accident detection dataset used by Ijjina *et al.* [30]. However, these datasets are not publicly available yet.

X. PERFORMANCE EVALUATION METHODS

It is crucial to evaluate the performance of different components of visual traffic surveillance systems. In this section, we briefly discuss the commonly used performance evaluation methods for the main components of these systems.

A. Vehicle Detection and Tracking

Performance evaluation measure is critical in the research of object detection and tracking. Different metrics have been proposed, among which some are widely used. In object detection, mean average precision (mAP) [115] is the most popular evaluation metric and has been adopted by a series of popular object detection challenges, such as the PASCAL VOC Challenge [41] and the COCO Object Detection Challenge [44], and most object detection papers. In the COCO Object Detection Challenge, mean average recall (mAR) was also adopted. In terms of multiple object tracking, the most widely used evaluation metric is multiple object tracking accuracy (MOTA) [116]. MOTA has a good expressiveness because it combines false negative, false positive and ID switches, thus it has been adopted as a main metric in MOTChallenge [117]. In addition, the multiple object tracking precision (MOTP) [118], which measures the average dissimilarity between all true positives and their corresponding ground-truth targets, and the IDF1 measure [117], which emphasizes the track identity preservation capability over the entrie sequence, have also been utilized in MOTChallenge [117].

However, the performance evaluation methods for existing visual traffic surveillance systems are different from general object detection and tracking. For example, Seong *et al.* [31] used Root Mean Square Error (RMSE) to compare the position of vehicles in each frame with the ground truth. Sochor *et al.* [27] evaluated the vehicle detection and tracking performance using the False Positives Per Minute (FPPM) and mean recall in vehicle counting. Ren *et al.* [14] manually labeled a dataset of 100 images using bounding boxes around objects and then computed the IoU between each detected bounding box and the ground truth. Afterwards, different rules were set to obtain true positive, false positive, and false negative. These values were then used to compute precision and recall. In summary, there is no consensus on a unified method to evaluate vehicle detection and tracking performance in visual traffic surveillance systems.

B. Traffic Camera Calibration

The evaluation of traffic camera calibration methods is not straightforward [26]. For example, Zhang *et al.* [102]

evaluated the calibration performance by comparing the focal length and the first vanishing point with the ground truth. However, this method requires to manually annotate the first vanishing point and compute the focal length with a traditional method using a calibration board. Therefore, this method is not feasible in practice, where access to traffic cameras is restricted.

Camera parameters can be utilized to backproject image points to 3D space. Therefore, the calculated Euclidean distance between backprojected points $\overline{P_1}$ and $\overline{P_2}$ (see Fig. 7) should be similar to the actual value, if the camera parameters are correct. Therefore, the performance evaluation of camera calibration is usually achieved using distance measurement [13], [26], [80], [90], [102] in real-world coordinates.

C. Speed Measurement

Although various speed measurement methods have been proposed, it was difficult to compare these methods due to the lack of standard datasets with associated ground truth. To solve this issue, Soochor *et al.* [80] created the BrnoCompSpeed dataset that contains ground-truth speed for performance comparison. This dataset has been employed in several studies [13], [35], [49], [90], [119], [120] and has almost become a standard dataset for speed measurement performance evaluation.

D. Evaluation Based on Traffic Simulation Environment

Some researchers have proposed to use a simulation environment, because it is easier to generate traffic videos and ground-truth vehicle information in simulation. For example, Clausse *et al.* [21] utilized CARLA to generate traffic videos with associated ground-truth information. They extracted vehicle trajectories from these videos and evaluated the accuracy using the ground truth. Specifically, they utilized RMSE to compare the position, velocity, and heading of vehicles. Performing evaluation using a traffic simulation environment is promising, but the sim-to-real gap should be properly handled.

XI. Future Research Directions

A. Adopting the Latest Object Detection and Tracking Methods

Object detection and tracking methods have a significant impact on the performance of visual traffic surveillance systems. However, the commonly adopted detection and tracking methods in existing visual traffic surveillance systems are a bit outdated. For example, both Mask R-CNN and Deep SORT were published in 2017. Many object detection and tracking methods have been proposed since then. Therefore, it would be beneficial to apply those new detection and tracking methods to improve the performance of visual traffic surveillance systems in the future.

B. Adopting 3D Detection Methods

3D bounding boxes of vehicles have been used in many existing studies. These 3D bounding boxes are built based on 2D detection results [13], [26], e.g., 2D bounding boxes or masks, using traditional methods. In recent years, some deep learning-based 3D detection methods [62], [121], [122] have been proposed. The 3D bounding boxes provided by these methods might be more accurate than those obtained using traditional methods. Therefore, it is worth trying to adopt these 3D detection methods in visual traffic surveillance systems.

C. Enhancing the Running Speed of Traffic Surveillance Systems

Some components in visual traffic surveillance systems are relatively time-consuming. For example, Mask R-CNN and Deep SORT can only reach 5 and 20 FPS, respectively. Obviously, systems that adopt these methods can not meet the real-time requirement. Therefore, improving the computing speed is essential for real-time practical applications.

D. Deep Learning-Based Calibration Methods

Automatic camera calibration methods can improve the generalization and scaling ability of visual traffic surveillance systems. However, existing automatic calibration methods are limited. Deep learning can be a potential solution for automatic calibration, although related research is still in an early stage. We expect that this will be a good research direction and will attract more researchers in the near future.

E. Including Other Road Users Into the System

Most existing visual traffic surveillance studies only considered vehicles. However, it is also crucial to include other road users, such as cyclists and pedestrians. This is particularly important to improve traffic safety and facilitate the development of autonomous vehicles. For example, the motion patterns of road users can be learned and replicated in simulations to facilitate the training and evaluation of autonomous vehicles.

F. Handling Low-Resolution Videos

Most existing visual traffic surveillance systems are designed to work with high-resolution videos. For example, the videos provided in the AI City Challenge for traffic flow analysis are high-resolution (1920×1080) videos. However, many practical traffic cameras have low resolution. For example, the resolution of the videos provided by TfL is only 352×288. The performance of the algorithms trained using high-resolution videos may significantly degrade when applied to low-resolution videos. It is therefore beneficial to devote more effort to low-resolution videos to improve the transferability and practical implications..

G. Handling Adverse Weather Conditions

Most aforementioned studies require good lighting and weather conditions. Their performance may degrade significantly when adverse conditions are present. Therefore, it would be beneficial to develop algorithms that are more robust to these adverse conditions. A potential solution is to integrate other types of cameras to work together. For example,

thermal cameras are insensitive to lighting conditions, thus they have been combined with RGB cameras to assist pedestrian detection [123], [124] and object tracking [125]–[127]. However, they have rarely been adopted in visual traffic surveillance systems.

XII. Conclusion

This paper presents a comprehensive review of monocular visual traffic surveillance systems, covering three main components, three important applications, datasets, and performance evaluation methods. It is found that deep learning has significantly inspired modern object detection and tracking methods, leading to substantial performance improvements. However, although researchers have started to develop deep learning-based calibration methods, traditional camera calibration methods are more commonly adopted in visual traffic surveillance systems. Therefore, it can be noted that deep learning has a limited impact on traffic camera calibration. In addition, we find that deep learning has enabled new applications of visual traffic surveillance systems, such as behavior learning of vehicles. Deep learning has also been applied to some traditional applications, such as accident detection and prediction. Based on the reviewed studies and outlined future research directions, we expect deep learning will play a more critical role in visual traffic surveillance systems.

References

[1] D. J. Dailey, F. W. Cathey, and S. Pumrin, "An algorithm to estimate mean traffic speed using uncalibrated cameras," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 2, pp. 98–107, Jun. 2000.

[2] W. Hu, X. Xiao, D. Xie, T. Tan, and S. Maybank, "Traffic accident prediction using 3-D model-based vehicle tracking," *IEEE Trans. Veh. Technol.*, vol. 53, no. 3, pp. 677–694, May 2004.

[3] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 34, no. 3, pp. 334–352, Aug. 2004.

[4] Y. Wang, Y. Zou, H. Shi, and H. Zhao, "Video image vehicle detection system for signaled traffic intersection," in *Proc. 9th Int. Conf. Hybrid Intell. Syst.*, vol. 1, 2009, pp. 222–227.

[5] R. Kachach and J. M. Cañas, "Hybrid three-dimensional and support vector machine approach for automatic vehicle tracking and classification using a single camera," *J. Electron. Imag.*, vol. 25, no. 3, Jun. 2016, Art. no. 033021.

[6] S. R. E. Datondji, Y. Dupuis, P. Subirats, and P. Vasseur, "A survey of vision-based traffic monitoring of road intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2681–2698, Oct. 2016.

[7] N. C. Mithun, N. U. Rashid, and S. M. M. Rahman, "Detection and classification of vehicles from video using multiple time-spatial images," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1215–1225, Feb. 2012.

[8] Z. Dong, Y. Wu, M. Pei, and Y. Jia, "Vehicle type classification using a semisupervised convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2247–2256, Aug. 2015.

[9] A. Fedorov, K. Nikolskaia, S. Ivanov, V. Shepelev, and A. Minbaleev, "Traffic flow estimation with data from a video surveillance camera," *J. Big Data*, vol. 6, no. 1, pp. 1–15, Dec. 2019.

[10] Z. Yu, Q. Jiang, and X. Li, "MPP: A novel algorithm for estimating vehicle space headways from a single image," *J. Adv. Transp.*, vol. 2020, pp. 1–16, Feb. 2020.

[11] D. C. Luvizon, B. T. Nassu, and R. Minetto, "A video-based system for vehicle speed measurement in urban roadways," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1393–1404, Jun. 2016.

[12] A. P. Shah, J.-B. Lamare, T. Nguyen-Anh, and A. Hauptmann, "CADP: A novel dataset for CCTV traffic camera based accident analysis," in *Proc. 15th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Nov. 2018, pp. 1–9.

[13] L. Yu, D. Zhang, X. Chen, and A. Hauptmann, "Traffic danger recognition with surveillance cameras without training data," in *Proc. 15th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Nov. 2018, pp. 1–6.

[14] X. Ren, D. Wang, M. Laskey, and K. Goldberg, "Learning traffic behaviors by extracting vehicle trajectories from online video streams," in *Proc. IEEE 14th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2018, pp. 1276–1283.

[15] F. Behbahani *et al.*, "Learning from demonstration in the wild," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 775–781.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[18] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional Siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 850–865.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1097–1105.

[20] Z. Tang, G. Wang, H. Xiao, A. Zheng, and J.-N. Hwang, "Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 108–115.

[21] A. Clausse, S. Benslimane, and A. de La Fortelle, "Large-scale extraction of accurate vehicle trajectories for driving behavior learning," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 2391–2396.

[22] C. Wang, A. Musaev, P. Sheinidashtegol, and T. Atkison, "Towards detection of abnormal vehicle behavior using traffic cameras," in *Proc. Int. Conf. Big Data*. Cham, Switzerland: Springer, 2019, pp. 125–136.

[23] A. Abdelhalim and M. Abbas, "Towards real-time traffic movement count and trajectory reconstruction using virtual traffic lanes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 592–593.

[24] Z. Yang and L. S. C. Pun-Cheng, "Vehicle detection in intelligent transportation systems and its applications under varying environments: A review," *Image Vis. Comput.*, vol. 69, pp. 143–154, Jan. 2018.

[25] T. Yuan *et al.*, "Machine learning for next-generation intelligent transportation systems: A survey," 2019. [Online]. Available: https://hal.inria.fr/hal-02284820v1/document

[26] M. Dubská, A. Herout, and J. Sochor, "Automatic camera calibration for traffic understanding," in *Proc. Brit. Mach. Vis. Conf.*, 2014, vol. 4, no. 6, p. 8.

[27] J. Sochor, R. Juránek, and A. Herout, "Traffic surveillance camera calibration by 3D model bounding box alignment for accurate vehicle speed measurement," *Comput. Vis. Image Understand.*, vol. 161, pp. 87–98, Aug. 2017.

[28] M.-C. Chang, Y. Wei, N. Song, and S. Lyu, "Video analytics in smart transportation for the AIC'18 challenge," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 61–68.

[29] L. Yu, P. Chen, W. Liu, G. Kang, and A. G. Hauptmann, "Training-free monocular 3D event detection system for traffic surveillance," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 3838–3843.

[30] E. P. Ijjina, D. Chand, S. Gupta, and K. Goutham, "Computer vision-based accident detection in traffic surveillance," in *Proc. 10th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, Jul. 2019, pp. 1–6.

[31] S. Seong, J. Song, D. Yoon, J. Kim, and J. Choi, "Determination of vehicle trajectory through optimization of vehicle bounding boxes using a convolutional neural network," *Sensors*, vol. 19, no. 19, p. 4263, 2019.

[32] Y. Yu, M. Xu, and J. Gu, "Vision-based traffic accident detection using sparse spatio-temporal features and weighted extreme learning machine," *IET Intell. Transp. Syst.*, vol. 13, no. 9, pp. 1417–1428, Sep. 2019.

[33] D. Singh and C. K. Mohan, "Deep spatio-temporal representation for detection of road accidents using stacked autoencoder," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 3, pp. 879–887, Mar. 2019.

[34] X. Huang, P. He, A. Rangarajan, and S. Ranka, "Intelligent intersection: Two-stream convolutional networks for real-time near-accident detection in traffic video," *ACM Trans. Spatial Algorithms Syst.*, vol. 6, no. 2, pp. 1–28, 2020.

[35] B. Zhang and J. Zhang, "A traffic surveillance system for obtaining comprehensive information of the passing vehicles based on instance segmentation," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 11, pp. 7040–7055, Nov. 2021.

[36] Q. Cai, A. Mitiche, and J. K. Aggarwal, "Tracking human motion in an indoor environment," in *Proc. Int. Conf. Image Process.*, vol. 1, 1995, pp. 215–218.

[37] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 91–99.

[38] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.

[39] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.

[40] J. R. R. Uijlings *et al.*, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.

[41] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and W. Zisserman, "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2010.

[42] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[43] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7263–7271.

[44] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 740–755.

[45] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, May 2015, pp. 1–14.

[46] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[47] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[48] M. Naphade *et al.*, "The 5th AI city challenge," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 4263–4273.

[49] V. Kocur and M. Ftáčnik, "Traffic camera calibration via vehicle vanishing point detection," 2021, *arXiv:2103.11438*.

[50] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6569–6578.

[51] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[52] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[53] O. Messoussi *et al.*, "Vehicle detection and tracking from surveillance cameras in urban scenes," in *Proc. Int. Symp. Vis. Comput.* Cham, Switzerland: Springer, 2021, pp. 191–202.

[54] H. Perreault, G.-A. Bilodeau, N. Saunier, and M. Héritier, "SpotNet: Self-attention multi-task network for object detection," in *Proc. 17th Conf. Comput. Robot Vis.*, 2020, pp. 230–237.

[55] R. Juranek, A. Herout, M. Dubská, and P. Zemcik, "Real-time pose estimation piggybacked on object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2381–2389.

[56] M. Dubská, A. Herout, R. Juránek, and J. Sochor, "Fully automatic roadside camera calibration for traffic surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1162–1171, Jun. 2015.

[57] H. Van Pham and B.-R. Lee, "Front-view car detection and counting with occlusion in dense traffic flow," *Int. J. Control. Autom. Syst.*, vol. 13, no. 5, pp. 1150–1160, Oct. 2015.

[58] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS—Improving object detection with one line of code," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5561–5569.

[59] H. Nguyen, "Improving faster R-CNN framework for fast vehicle detection," *Math. Problems Eng.*, vol. 2019, pp. 1–11, Nov. 2019.

[60] W. Zhang, Y. Zheng, Q. Gao, and Z. Mi, "Part-aware region proposal for vehicle detection in high occlusion environment," *IEEE Access*, vol. 7, pp. 100383–100393, 2019.

[61] Y. Fang, C. Wang, W. Yao, X. Zhao, H. Zhao, and H. Zha, "On-road vehicle tracking using part-based particle filter," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 12, pp. 4538–4552, Dec. 2019.

[62] H.-N. Hu *et al.*, "Joint monocular 3D vehicle detection and tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5390–5399.

[63] L. Unzueta, M. Nieto, A. Cortes, J. Barandiaran, O. Otaegui, and P. Sanchez, "Adaptive multicue background subtraction for robust vehicle counting and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 527–540, Jun. 2012.

[64] H. W. Kuhn, "The Hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, nos. 1–2, pp. 83–97, 1955.

[65] E. Bochinski, V. Eiselein, and T. Sikora, "High-speed tracking-by-detection without using image information," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2017, pp. 1–6.

[66] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3464–3468.

[67] R. E. Kalman, "A new approach to linear filtering and prediction problems," *ASME J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, 1960.

[68] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 3645–3649.

[69] Z. Lu, V. Rathod, R. Votel, and J. Huang, "RetinaTrack: Online single stage joint detection and tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 14668–14678.

[70] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "FairMOT: On the fairness of detection and re-identification in multiple object tracking," *Int. J. Comput. Vis.*, vol. 129, no. 11, pp. 3069–3087, Nov. 2021.

[71] E.-J. Kim *et al.*, "Extracting vehicle trajectories using unmanned aerial vehicles in congested traffic conditions," *J. Adv. Transp.*, vol. 2019, Apr. 2019, Art. no. 9060797.

[72] W. Xin, J. Hourdos, and P. G. Michalopoulos, "Vehicle trajectory collection and processing methodology and its implementation," in *Proc. 87th Annu. Meeting Transp. Res. Board*, 2008.

[73] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Hoboken, NJ, USA: Wiley, 2006.

[74] S. H. Hamdar and H. Mahmassani, "Driver car-following behavior: From discrete event process to continuous set of episodes," in *Proc. Transp. Res. Board 87th Annu. Meeting*, 2008, p. 37.

[75] C. Thiemann, M. Treiber, and A. Kesting, "Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2088, no. 1, pp. 90–101, Jan. 2008.

[76] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2118–2125.

[77] T. N. Schoepflin and D. J. Dailey, "Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 90–98, Jun. 2003.

[78] K. Wang, H. Huang, Y. Li, and F.-Y. Wang, "Research on lane-marking line based camera calibration," in *Proc. IEEE Int. Conf. Veh. Electron. Saf.*, Dec. 2007, pp. 1–6.

[79] F. W. Cathey and D. J. Dailey, "A novel technique to dynamically measure vehicle speed using uncalibrated roadway cameras," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2005, pp. 777–782.

[80] J. Sochor *et al.*, "Comprehensive data set for automatic single camera visual speed measurement," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 5, pp. 1633–1643, May 2019.

[81] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *Int. J. Comput. Vis.*, vol. 4, no. 2, pp. 127–139, Mar. 1990.

[82] S. C. Lee and R. Nevatia, "Robust camera calibration tool for video surveillance camera in urban environment," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2011, pp. 62–67.

[83] X. He and N. H. C. Yung, "New method for overcoming ill-conditioning in vanishing-point-based camera calibration," *Opt. Eng.*, vol. 46, no. 3, 2007, Art. no. 037202.

[84] R. Cipolla, T. Drummond, and D. Robertson, "Camera calibration from vanishing points in image of architectural scenes," in *Proc. Brit. Mach. Vis. Conf.*, vol. 99, 1999, pp. 382–391.

[85] F. Lv, T. Zhao, and R. Nevatia, "Camera calibration from video of a walking human," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1513–1518, Sep. 2006.

[86] N. Krahnstoever and P. R. S. Mendonça, "Autocalibration from tracks of walking people," in *Proc. Brit. Mach. Vis. Conf.*, 2006, pp. 1–10.

[87] Z. Zhang, M. Li, K. Huang, and T. Tan, "Practical camera auto-calibration based on object appearance and motion for traffic scene visual surveillance," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[88] N. Wang, H. Du, Y. Liu, Z. Tang, and J.-N. Hwang, "Self-calibration of traffic surveillance cameras based on moving vehicle appearance and 3-D vehicle modeling," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 3064–3068.

[89] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[90] V. Bartl, J. Špaňhel, P. Dobeš, R. Juránek, and A. Herout, "Automatic camera calibration by landmarks on rigid objects," *Mach. Vis. Appl.*, vol. 32, no. 1, pp. 1–13, Jan. 2021.

[91] C.-K. Chang, J. Zhao, and L. Itti, "DeepVP: Deep learning for vanishing point detection on 1 million street view images," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4496–4503.

[92] X. Li, L. Zhu, Z. Yu, B. Guo, and Y. Wan, "Vanishing point detection and rail segmentation based on deep multi-task learning," *IEEE Access*, vol. 8, pp. 163015–163025, 2020.

[93] O. Bogdan *et al.*, "DeepCalib: A deep learning approach for automatic intrinsic calibration of wide field-of-view cameras," in *Proc. Eur. Conf. Vis. Media Prod.*, 2018, pp. 1–10.

[94] Y. Hold-Geoffroy *et al.*, "A perceptual measure for deep single image camera calibration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2354–2363.

[95] M. Lopez, R. Mari, P. Gargallo, Y. Kuang, J. Gonzalez-Jimenez, and G. Haro, "Deep single image camera calibration with radial distortion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 11817–11825.

[96] A. Gordon, H. Li, R. Jonschkowski, and A. Angelova, "Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8977–8986.

[97] S. K. Bashetty, H. B. Amor, and G. Fainekos, "DeepCrashTest: Turning dashcam videos into virtual crash tests for automated driving systems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 11353–11360.

[98] J. Lee, M. Sung, H. Lee, and J. Kim, "Neural geometric parser for single image camera calibration," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 541–557.

[99] C. Zhang, F. Rameau, J. Kim, D. M. Argaw, J.-C. Bazin, and I. S. Kweon, "DeepPTZ: Deep self-calibration for PTZ cameras," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2020, pp. 1041–1049.

[100] G. S. Fung, N. H. Yung, and G. K. Pang, "Camera calibration from road lane markings," *Opt. Eng.*, vol. 42, no. 10, pp. 2967–2977, 2003.

[101] Y. Zheng and S. Peng, "A practical roadside camera calibration method based on least squares optimization," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 831–843, Apr. 2014.

[102] Z. Zhang, T. Tan, K. Huang, and Y. Wang, "Practical camera calibration from moving objects for traffic scene surveillance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 3, pp. 518–533, Mar. 2013.

[103] X. He and N. C. Yung, "A novel algorithm for estimating vehicle speed from two consecutive images," in *Proc. IEEE Workshop Appl. Comput. Vis. (WACV)*, Feb. 2007, p. 12.

[104] A. Juliani *et al.*, "Unity: A general platform for intelligent agents," 2018, *arXiv:1809.02627*.

[105] Y.-K. Ki and D.-Y. Lee, "A traffic accident recording and reporting model at intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 2, pp. 188–194, Jun. 2007.

[106] B. Maaloul, A. Taleb-Ahmed, S. Niar, N. Harb, and C. Valderrama, "Adaptive video-based algorithm for accident detection on highways," in *Proc. 12th IEEE Int. Symp. Ind. Embedded Syst. (SIES)*, Jun. 2017, pp. 1–6.

[107] M. Bortnikov, A. Khan, A. M. Khattak, and M. Ahmad, "Accident recognition via 3D CNNs for automated traffic monitoring in smart cities," in *Proc. Sci. Inf. Conf.* Cham, Switzerland: Springer, 2019, pp. 256–264.

[108] D. Tian, C. Zhang, X. Duan, and X. Wang, "An automatic car accident detection method based on cooperative vehicle infrastructure systems," *IEEE Access*, vol. 7, pp. 127453–127463, 2019.

[109] E. Batanina, I. E. I. Bekkouch, Y. Youssry, A. Khan, A. M. Khattak, and M. Bortnikov, "Domain adaptation for car accident detection in videos," in *Proc. 9th Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Nov. 2019, pp. 1–6.

[110] J. Li, S. Gong, and T. Xiang, "Scene segmentation for behaviour correlation," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer-Verlag, 2008, pp. 383–395.

[111] C. C. Loy, T. Xiang, and S. Gong, "Modelling multi-object activity by Gaussian processes," in *Proc. Brit. Mach. Vis. Conf.*, 2009, pp. 1–11.

[112] T. Hospedales, S. Gong, and T. Xiang, "A Markov clustering topic model for mining behaviour in video," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 1165–1172.

[113] C. H. Bahnsen and T. B. Moeslund, "Rain removal in traffic surveillance: Does it matter?" *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 8, pp. 2802–2819, Aug. 2019.

[114] F.-H. Chan *et al.*, "Anticipating accidents in dashcam videos," in *Proc. Asian Conf. Comput. Vis.* Springer, 2016, pp. 136–153.

[115] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," 2019, *arXiv:1905.05055*.

[116] R. Stiefelhagen *et al.*, "The CLEAR 2006 evaluation," in *Proc. Int. Eval. Workshop Classification Events, Activities Relationships*. Berlin, Germany: Springer, 2006, pp. 1–44.

[117] P. Dendorfer *et al.*, "MOTChallenge: A benchmark for single-camera multiple target tracking," *Int. J. Comput. Vis.*, vol. 129, no. 4, pp. 845–881, Apr. 2021.

[118] P. Dendorfer *et al.*, "MOT20: A benchmark for multi object tracking in crowded scenes," 2020, *arXiv:2003.09003*.

[119] V. Bartl, R. Juránek, J. Španhel, and A. Herout, "PlaneCalib: Automatic camera calibration by multiple observations of rigid objects on plane," in *Proc. Digit. Image Comput., Techn. Appl.*, 2020, pp. 1–8.

[120] M. Zhu, S. Zhang, Y. Zhong, P. Lu, H. Peng, and J. Lenneman, "Monocular 3D vehicle detection using uncalibrated traffic cameras through homography," 2021, *arXi:2103.15293*.

[121] G. Brazil and X. Liu, "M3D-RPN: Monocular 3D region proposal network for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9287–9296.

[122] X. Zhou, Y. Peng, C. Long, F. Ren, and C. Shi, "MoNet3D: Towards accurate monocular 3D object localization in real time," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 11503–11512.

[123] S. Hwang, J. Park, N. Kim, Y. Choi, and I. S. Kweon, "Multispectral pedestrian detection: Benchmark dataset and baseline," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1037–1045.

[124] C. Li, D. Song, R. Tong, and M. Tang, "Illumination-aware faster R-CNN for robust multispectral pedestrian detection," *Pattern Recognit.*, vol. 85, pp. 161–171, Jan. 2019.

[125] X. Zhang, P. Ye, S. Peng, J. Liu, K. Gong, and G. Xiao, "SiamFT: An RGB-infrared fusion tracking method via fully convolutional Siamese networks," *IEEE Access*, vol. 7, pp. 122122–122133, 2019.

[126] X. Zhang, P. Ye, S. Peng, J. Liu, and G. Xiao, "DSiamMFT: An RGB-T fusion tracking method via dynamic Siamese networks using multi-layer feature fusion," *Signal Process., Image Commun.*, vol. 84, May 2020, Art. no. 115756.

[127] X. Zhang, P. Ye, H. Leung, K. Gong, and G. Xiao, "Object fusion tracking based on visible and infrared images: A comprehensive review," *Inf. Fusion*, vol. 63, pp. 166–187, Nov. 2020.

Xingchen Zhang (Member, IEEE) received the B.Sc. degree from the Huazhong University of Science and Technology in 2012 and the Ph.D. degree from the Queen Mary University of London in 2018. He is currently a Research Associate and an Assistant Supervisor at the Department of Electrical and Electronic Engineering, Imperial College London. His main research interests include human motion prediction, human intention prediction, image fusion, and object tracking. He was a recipient of the Marie Skłodowska-Curie Individual Fellowship.

Yuxiang Feng (Member, IEEE) received the B.Sc. degree in mechanical engineering from Tongji University in 2013 and the M.Sc. degree in mechatronics and the Ph.D. degree in automotive engineering from the University of Bath in 2014 and 2019, respectively. Since 2019, he has been a Post-Doctoral Research Associate at the Transport Systems and Logistics Laboratory, Department of Civil and Environmental Engineering, Imperial College London. His main research interests include environment perception, sensor fusion, and artificial intelligence for robotics and autonomous vehicles.

Panagiotis Angeloudis received the M.Eng. degree in civil and environmental engineering and the Ph.D. degree in transport operations from Imperial College London in 2005 and 2009, respectively.

He is currently a Reader and the Director of the Transport Systems and Logistics Laboratory, part of the Centre for Transport Studies and the Department of Civil and Environmental Engineering, Imperial College London. His research interests lie in transport systems and networks operations, with a focus on the efficient and reliable movement of people and goods across land, sea, and water. He was recently appointed by the U.K. Department for Transport to the Expert Panel for Maritime 2050 and was a member of the U.K. Government Office of Science Future of Mobility Review Team.

Yiannis Demiris (Senior Member, IEEE) received the B.Sc. (Hons.) degree in artificial intelligence and computer science and the Ph.D. degree in intelligent robotics from the Department of Artificial Intelligence, The University of Edinburgh, Edinburgh, U.K., in 1994 and 1999, respectively. He is currently a Professor with the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K., where he is the Royal Academy of Engineering Chair in emerging technologies and the Head of the Personal Robotics Laboratory. His current research interests include human–robot interaction, machine learning, user modeling, and assistive robotics. He is a fellow of the Institution of Engineering and Technology (IET) and the British Computer Society (BCS). He was a recipient of the Rector's Award for Teaching Excellence in 2012 and the FoE Award for Excellence in Engineering Education in 2012.