IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 7 August 2024, accepted 29 August 2024, date of publication 2 September 2024, date of current version 10 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3453249

RESEARCH ARTICLE

Multi-Vehicle Tracking and Counting Framework in Average Daily Traffic Survey Using RT-DETR and ByteTrack

YUSUF GLADIENSYAH BIHANDA, CHASTINE FATICHAH, (Member, IEEE), AND ANNY YUNIARTI, (Member, IEEE)

Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya 60117, Indonesia

Corresponding author: Chastine Fatichah (chastine@if.its.ac.id)

This work was supported in part by the Ministry of Education, Culture, Research, and Technology Republic of Indonesia through the Penelitian Terapan Scheme under Grant 55/IT2/T/HK.00.01/2023; and in part by the patent under Grant S00202402259.

ABSTRACT The average daily traffic survey is essential for repairing and maintaining road sections. This method is generally conducted using a semi-manual approach that counts vehicles using CCTV. This approach is not effective or efficient because of the potential for human error. This paper have three contributions as follows. First, this paper proposes a framework by applying the RT-DETR architecture for vehicle detection and ByteTrack for vehicle tracking and counting in an average daily traffic survey. Second, this paper proposes a multi-vehicle voting algorithm to filter false identification of the same vehicle during the tracking process prior to vehicle counting. Third, to demonstrate the robustness of the proposed frameworks, we evaluated its performance using seven CCTV camera videos taken from diverse scenes during the day and night. RT-DETR Resnet 101, which is trained in an average daily traffic survey dataset, outperforms all object detection architectures with mAP@50 value of 0.992 and mAP@50-95 value of 0.891. RT-DETR Resnet101 also achieve best F1-score among all object detectors with value of 0.91. This framework using a combination of RT-DETR and ByteTrack also succeeded in counting vehicles with various video backgrounds, with an average counting accuracy value above 83% for all conditions. We compare the effect of using a multi-vehicle voting algorithm with and without showed that counting accuracy increased for each combination with an average increase value of 0.78. RT-DETR also has best performance compared to another object detection methods in detecting vehicles experiencing motion blur, especially in nighttime video scenes. In addition, ByteTrack roles in tracking vehicle objects show its robustness to handle occlusion and vehicle ID switch.

INDEX TERMS Multi-vehicle tracking counting framework, average daily traffic survey, RT-DETR, ByteTrack, voting algorithm.

I. INTRODUCTION

Roads are vital for human life because they provide access to economic growth and mobility from one place to another. However, as time progresses, roads require regular maintenance and repairs to be optimally used. To determine how repairs and maintenance are performed, several aspects must be considered, starting from the condition of the road itself to the surrounding environment. An average daily traffic survey can be used to choose which road sections need repair and maintenance. [1].

This survey was conducted by installing one or several Closed-circuit Television (CCTV) cameras next to a road section for a semi-manual approach. Different from the annual average daily traffic [2] survey, the purpose of the average daily traffic survey was to determine how many vehicles passed a road section during 40 hours instead of 365 days. The installed CCTV is monitored regularly to avoid adverse events. Twelve types of vehicles must be identified before counting. These vehicles are classified on the basis of

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

the number of wheel axles and their shape. After 40 hours of CCTV camera recording, the CCTV camera video was taken for manual counting [3]. However, based on the observation results, this survey semi-manual approach is not effective or efficient. Human error factor, such as fatigue can lead to error in manual vehicle identification and counting.

In addition, some approaches have proposed a vehicle counting framework as part of the vehicle counting task. These approaches use a road region of interest [4], a virtual line [5], [6], [7], [8], a virtual detection zone [9], [10], [11], [12], or combination of virtual line and virtual detection zone [13] to count vehicles. However, these existing aproaches use CCTV camera positions facing the front or rear of the vehicle. Hence, it is difficult to identify number of wheel axles on the vehicle. Existing approaches also focus on freeway settings, whereas the average daily traffic survey was conducted in a non-freeway setting. Another approaches propose a vehicle counting framework for an average daily traffic survey using inception model [14] and in edge computing device [15]. These studies count vehicles which appear in each frame; therefore, the same vehicle is counted as many times as the number of occurences in the set of video frames. However, this approach is constrained by partial or complete object occlusion. This approach also does not provide mitigation for matching similar vehicle when vehicle disappear then then reappear in several frame later.

Based on the abovementioned problem, existing studies have not solved the problems associated to average daily traffic survey which CCTV camera positions is facing front or rear view of road, not the side of the road according to average daily traffic survey manner. In addition, most of the datasets used for count and track vehicles did not represent vehicle objects in average daily traffic survey manner, especially in Indonesia. Existing approaches relevant to average daily traffic survey also did not address the problems, namely, repeated counting of same vehicles.

In this article, the contributions are as follows:

1) We introduce a multi-vehicle tracking and counting framework for the side-view camera position of CCTV video in a non-freeway setting. We apply RT-DETR [16] architecture to identify vehicles and ByteTrack [17] to track and count vehicles in multi-object manner. The application of RT-DETR architecture in vehicle tracking and counting framework is proven to identify vehicle in video conditions during day and night video scenes.
2) To ease the vehicle counting process and eliminate bias in vehicle class results, we propose a new multi-vehicle voting algorithm to determine the dominant class appearing in each tracked vehicle. Inspired from [8], this multi-vehicle voting algorithm triggered only a vehicle passing through a vehicle counting line. Based on our ablation study, our proposed multi-vehicle counting algorithm increased the vehicle counting accuracy to an average value of 0.78.
3) To test the robustness of our vehicle counting framework, we tested six CCTV camera videos taken in various settings during the day and night. We also tested this proposed framework in one extra video scenes which contains rare vehicle classes. Our experimental results demonstrate the robustness of the RT-DETR and ByteTrack method combinations compared with other method combinations either in optimal condition or non-optimal condition such as motion blur and occlusion.

II. RELATED WORKS

Vehicle detection, tracking, and counting are essential tasks, especially for solving problems in road maintenance and transportation systems. This section provides an overview of vehicle detection, tracking, and counting methods.

A. VEHICLE DETECTION

The application of deep learning to vehicle detection tasks has given rise to various studies that use state-of-the-art deep learning architectures. Naufal et al. [18] proposed a method for detecting empty parking spaces using Mask RCNN [19] architecture on CCTV camera videos. Because the video background experiences varying light conditions, an exposure fusion framework was also proposed to combine contrast result and video exposure to enhance light conditions. Then Asy'ari [20] applied image stitching method by using YOLOv5 [21] to detect empty parking spaces on two overlapping CCTV cameras. Each feature in the two videos is matched based on the basis of the object features, and stitching is then performed. Luo et al. apply image enhancement to reduce the impact of varying illumination using the Faster R-CNN [22] architecture. This architecture was modified to improve detection performance in multiscale and occluded objects using feature enrichment and architecture search in its backbone [23]. Yang et al. [7] add attention mechanism in SSD [24] to combine multiscale features and expand the receptive areas of shallow features.

Meanwhile, transformer-based architecture, namely Detection Transformer (DETR) [25] shows promising results when compared with convolution-based architecture in vehicle detection tasks [26]. Thanks to its self-attention mechanism, DETR [25] predicts fixed number of bounding boxes by pay attention to different parts of images. Then, the Hungarian Algorithm [27] is used to associate predicted bounding boxes with ground-truth bounding boxes. However, this architecture needs a long training epoch to converge compared with other architectures. Another problem is that it has poor performance in small object detection because of the limitation of attention modules in processing image feature maps.

Zhu et al. [28] tries to overcome these problems with deformable attention module, which assigns only small fixed number of keys for each query. This strategy significantly reduces complexity and maintains spatial resolution for small object detection based on its experiment.

Zhang et al. [29] modify cross attention module to ease the matching process between object queries and target features. this modification makes thr architecture converge faster than vanilla version of DETR. In the vehicle detection task, Deshmukh et al. [30] used Swin Transfomer [31] to solve the multi-scale feature extraction problem while maintaining its performance. However, the transformer-based architecture still struggles with its speed when measured using frame-per-second (FPS) metrics.

To overcome high computational and memory complexity while maintaining its performance, Lv et al. [16] proposed Real-Time Detection Transformer (RT-DETR). RT-DETR uses an efficient hybrid encoder to process multiscale features and IoU-aware query selection to overcome the incosistent distribution of classification score and location confidence. Experimental results showed that RT-DETR outperfoms real-time and end-to-end object detector in terms of its average precision and FPS measurement. Meanwhile, YOLOv9 [32] proposed concept of programable gradient information in the YOLO architecture to solve the deep supervision problem in object detection tasks. Inference results on MS COCO [33] prove that YOLOv9 beats RT-DETR in terms of accuracy with fewer parameters.Therefore, we interested in comparing YOLOv9 with RT-DETR in our proposed vehicle counting framework, specifically in the vehicle detection phase.

B. VEHICLE TRACKING

The object tracking task is one of the most important parts of the vehicle calculation framework that assist the vehicle calculation process in a video frame. One of the object-tracking approaches is tracking by detection (TBD). TBD relies heavily on the performance of the object detection model because the performance of the model determines the result of the object search. Early examples of this approach are SORT [34] and Deep-SORT [34]. SORT is a pioneering TBD approach that utilizes Kalman Filter [35] in predicting the location of objects in the next frame. This method also uses Hungarian Matching [27] Algorithm to perform data association on tracked objects. However, SORT has the disadvantage of frequent ID switch of same object and is not resistant to occlusion. Furthermore, Deep-SORT is proposed by introducing a deep association metric to handle the weaknesses in SORT. SORT and Deep-SORT-based approaches have also been applied to vehicle tracking tasks [13], [36], [37]. Another method that uses the TBD approach is ByteTrack [17] by utilizing all bounding box detection results, both high and low values. This method consists of two stages to perform object matching on existing frames. This two-stage matching process allows to minimize occlusion and also differences in object size due to objects position that move away or approach the camera. ByteTrack has also been implemented in vehicle tracking tasks with different detection object models [26]. Based on the explanation above, our proposed vehicle calculation framework applies ByteTrack to help calculate vehicles.

C. VEHICLE COUNTING FRAMEWORK

Researchers have proposed several methods to fulfil the requirements for vehicle counting task as part of vehicle counting framework in Intelligent Transportation System (ITS). This methods are divided into four types based on how to count vehicles, namely region of interest mehod, virtual line method, virtual zone method, and object appearance method. Region of interest method mark some parts of video to count vehicles with segmentation algorithm. This approach can be used to count vehicle on the road so vehicle outside the road will not counted as vehicle [4].

Virtual line method define single or multiple line in frame to count vehicles. This line can be drawn vertically or horizontally, depending on the conditions in a frame. Song et al. [5] proposed vehicle counting system using virtual line method which use ORB algorithm [38] to track and count vehicles from YOLOv3 [39] detection results. This system also applies feature extraction on freeways to limit area for counting vehicles. Liu et al. [6] aplies weak camera calibration before track and count process to improve vehicle counting in each lane of road. Different from above methods, Yang et al. [7] define state-based vehicle counting with two virtual line to prevent overtracking and overdetection. Azimjonov et al. [8] define virtual line manually before perform vehicle counting system. This system also applies shake filter to calibrate hanging camera and voting algorithm to determine dominant vehicle class under same ID in consecutive frames.

Besides, virtual detection method used polygon area to track and count vehicle. This method can be drawn manually by user input or automatically when conduct vehicle counting system. Virtual detection method can be used to track and count vehicles from its tracjectory in each lane [9]. Lin et al. [10] used combination of virtual zone and Gaussian Mixture Model (GMM) to conduct vehicle counting. Neupane et al. [11] add transfer learning using different dataset to solve domain shift problem in vehicle counting. Virtual zone defined in each lane of a road to track and count vehicle. Meanwhile, tracking algorithm like SORT [34] applies in virtual counting zone to solve multiple ID switch in intersections [12]. Another research tries to combine virtual zone and virtual line to count vehicle using SORT [34] and DeepSORT [34] tracking algorithm and YOLOv4 [40] object detector [13]. Virtual zone used to help track vehicle in each lane, while virtual line assign vehicle counting task.

In the meantime, object appearance method carried out to count vehicle based on tracking algorithm. Using Kalman Filter and YOLOv5 detector, this method can count vehicles in occlusion condition [41]. Then, fisheye camera also used to count vehicles based on the virtual zone that the vehicle passes thorugh using YOLOv7 [42] and SORT [34] at road intersections [43]. Vehicle counting using virtual detection zone also conducted at road intersections using PP-YOLO [44] and DeepSORT [34] tracking algorithm [45].

Vehicle counting method also used to set traffic light green light time using YOLOv3 [39] and DeepSORT tracking algorithm [34] and can give reccomendation of green light duration in intersection [46]. Another approach use YOLOv6 [47] with custom backbone to count vehicle in congestion to solve traffic management problem [48].

However, none of the approaches can solve the vehicle counting task in an average daily traffic survey, either using virtual detection method or virtual line method. The average daily traffic survey itself must conducted at straight road section with no intersection and also conducted in non-freeway setting. In addition, all vehicle types used in the abovemetioned approaches are not as same as the average daily traffic survey vehicles, especially the vehicles used in Indonesia.

To solve those problems, Rifai et al. [14] used Inception [49] model to count vehicle as an average daily traffic survey manner. Another approach used edge computing and combination of SSD [24] with Mobilenetv2 [50] to count vehicle [15]. Another approach attempted to count vehicle in the same manner as average daily traffic survey using YOLOv8 [51] and virtual detection zones in each road lane [52]. However, none of approaches applies occlusion handling and vehicle counting conducted in every frame so same vehicle would be count as many as it appearance in entire video frames. Thus, those approaches lack a tracking algorithm to help track and count a vehicle based on real-time conditions, not based on the number of appearances in entire frames. In addition, those approaches only count vehicles based on large classes, such as cars, motorcycles, busses, and trucks, not vehicle types according to the average daily traffic survey manner.

III. PROPOSED FRAMEWORK

Figure 1 shows how the proposed multi-vehicle tracking and counting framework works. This illustration contains three major phases, which will be discussed in separate subsections: a multi-vehicle detection phase, a multi-vehicle tracking phase, and a new multi-vehicle voting algorithm which integrated into the multi-vehicle counting phase.

A. MULTI-VEHICLE DETECTION

Before performing the multi-vehicle detection phase, a vehicle model first needs to be generated as illustrated by Figure 2. After going through the training phase with RT-DETR, the final result of training phase is a RT-DETR model that will be used in multi-vehicle detection phase.

The multi-vehicle detection phase starts with obtained CCTV video from a road section was split into a collection of video frames. This collection was then detected by an object detector. RT-DETR was then used as object detector because of its robustness in many aspects. The frame is detected by RT-DETR and produces a set of bounding boxes, class predictions, and prediction scores of the objects. Figure 3 illustrate RT-DETR architecture based on its original

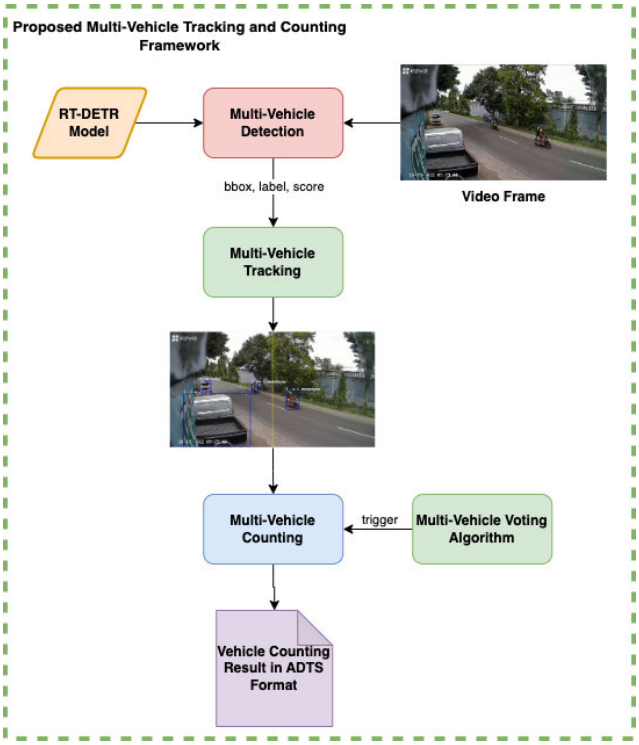

FIGURE 1. Illustration of the proposed multi-vehicle tracking and counting framework.

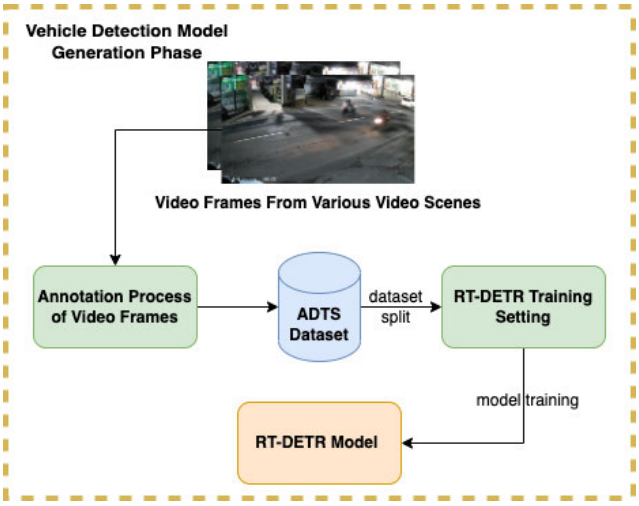

FIGURE 2. Illustration of vehicle model generation phase.

paper [16]. RT-DETR consists of a ResNet [53] backbone, an encoder, and a transformer decoder.

First, an image is fed into the Resnet backbone and the extracted output features of the last three stages of backbone (denoted as $S3$, $S4$, $S5$) are used as input to the efficient hybrid encoder. This encoder plays an important role in transforming features from backbone. Intrascale feature interaction (AIFI) reduces computational redundancy, which only performs intrascale interaction of backbone stage $S5$. Then, the Cross-scale Feature-fusion (CCFF) module based

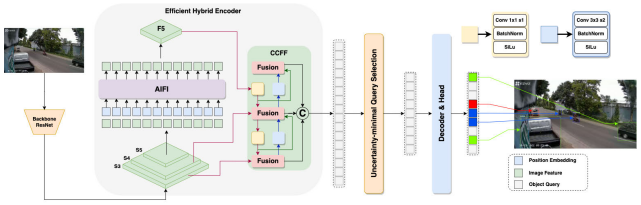

FIGURE 3. RT-DETR architecture adopted from its original paper [16].

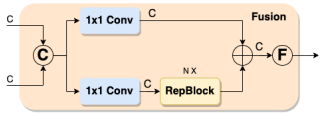

FIGURE 4. RT-DETR fusion block from its original paper [16].

on CNN inserts several fusion blocks into the fusion path. Figure 4 illustrate fusion block inside this module. Fusion block tasked to fuse adjancent feature into a new feature.

The Hybrid encoder produces an output sequence that is taken into Uncertainty-minimal Query Selection. This module selects a fixed number of image features as input of the transformer decoder. Finally, a transformer decoder with auxiliary prediction heads generates bounding boxes, confidence scores, and class predictions by iteratively optimizing its object queries.

B. MULTI-VEHICLE TRACKING

After obtaining a set of object bounding boxes, scores, and class predictions from the object detection phase, this information is entered into the tracking module to track all objects in the entire frame. Figure 5 shows how tracking module works in the proposed framework using ByteTrack inspired by the original paper pseudocode [17] and ByteTrack illustration from [54]. Firstly, a set of object information from each frame is divided into two types based on the bounding box confidence score. If the confidence score is higher than the threshold, this bounding box is categorized as a high confidence score bounding box; otherwise, it is categorized as a low confidence score bounding box.

Kalman filter is adopted to predict new location of each object track in current frame. First data association was also performed between high confidence score bounding box and all tracks in each frame (whether active or lost tracks). This association computes similarity between high confidence score bounding box and predicted tracks from kalman filter using IoU distance. For matched tracks will be updated using kalman filter and taken into list of active tracks. Unmatched measurement from first association treated as initialization of new track and taken into list of active tracks. Then, unmatched tracks from the first association go to second data association, which computes the similarity between unmatched tracks from first association and low confidence score of bounding box. Matched track from second data association updated using kalman filter and taken into list of active tracks.

While unmatched tracks from second data association taken into list of lost tracks. After n number of consecutives frame, this list of lost tracks is removed. Finally, list of active tracks produces output tracks that contain information about object bounding box, object ID, and object label. This information is fed as input of a new multi-vehicle voting algorithm to determine dominant object class label.

C. MULTI-VEHICLE COUNTING

All tracked vehicles must pass through a counting line to be listed in vehicles counting result. The line is positioned vertically at the middle of the frame. The purpose of this counting line is to identify which vehicle ID centroid that passes through the counting line. If one or more vehicles pass through the counting line, the framework determines the location of the intersection point between the vehicle centroid tracking line and the counting line. If the location of this intersection point is obtained, the vehicle is moved to the multi-vehicle voting algorithm tho determine the dominant class of the vehicle. It is noteworthy that tracking algorithm must maintain their tracking line from each vehicle ID to be counted by the proposed framework.

D. MULTI-VEHICLE VOTING ALGORITHM

Inspired from [8], we propose a new multi-vehicle voting algorithm to determine the dominant class of vehicles which passing through the counting line. Different from [8] approach which initiates voting algorithm after 30 consecutive frames, the proposed multi-vehicle voting algorithm initiates when one or more vehicles pass through the counting line.

Multi-vehicle voting algorithm is necessary because an object detector may produce incorrect detection results in vehicle class prediction. Sometimes, object detector produce different vehicle class prediction when an object appear in video frames. First, an object is detected as the light truck class, but when near the counting line, it detected as the medium truck class. To the best of our knowledge, previous approaches of vehicle counting based on a virtual counting line did not mention about filtering dominant objects from entire video frames, except the approach from [8]. Algorithm 1 shown pseudo-code of proposed multi-vehicle voting algorithm.

When one or more objects pass through the counting line, the framework retrieves class prediction information of objects that pass through the counting line based on object ID. This information is stored in a temporary list, and the list searches for most class prediction results. After obtaining the most class prediction result based on object ID, object ID information was updated with new class prediction based on multi-vehicle voting algorithm result and counted as vehicle.

At the end of the video frame, the framework will produce a file in pdf format that contains all counting results from each vehicle class in the video that has been processed before. In addition to counting results, this file contains timestamps

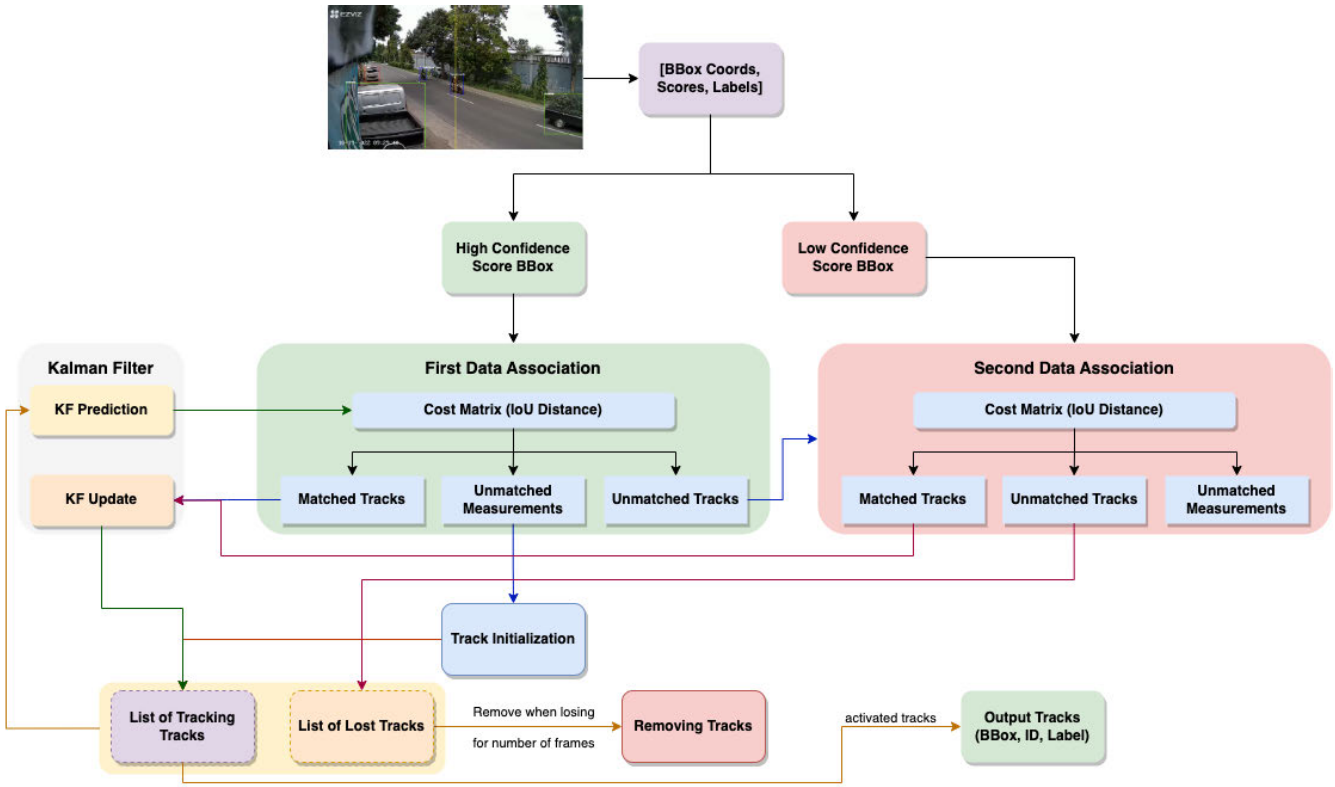

FIGURE 5. ByteTrack tracking algorithm adopted from [17] and [54].

Algorithm 1 Pseudocode of Multi-Vehicle Voting Algorithm for Filter Dominant Vehicle Class Label for Vehicle Pass-Through Counting Line

Input: Vehicle object info array O_l, vehicle object id intersect $O_d i$
Output: Dominant class of vehicle object C_d
begin
  $temp \leftarrow []$;
  for $key = 0$ *and* $val = 0$ **to** *items of* 0_l **do**
    for $i = 0$ **to** *len of val* **do**
      if $O_l[i][2] = O_d i$ **then**
        $temp \leftarrow temp \cup O_l[i][3]$
      end
    end
  end
  search max value of temp;
  $C_d \leftarrow$ max value of *temp*;
end

of the counting process and FPS results for every video that has been processed before.

IV. EXPERIMENTS AND RESULTS

In this section, the average daily traffic survey dataset used in this study is briefly explained, and the experiment settings for training the object detection model and hyperparameters used in the tracking module are described. This section also describes the evaluation results using various object detector in the vehicle detection task and vehicle counting results each video scenario. Therefore, this section is divided into sub-sections average daily traffic survey dataset, experiment settings, vehicle detection results, and vehicle tracking and counting results.

A. AVERAGE DAILY TRAFFIC SURVEY DATASET

This study used a dataset of average daily traffic survey [3] which contains 12 vehicle classes. The division of these vehicle classes is based on vehicle shape and number of wheel axles. Figure 6 illustrate of each vehicle class in this dataset. It is worth noting that 6a-light truck class is slightly different with 6b-medium truck. The difference between these two classes is located in its rear wheel shape, which if we look closely 6a-light truck has only one pair rear wheel axle while 6b-medium truck has two pair rear wheel axle. This dataset was obtained from various video of a CCTV camera placed on the side of a road section and taken both during the day and night.

All videos recorded in this dataset were placed only in a relatively straight road because the average daily survey focused only on one section of road. This setup also explains that all videos were recorded not at intersection or on highways. The background condition of the road were limited to only during daytime and nighttime condition using a

FIGURE 6. Example of each vehicle class in proposed framework.

TABLE 1. Number of objects in dataset of average daily traffic survey.

Vehicle Class	Number of Objects
1-Motorcycle	16,852
2-Car	14,319
3-Passenger Car	8,050
4-Pickup	10,434
5a-Small Bus	409
5b-Big Bus	1,290
6a-Light Truck	2,904
6b-Medium Truck	7,467
7a-Heavy Truck	2,108
7b-Double Truck	163
7c-Trailer Truck	1,581
8-No Motor Vehicle	4,097
Total Objects	**67566**

CCTV camera installed on a straight pole. This dataset not included extreme condition such as extreme weather condition or vehicle backlight which, difficult to identify the vehicle that passes through the videos.

These videos were divided into frames, and the division result generated 46,610 frames. These generated frames then annotated on the basis of their vehicle class. Table 1 described annotation result from each class of dataset in general. After dataset annotation phase, this dataset was divided into training set and evaluation set with percentage of 80% and 20%, respectively.

B. EXPERIMENT SETTINGS

This experiment was conducted with a single NVIDIA GeForce RTX 4070 12GB GPU graphics, an Intel Core I5-13600K CPU, and 32 GB RAM. This experiment used three object detection models namely RT-DETR [16], YOLOX [55], and YOLOv9 [32]. To train the object detection model, this experiment used 36 epochs, 8 batch size, and video frame resized to 640*640. Other hyperparameters, including the augmentation setups, follow default recipe from the original paper. For the tracking module, this experiment used two tracking algorithms namely ByteTrack [17] and SORT [34].

FIGURE 7. Video scenes used for testing the proposed framework.

Similar to the object detection model, this tracking experiment follows the default recipe of hyperparameters from each original paper. Some modification of the tracking algorithm need to be conducted specifically for considering class prediction results from the object detection model. This experiment used 0.7 as score threshold of bounding box as the input of tracking algorithm. To count vehicles, this experiment used six videos taken from various scenes and different locations. There are four day condition videos and two night conditions videos. For the night video, this experiment used video scenes with sufficient light. It is worth noting that this experiment also conducted in video scenes with various camera positions but was still able to monitor wheel axles from each vehicle. In addition, this experiment was conducted to process 3000 video frames from each video. Figure 7 shows video scenes used in this experiment. We also conducted manual vehicle counting as ground-truth for the proposed framework.

However, we found that these six videos lacks of small bus, big bus, and double truck vehicle class. Thus, we add an extra video scene to fulfil ground-truth gap. Figure 8 shows extra video scene use in this experiment. In addition, Table 2 shows actual counting results from each videos used in this experiment. These seven videos were different from those used as the average daily traffic survey dataset.

C. VEHICLE DETECTION RESULT

Table 3 shows comparison of the object detection models in terms of mean average precision (mAP). Result shows that

TABLE 2. Actual counting results from video scenes.

Class	Actual Counting Result						
	Video 1 (Day)	Video 2 (Day)	Video 3 (Day)	Video 4 (Day)	Video 5 (Night)	Video 6 (Night)	Video 7 (Addition)
1-Motorcycle	35	58	36	46	209	113	425
2-Car	12	12	4	3	117	20	191
3-Passenger Car	0	0	0	0	6	0	10
4-Pickup	7	1	3	3	7	5	44
5a-Small Bus	0	0	0	0	0	0	0
5b-Big Bus	0	0	0	0	0	0	2
6a-Light Truck	4	0	1	0	2	0	2
6b-Medium Truck	4	4	3	0	13	10	27
7a-Heavy Truck	0	0	1	0	0	1	5
7b-Double Truck	0	0	1	0	0	1	5
7c-Trailer Truck	0	0	0	0	2	0	2
8-No Motor Vehicle	2	1	0	1	0	0	1

FIGURE 8. Extra video scene.

TABLE 3. Mean Average Precision (mAP) from each object detection model.

Object Detector	Params(M)	mAP@50	mAP@50-95
RT-DETR R50 [16]	**42**	**0.992**	0.888
RT-DETR R101 [16]	76	**0.992**	**0.891**
YOLOX-l [55]	54	0.988	0.865
YOLOX-x [55]	99	0.988	0.869
YOLOv9-c [32]	51	0.988	0.889
YOLOv9-e [32]	69	0.988	0.890

RT-DETR ResNet101 outperfoms all variants of YOLOX and YOLOv9 with an mAP@50-95 score 0.891. RT-DETR for all variants also outperforms all variants of YOLOX and YOLOv9 with an mAP@50 score of 0.992. In addition, wel also calculated F1 score for all object detectors, and the results are shown in Table 4. Result shows that RT-DETR Resnet101 outperforms all object detectors in terms of its Mean Average Recall (mAR) and F1-Score, with scores of 0.930 and 0.91, respectively. From this point, each best model type from each object detection model taken to conduct an experiment in tracking and counting of the proposed framework. Specifically, the tracking and counting experiments used RT-DETR with ResNet101 backbone version, YOLOX-x version, and YOLOv9-e version.

We also conduct analysis of each best models in each vehicle class to produce average precision result. Table 5 shows each vehicle class average precision results from each best models. Result shows that YOLOv9-e outperforms all

TABLE 4. F1-score of each object detector.

Object Detector	mAP	mAR	F1-Score
RT-DETR R50 [16]	0.888	0.922	0.905
RT-DETR R101 [16]	**0.891**	**0.930**	**0.910**
YOLOX-l [55]	0.865	0.897	0.881
YOLOX-x [55]	0.869	0.899	0.884
YOLOv9-c [32]	0.889	0.910	0.899
YOLOv9-e [32]	0.890	0.913	0.901

TABLE 5. Average Precision each vehicle class from the best object detection model.

Class	Average Precision Result		
	RT-DETR R101	YOLOX-x	YOLOv9-e
1-Motorcycle	**0.891**	0.760	0.799
2-Car	**0.992**	0.858	0.885
3-Passenger Car	**0.983**	0.876	0.908
4-Pickup	0.745	0.871	**0.905**
5a-Small Bus	0.824	0.887	**0.924**
5b-Big Bus	0.898	0.895	**0.934**
6a-Light Truck	0.837	0.893	**0.922**
6b-Medium Truck	0.919	0.897	**0.932**
7a-Heavy Truck	0.921	0.912	**0.949**
7b-Double Truck	0.788	0.892	**0.920**
7c-Trailer Truck	0.869	0.917	**0.959**
8-No Motor Vehicle	**0.928**	0.771	0.816

models in pickup, bus, and truck classes, while RT-DETR Resnet101 outperfoms all models in private vehicle classes such as motorcycle, car, passenger car, and no motor vehicle classes. This result also shows that RT-DETR Resnet101 has good performance if the number of objects in dataset plenty enough, otherwise the result will degraded.

D. VEHICLE TRACKING AND COUNTING RESULT

First, every bounding box was produced by each object detector rescaled to the original frame size. Information about the object is fed to the tracking module. Table 6 described counting results from various combination object detection models and tracking algorithms in addition to the

TABLE 6. Counting results using various object detectors and trackers.

Video	Detector	Tracker	FPS	Counting Result Accuracy (%)											
				1-Motor Cycle	2-Car	3-Passenger Car	4-Pickup	5a-Small Bus	5b-Big Bus	6a-Light Truck	6b-Medium Truck	7a-Heavy Truck	7b-Double Truck	7c-Trailer Truck	8-No Motor Vehicle
Video 1 (Day)	RT-DETR R101	ByteTrack	1.99	31	11	0	7	0	0	1	5	0	0	0	1
		SORT	1.96	29	11	0	7	0	0	1	6	0	0	0	1
	YOLOX-x	ByteTrack	5.10	32	12	0	7	0	0	0	7	0	0	0	0
		SORT	**5.21**	31	10	0	7	0	0	0	7	0	0	0	1
	YOLOv9-e	ByteTrack	3.37	19	10	0	5	0	0	0	7	0	0	0	0
		SORT	3.42	17	7	0	5	0	0	0	7	0	0	0	0
Video 2 (Day)	RT-DETR R101	ByteTrack	1.99	50	10	0	1	0	0	0	4	0	0	0	1
		SORT	1.95	51	11	0	1	0	0	0	3	0	0	0	1
	YOLOX-x	ByteTrack	5.02	51	10	0	1	0	0	0	4	0	0	0	1
		SORT	**5.18**	50	10	0	1	0	0	0	3	0	0	0	0
	YOLOv9-e	ByteTrack	3.39	46	11	0	1	0	0	0	4	0	0	0	1
		SORT	3.38	44	10	0	1	0	0	0	4	0	0	0	0
Video 3 (Day)	RT-DETR R101	ByteTrack	1.99	35	4	0	4	0	0	1	4	0	0	0	0
		SORT	1.98	34	3	0	3	0	0	1	4	0	0	0	0
	YOLOX-x	ByteTrack	**5.17**	35	4	0	4	0	0	1	2	2	0	0	0
		SORT	5.16	35	4	0	3	0	0	1	4	0	0	0	0
	YOLOv9-e	ByteTrack	3.4	22	3	0	4	0	0	0	5	0	0	0	0
		SORT	3.42	23	4	0	2	0	0	0	4	1	0	0	0
Video 4 (Day)	RT-DETR R101	ByteTrack	1.99	40	2	0	3	0	0	0	0	0	0	0	1
		SORT	1.95	39	1	0	3	0	0	0	0	0	0	0	1
	YOLOX-x	ByteTrack	**5.23**	40	2	0	3	0	0	0	0	0	0	0	0
		SORT	5.07	35	1	1	2	0	0	0	0	0	0	0	1
	YOLOv9-e	ByteTrack	3.36	23	0	0	4	0	0	0	0	0	0	0	0
		SORT	3.43	22	1	0	2	0	0	0	1	0	0	0	1
Video 5 (Night)	RT-DETR R101	ByteTrack	1.92	48	107	3	5	0	0	2	10	1	0	2	0
		SORT	1.91	64	91	4	4	0	0	0	8	1	0	1	0
	YOLOX-x	ByteTrack	4.9	44	63	2	6	0	0	1	5	5	8	0	0
		SORT	**4.93**	37	55	1	3	0	0	1	5	2	5	0	0
	YOLOv9-e	ByteTrack	3.27	22	59	0	2	0	0	1	4	3	0	0	0
		SORT	3.31	33	48	0	2	0	0	1	4	2	0	0	0
Video 6 (Night)	RT-DETR R101	ByteTrack	1.94	82	17	2	4	0	0	0	8	1	0	0	0
		SORT	1.90	82	16	2	3	0	0	0	8	1	0	0	0
	YOLOX-x	ByteTrack	**5.12**	73	17	1	4	0	0	0	9	1	0	0	0
		SORT	4.95	72	15	2	4	0	0	0	7	1	0	0	0
	YOLOv9-e	ByteTrack	3.27	21	14	2	5	0	0	0	5	1	0	0	0
		SORT	3.31	21	14	1	5	0	0	0	5	1	0	0	0
Video 7 (Addition)	RT-DETR R101	ByteTrack	1.7	190	152	10	36	0	2	1	20	8	0	0	0
		SORT	1.8	231	127	7	32	0	2	0	18	6	0	0	0
	YOLOX-x	ByteTrack	**4.13**	195	135	11	33	0	4	2	17	7	1	0	0
		SORT	**4.13**	218	114	4	28	0	3	1	9	4	1	0	0
	YOLOv9-e	ByteTrack	2.87	188	122	4	35	0	4	2	16	4	0	0	3
		SORT	2.86	194	98	3	26	0	2	1	11	3	0	1	1

TABLE 7. Counting accuracy results from the combination of object detector and tracker compared with actual counting result.

Video	Detector	Tracker	Counting Result Accuracy (%)												
			1-Motor Cycle	2-Car	3-Passenger Car	4-Pickup	5a-Small Bus	5b-Big Bus	6a-Light Truck	6b-Medium Truck	7a-Heavy Truck	7b-Double Truck	7c-Trailer Truck	8-No Motor Vehicle	Avg Acc (%)
Video 1 (Day)	RT-DETR R101	ByteTrack	88.57	91.67	100	100	100	100	25	80	100	100	100	50	**86.27**
		SORT	82.86	91.67	100	100	100	100	25	66.67	100	100	100	50	84.68
	YOLOX-x	ByteTrack	91.43	100	100	100	100	100	0	57.14	100	100	100	0	79.05
		SORT	88.57	83.33	100	100	100	100	0	57.14	100	100	100	50	81.59
	YOLOv9-e	ByteTrack	54.29	83.33	100	71.43	100	100	0	57.14	100	100	100	0	72.18
		SORT	48.57	58.33	100	71.43	100	100	0	57.14	100	100	100	0	69.62
Video 2 (Day)	RT-DETR R101	ByteTrack	86.21	83.33	100	100	100	100	100	100	100	100	100	100	97.46
		SORT	87.93	91.67	100	100	100	100	100	75	100	100	100	100	96.22
	YOLOX-x	ByteTrack	87.93	83.33	100	100	100	100	100	100	100	100	100	100	**97.61**
		SORT	86.21	83.33	100	100	100	100	100	75	100	100	100	0	87.05
	YOLOv9-e	ByteTrack	79.31	91.67	100	100	100	100	100	100	100	100	100	100	97.58
		SORT	75.86	83.33	100	100	100	100	100	100	100	100	100	0	88.26
Video 3 (Day)	RT-DETR R101	ByteTrack	97.22	100	100	75	100	100	100	75	0	100	100	100	87.27
		SORT	94.44	75	100	100	100	100	100	75	0	100	100	100	87.04
	YOLOX-x	ByteTrack	97.22	100	100	75	100	100	100	66.67	50	100	100	100	**90.74**
		SORT	97.22	100	100	100	100	100	100	75	0	100	100	100	89.35
	YOLOv9-e	ByteTrack	61.11	75	100	75	100	100	0	60	0	100	100	100	72.59
		SORT	63.89	100	100	66.67	100	100	0	75	100	100	100	100	83.80
Video 4 (Day)	RT-DETR R101	ByteTrack	86.96	66.67	100	100	100	100	100	100	100	100	100	100	**96.14**
		SORT	84.78	33.33	100	100	100	100	100	100	100	100	100	100	93.18
	YOLOX-x	ByteTrack	86.96	66.67	100	100	100	100	100	100	100	100	100	0	87.80
		SORT	76.09	33.33	0	66.67	100	100	100	100	100	100	100	100	81.34
	YOLOv9-e	ByteTrack	50	0	100	75	100	100	100	100	100	100	100	0	77.08
		SORT	47.83	33.33	100	66.67	100	100	100	0	100	100	100	100	78.98
Video 5 (Night)	RT-DETR R101	ByteTrack	22.97	91.45	50	71.43	100	100	100	76.92	0	100	100	100	**76.06**
		SORT	30.62	77.78	66.67	57.14	100	100	0	61.54	0	100	50	100	61.98
	YOLOX-x	ByteTrack	21.05	53.85	33.33	85.71	100	100	50	38.46	0	0	0	100	48.53
		SORT	17.70	47.01	16.67	42.86	100	100	50	38.46	0	0	0	100	42.72
	YOLOv9-e	ByteTrack	10.53	50.43	0	28.57	100	100	50	30.77	0	100	0	100	47.52
		SORT	15.79	41.02	0	28.57	100	100	50	30.77	0	100	0	100	47.18
Video 6 (Night)	RT-DETR R101	ByteTrack	72.57	85	0	80	100	100	100	80	100	100	100	100	84.80
		SORT	72.57	80	0	60	100	100	100	80	100	100	100	100	82.71
	YOLOX-x	ByteTrack	64.60	85	0	80	100	100	100	90	100	100	100	100	**84.97**
		SORT	63.72	75	0	80	100	100	100	70	100	100	100	100	82.39
	YOLOv9-e	ByteTrack	18.58	70	0	100	100	100	100	50	100	100	100	100	78.22
		SORT	18.58	70	0	100	100	100	100	50	100	100	100	100	78.22
Video 7 (Addition)	RT-DETR R101	ByteTrack	44.71	79.58	100	81.81	100	100	50	74.1	62.5	0	0	0	57.72
		SORT	54.35	66.49	70	72.72	100	100	0	66.67	83.33	0	0	0	51.13
	YOLOX-x	ByteTrack	45.88	70.68	90.91	75	100	50	100	62.96	71.43	100	0	0	**63.91**
		SORT	51.29	59.69	40	63.63	100	66.67	50	33.33	80	100	0	0	53.72
	YOLOv9-e	ByteTrack	44.24	63.87	40	79.54	100	50	100	59.26	80	0	0	33.33	54.19
		SORT	45.65	51.31	30	59.1	100	100	50	40.74	60	0	50	100	57.23

combination of RT-DETR with ByteTrack. To show the robustness of each combination, each counting result was compared with the actual counting result from table 2 to produce accuracy percentage with equation 1 adopted from [8], where *Acc* is the accuracy of counting result, *ES* is the object counted by proposed framework, and *GT* is the object counting ground-truth. It is worth noting that this work used equation 1 because sometimes estimated counting result may sometimes exceed the vehicle counting ground truth. In another case, the estimated counting result produces a value of zero, but ground-truth value is not zero.

$$Acc(GT, ES) = \begin{cases} 100 & \text{if } GT = ES \\ (\frac{ES}{GT}) \times 100 & \text{if } GT > ES \\ (\frac{min(GT, ES)}{max(GT, ES)}) \times 100 & \text{if } GT < ES \end{cases} \tag{1}$$

After the accuracy of each video and each combination of detector and tracking acquired, the average accuracy is determined using a simple average calculation represented in equation 2, where *Avg* is the accuracy of the entire class from each video and each combination of detector and tracker, and $num_{classes}$ is the total number of vehicle classes, which by default is 12.

$$Avg = \frac{\sum Acc}{num_{classes}} \tag{2}$$

Then, each accuracy and average accuracy result plotted in table 7.

V. DISCUSSION

To provide a brief insight from the proposed framework test videos as shown in Figure 7 and from an extra video as shown in Figure 8, each video case can be listed as follows. Video 1 is located in a non-crowded area with two-lane roads. The vehicle that pass by in this video are quite varied and are dominated by motorcycle and car classes. Some vehicles belonging to the pickup class can be seen in this video. The trucks class in the video is dominated by light and medium trucks. Video 2 captures the relatively slow vehicle traffic due to the small road and the slow speed of truck in video. Video 3 has almost the same conditions as video 1, except that in this video, the heavy truck class can be found occasionally passing through the road. In video 4, there is a slight challenge in that the visibility of the camera is obstructed by other objects; therefore, the proposed framework cannot detect vehicles in a long-distance manner. Video 5 is set on a busy road with truck classes often found here. However, in this video, the passing vehicles experience motion blur because of high speed, which is challenging for the proposed framework. This also happens in video 6, although the road conditions in video 6 are relatively not crowd than in video 5. Finally, video 7 had the same settings as video 5. The difference is video 7 was recorded in the daytime setting while video 5 was recorded in the nighttime setting. In addition, all vehicles in video 7 did not experience motion blur, as observed in video 5 and 6. Thus, it can be concluded that motion blur occurs only in nighttime scenes.

Based on the experimental results, CCTV camera video must be clear and has sufficient light to track and count vehicles, especially during night conditions. Camera CCTV position also plays an important role in the robustness of the proposed framework, such as the height of camera from the road section until the field of view. However, CCTV camera sometimes experience object motion blur caused by the velocity of object, especially in nighttime video scene. This phenomenon leads to RT-DETR ResNet 101 rarely failed to identify objects. Compared with YOLOX-x and YOLOv9-e,RT-DETR ResNet 101 has better performance in the detection of motion blur objects.

Figure 9 shows the robustness of RT-DETR ResNet 101 in the detection of motion blur objects. It can be seen that RT-DETR with ResNet 101 as the backbone can detect accurately three objects in the scene, while YOLOX-x cannot detect white car object and YOLOv9-e failed to detect all objects in the scene. RT-DETR ResNet 101 also can detect small object correctly even though it does not provide stable bounding box location during frame changes. However, one-to-one assignment in transformer architecture affects RT-DETR ResNet 101 performancee speed in processing each frame of video so RT-DETR ResNet 101 not suitable for real-time video object detection task.

From the average accuracy result for each variant, both RT-DETR ResNet 101 and YOLOX-x have the best performance in some videos. However, YOLOX-x suffers from a combination of crowded fast moving objects, which leads to degraded performance in predicting bounding-box location, as shown in video 5. While RT-DETR ResNet 101 has better performance when crowded fast moving objects appear in the scene. Then, RT-DETR ResNet 101 and ByteTrack combination is more resistant to ID switch than other variants because ByteTrack employs a two-stage association method that is resistant to short-term occlusion. While RT-DETR ResNet 101 give more stable detection in each frame, it can improve ByteTrack tracking robustness.

YOLOv9-e is not sufficiently stable in detecting objects in all videos scenes, and this phenomenon leads to degraded performance of the tracking module. Another disadvantage of YOLOv9-e is that it often experiencing misprediction in clear objects. Figure 10 shows object detector performance in daytime scenes, which is the best condition for experiment. Although YOLOX-x has the best FPS among all object detectors, it is sometimes not resistant to ID switching when an object experiences short-term occlusion. Another weakness of YOLOX is that it sometimes gives false class prediction for some objects. Figure 11 shows comparison between all object detectors with ByteTrack as the tracking module when experiencing short-term occlusion. It can be seen that when an object motorcycle experiences ID switch in both YOLOX-x and YOLOv9-e, while in RT-DETR ResNet 101 it can maintain the same ID after short-term occlusion.

FIGURE 9. Comparison of object detectors when experiencing motion blur.

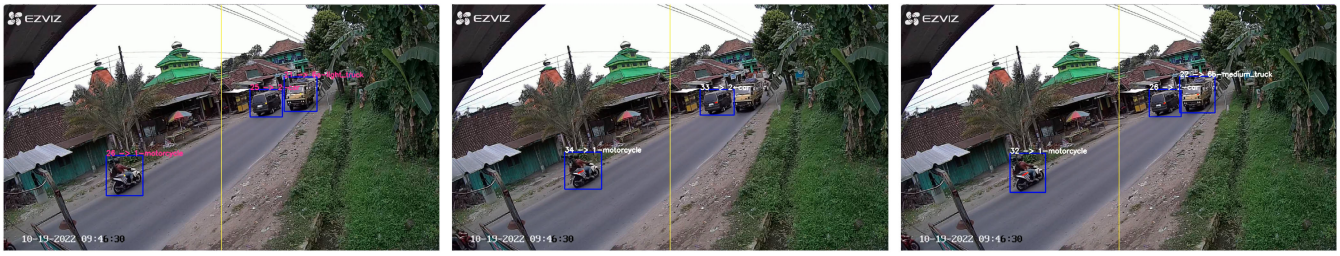

FIGURE 10. Comparison of object detector performance in daytime video scene.

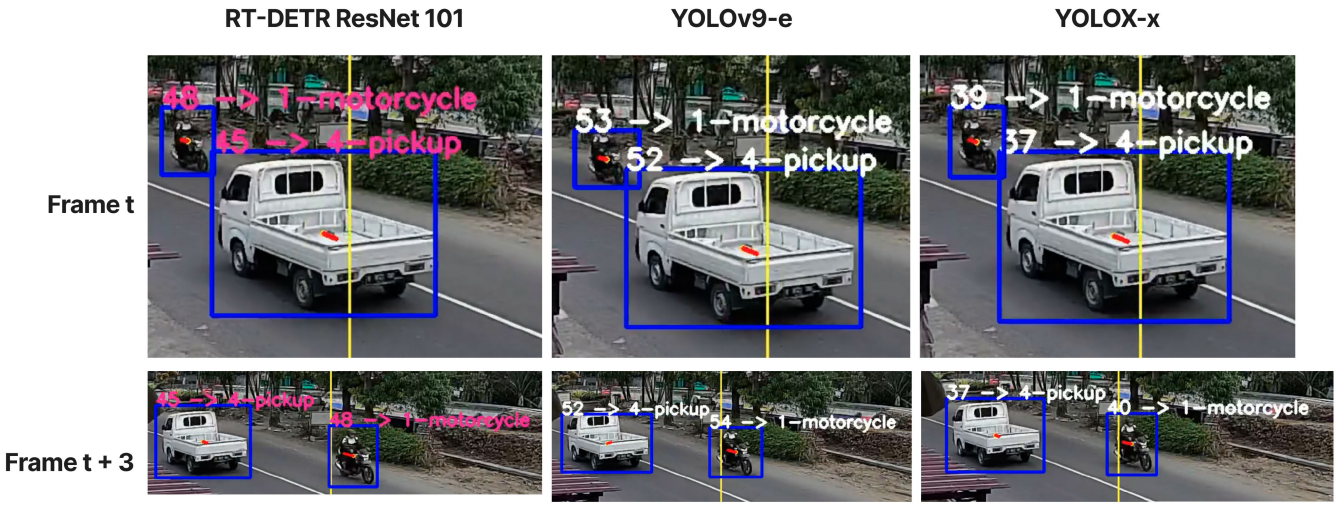

FIGURE 11. Comparison of ByteTrack with different object detector in short-term occlusion.

For the tracking module, ByteTrack clearly outperfoms SORT in terms of tracking robustness. While ByteTrack can maintain the same ID when the object detector is not sufficiently enough in predicting object location, SORT performance would be degraded when this phenomenon occurs. This is because SORT does not rely on appearance feature but only on the association matrix. Thanks to the Byte algorithm, ByteTrack can produce fewer IDs and resistant to short-term occlusion compared to SORT.

Besides ByteTrack robustness against SORT in constantly tracking object and when handling object short-term occlusion, there are several limitations from ByteTrack, which are described as follows: (1) When objects experience motion blur, sometimes ByteTrack experiences ID switch for the same object due to blurred appearance of objects. (2) While its robustness proved in short-term occlusion, sometimes ByteTrack gives a new ID to same object when experiencing long-term occlusion, this phenomenon depicted in Figure 12.

VI. ABLATION ANALYSIS

We compared the robustness of each detector and tracking algorithm combination on vehicle counting accuracy based on Table 7 results to obtain average results from all video scenes. Table 8 shows counting accuracy results of each

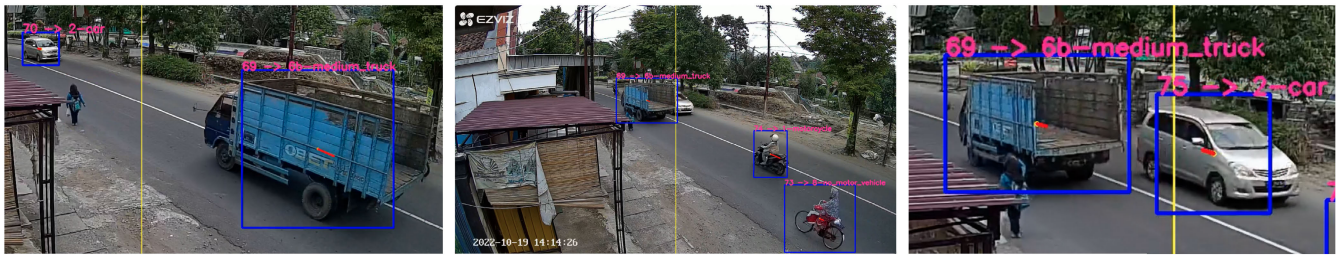

FIGURE 12. Long-term occlusion problem in video scene. Condition before the object car experiencing long-term occlusion (left), when long-term occlusion occurs (middle), and ID change after long-term occlusion (right).

TABLE 8. Counting accuracy results of each combination from all video scenes including extra video scene.

Detector	Tracker	FPS All	Acc All (%)
RT-DETR R101	ByteTrack	1.93	**83.67**
	SORT	1.92	79.56
YOLOX-x	ByteTrack	**4.95**	78.94
	SORT	4.94	74.02
YOLOV9-e	ByteTrack	3.28	71.34
	SORT	3.31	71.90

TABLE 9. Counting accuracy results of each video from all combination of detectors and trackers.

Video Name	Acc All (%)
Video 1 (Day)	78.89
Video 2 (Day)	**94.03**
Video 3 (Day)	85.13
Video 4 (Day)	87.73
Video 5 (Night)	54
Video 6 (Night)	81.88
Video 7 (Addition)	56.32

TABLE 10. The effect of using multi-vehicle voting algorithm.

Detector	Tracker	Counting Accuracy (%)	
		without voting	with voting
RT-DETR R101	ByteTrack	83.33	83.67 (+ 0.34)
	SORT	78.98	79.56 (+ 0.58)
YOLOX-x	ByteTrack	77.57	78.94 (+ 1.37)
	SORT	73.63	74.02 (+ 0.39)
YOLOv9-e	ByteTrack	70.13	71.34 (+ 1.21)
	SORT	71.11	71.90 (+ 0.79)

combination from all video scenes including an extra video. From table 8 can be concluded that ByteTrack outperforms SORT in all video scenes including extra video scene for the tracking module with an average accuracy above 83%. Meanwhile, YOLOX-x outperforms all object detector in terms of frame-per-second (FPS) in all video scenes including extra video scene. Self-attention inside RT-DETR encoder and decoder makes RT-DETR has the lowest FPS compared to other detectors because self-attention itself has a huge computational cost compared to convolution.

The counting accuracy of each video from all combinations of detectors and trackers is shown in Table 9. It can be concluded that video 2 (Day) has highest counting accuracy among all video scenes. Video 2 (Day) had non-crowded traffic, and also occlusion rarely occured in this video scene. Occlusion occurs in this scene categorized as short-term occlusion; thus, the TBD approach can handle this phenomenon with good results.

To demonstrate the impact of using the multi-vehicle voting algorithm, we compared the effects of enabling and obstructing the multi-vehicle voting algorithm in terms of counting accuracy. The proposed framework only uses a previous neighbor video frame information to determine vehicle labels for not using multi-vehicle voting algorithm. The result of this comparison depicted in Table 10. It can be clearly seen that the addition of multi-vehicle voting algorithm increase counting accuracy with average value of 0.78. Multi-vehicle voting algorithm also can filter confusion for vehicle labels decision. Occasionally, objects which first detected in frame not have same vehicle label compared when object near counting line. This happen due to object detector confusion for predicting vehicle object. If object detector not robust enough, it can lead to false detection when reaching counting line.

VII. CONCLUSION AND FUTURE WORK

An average daily traffic survey framework for tracking and counting vehicles was proposed in this work. RT-DETR with ByteTrack was used as the core process for tracking and counting vehicle. The experimental result shows that RT-DETR ResNet 101 combination achieves high accuracy compared to all object detector variants with an mAP@50-95 value of 0.891. In addition, RT-DETR ResNet 101 outperfomed all object detectors in terms of the F1-Score (0.91). The average precision result for each vehicle class shows that RT-DETR ResNet 101 achieve the best results in motorcycle, car, passenger car, and non-motor vehicle classes, while YOLOv9-e achieve best results in pickup, all bus variant, and all truck variant classes.

This study also demonstrate the effect of a multi-vehicle voting algorithm that increases counting accuracy by approximately 0.78 compared to using only the first previous frame information. The result also shows that the proposed

multi-vehicle voting algorithm can eliminate misclassified object detector when predicting vehicle object labels.

Despite YOLOX-x variant outperform RT-DETR ResNet 101 in several video scenes and in processing speed measured by FPS, it can be seen from discussion that RT-DETR ResNet 101 has better performance in detecting each vehicle, especially in nighttime scene videos when motion blur occurs. RT-DETR ResNet 101 also detect objects more stable than YOLOX and YOLOv9, which leads to robustness of the tracking module in counting vehicles. For the tracking and counting vehicle task, ByteTrack outperforms SORT in all video scenes with an average accuracy value above 75%. It can also be seen that ByteTrack is more resistant to occlusion, which sometimes occurs during the processing of video frames, than SORT. In addition, SORT rejects some object bounding box, which leads to ID switch on same object whether occlusion happen or not.

This work also shows that the proposed multi-vehicle tracking and counting framework can solve previous problem in average daily traffic survey, which is still conducted using a manual or a semi-manual approach. Based on these findings, we are excited to improve this framework capability in counting vehicle when long-term occlusion occurs. Another future work is to improve the speed of RT-DETR ResNet 101 when processing each video frame.

ACKNOWLEDGMENT

The authors would like to express their gratitude to the partner company for providing the testing video for this work.

REFERENCES

[1] L. F. Huntsinger, "Part 2—Transportation systems planning," in *Highway Engineering*, 2nd ed., D. J. Findley, C. M. Cunningham, T. H. Brown, L. M. Cahill, G. Yang, and L. F. Huntsinger, Eds., London, U.K.: Butterworth, 2022, pp. 17–81.

[2] K. Molugaram and G. S. Rao, "Preliminaries," in *Statistical Techniques for Transportation Engineering*, K. Molugaram and G. S. Rao, Eds., London, U.K.: Butterworth, 2017, ch. 2, pp. 25–91.

[3] Y. G. Bihanda, C. Fatichah, and A. Yuniarti, "Comparative analysis of ConvNext and mobilenet on traffic vehicle detection," in *Proc. IEEE 8th Int. Conf. Softw. Eng. Comput. Syst. (ICSECS)*, Aug. 2023, pp. 101–105.

[4] A. Stanley and R. Munir, "Vehicle traffic volume counting in CCTV video with YOLO algorithm and road HSV color model-based segmentation system development," in *Proc. 15th Int. Conf. Telecommun. Syst., Services, Appl. (TSSA)*, Nov. 2021, pp. 1–5.

[5] H. Song, H. Liang, H. Li, Z. Dai, and X. Yun, "Vision-based vehicle detection and counting system using deep learning in highway scenes," *Eur. Transp. Res. Rev.*, vol. 11, no. 1, p. 51, Dec. 2019.

[6] C. Liu, D. Q. Huynh, Y. Sun, M. Reynolds, and S. Atkinson, "A vision-based pipeline for vehicle counting, speed estimation, and classification," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 12, pp. 7547–7560, Dec. 2021.

[7] T. Yang, R. Liang, and L. Huang, "Vehicle counting method based on attention mechanism SSD and state detection," *Vis. Comput.*, vol. 38, no. 8, pp. 2871–2881, Aug. 2022.

[8] J. Azimjonov and A. Özmen, "Vision-based vehicle tracking on highway traffic using bounding-box features to extract statistical information," *Comput. Electr. Eng.*, vol. 97, Jan. 2022, Art. no. 107560.

[9] K. N. Bui, H. Yi, and J. Cho, "A vehicle counts by class framework using distinguished regions tracking at multiple intersections," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 2466–2474.

[10] C.-J. Lin, S.-Y. Jeng, and H.-W. Lioa, "A real-time vehicle counting, speed estimation, and classification system based on virtual detection zone and YOLO," *Math. Problems Eng.*, vol. 2021, Nov. 2021, Art. no. 1577614.

[11] B. Neupane, T. Horanont, and J. Aryal, "Real-time vehicle classification and tracking using a transfer learning-improved deep learning network," *Sensors*, vol. 22, no. 10, p. 3813, May 2022.

[12] T. Zhonglin, M. N. A. Wahab, M. F. Akbar, A. S. A. Mohamed, M. H. M. Noor, and B. A. Rosdi, "SFFSORT multi-object tracking by shallow feature fusion for vehicle counting," *IEEE Access*, vol. 11, pp. 76827–76841, 2023.

[13] W.-C. Chen, M.-J. Deng, P.-Y. Liu, C.-C. Lai, and Y.-H. Lin, "A framework for real-time vehicle counting and velocity estimation using deep learning," *Sustain. Comput., Informat. Syst.*, vol. 40, Dec. 2023, Art. no. 100927.

[14] M. I. Rifai, R. W. Sudibyo, A. D. S. Kurniawan, M. Z. S. Hadi, H. Mahmudah, and N. Sa'Adah, "Automatic vehicle classification and counting system using inception model," in *Proc. 14th Int. Conf. Inf. Technol. Electr. Eng. (ICITEE)*, Oct. 2022, pp. 275–279.

[15] R. W. Sudibyo, H. Mahmudah, M. Z. S. Hadi, and N. Sa'adah, "Edge computing-based automated vehicle classification system using the MobileNet V2 model," *Indonesian J. Comput. Sci.*, vol. 11, no. 3, pp. 817–828, Dec. 2022.

[16] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen, "DETRs beat YOLOs on real-time object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2024, pp. 16965–16974.

[17] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "ByteTrack: Multi-object tracking by associating every detection box," in *Proc. 17th Eur. Conf. Comput. Vis. (ECCV)*, Tel Aviv, Israel. Berlin, Germany: Springer-Verlag, Oct. 2022, pp. 1–21.

[18] A. Naufal, C. Fatichah, and N. Suciati, "Preprocessed mask RCNN for parking space detection in smart parking systems," *Int. J. Intell. Eng. Syst.*, vol. 13, no. 6, pp. 255–265, Dec. 2020.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988.

[20] M. F. Asyari, C. Fatichah, and N. Suciati, "Parking space availability detections from two overlapping cameras using YOLOv5 and image stitching methods," *Int. J. Intell. Eng. Syst.*, vol. 16, no. 4, 2023.

[21] G. Jocher et al., "Ultralytics/YOLOv5: v3.1—Bug fixes and performance improvements," Ultralytics, USA, Rep. v7.0, 2020.

[22] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[23] J.-Q. Luo, H.-S. Fang, F.-M. Shao, Y. Zhong, and X. Hua, "Multi-scale traffic vehicle detection based on faster R–CNN with NAS optimization and feature enrichment," *Defence Technol.*, vol. 17, no. 4, pp. 1542–1554, Aug. 2021.

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Berg, and A. C. Fu, "SSD: Single shot multibox detector," in *Computer Vision—ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham, Switzerland: Springer, 2016, pp. 21–37.

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. 16th Eur. Conf. Comput. Vis.*, Glasgow, U.K. Berlin, Germany: Springer-Verlag, Aug. 2020, pp. 213–229.

[26] A. Zarindast and A. Sharma, "Opportunities and challenges in vehicle tracking: A computer vision-based vehicle tracking system," *Data Sci. Transp.*, vol. 5, no. 1, p. 3, Feb. 2023.

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

[28] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–16.

[29] G. Zhang, Z. Luo, Y. Yu, K. Cui, and S. Lu, "Accelerating DETR convergence via semantic-aligned matching," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 939–948.

[30] P. Deshmukh, G. S. R. Satyanarayana, S. Majhi, U. K. Sahoo, and S. K. Das, "Swin transformer based vehicle detection in undisciplined traffic environment," *Expert Syst. Appl.*, vol. 213, Mar. 2023, Art. no. 118992.

[31] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9992–10002.

[32] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "YOLOv9: Learning what you want to learn using programmable gradient information," 2024, *arXiv:2402.13616*.
[33] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision—ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham, Switzerland: Springer, 2014, pp. 740–755.
[34] 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.
[35] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.
[36] M. H. Ashraf, F. Jabeen, H. Alghamdi, M. S. Zia, and M. S. Almutairi, "HVD-Net: A hybrid vehicle detection network for vision-based vehicle tracking and speed estimation," *J. King Saud Univ., Comput. Inf. Sci.*, vol. 35, no. 8, Sep. 2023, Art. no. 101657.
[37] L. Lin, H. He, Z. Xu, and D. Wu, "Realtime vehicle tracking method based on YOLOv5 + DeepSORT," *Comput. Intell. Neurosci.*, vol. 2023, no. 1, Jun. 2023, Art. no. 7974201.
[38] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2564–2571.
[39] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.
[40] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.
[41] J. Jiao and H. Wang, "Traffic behavior recognition from traffic videos under occlusion condition: A Kalman filter approach," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2676, no. 7, pp. 55–65, Jul. 2022.
[42] C.-Y. Wang, A. Bochkovskiy, and H.-Y.-M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 7464–7475.
[43] M. Adl, R. Ahmed, C. Vidal, and A. Emadi, "Enhanced vehicle movement counting at intersections via a self-learning fisheye camera system," *IEEE Access*, vol. 12, pp. 77947–77958, 2024.
[44] X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, and S. Wen, "PP-YOLO: An effective and efficient implementation of object detector," 2020, *arXiv:2007.12099*.
[45] J. Lu, M. Xia, X. Gao, X. Yang, T. Tao, H. Meng, W. Zhang, X. Tan, Y. Shi, G. Li, and E. Ding, "Robust and online vehicle counting at crowded intersections," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 3997–4003.
[46] W. Karoon, P. Chuasuai, P. Thipprasert, N. Khongchu, P. Kunakornjittirak, and T. Siriborvornratanakul, "Adaptive traffic light control using vision-based deep learning for vehicle density estimation," in *Proc. 6th Asia–Pacific Inf. Technol. Conf.* New York, NY, USA: Association for Computing Machinery, Jan. 2024, pp. 37–42.
[47] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, Y. Li, B. Zhang, Y. Liang, L. Zhou, X. Xu, X. Chu, X. Wei, and X. Wei, "YOLOv6: A single-stage object detection framework for industrial applications," 2022, *arXiv:2209.02976*.
[48] Y. Said, Y. Alassaf, Y. Alsariera, R. Ghodhbani, T. Saidani, O. B. Rhaiem, and M. K. Makhdoum, "Traffic flow management by detecting and estimating vehicles density based on object detection model," *Neural Comput. Appl.*, vol. 36, no. 19, pp. 11495–11505, Jul. 2024.
[49] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.
[51] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLO," Ultralytics, USA, Rep. 8.0.0, Jan. 2023.
[52] D. Darmadi, P. Pratikso, and M. Rahmat, "Traffic counting using YOLO version-8 (case study of Jakarta–Cikampek toll Road)," *ASTONJADRO*, vol. 13, no. 1, pp. 115–124, Jan. 2024.
[53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[54] L. V. Ma, M. I. Hussain, J. Park, J. Kim, and M. Jeon, "Adaptive confidence threshold for ByteTrack in multi-object tracking," in *Proc. 12th Int. Conf. Control, Autom. Inf. Sci. (ICCAIS)*, Nov. 2023, pp. 370–374.
[55] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: Exceeding YOLO series in 2021," 2021, *arXiv:2107.08430*.

YUSUF GLADIENSYAH BIHANDA received the bachelor's degree in informatics engineering from the University of Brawijaya, Malang, Indonesia, in 2022. He is currently pursuing the master's degree in informatics engineering with Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia.

From 2019 to 2022, he was a Research Assistant with the University of Brawijaya. He is currently a Research Assistant with Institut Teknologi Sepuluh Nopember. His research interests include multi-object detection, multi-object tracking, and integration software with artificial intelligence.

CHASTINE FATICHAH (Member, IEEE) received the Ph.D. degree from Tokyo Institute of Technology, Japan, in 2012. She is currently a Professor with Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia. She has published more than 110 computer science-related journal articles and conference papers. Her research interests include artificial intelligence, image processing, and data mining.

ANNY YUNIARTI (Member, IEEE) received the master's degree from The University of Western Australia, Australia, in 2008, and the Ph.D. degree from Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, in 2022. She is currently an Associate Professor with Institut Teknologi Sepuluh Nopember. She has published more than 70 computer science-related journal articles and conference papers. Her research interests include 3D reconstruction from 2D images, deep learning, and computer vision.