A Vehicle Counts by Class Framework using Distinguished Regions Tracking at Multiple Intersections

Khac-Hoai Nam Bui, Hongsuk Yi, and Jiho Cho

Korea Institute of Science and Technology Information, Korea
hoainam.bk2012@gmail.com, {hsyi, jhcho}@kisti.re.kr

Abstract

*Turning movement counting plays an important step for traffic analysis at complex areas (*e.g. *intersections). Specifically, accurate and detailed traffic flow information enables the traffic control system to be more efficient and valuable. Recently, with the successful development of Deep Learning for vehicle detection and tracking, the current research focuses on video-based traffic analysis which is regarded as an emergent approach to monitoring vehicle movements. In this study, we present a comprehensive vehicle counting framework by integrating state-of-the-art techniques of object detection and tracking such as Yolo and DeepSort. Furthermore, in order to improve the vehicle counting problem, we propose a distinguished region tracking approach for the vehicle trajectory monitoring, which is able to work well with various scenarios, especially in complex areas with complicated movements. Regarding the experiment, the proposed framework is evaluated on the CVPR AI City Challenge 2020 dataset. Accordingly, our method is able to achieve around 85% of the accuracy which places to the top 10 of the leaderboard in Track 1 of the Challenge.*

1. Introduction

Vehicle counting is an important technique to estimate the traffic density in a certain area. Specifically, traffic conditions are able to determine based on the counting results for providing smart traffic control systems [2]. Technically, there are different techniques to measure traffic conditions such as manual vehicle counting, inductive-loop traffic detectors, magnetic sensors, and video vehicle detection [10]. Among the aforementioned methods, video-based counting using computer vision techniques (*e.g.* object detection and tracking) has recently attracted more attention for the following reasons:

- The recent Deep Learning (DL) models have achieved great success in terms of detecting and tracking moving objects in video sequences.
- Many smart applications of video-based traffic monitoring can be applied, for instance, re-identification and anomaly detection.
- Evaluations are effortlessly executed to verify the performance of the proposed systems.

Figure 1 depicts the general pipeline of video-based vehicle counting system. Specifically, a set of Region of Interests (ROI) is defined and pre-processed for reducing noise and determining the considered areas. Sequentially, detection and tracking methods are executed to identify the vehicle information (*e.g.* location and type of vehicle) and track vehicles across subsequent frames. Finally, vehicles will be counted and recorded based on the virtual lines which are assigned in different directions of the considered scenario.

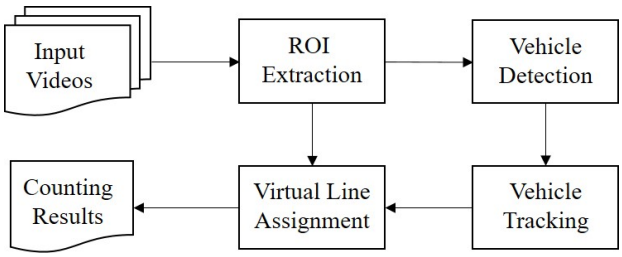

Figure 1. The pipeline of the video-based vehicle counting system. Accordingly, the tracking-by-detection paradigm is adopted for multiple vehicle tracking problems.

Recently, many studies have proposed the video-based vehicle counting framework following the aforementioned concept. However, video-based vehicle counting is still a very challenging issue due to the two main reasons: i) Vehicle tracking is a difficult task, especially for movement counting problem which requires a long-time tracking; ii) Scalability is a critical problem which requires the counting systems are able to work well with various situations of traffic flow across different types of scenarios/intersections.

In this study, we present a practice-based approach for vehicle counting in order to deal with the aforementioned problems in which we focus on the Multi-Class Multi-Movement Vehicle Counting (CMVC) problem at multiple intersections/scenarios. In particular, instead of focusing on long-time range tracking, we are able to divide the considered scenario into distinguished regions for vehicle monitoring. Consequently, the movement counting is calculated following the linking regions which are provided based on the geographical features of each scenario. For more detail, Figure 2 depicts the flowchart of our proposed framework. Specifically, regarding the multiple vehicle tracking,

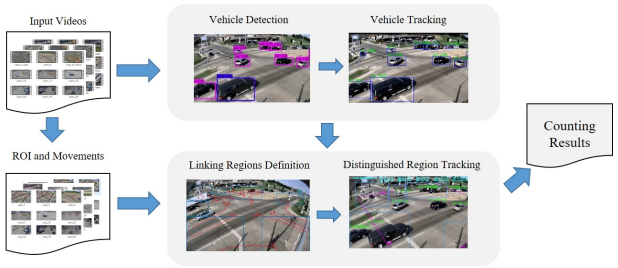

Figure 2. The main steps of the proposed framework.

we apply the state-of-the-art techniques of object detection and tracking such as Yolo [24] and DeepSort [27], respectively. Then, a distinguished regions tracking approach is executed to count the vehicles. Furthermore, in order to aim with the scope of the Challenge, instead of using external datasets, we extract appearance features of vehicles from the NVIDIA AI City Challenge dataset [25].

The rest of this paper is organized as follows: In Section 2, recent state-of-the-art methods for object detection and tracking processes are presented. Moreover, previous works on video-based vehicle counting systems are also reviewed. The methodology of the proposed framework for the CMVC problem is presented in Section 3. Section 4 shows the results of our proposed framework which is evaluated on the CVPR AI City Challenge 2020 (AIC 2020) dataset [1]. Discussions and future works are concluded in Section 5.

2. Literature Review

2.1. Object Detection and Tracking Methods

Recently, advanced methods of object detection and tracking enable many smart applications to improve people's life such as face recognition, self-driving cars, and emergency alert systems. Specifically, Convolutional Neural Network (CNN)-based methods has been observed widely successful for the object detection [14]. Technically, the recent object detection methods can be classified into two approaches which are *single-stage* and *two-stage* detectors. In particular, *two-stage* methods, which are the evolution of Regions with Convolutional Neural Networks (R-CNN) series (*e.g.* Fast R-CNN [11], R-FCN [6], and Mask R-CNN [12]), perform the detection process into two stages: i) generating a set of regions of interests by using regional proposal network; ii) then, optimizing the regressions process on the region candidates for the detection. On the other hand, *single-stage* (*e.g.* SSD [19], YOLO [23], and RetinaNet [18]) methods skip the region proposal stage and execute the detection directly over a dense sampling of possible locations. Consequently, it depends on the objective and target of the detection to adopt the appropriate method. Specifically, *two-stage* methods are able to achieve the higher accuracy but lower speed comparing with *single-stage* methods.

Regarding the object tracking process, SORT [29] is the well-known method which was ranked as the best open-source on the multiple object tracking (MOT) benchmark [20]. Specifically, the algorithm implements a visual MOT framework based on rudimentary data association and state estimation techniques. Sequentially, DeepSORT [28], an extension of SORT, has proposed to improve the performance of the tracking-by-detection paradigm by incorporating deep features for tracking detected objects. Regarding the feature extraction of vehicles, TC [26], an offline method, has recently emerged as a promising approach for tracking vehicles across multiple videos/cameras by incorporating various semantic features (*e.g.* trajectory smoothness, velocity change, and temporal information) for the data association. Specifically, this method is able to achieve high performance with both single and multiple camera tracking, which was the first rank of vehicle re-identification in the AIC 2018 [21].

2.2. Video-based Vehicle Counting Systems

Object counting system using data from monocular cameras becomes a promising application to provide social security, crowd, and disaster prevention [16]. Specifically, the object can be counted when it passes a certain area (*e.g.* ROI). Accordingly, this concept has been widely applied for people counting, which detects and tracks each individual to count the number of people passing considered areas [17, 15]. Recently, video-based vehicle counting has emerged as a new challenge since the difficulties of tracking and re-identification problems of vehicles [22]. Dai *et al.* [7] proposed a vehicle counting framework including three-component processes which are object detection, tracking, and trajectory processing to obtain the information of traffic conditions. Liu *et al.* [9] adopt the virtual loop and detection line for multiple movement counting. However, the problem of identity switches because of the long-range vehicle tracking and the scalability for adopting the aforementioned methods in different scenarios are still open research issues.

3. Methodology

3.1. Multiple Vehicle Tracking Method

The standard approach for the MOT problem follows the tracking-by-detection paradigm in which given a set of detection results, the tracking process is executed by associating the corresponding objects and assigning them in the same ID across the subsequent frames [4, 5]. Particularly, tracking vehicles across entire the video is more difficult since the high similarity of the vehicle's appearance features and the frequent occlusion of moving vehicles. Therefore, vehicle detection and tracking recently become the main challenge in this research field. In this study, we adopt two well-known methods which are Yolo and DeepSORT for detecting and tracking vehicles across a certain video. More detail of our proposed framework is explained in the following sections.

Vehicle Detection. This process detects and extracts the information of vehicles in each frame. The output is the set of detected results which are formatted as follows:

$$< x, y, w, h, class, confidence > \quad (1)$$

where (x, y, w, h) represents the location of the detected vehicle (i.e, bounding box). Moreover, *class* and *confidence* are the type and score of the detected object, respectively. In the proposed framework, we adopt Yolo model, which follows the *single-stage* approach, to improve the faster speed of the detection process. Moreover, the lasted version of the method (Yolov3 [24]) has been released which is able to achieve high accuracy of the detection by conducting 53 convolutional layers. Furthermore, in order to improve the effectiveness of this process, we only extract the objects which belong to the vehicle's types. Specifically, the bounding boxes are extracted if they follow two conditions:

- The *class* of the detected object is the vehicle's type (*e.g*. Car, Truck, and Bus). Specifically, two-wheeled vehicles such as Bicycle, Motorized bicycle, and Motorbike are not taken into account in this study.
- The *confidence* of the detected object is greater than a given threshold. Specifically, the appropriate threshold is defined based on the observation during the experiment.

Vehicle Tracking. Regarding the tracking process, since the objective focuses on tracking vehicles in independent scenarios/videos, we adopt DeepSORT, an online method, for the multiple vehicle tracking process which has been proved the effectiveness to track multiple objects in a certain video [28]. Specifically, the output of this process is defined as follows:

$$< x, y, w, h, \dot{x}, \dot{y}, \dot{w}, \dot{h} > \quad (2)$$

where $\dot{x}, \dot{y}, \dot{w}, \dot{h}$ are the parameters of the new bounding box, which are predicted and updated by using Kalman Filter algorithm, an essential component of DeepSORT.

Furthermore, regarding the deep association metric model, in order to cover with the scope of the challenge, instead of using external data, we take 36,935 images from 333 vehicle identities which are provided by AIC 2020 [1] for exporting the appearance descriptor of vehicles. Figure 3 shows the results of the classification accuracy for extracting the appearance descriptor, by using the deep cosine metric learning [27]. Specifically, we stop the training at 200,000 iterations which can provide high performance.

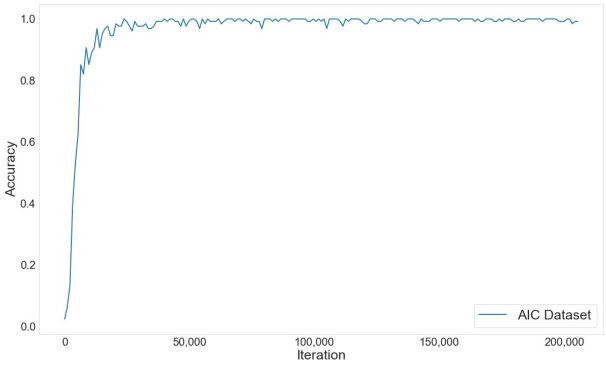

Figure 3. The classification accuracy of the trained model on AIC dataset using deep cosine metric learning.

3.2. Distinguished Regions-based Vehicle Counting

In order to monitor and count vehicle movements, we have to detect and track vehicles across the entire video. Particularly, long-term tracking is the cause of identity switches problem [8]. Therefore, in this study, we take an investigation on exploiting the region of interest for vehicle tracking in terms of reducing the range of vehicle monitoring to improve the performance of the counting problem. Specifically, for each input video, a set of distinguished regions are defined based on the geographical features and the number of movements of that scenario. For instance, Figure 4 depicts an example of the set of regions to monitor vehicle movements in a certain scenario. Conceptually, this approach is a well-known method and has been applied for various applications of video-based moving object tracking [13, 3]. However, in this study, we exploit in more detail how this method can affect the performance and the interaction between the effectiveness and efficiency of the counting problem in order to enable the scalability problem of this method to work well with various scenarios.

Monitoring Vehicle Movements using Linking Regions. Normally, the vehicles will be counted using virtual line assignments corresponding with the number of movements/directions in each scenario. Accordingly, when the vehicle passes the line, the predefined movement will be

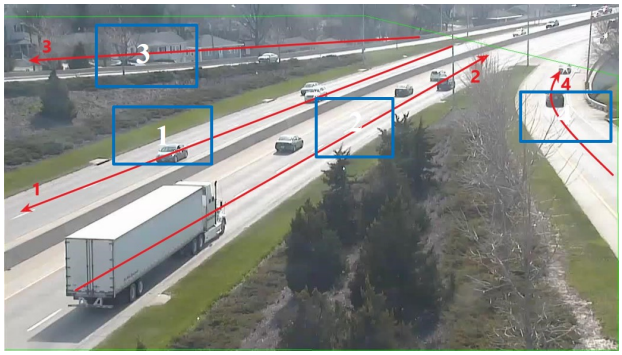

Figure 4. An example of distinguished region tracking. There are 4 regions to monitoring vehicles corresponding with 4 vehicle movements. The red lines are the predefined movements of the scenario.

counted following the corresponding region tracking of the vehicle. However, in this framework, we adopt the counting process based on the linking information between regions (virtual loops) instead of using virtual lines in order to improve the performance of the counting. For instance, Figure 5 demonstrates the vehicle monitoring using distinguished regions of the scenario in Figure 4.

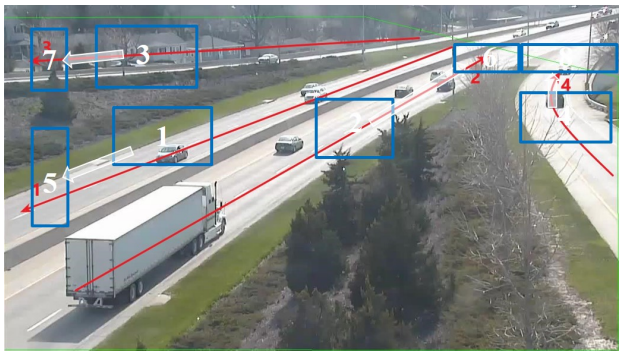

Figure 5. An example of movement counting based on the linking between regions. Four movements in this scenario correspond with four pair regions [1,5], [2,6], [3,7], and [4,8].

Specifically, using the linking regions for vehicle counting is able to achieve better results comparing with virtual line assignment because of two reasons: i) Reducing the identity switches problem by tracking vehicles in the short-term range; ii) Avoiding the occlusion problem cause of the wrong counting in case of multiple vehicles passing at the same time. For more detail, Figure 6 shows the results of vehicle counting performance based on the virtual line and distinguished regions at the same intersection.

Formulation of the CMVC Problem. The output of the CMVC problem at multiple intersections is the list of results which are formatted as follows:

$$< Vdo_{id}, Fra_{id}, Mov_{id}, Cla_{id} > \quad (3)$$

where Vdo_{id} is the video numeric identifier of the track scenarios which is sorted in alphanumeric order. Fra_{id} and Mov_{id} represent the frame count and movement numeric identifier of the Vdo_{id} of the detected vehicle, respectively. Cla_{id} denotes the type of the vehicle.

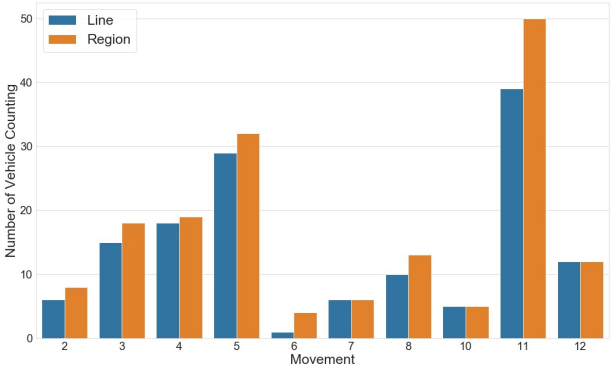

Figure 6. The comparison between virtual line and distinguished regions for the multi-movement counting in which the considered scenario has 12 movements and the number of frames of the input video is around 2000. As shown in the Figure, by using the distinguished regions, we are able to improve from 5% to 20% of the counting performance in each movement.

Particularly, Mov_{id} of a vehicle v is calculated based on the tracking information of linking regions corresponding with the predefined movements of each scenario. Specifically, the vehicle v can be tracked in the region r in case of the following condition:

$$r = \{v \in \mathcal{V} \mid \forall v \in 1...m : p_t^r \leq p_c^v \leq p_b^r\} \quad (4)$$

where p_c^v represents the centroid point of vehicle v, p_t^r and p_b^r are the top left corner and bottom right corner points of region r, respectively. For more detail, Algorithm 1 demonstrates the counting method of the proposed framework.

Conflict Regions Problem. In complex scenarios, vehicles might move into multiple regions which is the cause of the multi-counting problem. Consequently, a vehicle can be recorded in more than one movement. For instance, in the scenario of Figure 5, we are able to determine distinguished regions in which vehicles move independently on different movements, however, in the scenario of Figure 7, the regions are conflicted in which vehicles need to across multiple regions in some specific directions. Therefore, a set of tracking list $\mathbb{T}$ is defined to deal with this issue. Specifically, vehicles are only recorded in the first region that they move in (Line 10-13, Algorithm 1) in which the demonstration of this process is illustrated in Figure 8. Consequently, the Equation 4 can be reformulated as follows:

$$r = \{v \in \mathcal{V}, v \notin \mathbb{T} \mid \forall v \in 1...m : p_t^r \leq p_c^v \leq p_b^r\} \quad (5)$$

Furthermore, another issue of the conflict regions' problem is the wrong counting of the movement for vehicles

Algorithm 1: Multi-Class Multi-Movement Vehicle Counting

Data: Video Vdo^i; Set $\mathcal{R}^i$ of distinguised regions; Set $\mathcal{M}^i$ of movements
Result: Number of vehicles passing in each movement

Frame = 1
while $True$ **do**
  Set T = Set of Bounding Boxes
  for t_{cur}, $t_{pre} \in T$ **do**
    $(x, y, w, h, class)$ = $(t_{cur}[0])$, $t_{cur}[1]$,$t_{cur}[2]$, $t_{cur}[3]$, $t_{cur}[4]$)
    $(\dot{x}, \dot{y}, \dot{w}, \dot{h})$ = $(t_{pre}[0])$, $t_{pre}[1]$,$t_{pre}[2]$, $t_{pre}[3]$)
    $p^v_{cur} = (x + \frac{w-x}{2}; y + \frac{h-y}{2})$
    $p^v_{pre} = (\dot{x} + \frac{\dot{w}-\dot{x}}{2}; \dot{y} + \frac{\dot{h}-\dot{y}}{2})$
    $Cla^v = t_{cur}[4]$
    while $r \in \mathcal{R}$ **do**
      if *InZone*$(p^v_{cur}, p^r_0, p^r_1)$ *and* $v \notin \mathbb{T}$ **then**
        $\mathbb{T} \leftarrow v, r$
      end
    end
    while $r \in \mathcal{R}$ **do**
      if *InZone*$(p^v_{cur}, p^r_0, p^r_1)$ **then**
        if $v \in \mathbb{T}$ **then**
          Mov^v = CheckMovement(r^v, r)
          $result.write(Fra, Mov^v, Cla^v)$
        end
      end
    end
  end
  Frame += 1
end
Function InZone(A,B,C)
return ($A[x] > B[x]$ and $A[x] < C[x]$ and $A[y] > B[y]$ and $A[y] < C[y]$)
Function CheckMovement(r_s, r_d)
while $mo \in \mathcal{M}$ **do**
  if $mo[1] = r_s$ *and* $mo[2] = r_d$ **then**
    return $mo[0]$
  end
end

that are not be tracked in the original region. For example, in the scenario of Figure 7, the vehicles move in the mov_5 might be counted by mov_1 in case they can not be detected in the original region r_4. In this regard, in order to improve the performance and reduce the wrong movement of the counting, additional regions are set between conflict regions. Consequently, several movements will have more than one pair of regions for vehicle monitoring. Specifically, the demonstration of this process is shown in Figure 9.

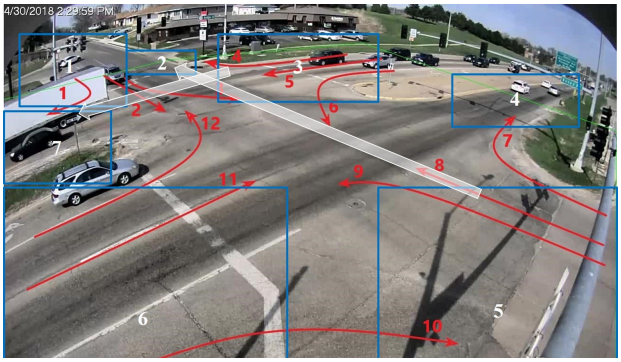

Figure 7. An example of vehicles across multiple regions. For instance, vehicles moving in mov_5 might across r_3 and r_1 (white arrow), therefore, the vehicles move in mov_5 will be counted at r_7 with two movements which are mov_5 ([3,7]) and mov_1 ([1,7]), similarity occurs with mov_8 ([5,2]) and mov_4 ([3,2]).

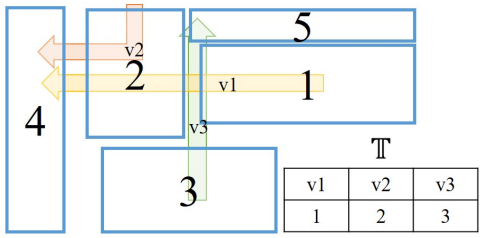

Figure 8. The description of vehicles across multiple regions using tracking list $\mathbb{T}$. Accordingly, vehicles are only tracked in the first region that the vehicle moves in.

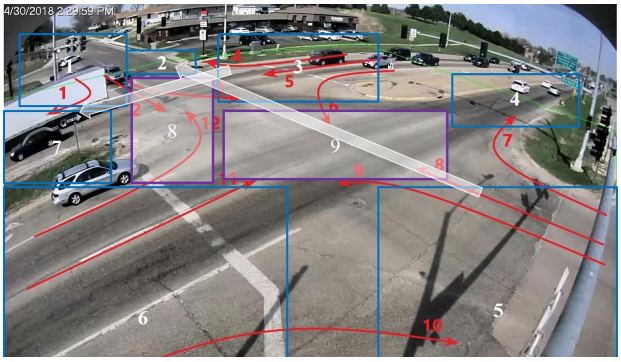

Figure 9. An example of additional regions for improving the accuracy of the counting. Additional regions (purple rectangles) are set among conflicted regions in order to obtain the vehicles that are not detected in the original regions. Consequently, several movements have more than one pair of regions, for example, mov_5 includes two pairs of linking regions which are [3,7] and [8,7].

Optimizing Number of Regions. The disadvantage of counting vehicles by using distinguished regions comparing

with the virtual line is time-consuming in which the number of regions is the cause of increasing the computational time. Figure 10 demonstrates the result of our experiment at a certain scenario in order to investigate the effect of the number of distinguished regions with the computational time. In

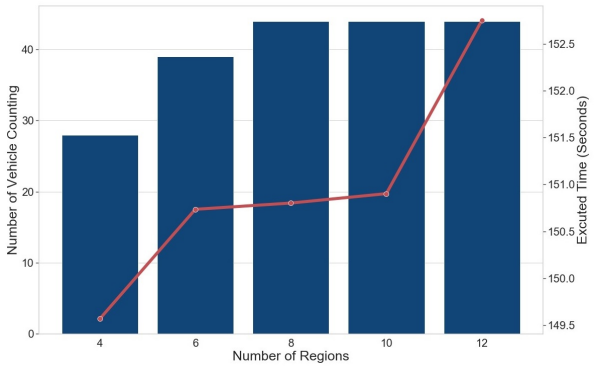

Figure 10. The effect of the number of distinguished regions on the computational time of the counting process at a certain scenario.

this regard, an effective process for determining the number of regions in each scenario needs to be taken into account. For instance, considering the scenario in Figure 10, by setting 8 regions is able to give the best performance.

4. Experiment

Datasets. The dataset contains 31 video clips capturing from 20 unique camera views which are provided by AIC 2020 [1]. Some scenarios provide multiple video clips to cover different conditions (*e.g*. lighting and weather). The resolution and frame rate are at least 720p and 10 FPS. Furthermore, each scenario includes a detailed instruction document which describes the ROI and movements of interest (MOI). Consequently, our first process is to manually define the set of distinguished regions corresponding with the MOI in each scenario that integrates into the input data.

Experiment Setup. Regarding the experiment, some important parameters are listed in Table 1. Accordingly,

Parameter	Values
Input Video	31
Total Time	17913 seconds ($\approx$ 5 hours)
Number of Class	02 (Car and Truck)
Score Detection Threshold	0.6
IoU Detection Threshold	0.5
Training Image Size	128 x 64
Base Factor Score (σ)	0.824664

Table 1. Parameter setting of the proposed framework.

two classes of vehicles are car and $truck$ which are required for detection and tracking. Specifically, class car includes four-wheel vehicles (*e.g*. sedan car, SUV, van, bus, and small trucks) and $truck$ refers to freight trucks (*e.g*. garbage truck, tractor-trailer, and 18-wheeler vehicle). Furthermore, object detection and IoU thresholds are set 0.6 and 0.5, respectively. For training appearance features of vehicles, all input images are reshaped with the same size (128 x 64). Moreover, the total score of the challenge includes efficiency (computational time) which requires the information of the execution system. Consequently, the *Efficiency Base Factor Score* of our executed system is computed, by using *pyperformance* package [1]. Specifically, the script for extracting *Efficiency Base Factor Score* is provided by AIC 2020 [1].

Experiment Results. Figure 11 shows the screenshots of the proposed framework for the CMVC problem with different intersections/scenarios. Particularly, our proposed framework is able to work well with various scenarios including traffic flow density, multi-directions, video resolution, lighting and weather conditions. Accordingly, our method belongs to the top 10 of the leaderboard in the challenge which is shown in Table 2.

Rank	Team ID	Score
1	99	0.9389
2	110	0.9346
3	92	0.9292
4	26	0.8936
5	22	0.8852
6	74	0.8829
7	**6 (Ours)**	**0.8540**
8	119	0.8254
9	80	0.8064
10	65	0.7933

Table 2. The resulting scores of top 10 on Track 1 Leaderboard.

Specifically, given by the AIC 2020 [1], the total score is computed based on the combination between *efficiency* ($S_{efficiency}$) and *effectiveness* ($S_{effectiveness}$) scores, which is calculated as follows:

$$Score = \alpha S_{efficiency} + \beta S_{effectiveness} \quad (6)$$

where α = 0.3 and β = 0.7, respectively. The *Efficiency Score* is calculated based on the execution time as follows:

$$S_{efficiency} = \max(0, 1 - \frac{t^{ex} \times \sigma}{5 \times t^{vdo}}) \quad (7)$$

where t^{ex}, t^{vdo}, and σ are execution time, total time of videos, and *Base Factor Score* of the executed system, respectively. On the other hand, *Effectiveness Score* is computed as a weighted average of normalized weighted

[1]https://pyperformance.readthedocs.io

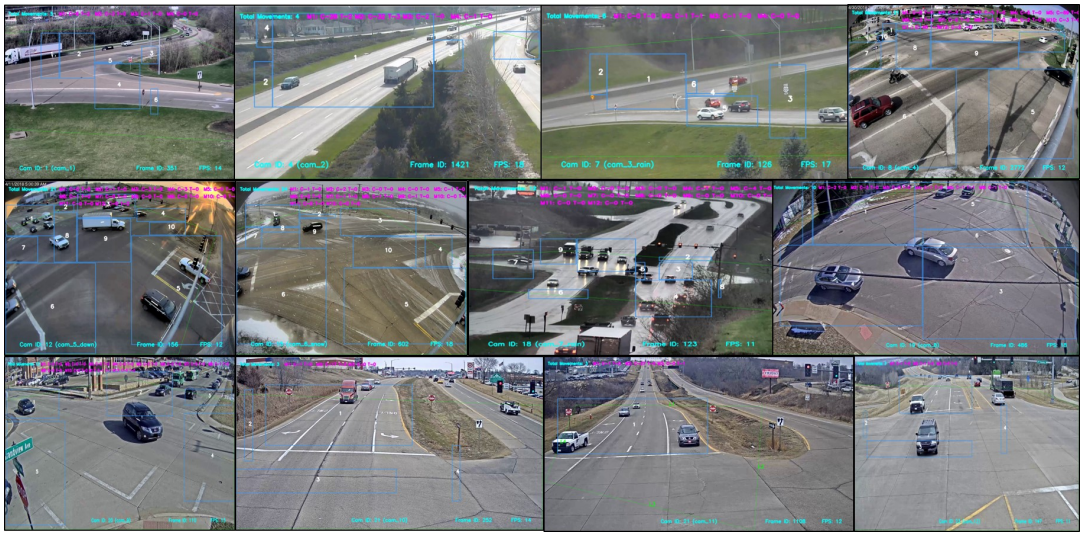

Figure 11. Screenshots of the proposed framework for the CMVC problem with different scenarios.

root mean square error scores ($nwRMSE$) across all input videos, movements and vehicle classes. Particularly, $nwRMSE$ score is the weighted $RMSE$ ($wRMSE$) between the predicted ($\mathcal{C}^{count}$) and true cumulative ($\mathcal{C}^{true}$) vehicle counts, which is formulated as follows:

$$nwRMSE = \begin{cases} 0, & \text{if } wRMSE > \mathcal{C}^{true} \\ 1 - \dfrac{wRMSE}{\mathcal{C}^{count}}, & \text{otherwise.} \end{cases} \tag{8}$$

where $wRMSE$ is computed following k segments of each video in order to reduce the labeling discrepancies problem which is formulated as follows:

$$wRMSE = \sqrt{\sum_{i=1}^{k} w_i (\mathcal{C}_i^{count} - \mathcal{C}_i^{true})^2} \tag{9}$$

where

$$w_i = \frac{2i}{k(k+1)} \tag{10}$$

As shown in the result table, our proposed method is able to achieve around 0.85/1 of the overall score. Specifically, Table 2 demonstrates in more detail the evaluation of our method. Accordingly, the accuracy of our method is able to achieve around 85% across multiple scenarios.

mwRMSE	$S_{efficiency}$	$S_{effectiveness}$	Score
7.0731	0.8538	0.8543	0.8540

Table 3. The detailed evaluation of the proposed framework.

5. Conclusion and Future Work

In this paper, we propose a comprehensive framework for multi-class multi-movement vehicle counting across multiple intersections/scenarios. Specifically, the long-time range tracking can be reduced in order to improve the performance of the counting system by using distinguished regions for vehicle tracking. Consequently, the framework is able to work well with various scenarios with different numbers of movements and lighting conditions. According to the experiment, our method is able to achieve promising results compared with the competing methods of the AIC 2020 Track 1. From our point of view, there are several issues that can improve the performance of the proposed framework such as i) training appropriate datasets for the detection process which is able to detect several specific objects (*e.g.* tractor-trailer and 18-wheeler vehicle); ii) proposing an optimization approach for determining the locations of distinguished regions to improve the tracking problem; and iii) determining the distinguished region is still a manual process, an automated process is able to significantly improve the scalability of the proposed method. The aforementioned problems are interesting issues that we take into account as the future work regarding this study.

Acknowledgment

This work was partly supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2018-0-00494, Development of deep learning-based urban traffic congestion prediction and signal control solution system) and Korea Institute of Science and Technology Information (KISTI) grant funded by the Korea government (MSIT) (K-19-L02-C07-S01).

References

[1] AI City Challeenge 2020 Official Website (AIC2020). *https://www.aicitychallenge.org/*. Accessed: March 09th, 2020.

[2] Khac Hoai Nam Bui, O-Joun Lee, Jason J. Jung, and David Camacho. Dynamic traffic light control system based on process synchronization among connected vehicles. In *Proceedings of the International Symposium on Ambient Intelligence (ISAmI)*, pages 77–85, 2016.

[3] Khac Hoai Nam Bui, Hongsuk Yi, Heejin Jung, and Jiho Cho. A multi-class multi-movement vehicle counting framework for traffic analysis in complex areas using cctv systems. *Energies*, 13(8):2036, 2020.

[4] Visesh Chari, Simon Lacoste-Julien, Ivan Laptev, and Josef Sivic. On pairwise costs for network flow multi-object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5537–5545, 2015.

[5] Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, Luigi Troiano, Roberto Tagliaferri, and Francisco Herrera. Deep learning in video multi-object tracking: A survey. *Neurocomputing*, 381:61–88, 2020.

[6] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: object detection via region-based fully convolutional networks. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, pages 379–387, 2016.

[7] Zhe Dai, Huansheng Song, Xuan Wang, Yong Fang, Xu Yun, Zhaoyang Zhang, and Huaiyu Li. Video-based vehicle counting framework. *IEEE Access*, 7:64460–64470, 2019.

[8] Patrick Dendorfer, Seyed Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian D. Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. CVPR19 tracking and detection challenge: How crowded can it get? *CoRR*, abs/1906.04567, 2019.

[9] Liu Fei, Zeng Zhiyuan, and Jiang Rong. A video-based real-time adaptive vehicle-counting system for urban roads. *Plos One*, 12(11):e0186098, 2017.

[10] Mohammad Ghanim and Khaled Shaaban. Estimating turning movements at signalized intersections using artificial neural networks. *IEEE Transaction on Intelligent Transportation Systems*, 20(5):1828–1836, 2019.

[11] Ross B. Girshick. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.

[13] Hung-Min Hsu, Tsung-Wei Huang, Gaoang Wang, Jiarui Cai, Zhichao Lei, and Jenq-Neng Hwang. Multi-camera tracking of vehicles based on deep features re-id and trajectory-based camera link models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 416–424, 2019.

[14] Licheng Jiao, Fan Zhang, Fang Liu, Shuyuan Yang, Lingling Li, Zhixi Feng, and Rong Qu. A survey of deep learning-based object detection. *IEEE Access*, 7:128837–128868, 2019.

[15] S. Kanagamalliga and S. Vasuki. An efficient algorithm for tracking and counting pedestrians based on feature points in video surveillance applications. *Journal of Intelligent and Fuzzy Systems*, 36(1):67–78, 2019.

[16] Di Kang, Zheng Ma, and Antoni B. Chan. Beyond counting: Comparisons of density maps for crowd analysis tasks - counting, detection, and tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(5):1408–1422, 2019.

[17] Mehmet Kemal Kocamaz, Jian Gong, and Bernardo Rodrigues Pires. Vision-based counting of pedestrians and cyclists. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–8, 2016.

[18] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017.

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 21–37, 2016.

[20] Anton Milan, Laura Leal-Taixé, Ian D. Reid, Stefan Roth, and Konrad Schindler. MOT16: A benchmark for multi-object tracking. *CoRR*, abs/1603.00831, 2016.

[21] Milind Naphade, Ming-Ching Chang, Anuj Sharma, David C. Anastasiu, Vamsi Jagarlamudi, Pranamesh Chakraborty, Tingting Huang, Shuo Wang, Ming-Yu Liu, Rama Chellappa, Jenq-Neng Hwang, and Siwei Lyu. The 2018 NVIDIA AI city challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 53–60, 2018.

[22] Milind Naphade, Zheng Tang, Ming-Ching Chang, David C. Anastasiu, Anuj Sharma, Rama Chellappa, Shuo Wang, Pranamesh Chakraborty, Tingting Huang, Jenq-Neng Hwang, and Siwei Lyu. The 2019 AI city challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 452–460, 2019.

[23] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[24] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018.

[25] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David C. Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 8797–8806, 2019.

[26] Zheng Tang, Gaoang Wang, Hao Xiao, Aotian Zheng, and Jenq-Neng Hwang. Single-camera and inter-camera vehicle

tracking and 3d speed estimation based on fusion of visual and semantic features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 108–115, 2018.

[27] Nicolai Wojke and Alex Bewley. Deep cosine metric learning for person re-identification. In *Proceeding of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, page 7, 2018.

[28] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *Proceeding of the International Conference on Image Processing (ICIP)*, pages 3645–3649, 2017.

[29] Liang Zheng, Zhi Bie, Yifan Sun, Jingdong Wang, Chi Su, Shengjin Wang, and Qi Tian. MARS: A video benchmark for large-scale person re-identification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 868–884, 2016.