Robust and Online Vehicle Counting at Crowded Intersections

Jincheng Lu Meng Xia Xu Gao Xipeng Yang Tianran Tao Hao Meng
Wei Zhang Xiao Tan Yifeng Shi Guanbin Li Errui Ding
{lujincheng01, gaoxu03, yangxipeng01, taotianran, menghao05, zhangwei99, tanxiao01, shiyifeng, dingerrui}@baidu.com
xiam9@mail2.sysu.edu.cn, liguanbi@mail.sysu.edu.cn

Abstract

In this paper, we propose an online movement-specific vehicle counting system to realize robust traffic flow analysis at crowded intersections. Our proposed framework adopts PP-YOLO as the vehicle detector and adapts the Deep-Sort algorithm to perform multi-object tracking. In order to realize online and robust vehicle counting, we further adopt a shape-based movement assignment strategy to differentiate movements and carefully designed spatial constraints to effectively reduce false-positive counts. Our proposed framework achieves the overall S1-score of 0.9467, ranking the first in the AICITY2021-track1 challenge.

1. Introduction

It is essential to count vehicles in the traffic scene, which is expected to mitigate traffic congestion and elevate the efficiency of the traffic light. Online vehicle counting by movements of interest (MOI) is to find the number of vehicles that are corresponding to MOI in a period of time in an online pattern. Nevertheless, accurate vehicle counting is a challenging task at crowded intersections, due to the difficulties such as the occlusions between different vehicles and poor weather conditions. Besides, the arrival time when vehicles move out of the region of interest (ROI) should also be calculated.

There are two solutions to the traditional vehicle counting problem. The first one is frame-wise vehicle counting [9, 1, 28, 31], which only counts vehicles in a single frame rather than consecutive frames. Two strategies are used, including the density-aware strategy and detection-based strategy. Density-aware strategy [3, 10, 2] has been used, which uses density estimation algorithms to regress the number of vehicles. Besides, based on the recent progress of deep learning methods for object detection [6, 27], more studies turn to the detection-based strategy [8, 26, 9, 1, 28], which is to detect vehicles then count the detected vehicles afterwards. Nevertheless, due to the mutual occlusions between vehicles and the occlusions caused by roadside trees, both density-aware and detection-based approaches usually miss the occluded vehicles. The second one is instance-wise vehicle counting, which counts vehicles in consecutive frames. Hence, those vehicles that are not detected can be counted by the usage of consecutive frame knowledge. To be specific, these methods [7, 17] mainly follow the detection-tracking-counting (DTC) framework, which performs multiple object tracking based on the detection results, then counts the vehicles according to the tracking results afterwards.

Different from the above mentioned traditional vehicle counting problems, this paper tackles the task of online movement-specific vehicle counting. Specifically, online movement-specific vehicle counting requires to not only count the total vehicle number for each MOI, but also record the timestamp when each vehicle moves out of the ROI and stream out the counting result in a limited time. Our proposed approach mainly follows the DTC pipeline, in which we choose PP-YOLO [20] and DeepSORT [30] as the baseline methods for vehicle detection and multi-object tracking, respectively. Though been carefully fine-tuned, this tracking-detection pipeline still performs poorly: the partially occluded vehicles are usually missed by the detector in crowded traffics, which severely increases the id-switches in the tracking results. To remedy the defect of the detector, we propose a detection augmentation method, which aims to generate additional detections with high confidence for missing/occluded objects to prevent identity switches. Specifically, we propose two augment detections strategies, including the detection re-match strategy and the single object tracking (SOT) strategy. Besides, due to the fact that the velocity of vehicles frequently changes sharply in the intersections, it is difficult to set a fixed threshold for the Mahalanobis distance in DeepSORT. For instance, when the vehicle accelerates sharply, the variance could be very large. As a consequence, the Mahalanobis distance could be extremely small. To avoid such situation, we propose a Mahalanobis distance smoothness method for a reasonable distance. We also propose tracking spatial constraints to remove the interference of vehicles outside the ROI. The

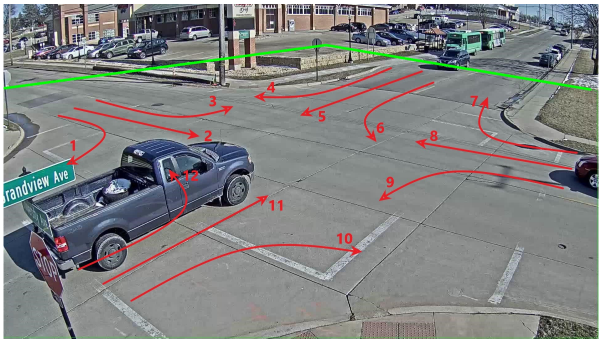

Figure 1. The visualization of the movements of interest (MOI) and region of interest (ROI) at one intersection. Each arrow line indicates one movement and the green lines outline the ROI.

tracklets predicted as having exited the ROI will enter into the online vehicle couting module.

Furthermore, to deal with movement-specific vehicle counting problem, we propose a shape-based movement assignment method. The main idea of this method is to generate a typical trajectory for each MOI, and calculate the shape similarity between a tracklet trajectory and each typical trajectory. The optimal movement of one trajectory can be generated by the typical trajectory with the best shape similarity among the typical trajectory set. Unlike [15] which also uses vehicle trajectories, our method does not need zones to delimit trajectory and the trajectories in our method is mostly selected from tracklets set in videos. Besides, the distance measurement method between tracked vehicle and trajectories in our method is more efficient. Our method is evaluated on AICity 2021 Track-1 Dataset. Experimental results show the effectiveness and efficiency of our method.

The main contributions are as follows: (1) We propose a DTC system for online movement-specific vehicle counting problem, which is a new and challenging task. (2) The detection augmentation method, Mahalanobis distance smoothness method and tracking spatial constraints are proposed to improve the multi-object tracking performance. (3) A shape-based movement assignment method with counting spatial constraints is carefully designed to accurately categorize each trajectory into different movements.

2. Methodology

In this paper, we propose an online movement-specific vehicle counting system to realize robust traffic flow analysis at crowded intersections. Figure 2 shows the framework of our proposed system. With a video stream as input, we sequentially detect vehicles in each frame, track their trajectories and stream out the online vehicle count. In subsequent sections, we will elaborate each step in detail.

2.1. Object Detection

We adopt PP-YOLO [20] with a Resnet50 [11] backbone as our frame-wise vehicle detector. PP-YOLO is an optimized model based on YOLOv3[23], which combines various existing tricks, including Grid Sensitive[5], Matrix NMS[29], CoordConv[18] and Spatial Pyramid Pooling[12] to improve performance and inference speed. Implementation details could be found in [22]. We adopt a COCO pre-trained model and then perform finetuning on our annotated AICity 2021 Dataset.

2.2. Online Multi-Object Tracking

Next, we perform online multi-vehicle tracking to get the trajectories inside ROI. We adopt DeepSORT [30] as the baseline method, which contains three major steps: motion prediction, feature extraction and data association.

2.2.1 Motion Prediction

As in [30], we use Kalman filter to predict motion and update state in the eight-dimensional tracklet state space ($x, y, a, h, \dot{x}, \dot{y}, \dot{a}, \dot{h}$), including the bounding box center position (x,y), aspect ratio a which is the ratio of width to height, bounding box height h and their respective velocities in image coordinates. The unmatched detections are applied to initialize new tracklets and the matched detections are applied to update the corresponding tracklets.

2.2.2 Feature Extraction

To reduce computational complexity, we adopt a feature combination with color histogram feature, motion feature and shape feature [32] instead of CNN appearance descriptor. The similarity distance between tracks and detections is calculated as:

$$d_{\text{appearance}}(i,j) = 1 - cosine(C_{\text{i}}, C_{\text{j}}) \tag{1}$$

$$d_{\text{motion}}(i,j) = 1 - e^{-w1 * ((\frac{X_{\text{i}} - X_{\text{j}}}{W_{\text{j}}})^2 + (\frac{Y_{\text{i}} - Y_{\text{j}}}{H_{\text{j}}})^2)} \tag{2}$$

$$d_{\text{shape}}(i,j) = 1 - e^{-w2 * (\frac{|W_{\text{i}} - W_{\text{j}}|}{W_{\text{i}} + W_{\text{j}}} + \frac{H_{\text{i}} - H_{\text{j}}}{H_{\text{i}} + H_{\text{j}}})} \tag{3}$$

$$d(i,j) = d_{\text{appearance}}(i,j) + d_{\text{motion}}(i,j) + d_{\text{shape}}(i,j) \tag{4}$$

where we denote i, j as a tracked object and a detected object, C denotes the color histogram feature, X, Y as the bounding box location coordinates and W, H denote the bounding box width and height, respectively. The cosine in equation (1) represents the cosine distance. We combine the three similarities with weights w1 and w2 as the final similarity distance.

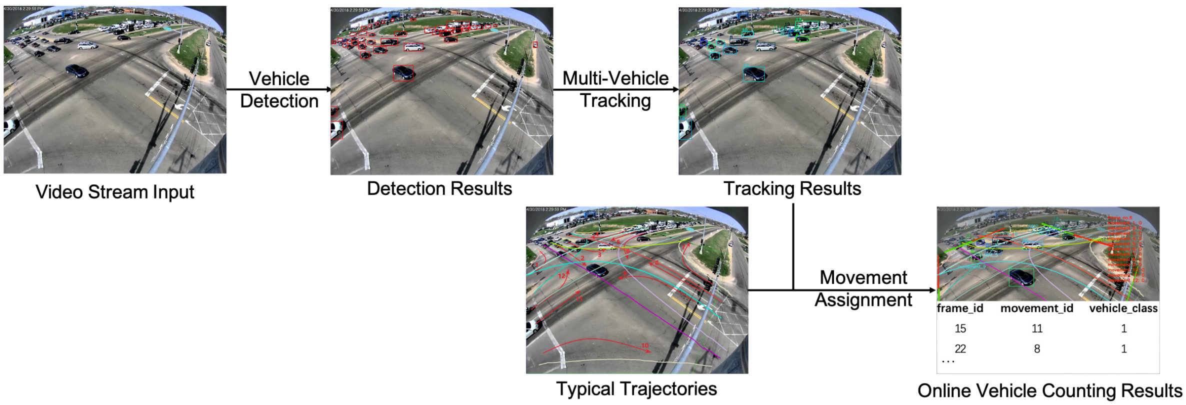

Figure 2. Our proposed Online detection-tracking-counting framework. Best viewed in color.

2.2.3 Data Association

We adopt the Matching Cascade algorithm from [30] to get the track ID for each detected vehicle. Hungarian algorithm [16] is performed to match the detections and tracklets based on the similarity distance we calculate above. The Mahalanobis distance [21] gate is used to disregarded infeasible assignments based on possible object locations inferred by the Kalman filter. Same as [19], we add identity matrix to the denominator to smooth the Mahalanobis distance result and avoid sudden change during non-linear motion stage. After Matching Cascade, we run intersection over union (IoU) association as proposed in the original SORT algorithm [4] to avoid mismatch caused by sudden appearance changes.

Detection Augmentation Strategy As we proposed in [19], additional detection augmentation strategies are adopted to improve the vehicle detection performance in some severe occluded scenes. It is very likely that only one object could be detected in two heavily occluded vehicles. In this case, two tracklets will compete for the same detection in the matching phase. One of the tracklets will switch into the unmatched state and easily cause ID switch. Due to this, we propose a detection rematch strategy. If both tracklets meet the IoU distance threshold, we will copy a new detection and update the Kalman state space of the two tracklets by the weighed position between the last historical position and the matched detection. For the unmatched tracklets after above stage, we used the single object tracking strategy to predict their position. The predicted position will be used as matched detection hypothesis and for the updating of Kalman state space. The single object tracking strategy will be stopped when the tracklets exceed a predefined maximum age, or the predicted confidence is lower than the predefined threshold.

Tracking Spatial Constraints We propose to utilize spatial constrains to better connect vehicle tracking and movement-specific counting tasks. ROI masks generated according to official ROI definition are used to filter the vehicles completely outside the ROI. It could bring us benefits from two aspects. On the one hand, it avoids the interference of external vehicles on tracking. On the other hand, it allows us to easily get the last frame before the vehicle completely exits the ROI. In order to avoid id switch at the ROI boundary, we mark a tracklet out if its location predicted by kalman filter is completely outside the ROI. When a tracklet has not been updated for a long time or has been marked out, we suppose its trajectory is ended and perform online vehicle counting to count the movement-specific vehicle number.

2.3. Online Vehicle Counting

For each ended tracklet, we assign one movement based on the shape similarity between the tracklet and the typical trajectories. After that, the certain frame ID when the vehicle existed the ROI together with movement type and vehicle class name are streamed out for online traffic flow analysis.

2.3.1 Trajectory Modeling

The semi-automatic selected trajectories we proposed in [19] are used as typical trajectories for each movement. Figure 3 visualizes the typical trajectories.

2.3.2 Movement Assignment

We mainly utilize shape-based trajectory similarity and spatial constraints to get the correct movement type.

Shape-based Trajectory Similarity Follow [19], we get shape similarities between a tracklet and typical trajectories by calculating Hausdorff distance [25] and angle between directions. We set thresholds for the Hausdorff distance and direction separately to remove wrong movement assignments.

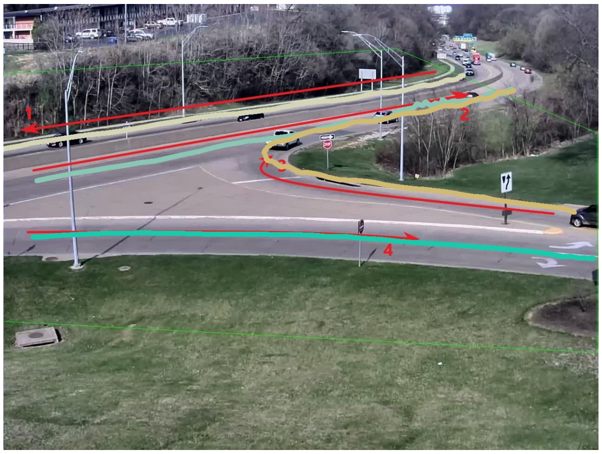

Figure 3. The visualization of the typical trajectories for each movement. The arrow lines indicate movements and the colored lines indicate the selected typical trajectories.

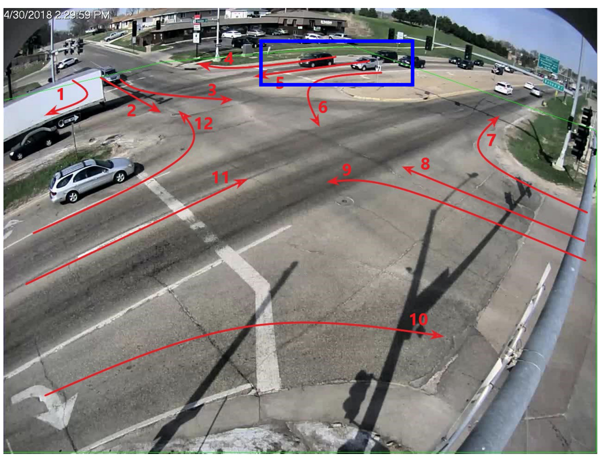

Figure 4. An example of counting spatial constraints. The tracklets whose last detected location is in the blue box area will not be counted because the area is a traffic light waiting zone near the start line.

Counting Spatial Constraints Furthermore, we set spatial constraints to disregard infeasible vehicle counts. False detection or tracking id switch could still happen in severe weather or heavy traffic scenes and lead to incomplete tracklet trajectory. The trajectory fragments are likely to match the wrong movement type. We manually collected some camera-specific spatial constraints to facilitate more robust movement assignment. Tracklets with too short length comparing to the typical trajectories or last detected in the traffic light waiting area are ignored when counting. An example of counting spatial constraints is shown in Figure 4. The movement assignment with smallest Hausdorff distance and satisfied counting spatial constraints is determined as the final matching result.

2.3.3 Counting Number

Finally, given the tracklet and the corresponding movement assignment, we record the frame ID that a certain vehicle exiting the ROI as the counting output. The results are streamed out for further real-time traffic flow analysis.

3. Experiments

3.1. Datasets

AICity 2021 Dataset The dataset is composed of about 9 hours video which is divided into 31 video clips. The video clips are captured from 20 unique camera views of typical traffic situation including intersection single approaches, full intersections, highway segments and city streets. Some of them are captured under various lighting and weather conditions including dawn, rain and snow. Region of interest (ROI) and movements of interest (MOI) of each camera view are annotated in detail. The 9 hours of video in Track-1 are split into datasets A and B. Dataset A is provided with instruction document and a small subset of ground truth labels for demonstration purpose and it can be used for training and validation. Meanwhile, dataset B is reserved for later testing. We will show our experiment results on dataset A since dataset B is not available to participants. We picked 17918 frames from AICity2021 Track-1 dataset A and 3516 frames from AICity2021 Track-3 dataset and annotated them with vehicle detection label.

3.2. Evaluation Metrics

Evaluation Metrics for vehicle Counting Following the official guide, AICity2021 dataset is evaluated with Track-1 efficiency score ($S1_{efficiency}$) and Track-1 effectiveness score ($S1_{effectiveness}$):

$$\begin{gathered} S1 = \alpha S1_{\text{efficiency}} + \beta S1_{\text{effectiveness}} \\ \text{where } \alpha = 0.3, \beta = 0.7 \end{gathered} \tag{5}$$

Since counting accuracy and the program efficiency should be combined to evaluate the performance of our model, weighted combination of $S1_{efficiency}$ and $S1_{effectiveness}$ shown above can be more accurate.

The $S1_{efficiency}$ score is calculated based on the total running time and adjusted by a efficiency base factor which is measured on specific test systems the experiments are executed on. Finally $S1_{efficiency}$ is normalized within $[0, 1.1 \times video\,play - back\,time]$.

The $S1_{effectiveness}$ score is calculated as a weighted average of normalized weighted root mean square error scores (nwRMSE) across all videos, movements and vehicle classed in test set, with proportional weights based on the number of vehicles of the given class in the movement. Each video is split into k continual segments and we consider the cumulative vehicle counts from the start of the video to the end of each segment. The nwRMSE score is the weighted RMSE (wRMSE) between the predicted and true cumulative vehicle counts, normalized by the true count of vehicles of that type in that movement. To further reduce that impact of errors on early segments, the wRMSE score weighs each record incrementally in order to give more weight to recent records.

$$
\begin{aligned}
wRMSE &= \sqrt{\sum_{i=1}^{k} w_i \left(\widehat{x_l} - x_i\right)^2} \\
\text{where } w_i &= \frac{i}{\sum_{j=1}^{k} j} = \frac{2i}{k(k+1)}
\end{aligned} \tag{6}
$$

3.3. Implementation details

We employ PP-YOLO with backbone ResNet50 as our detection network. The pre-trained parameters on COCO dataset from [22] is used. The model is fine-tuned on AICity2021 dataset with momentum optimizer for 100000 iterations. Learning rate is initialized as 0.0025 and is decreased by 0.1 at the 20000 and 60000 iters. The input resolution during training is random selected from 320 to 608. TensorRT INT8 mode is used during the inference period.

We run our experiments on a machine with 1 NVIDIA GTX1080Ti GPU which has 0.874121 AICity Track1 efficiency base factor. GPU is only used for running vehicle detection.

3.4. Ablation Experiments

The ablation study on the AICity Track-1 dataset A is intended to show two aspects:(1) the affect of detection on counting results; (2) the effectiveness of online tracking and counting strategies.

Comparisons between detection Models. As a key part of three-stage vehicle counting method, detection model can be crucial to the whole process, so we trained and tested several potential detection models on AICity 2021 Dataset, including PP-YOLO with MobileNet V3 [14] / ResNet50 [13] as backbone and Faster R-CNN [24] with ResNet50 as backbone, which we used in [19]. The test results are listed in Table 1. Considering that both performance and speed are vital to vehicle counting task, we choose PP-YOLO with backbone ResNet50 as our detector.

Online tracking and counting strategies. In this section, we will introduce our tracking and counting strategies. The last two stages of vehicle counting pipeline determine the final counting result. In Table 2, we show our results with different combination of strategies which bring further improvement. Following previous work, the baseline represents the method used in [19]. *TSC* denotes tracking spatial constraint, including filtering the detection results with 0-1 ROI mask generated from ROI provided by official and marking the end of tracks which are predicted out of the boundaries of ROI. It reminds what we should focus and also rules out the detection results out of ROI which may cause false positive counts. $S1_{effectiveness}$ of method with *TSC* surpasses the baseline with 5.6%. At the same time, another constraint *CSC* denotes counting spatial constraint. Tracks in restricted area (e.g. edge of MOI, the area around traffic lights) and tracks with too short time span should not be taken into consideration. These strategies can apparently improve our effectiveness.

model	mAP	FPS
Faster R-CNN(Res50)	0.399	19.5
PPYOLO(MobileNet v3)	0.347	120.5
PPYOLO(Res50)	0.452	72.9

Table 1. Detection results based on different detection model. The test set is split from AICity2021 Dataset with 8205 images. The mAP is evaluation results of IoU=0.5:0.95, the fps is calculated without pre-processing and post-processing

Method	$S1_{effectiveness}$
baseline	86.44
baseline + TSC	92.06
baseline + TSC + CSC	93.44

Table 2. Comparison with different methods in tracking and counting stage. TSC means tracking spatial constraint and CSC means counting spatial constraint.

TeamID	S1 score
37(Ours)	**0.9467**
5	0.9459
8	0.9263
19	0.9249
118	0.9235

Table 3. Top 5 overall scores of the vehicle counting task in AICity2021 track 1. Our proposed method outperforms all the other competitors in terms of the overall score $S1$.

3.5. Overall Score on AICity2021 Track-1 Dataset

Comparisons of the overall scores As shown in Table 3, our proposed vehicle counting method shows superiority on effectiveness and efficiency, outperforms all the other competitors in terms of the overall score $S1$.

4. Conclusion

In this paper, we present our approach for the CVPR2021 Workshop AICity Challenge Track1. A robust online movement-specific vehicle counting system is proposed for traffic flow analysis. Our online tracking and counting strategies have shown good performance at crowded intersection traffic scenes in both effectiveness and efficiency.

References

[1] A. Abdagic, O. Tanovic, A. Aksamovic, and S. Huseinbegovic. Counting traffic using optical flow algorithm on video footage of a complex crossroad. pages 41–45, 2010.

[2] Shubhra Aich and Ian Stavness. Leaf counting with deep convolutional and deconvolutional networks. pages 2080–2089, 2017.

[3] J. Barandiaran, B. Murguia, and F. Boto. Real-time people counting using multiple lines. pages 159–162, 2008.

[4] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016.

[5] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[6] S. Bouaich, M. A. Mahraz, J. Riffi, and H. Tairi. Vehicle counting system in real-time. In *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, pages 1–4, 2018.

[7] Z. Dai, H. Song, X. Wang, Y. Fang, X. Yun, Z. Zhang, and H. Li. Video-based vehicle counting framework. *IEEE Access*, 7:64460–64470, 2019.

[8] Luca Del Pizzo, Pasquale Foggia, Antonio Greco, Gennaro Percannella, and Mario Vento. Counting people by rgb or depth overhead cameras. *Pattern Recognition Letters*, 81:41–50, 2016.

[9] KV Embleton, CE Gibson, and SI Heaney. Automated counting of phytoplankton by pattern recognition: a comparison with a manual counting method. *Journal of Plankton Research*, 25(6):669–681, 2003.

[10] KV Embleton, CE Gibson, and SI Heaney. Automated counting of phytoplankton by pattern recognition: a comparison with a manual counting method. *Journal of Plankton Research*, 25(6):669–681, 2003.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019.

[15] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[16] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[17] Haoxiang Liang, Huansheng Song, Huaiyu Li, and Zhe Dai. Vehicle counting system using deep learning and multi-object tracking methods. *Transportation Research Record*, page 0361198120912742, 2020.

[18] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *arXiv preprint arXiv:1807.03247*, 2018.

[19] Zhongji Liu, Wei Zhang, Xu Gao, Hao Meng, Xiao Tan, Xiaoxing Zhu, Zhan Xue, Xiaoqing Ye, Hongwu Zhang, Shilei Wen, et al. Robust movement-specific vehicle counting at crowded intersections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 614–615, 2020.

[20] Xiang Long, Kaipeng Deng, Guanzhong Wang, Yang Zhang, Qingqing Dang, Yuan Gao, Hui Shen, Jianguo Ren, Shumin Han, Errui Ding, et al. Pp-yolo: An effective and efficient implementation of object detector. *arXiv preprint arXiv:2007.12099*, 2020.

[21] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[22] PaddlePaddle. Paddle detection. `https://github.com/PaddlePaddle/PaddleDetection/tree/release/2.0-rc/configs/ppyolo`.

[23] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[24] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[25] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.

[26] Frank Y Shih and Xin Zhong. Automated counting and tracking of vehicles. *International Journal of Pattern Recognition and Artificial Intelligence*, 31(12):1750038, 2017.

[27] Maojin Sun, Yan Wang, Teng Li, Jing Lv, and Jun Wu. Vehicle counting in crowded scenes with multi-channel and multi-task convolutional neural networks. *Journal of Visual Communication and Image Representation*, 49:412–419, 2017.

[28] L. Unzueta, M. Nieto, A. Cortes, J. Barandiaran, O. Otaegui, and P. Sanchez. Adaptive multicue background subtraction for robust vehicle counting and classification. *IEEE Transactions on Intelligent Transportation Systems*, 13(2):527–540, 2012.

[29] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic, faster and stronger. *arXiv preprint arXiv:2003.10152*, 2020.

[30] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[31] Honghong Yang and Shiru Qu. Real-time vehicle detection and counting in complex traffic scenes using background

subtraction model with low-rank decomposition. *IET Intelligent Transport Systems*, 12(1):75–85, 2017.

[32] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *Computer Vision – ECCV 2016 Workshops*, pages 36–42, Cham, 2016. Springer International Publishing.