Robust Movement-Specific Vehicle Counting at Crowded Intersections

Zhongji Liu* Wei Zhang* Xu Gao* Hao Meng* Xiao Tan Xiaoxing Zhu
Zhan Xue Xiaoqing Ye Hongwu Zhang Shilei Wen Errui Ding
{liuzhongji,zhangwei99,gaoxu03,menghao05,tanxiao01,zhuxiaoxing}@baidu.com
{xuezhan,yexiaoqing, zhanghongwu,wenshilei, dingerrui}@baidu.com
Baidu Inc.

Abstract

With the demands of intelligent traffic, vehicle counting has become a vital problem, which can be used to mitigate traffic congestion and elevate the efficiency of the traffic light. Traditional vehicle counting problems focus on counting vehicles in a single frame or consecutive frames. Nevertheless, they are not expected to count vehicles by movements of interest (MOI), which can be pre-defined by all possible states of vehicles, combining different lanes and directions. In this paper, we mainly focus on movement-specific vehicle counting problem. A detection-tracking-counting (DTC) framework is applied, which detects and tracks objects in the region of interest (ROI), then counts those tracked trajectories by movements. To be specific, we propose the detection augmentation method and the Mahalanobis distance smoothness method to improve the multi-object tracking performance. For vehicle counting, a shape-based movement assignment method is carefully designed to categorize each trajectory by movements. Experiments are conducted on both the AICity 2020 Track-1 Dataset and the Vehicle-Track Dataset, which is built in this paper. Experimental results show the effectiveness and efficiency of our method.

1. Introduction

Vehicle counting in the traffic scene is a crucial problem in computer vision, since it is expected to mitigate traffic congestion and elevate the efficiency of the traffic light. To be specific, the goal of vehicle counting by movements of interest (MOI) is to find the number of vehicles that are corresponding to MOI in a period of time, where MOI can be pre-defined by all possible states of vehicles, combining different lanes and directions (left-turning/right-turning/through). Furthermore, the arrival timestamp when vehicles move out of the region of interest (ROI) is expected to be given. An illustration of MOI and ROI is shown in Fig. 1. However, accurate vehicle counting is still challenging at crowded intersections, due to the difficulties such as the occlusions between different vehicles and poor weather conditions.

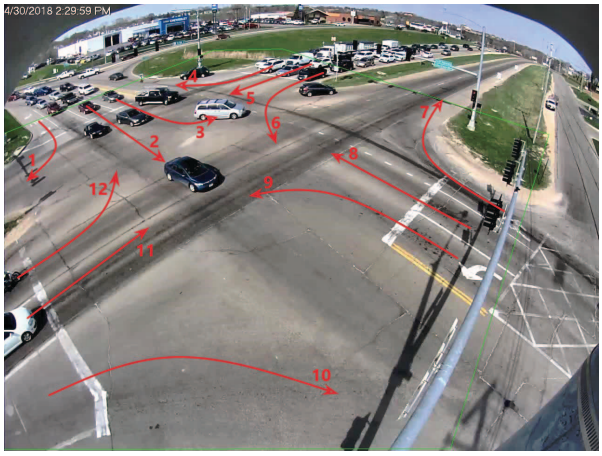

Figure 1. The visualization of the movements of interest (MOI) and region of interest (ROI) at one intersection. Each arrow line indicates one movement and the green lines outline the ROI.

Traditional vehicle counting problems can be divided into two categories. One is frame-wise vehicle counting [15, 1, 45, 50], which aims at counting vehicles in a single frame, regardless of the identity of each vehicle. Some studies use a density-aware strategy [5, 16, 2] , which uses density estimation algorithms to regress the number of vehicles. Density-aware strategy is suitable for heavy crowds counting. Besides, based on the recent progress of deep learning methods for object detection [8, 43], more studies turn to the detection-based strategy [14, 41, 15, 1, 45], which is to detect vehicles then count the detected vehicles afterwards. However, due to the mutual occlusions between vehicles and the occlusions caused by roadside trees, both density-aware and detection-based approaches usually miss the the occluded vehicles.

*equal contribution

2160-7516/20/$31.00 ©2020 IEEE
DOI 10.1109/CVPRW50498.2020.00315

Another category of traditional vehicle counting problems is instance-wise vehicle counting, which aims at counting vehicles in consecutive frames. Hence, those vehicles that are occluded and not detected can be counted by taking advantage of consecutive frame knowledge. Specifically, these methods [10, 29] 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 movement-specific vehicle counting. Specifically, 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. Our proposed approach mainly follows the DTC pipeline, in which we choose Faster R-CNN [38] and DeepSORT [48] 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 aim at generating 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.

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 each trajectory and each typical trajectory in the typical trajectory set. The optimal movement of one trajectory can be generated by the typical trajectory with the best shape similarity among the typical trajectory set. Different with [25] 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 both AICity 2020 Track-1 Dataset and the Vehicle-Track Dataset, which is built in this paper. Experimental results show the effectiveness and efficiency of our method.

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

2. Related Work

Object Detection. Object detection is one of the most fundamental and challenging problems in computer vision. Feature extraction is one of the primary tasks in object detection. Before the emergence of deep learning algorithms, most object detection algorithms [46, 20] are built based on manual features (such as HOG[11] and SIFT[27]), Then use the extracted features to train SVM, Adaboost and other classifiers to obtain detection results. However, these methods are not adaptive to rotations, since they are not robust in the complex scenes. The advent of deep learning has changed this situation. It uses a data-driven approach, which avoids manual feature extraction and allows the machine to automatically learn feature expressions from massive amounts of data, making the detection ability greatly improved. In the era of deep learning, object detection can be divided into two categories: "two-stage detection" and "one-stage detection", where the former (RCNN [23], Fast RCNN [22], Faster RCNN [38] etc.) regards the detection as a "coarse-to-fine" process while the later (YOLO [34], SSD [32], YOLOv2 [35], YOLOv3 [36], etc.) regards it as to "complete in one step"[55]. Meanwhile, the implementation of the excellent object detection algorithm cannot lack large labeled datasets. As a result, a number of well-known datasets and benchmarks are released by many research institutions, including the datasets of PASCAL VOC Challenges [18, 17], MS-COCO Detection Challenge [31], ImageNet Large Scale Visual Recognition Challenge [39], etc.

Multi-Object Tracking. Multi-Object tracking (MOT) is another very important task in computer vision. In recent years, with the improvement of object detection, many existing MOT studies adopts the tracking-by-detection strategy, which performs object detection first and then associate the detections afterwards. These studies can be grouped into offline methods and online methods. The offline methods can use detections from the whole frame sequences, and then conduct global optimization, containing graph-based methods and hierarchical methods. The former methods construct the MOT as a graph model, which can be optimized using k-shortest path [6], minimum cost flow [28, 13, 47], and subgraph decomposition[44, 12], while

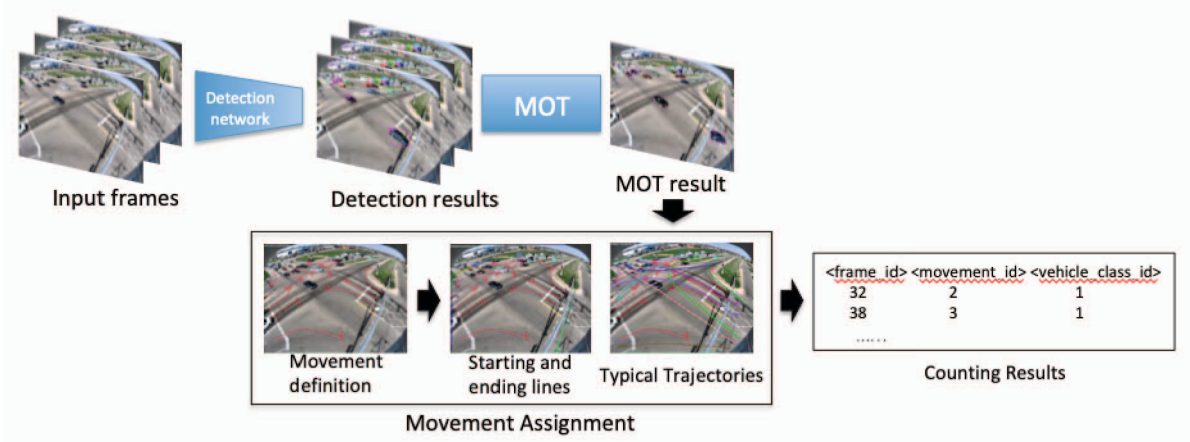

Figure 2. Our proposed detection-tracking-counting framework. Best viewed in color.

the latter methods construct the trajectories in a hierarchical manner. In comparison, Online object tracking only use the detections from current and previous frames. Recent online MOT methods model MOT problem as a data association problem between the tracked objects and the detections. The key is to evaluate the similarity between each tracked object and each detected object by using different mechanisms, including the attention mechanism [9, 21, 53], the single object tracking mechanism for pre-processing [9, 53]) and different neural networks(Recurrent Neural Networks [19, 33, 40, 54, 21] and Reinforcement Learning [49, 37]).

Vehicle Counting. The existing methods of vehicle counting can be mainly divided into two categories: density-aware approches [5, 16, 2] and detection-aware approches [10, 3, 42, 52, 50, 8, 43]. The density-aware method aims at learning a regression function by using object features to predict the counting results, while the features can either be done by manually crafting features [16] or be used to annotate data to train feature extractors [2]. However, these methods can only estimate the possible number of objects, but cannot accurately count the objects. Fortunately, the detection-aware methods can achieve more accurate counting results. Generally, the detection-aware approches can be grouped into two categories, including frame-wise vehicle counting [52, 14, 41, 8, 43] and instance-wise vehicle counting [10, 3, 42]. The goal of frame-wise vehicle counting is to count vehicles in a single frame, regardless of the identity of each vehicle. However, the frame-wise works can only estimate the possible number of vehicles in a single frame, but cannot accurately count the vehicles in the instance level. The instance-wise vehicle counting takes trajectories, which generated by detection/tracking systems, as input, and the counting techniques are applied to the trajectories in order to reduce the deviation between the number of tracks and the actual number of instances. Hence, those vehicles that are occluded and not detected can be counted by taking advantage of consecutive frame knowledge.

3. Methodology

As shown in Figure 2, our proposed detection-tracking-counting framework contains three major steps: frame-wise vehicle detection, online multiple vehicle tracking and typical trajectory-based vehicle counting. Given one video as input, our framework is able to output a list of counting results, in which each line records one counted vehicle. Each step in our proposed framework will be elaborated in detail in subsequent sections.

3.1. Object Detection

We choose a two-stage Faster R-CNN [38] as the detector, which adopts Resnet50 [24] as the backbone feature extractor. We further enhance the backbone with FPN [30] for better usage of the mutliscale context information. Also, we use a pretrained model on COCO and fine-tune this model on the manually annotated data in AICity 2020 Track-1 Dataset.

3.2. Online Multi-Object Tracking

After obtaining the detection results of each frame, we use DeepSort [48] as the baseline method for online multi-object tracking. We acknowledge that other offline trackers might perform better, we perform online tracking in our proposed method which realizes more applications in online traffic control scenarios.

3.2.1 Preliminary

DeepSort [48] adopts a single hypothesis tracking methodology with recursive Kalman filtering and frame-by-frame data association. Tracklet state space in Kalman filtering is defined in the eight dimensional state space (u, v, γ, h,

$\dot{x}, \dot{y}, \dot{\gamma}, \dot{h}$), including the bounding box center position (*u, v*), aspect ratio γ, height *h*, and their respective velocities in image coordinates. For each tracklet *k* we use a standard Kalman filter with constant velocity motion and linear observation model. Different from [48], we formulate a combination feature with color histogram feature, motion feature and shape feature [51] of objects to obtain similarity matrix between detections and predicted Kalman states of tracklets. To disregard infeasible assignments based on possible object locations, the (squared) Mahalanobis distance between detections and predicted Kalman states is calculated as:

$$d(i,j) = (\mathbf{d}_j - \mathbf{y}_i)^T \mathbf{S}_i^{-1} (\mathbf{d}_j - \mathbf{y}_i) \quad (1)$$

where we denote $\mathbf{S}_i$ as covariance matrix and the the projection of the *i*-th tracklet distribution into measurement space by $(y_i, \mathbf{S}_i)$ and the *j*-th bounding box detection by $\mathbf{d}_j$.

Then, we perform Hungarian algorithm to match the detections and tracklets based on the gated similarity matrix we calculate above. In the final stage, we run intersection over union association as proposed in the original SORT algorithm [7] to avoid mismatch caused by sudden appearance changes.

For each tracklet *k*, we count the number of frames since its last successful association with detection, and delete the tracklets that exceed a predefined maximum age A_{max}. Unmatched detections after matching will be initiated and these new tracklets are classified as tentative during their first N_INIT frames. Tracklets that are not successfully associated during their first N_INIT frames will be deleted.

However, the basic pipeline algorithm depends on the performance of detection model, hence it is hard to work well in complicated scenes in AICity task. For instance, when severe occlusion occurs, there is high probability that the detection of the occluded object is missing. Thus, we propose a series of methods as below to solve the problems in these complex scenarios.

3.2.2 Detection Augmentation Method

It is difficult to detect objects under low image quality or severe occlusion in heavy traffic scenarios. Lots of ID switches will occur and affect the final counting results seriously. To deal with such situation, we propose a detection augmentation method to supplement missing detections for data association. Specially, two strategies are proposed for detection augmentation, including the detection re-match strategy and the single object tracking strategy.

Detection Re-Match Strategy. There is a high probability that we can only get one detection when two objects are in severe occlusion. When two tracklets compete for the same detection in the matching stage, one of them will move into unmatched state, which could cause ID switches of vehicles. Hence, we propose the detection re-match strategy. If both tracklets satisfy the intersection over union association threshold, we simply duplicate this detection and update the Kalman state space of both these two tracklets by weighted position between the last history position and the matched detection. Figure 3 shows the visualization results without and with our detection re-match startegy.

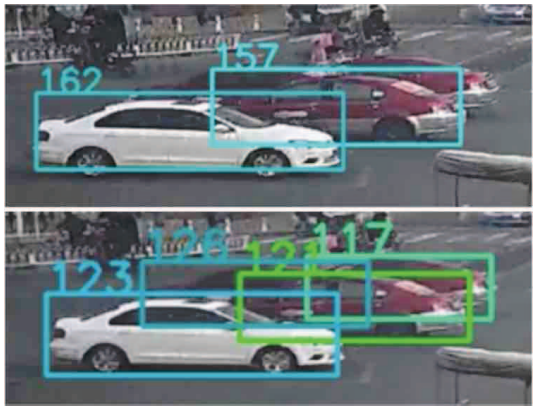

Figure 3. The examples of detection missing fixed when detection re-match strategy is used(bottom) comparing with the preliminary(upper).

Single Object Tracking Strategy. We use single object tracking strategy to predict and add possible position when no detection bounding box is available. For unmatched tracklets after above stage, we limit a search region by expanding the last history position and do template matching method in the current frame image to predict a possible position. The predicted position will be used as matched detection hypothesis and to update the Kalman state space. We maintain a counter which is incremented if the tracklets is recovered by single object tracking method and reset to 0 when the tracklet has been associated with a real detection bounding box. Tracklets exceed a predefined maximum age $\mathbf{Age}_{max}$, or the predict confidence is lower than the predefined threshold will stop using single object tracking strategy.

Mahalanobis Distance Smoothness Method The standard Kalman filter we used is based on constant velocity motion and linear observation model, which is not suitable for all situations in reality. In AICity task scenes, many vehicles will stop or start in the intersections where velocity of vehicle changes sharply. According to formula (1), the difference between $\mathbf{d}_j - \mathbf{y}_i$ will be relatively large while the state estimation uncertainty covariance $\mathbf{S}_i$ relatively maintains small at this motion stage, which will cause the Mahalanobis distance to exceed the predefined threshold. We modify the calculation formula by adding identity matrix to $\mathbf{S}_i$ to avoid sudden change during non-linear motion stage:

$$d(i,j) = (\mathbf{d}_j - \mathbf{y}_i)^T (\mathbf{S}_i + \alpha \mathbf{E})^{-1} (\mathbf{d}_j - \mathbf{y}_i) \quad (2)$$

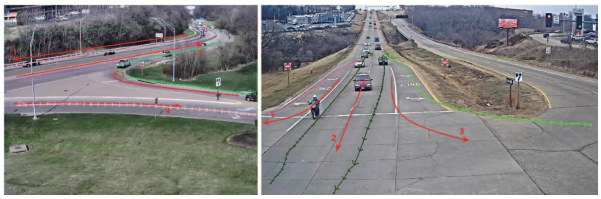

Figure 4. The visualization of the typical trajectories for each movements. The arrow lines indicate movements and the colored lines indicate the selected typical trajectories.

where α is a parameter to balance the original covariance matrix and the identity matrix E.

3.3. Vehicle Counting

In this section, we introduce our proposed shape-based vehicle counting method. Specifically, we first select the typical trajectories for each movement. Then, we assign one movement for each vehicle tracklet by measuring the distances between tracklets and the typical trajectories of the movements. Finally, the movement assignment and the frame ID when the certain vehicle is fully exiting the ROI are recorded as the final counting output.

3.3.1 Typical Trajectories Selection

We propose a semi-automatic approach for typical trajectory selection which combines a rule based computation and manual labeling. First, we label the entrance line and the exit line for each movement. For each movement, we subsequently collect a set of trajectories which hit both of the entrance line and exit line. After that, we manually select one or two trajectories as typical trajectories for each movement from the trajectory set. Besides, for those movements with empty trajectory set, we manually draw a set of points to serve as the typical trajectory. Figure 4 visualizes our selected typical trajectories.

3.3.2 Shape-Based Movement Assignment

Next, given a vehicle tracklet, we assign one movement based on the distance measurement between the tracklet and typical trajectories in the same camera.

Specifically, let $d_H(t, j)$ denote the Hausdorff distance between tracklet t and typical trajectory j of movement m. Both of t and j consist of the center points of all the bounding boxes in them. Then, the distance between tracklet t and movement m is:

$$d_{t,m} = \min_{j \in traj(m)} d_H(t, j) \tag{3}$$

Moreover, since Hausdorff distance regards the tracklet and trajectory as disordered point sets, we further propose to measure the direction similarities between the tracklet and typical trajectories. Specifically, let $\vec{v_t}$ denote the vector which points from the starting bounding box to the terminal bounding box in tracklet t. $t^{start}(*)$ and $t^{end}(*)$ denote the pixel coordinates of the center of the starting bounding box and the terminal bounding box, respectively. Similarly, we can define $\vec{v_j}$ using typical trajectories, which can be written as,

$$\vec{v_t} = \langle t^{end}(x) - t^{start}(x), t^{end}(y) - t^{start}(y) \rangle \tag{4}$$

$$\vec{v_j} = \langle j^{end}(x) - j^{start}(x), j^{end}(y) - j^{start}(y) \rangle \tag{5}$$

$\theta_{t,j}$ denotes the angle between $\vec{v_t}$ and $\vec{v_j}$. Then, the angle between tracklet t and movement m is:

$$\theta_{t,m} = \min_{j \in traj(m)} \theta(t, j) \tag{6}$$

Algorithm 1 shows our proposed Hausdorff distance-based movement assignment algorithm. In order to facilitate more robust movement assignment, we further propose to utilize spatial constrains which are manually collected. For example, for movement 1 in the right hand-side video in Figure 4, the center of the starting bounding box should not be located in the right half of the video frame.

Algorithm 1 Shape-based Movement Assignment

Input: Given one video:
tracklet indices $\mathcal{T} = \{1, ..., T\}$;
Movement indices $\mathcal{M} = \{1, ..., M\}$;
Threshold H;
Spatial constrain $SC(*)$;
Output: Movement assignment $M : \mathcal{T} \rightarrow \mathcal{M}$;
for each $t \in [1, T]$ **do**
  Compute distance vector $D_t = [d_{t,m}]$;
  Sort $\mathcal{M}$ using D_t in increasing order;
  for each $m \in [1, M]$ **do**
    if $d_{t,m} > H$ or $SC(t, m) < 0$ or $\theta_{t,m} > \pi/2$ **then**
      continue
    end if
    $M(t) = m$
    break
  end for
end for
return M

3.3.3 Vehicle Counting

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.

4. Experiments

4.1. Datasets

AICity 2020 Track-1 Dataset The data set contains 31 video clips (about 9 hours in total) captured from 20 unique camera views (some cameras provide multiple video clips to cover different lighting and weather conditions.). These cameras locate in a city in the United States as well as from state highways in Iowa. Each camera view comes with a detailed instruction document describing the region of interest (ROI) and movements of interest (MOI). The 9 hours of video in track 1 are split into two data sets A and B. Data set A (5 hours in total) along with all the corresponding instruction documents and a small subset of ground truth labels (for demonstration purpose) are made available to participating teams. Data set B will be reserved for later testing. Since the dataset B is not available to participates, we provide experimental results on datasetA in the subsquent sections. Detection model is fine-tuned on the AICity 2020 Track-1 DatasetA with pretrained model on COCO. We totally annotate 17918 frames from the video in AICity2020 Track-1 datasetA and only cars and trucks are labeled.

Vehicle-Track Dataset We introduce a new dataset, named Vehicle-Track Dataset, for vehicle tracking at multipule intersections. This dataset contains 11 video clips (about 1 hour in total) captured from 11 unique cameras at 8 different intersections from one Chinese City (some intersections contain more than one camera to cover different movements). Also, each video comes with one binary mask map describing the region of interest (ROI). For all of the 11 videos, all the vehicles in the ROI are manually labeled with the bounding box and tack Id. Besides, each vehicle is labeled with one of the four types: car, bus, truck, van.

4.2. Evaluation Metrics

Evaluation Metrics for vehicle Counting For AICity2020 dataset, we adopt the official evaluation metrics in AICity 2020 Challenge. The finally score S_1 is a weighted combination between efficiency score $S1_{efficiency}$ and effectiveness score $S1_{effectiveness}$:

$$S1 = 0.3 * S1_{efficiency} + 0.7 * S1_{effectiveness} \quad (7)$$

where $S1_{efficiency}$ measures the time consumption compared which the total length of all videos under certain computational resources. $S1_{effectiveness}$ is computed as a weighted average of normalized weighted root mean square error scores nwRMSE across all videos, movements, and vehicle classes in the test set, with proportional weights based on the number of vehicles of the given class in the movement. To reduce jitters due to labeling discrepancies, each video is split into segments and we consider the cumulative vehicle counts from the start of the video to the end of each segment. Detailed illustrations of efficiency score $S1_{efficiency}$ and effectiveness score $S1_{effectiveness}$ can be found on the official website of AICity2020 Challenge: `https://www.aicitychallenge.org/2020-data-and-evaluation/`.

Evaluation Metrics for Vehicle Tracking For our proposed Vehicle-Track Dataset, we adopt a standard metric, the CLEAR MOT [4], for multiple vehicle tracking evaluation.

4.3. Implementation details

We employ Resnet50 [24] with FPN [30] as the backbone network in Faster R-CNN [38]. The network is trained with standard SGD optimizer for 90000 iters. The learning rate is initialized as 0.02 and is decreased by 0.1 at the 30000 and 50000 iters. Random horizontal flip is applied to reduce overfitting. The input resolution is fixed to $800*800$.

Evaluation on Vehicle-track dataset is carried out using A_{max} = 20 frames and N_INIT = 3 frames in online tracking stage. Detections are threshold at a confidence score of 0.3. In our tracking strategies, $\mathbf{A}_{sot\_max}$ is set to 10 and minimize template match confidence is set to 0.9. The α in Mahalanobis distance smoothness method is set to 50 and the gating distance is adjusted from 9.4877 to 50. The threshold of similarity matrix is adjusted to 0.6.

We run our experiments on a server with 4 Tesla P4 GPUs which efficiency base factor calculated by AICity task1 tool is 0.4649. GPUs are only used for running detection model on all frames extracted from task1 videos.

4.4. Ablation Experiments

The ablation study on the AICity Track-1 datasetA is intended to show three aspects:(1) the affect of detection on counting results; (2) the effectiveness of online tracking strategies; (3) the suitable counting method for counting tracklets by typical trajectories.

Comparisons between detection Models. Beside the Faster R-CNN with ResNet50-FPN as backbone we use in the final submission, we also train other several detection model to test how the detection results affect our method, including Faster R-CNN with SENet154-FPN [26] as backbone and YOLOv3 [36]with ResNet50 as backbone. The test results are listed in Table 1 and Faster R-CNN with backbone Resnet50 is selected to get a trade-off between effectiveness and efficiency. Different detection results from several model get the similar effectiveness score which intimate that our tracking and counting methodology is robust to detection results.

Online tracking strategies. In this place, we evaluate the effects of our tracking strategies. In Table 2, *DeepSort* means the basic pipeline of DeepSort in which we replace the deep reid feature with the combination of color histogram feature, motion feature and shape feature. *DeepSort(+DA)* corresponds to using detection augmentation

model	$S1_{effectiveness}$	$S1_{efficiency}$
Faster R-CNN(SE154)	0.901492	0.773492
Faster R-CNN(Res50)	0.902973	0.948114
YOLOv3(Res50)	0.898484	0.955951

Table 1. Counting results based on different detection model. The effectiveness is calculated based on the manually annotated counting ground-truth for first 5 minutes of each video in the AICity 2020 Track-1 DatasetA.

Method	MOTA ↑	MOTP ↑	ID Sw. ↓
DeepSort	88.76	88.12	492
DeepSort(+DA)	88.69	88.12	483
DeepSort(+MDS)	88.90	88.12	439
Ours	88.81	88.12	431

Table 2. Comparison with different strategies in online multi-object tracking.

Method	$S1_{effectiveness}$	Run_time
line-based	86.44	174s
NCC-based	89.87	704s
shape-based	92.06	71s
shape-based+D+S	93.44	71s

Table 3. Comparison with different methods in counting stage.

method and *DeepSort(+MDS)* corresponds to using the mahalanobis distance smoothness method. We finally use both strategies in online multi-object tracking stage. With these strategies the identity switches reduce substantially while the tracking speed maintains as before.

Comparisons between different counting strategies To demonstrate the effectiveness of our proposed vehicle counting method, we conduct a comparison between alternative approaches and different settings based on our proposed Hausdorff distance-based method. As shown in Table 3, *line-based* denotes the counting strategy that only adopts the starting and ending line to identity movement for each tracklet. *NCC-based* denotes the similarity between tracklets and the typical trajectories are computed based on the normalized correlation. *shape-based* denotes our proposed counting strategy based on Hausdorff distance measurement. *shape+D+S* denotes the full strategy proposed in section 3.3 which further incorporates direction and spatial constrains.

4.5. Overall Score on AICity2020 Tack1 Dataset

Comparisons of the overall scores As shown in Table 4, our proposed vehicle counting method outperforms all the other competitors in terms of the overall score $S1$.

TeamID	S1 score
99(Ours)	**0.9389**
110	0.9346
92	0.9292
26	0.8936
22	0.8852

Table 4. Top 5 overall scores of the vehicle counting task in AICity2020 track 1. Our proposed method outperforms all the other competitors in terms of the overall score $S1$.

5. Conclusion

In this paper, we apply a detection-tracking-counting (DTC) framework for movement-specific vehicle counting problem. To improve the multi-object tracking performance, we propose the detection augmentation method and the Mahalanobis distance smoothness method. For vehicle counting, a shape-based movement assignment method is carefully designed to categorize each trajectory into different movements. Experimental results show the effectiveness and efficiency of our method.

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] G. Antonini and J. P. Thiran. Counting pedestrians in video sequences using trajectory clustering. *IEEE Transactions on Circuits and Systems for Video Technology*, 16(8):1008–1020, 2006.

[4] Andrew D. Bagdanov, Alberto Del Bimbo, Fabrizio Dini, Giuseppe Lisanti, and Iacopo Masi. Posterity logging of imagery for video surveillance. 2012.

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

[6] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819, 2011.

[7] 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.

[8] 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.

[9] Qi Chu, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, and Nenghai Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal atten-

tion mechanism. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4836–4845, 2017.

[10] 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.

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[12] Afshin Dehghan, Shayan Modiri Assari, and Mubarak Shah. Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4091–4099, 2015.

[13] Afshin Dehghan, Yicong Tian, Philip HS Torr, and Mubarak Shah. Target identity-aware network flow for online multiple target tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1146–1154, 2015.

[14] 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.

[15] 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.

[16] 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.

[17] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[18] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[19] K. Fang, Y. Xiang, X. Li, and S. Savarese. Recurrent autoregressive networks for online multi-object tracking. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 466–475, 2018.

[20] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[21] Xu Gao and Tingting Jiang. Osmo: Online specific models for occlusion in multiple object tracking under surveillance scene. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 201–210, 2018.

[22] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[23] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

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

[25] 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.

[26] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CoRR*, abs/1709.01507, 2017.

[27] Luo Juan and Luo Gwon. A comparison of sift, pca-sift and surf. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 8(3):169–176, 2007.

[28] Li Zhang, Yuan Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[29] 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.

[30] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[31] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[32] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[33] Anton Milan, S Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[34] Joseph Redmon, Santosh Divvala, Ross 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*, pages 779–788, 2016.

[35] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

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

[37] Liangliang Ren, Jiwen Lu, Zifeng Wang, Qi Tian, and Jie Zhou. Collaborative deep reinforcement learning for multi-object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 586–602, 2018.

[38] 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.

[39] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[40] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 300–311, 2017.

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

[42] Huansheng Song, Xuan Wang, Cui Hua, Weixing Wang, Qi Guan, and Zhaoyang Zhang. Vehicle trajectory clustering based on 3d information via a coarse-to-fine strategy. *Soft Computing*, 22(5):1433–1444, 2018.

[43] 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.

[44] Siyu Tang, Bjoern Andres, Miykhaylo Andriluka, and Bernt Schiele. Subgraph decomposition for multi-target tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5033–5041, 2015.

[45] 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.

[46] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001.

[47] X. Wang, E. Türetken, F. Fleuret, and P. Fua. Tracking interacting objects using intertwined flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2312–2326, 2016.

[48] 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.

[49] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE international conference on computer vision*, pages 4705–4713, 2015.

[50] 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.

[51] 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.

[52] C. Zeng and H. Ma. Robust head-shoulder detection by pca-based multilevel hog-lbp detector for people counting. pages 2069–2072, 2010.

[53] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 366–382, 2018.

[54] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 366–382, 2018.

[55] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*, 2019.