ELECTRICITY: An Efficient Multi-camera Vehicle Tracking System for Intelligent City

Yijun Qian* , Lijun Yu* , Wenhe Liu , and Alexander G. Hauptmann

Language Technologies Institute, Carnegie Mellon University
yijunqia@andrew.cmu.edu,lijun@cmu.edu,wenhel@andrew.cmu.edu,alex@cs.cmu.edu

Abstract

City-scale multi-camera vehicle tracking is an important task in the intelligent city and traffic management. It is quite challenging with large scale variance, frequent occlusion and appearance variance caused by viewing perspective difference. In this paper, we propose ELECTRICITY, an efficient multi-camera vehicle tracking system with aggregation loss and fast multi-target cross-camera tracking strategy. The proposed system contains four main modules. Firstly, we extract tracklets under single camera view through object detection and multi-object tracking modules which shared the detection features. After that, we match the generated tracklets through a multi-camera re-identification module. Finally, we eliminate isolated tracklets and synchronize tracking ids according to the re-identification results. The proposed system wins the first place in the City-Scale Multi-Camera Vehicle Tracking of AI City 2020 Challenge (Track 3)[1] with a score of 0.4585.

1. Introduction

With the continuous expansion of the city scale, the management of city has become more and more challenging. Thanks to the development of computer vision technology and surveillance network throughout the city, there are many new options for city management, especially in traffic management. Among them, multi-camera vehicle tracking is one of the important tasks. It aims to track the vehicles over large areas in multiple surveillance camera networks. Furthermore, it enables better transportation design and traffic flow optimization. Different from classical multiple object tracking (MOT) which only focuses on tracking objects within a single camera, multi-camera vehicle tracking needs to resort to multiple cameras.

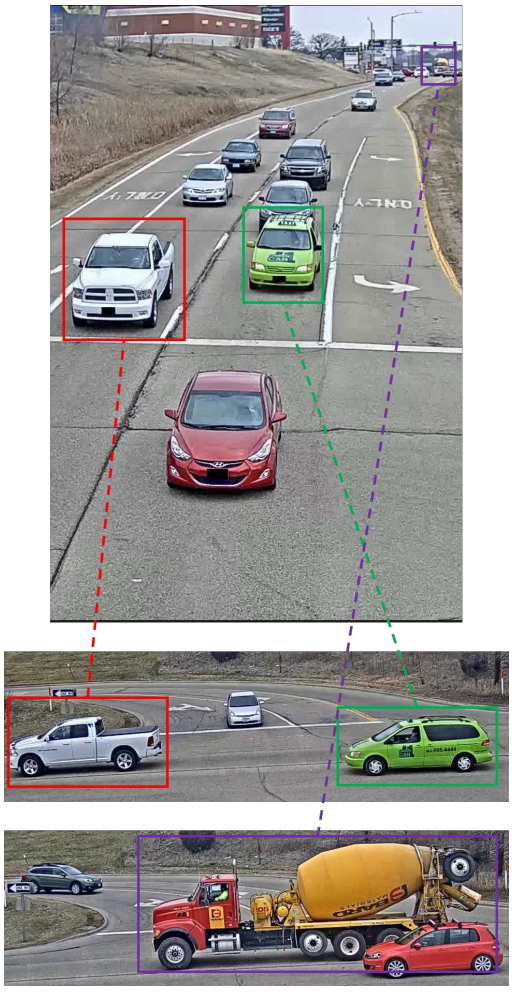

Figure 1: Multi-camera vehicle tracking needs to find out the same vehicles which appear in multiple cameras. Their appearance and size usually varies a lot due to the difference of viewing perspective and distance to cameras.

*Equal contribution
[1]https://www.aicitychallenge.org/

In fact, multi-camera vehicle tracking is a complicated task that includes detection, tracking, re-identification (ReID) and it is already a heated and leading-edge realm in computer vision research. Theses years have witnessed many successful works[[1],[24],[7],[23],[10],[9]], especially after the release of some public datasets[[25],[18],] and challenges in this realm. Although the former methods have achieved great performance on the public datasets, they usually use feature aggregation which costs lots of computational resource, such as GPUs. Moreover, most of them require annotations of large-scale datasets to train their models, which is difficult to collect. Meanwhile, there are still several challenges left in this realm:

1. The appearance of a specific vehicle usually varies a lot due to the difference of viewing perspectives and its distance to cameras. It brings challenges to accurate multi-camera ReID.

2. The synchronization procedure of tracking IDs tends to be a time-consuming work since it needs to match tracklets from different camera views.

To tackle these challenges, we proposed ELECTRICITY, an efficient and accurate multi-camera vehicle tracking system with aggregation loss and fast multi-target cross-camera tracking (MTMC) strategy. Given a set of videos under different camera views, the system firstly detects and tracks multiple vehicles within each single camera. After that, it re-identifies these tracklets among multiple camera views with a model trained by aggregation loss. Finally, the system synchronizes tracking results across multiple camera views and accelerate the procedure through using geometry information.

2. Related Research

2.1. Object Detection

Object Detection is one of the most popular tasks in the realm of computer vision. Generally, there are Image Object Detection (IOD) and Video Object Detetion (VOD). Given an image I as input, an Image Object Detection (IOD) model uses a feature network N_{feat} to extract features as $f = N_{feat}(I)$. Based on the extracted features, it introduces a sub-network N_{det} for detection, which generates a label y and a bounding box b. Presently, we could divide the current object detection works into two types, single stage methods and two stages methods. Single stage methods, such as SSD[14] and YOLO[20] does not implement Region Proposal Network (RPN)[21] for region of intereset (ROI) selection, thus they are much faster and the locations of the objects are generated by a single CNN network. Two stage methods, like Faster RCNN[21] and Mask RCNN[6] resort to RPN network for ROI generation and usually give out more accurate bounding boxes.

Figure 2: Vehicles usually get overlapped by surroundings under a surveillance perspective. Meanwhile the scale of vehicles varies a lot in this data set which also makes accurate detection a challenging task.

Meanwhile, to solve the blur, occlusion and out of focus in steaming videos, there are also video object detection (VOD) models which resort to information from contextual frames. For example, Qian *et al*. propose a video object detection model with adaptive feature aggregation weight. aggregation[19].

2.2. Tracking

Figure 3: Different from SOT, tracking ID in MOT usually change due to overlapping. Thus, rules, filters or smooth functions is needed.

Tracking is another challenging computer vision task and there are many successful works in this realm [29, 31, 15, 3]. Generally, it can be split into three categories, single object tracking (SOT), multiple object tracking (MOT)

and multi-target cross-camera tracking (MTMC). The difference between SOT and MOT is that, in MOT, there are multiple objects within the target scene. Thus, the model needs to tackle many difficult circumstances, for example, objects get occluded and objects with similar appearance. Most of existing methods regard it as a template-matching problem with various object features. For example, Sort [2] and Deep-Sort [27] both combine the Kalman filter and Hungarian algorithm. The difference is that Deep-Sort not only uses bounding box information, but resorts to features extracted by a deep CNN network as well. When it comes to vehicle MOT, considering the location of traffic cameras, the appearance feature and bounding box size of a specific vehicle vary a lot under different views. To solve this problem, BoxCars [22] uses 3D bounding box to get more accurate location. Meanwhile, when objects get overlapped, their tracking ids usually get switched. To solve this, Maksai*et al*. proposes a new training mode to improve this problem [17]. Compared with previous discussed two sub-tasks, MTMC, is more challenging in that it needs to track multiple objects appearing in multiple cameras. We will especially discussed how did we tackle this task in the following sections.

2.3. Object Re-identification

Object Re-identification (ReID) is usually regarded as a retrieval task which aims at matching targets in different scenes. Previously, many works focus on person ReID, for example, Hermans *et al*. implement triplet loss with hard mining in person ReID [8]. These years, vehicle Re-identification (vehicle ReID) has received more and more attention thanks to the grant application in city management and intelligent traffic. Usually, to enhance accuracy, some methods will resort to multiple formats of information like license plate, vehicle model, etc. For example, Liu *et al*. propose a two-branch retrieval pipeline to extract both vehicle model and instance differences[13]. And Liu *et al*. implement a method that fuses sift descriptors and other features to generate high level semantic information like number of doors and the number of seats. Meanwhile, in a later work, Liu *et al*. make the usage of license plate [16]. However, given the limitation of data usage, we can not resort to these attributes. So our model only relies on tracking id and camera id. Specific information will be provided in next section.

3. Methodology

3.1. Vehicle Detection

To perform multi-camera vehicle tracking, the first step is to reliably detect vehicles within a single image. We adopt the state-of-the-art instance segmentation network Mask R-CNN[6] as our frame-level vehicle detection model. It utilizes a powerful convolutional neural network as its backbone for feature extraction, such as ResNext-101[28] with feature pyramid network[11]. With region proposal network and region of interest (RoI) alignment, it produces feature representations for candidate detections. Through multiple output heads, we can get the object class, confidence score, bounding box, and segmentation mask of each detection.

Figure 4: In order to protect personal privacy, we cannot obtain license plate information from AI City Challenge 2020 Track 3 dataset. What's more, the resolution and viewing perspective make it difficult to detect the number of seats.

In the traffic scenerios, the objects we mainly focus on are vehicles. Here it is defined as the union of *Car*, *Bus*, *Truck* from the Microsoft COCO[12] dataset.

The original Mask R-CNN only performs non-maximum suppression (NMS)[5] within each class, which typically works fine. However, in our cases some vehicles could result in multiple detections, such as both *Car* and *Truck* for a pickup truck. Therefore, we additionally apply an inter-class non-maximum suppression(NMS) on the detections. All detections are sorted in the descending order according to their confidence scores. Then we select the detection one by one and skip if there exists a detection with intersection over union (IoU) of at least IoU_{nms}.

Samples of detected vehicles are shown in Figure 6.

3.2. Online Multi-target Single-camera Tracking

To track multiple targets within a single view, we follow the tracking-by-detection paradigm to associate frame-level detection results into tracklets. We utilize two state-of-the-art online multi-target tracking algorithms to associate detections into tracklets.

Deep SORT[27] is an online tracking algorithm proposed by Wojke *et al*. It incorporates a Kalman filter with a constant velocity model to estimate the location and speed of objects from noisy detections. One of its great advantage is including the deep visual features as association criterions, which is much more expressive than simple bounding boxes. To reduce computational complexity, we directly reuse the RoI features from the backbone of Mask R-CNN as the features for Deep SORT.

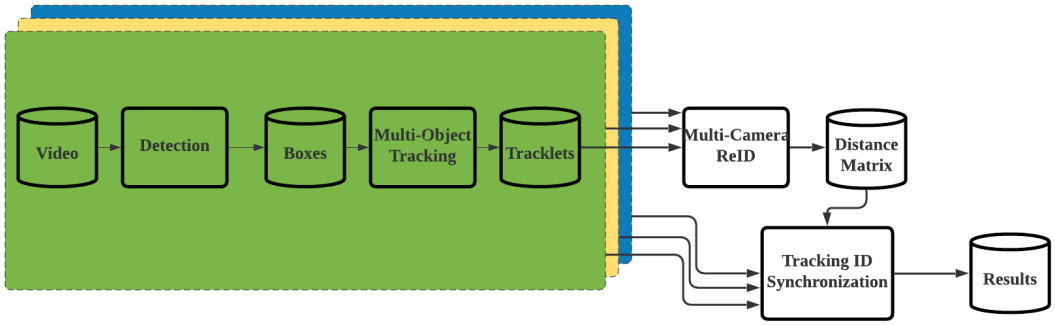

Figure 5: The pipeline of our MTMC model. In this figure, each colored box represents a single camera view.

Figure 6: Samples of detected vehicles in single cameras

Towards-Realtime-MOT[26] is a recent successor of tracking algorithms, which unifies detection and feature into a single model. As we are already doing so by reusing the Mask R-CNN feature, we only utilizes its association algorithm. Basically, new detections are assigned to existing tracklets based on feature similarity and compliance with spatial constraints. It first attempts to match all detections with previously confirmed tracklets based on feature similarity. A match would be rejected if they are not spatially adjacent. Then its second try is using all remaining detections to match all remaining confirmed tracklets based on bounding box intersection over union (IoU). The third round includes unconfirmed tracklets in the matching, which are typically tracklets of length 1. After all these matches, remaining detections will be recorded as new tracks.

3.3. Vehicle Re-identification

As is shown in Figure 7, $T_{B\times W\times H\times 3}$ is an input batch of tracklets, where $W \times H \times 3$ is the size of the image. It contains B images of V vehicle identities. We firstly use a deep CNN network N_{feat} to extract features $F_{B\times C}$ where C is the degree of output features. Then we train the network through an aggregation loss $loss_{agg}$ which aggregates cross entropy loss $loss_{xe}$ and triplet loss $loss_{tr}$ with hard mining. Meanwhile we use a margin loss to normalize and adjust the weight between $loss_{hp}$ and $loss_{hn}$. To better illustrate $loss_{agg}$, we need to first define hard negative and hard positive.

1. Hard negative defines the image of a vehicle which is quite similar to the images of other vehicles.

2. Hard positive means the image of a vehicle which is quite different from the other images of this vehicles.

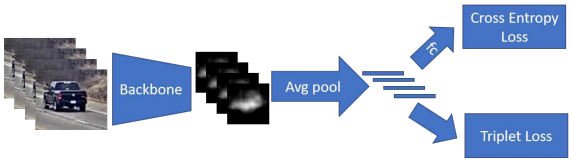

Figure 7: Training ReID model with aggregation loss

Thus, we have the assumption that if the model can successfully classify the hardest positive and hardest negative samples for each vehicle, it should gain a great capability to classify other easier samples correctly and overcome the variance of appearance caused by viewing perspective. $loss_{tr}$ is represented as:

$$loss_{tr} = \max(0, loss_{hp} + loss_{hn} + \lambda) \quad (1)$$

$$loss_{hp} = \sum_{i=1}^{V}\sum_{j=1}^{B_i}(\max_{k=1}^{B_i}(D(N_{feat}(I_j^i), N_{feat}(I_k^i)))) \quad (2)$$

$$loss_{hn} = -\sum_{i=1}^{V}\sum_{j=1}^{B_i}(\min_{\substack{p=1...v \\ q=1...B_p \\ p\neq i}}(D(N_{feat}(I_j^i), N_{feat}(I_q^p)))) \tag{3}$$

And our aggregation loss $loss_{agg}$ is represented as:

$$loss = \alpha \times loss_{tr} + \beta \times loss_{xe} \tag{4}$$

Where B_i represents the number of images belong to vehicle i. I_j^i represents the jth image of vehicle i. D represents the distance function. $loss_{hp}$ is the hardest positive loss and $loss_{np}$ is the hardest negative loss. So through the loss function above, we want the distance of hardest positive to be small and the distance of hardest negative to be large. For the distance function, we have tried multiple different distance like square euclidean and un-square euclidean, the specific results will be provided in next section.

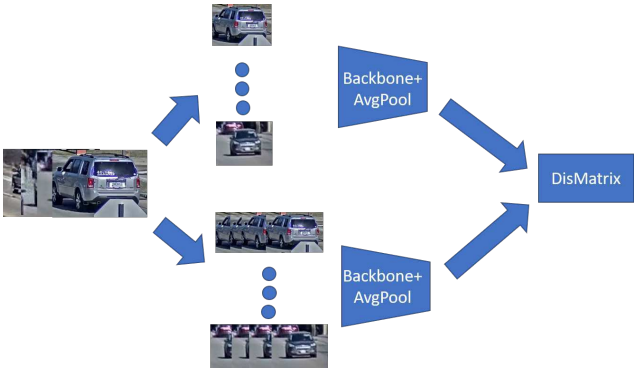

Figure 8: The pipeline of ReID model during inference

As is shown in Figure 8, when inference and given an input batch of tracking set T generated by MOT model, we firstly split it into query set $Q_{N\times W\times H\times 3}$ with N frames and gallery set $G_{M\times W\times H\times 3}$ with M frames. Specifically, set Q contains 1 frame from each track as the query for retrieval and the rest frames are split to set G. After that, we use a deep CNN network N_{feat} to extract features from both sets. Then, different from training split, we calculate the distance between each feature from set Q with each feature from set G, thus, we can get a distance matrix $Dis_{N\times M}$. Based on the distance matrix and previous output, we can generate final MTMC results.

3.4. Multi-target Cross-camera Tracking

We proposed an efficient fast MTMC strategy which accelerates the procedure and get more stable synchronization results through using geometry information. As is shown in Figure 5, given the distance matrix $Dis_{N\times M}$, we update previous MOT tracking result T under the following rules:

1. Sort the tracklets according to their camera id and only compare tracklets with those having adjacent camera ids.

2. Remove the tracklet T_i if the minimum distance between its query feature and all other gallery features of tracklets under different cameras is larger than $maxThresh$.

3. Update the tracking ids of T_i and T_j to be the same if the distance between q_i and T_i and q_j and T_j both less than $syncThresh$.

Where T_i represents the galleries of tracklets with tracking id i and q_i represents the query of tracklets with tracking id i. Meanwhile, here we will illustrate the reason of not comparing tracklets under a specific view with tracklets from all other views. According to the location of each camera and their surrounding roads, if the vehicle turns left or right at the intersections, it can not return to the main street in several minutes through calculation and each of the provided videos only lasts several minutes. Thus, we only need to calculate distance sequentially along the camera id. Usually our thoughts are constrained by algorithms and neglect the usage of logic deduction. In fact, such thoughts can be applied to many other tasks. For example, combination of deep learning models and Internet Information Retrieval for fake news detection. If we can find different versions of news or news reported several years before, then it has high probability to be a fake news.

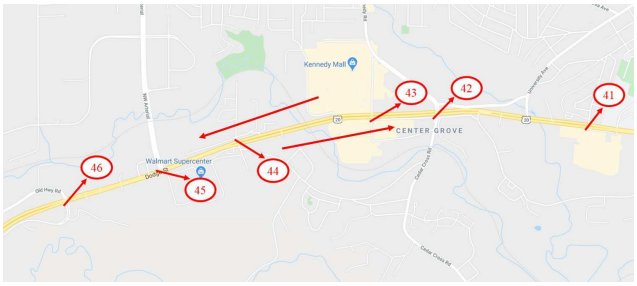

Figure 9: The location of each camera and their surrounding roads.

4. Experiments

4.1. Datasets and Evaluation Metric

Besides COCO[12] and ImageNet[4] which are used to pre-train our backbone networks of detection model and ReID model, we only use the training and validation data of AI city 2020 Track 3.

Figure 10: Visualization results of MTMC on five different kinds of vehicles. Each image represents a tracklet and each line represents these tracks from different camera views are classified by MTMC to be from the same vehicle. Yellow box represents flase negative and red box represents false positive.

Datasets In total, there are 6 scenes and 46 cameras in the AI City 2020 challenge dataset (Track 3). There are 3 scenes with 36 cameras in training set, 2 scenes with 23 cameras in validation set. Among the 23 cameras in validation set, 19 cameras also appear in training set. The last scene is used as test set and it contains 6 cameras and none of them appears in training set or validation set. What's more, the scales and resolutions of this test set is different from those of training set and validation set. The new scenes with different scale of objects in test set makes the competition much more challenging.

Evaluation Metric IDF_1 is the official evaluation matrix on leader board. Suppose IDTP represents number of true positive IDs, IDTN represents number of true negative IDs, IDFP represents number of false positive IDs and IDFN represents number of false negative IDs, IDF_1 can be represented as:

$$IDF_1 = \frac{2IDTP}{2IDTP + IDFP + IDFN} \tag{5}$$

4.2. Vehicle Detection

In this part, we directly import the pre-trained Mask-RCNN[6] from Detectron2 with weighted inter-class NMS as we used in Track 1 [30]. We tried several backbone networks and according to our experiments, res101 and res101x seems have similar performance on the official metric of validation set. For our final submissions on the leader board, we fix the settings of our detection model and only use res101 as the backbone.

Backbone Network	IDF1
res50	0.375
res101	0.388
res101x	0.381

Table 1: IDF1 results on validation set with different detection backbone networks

4.3. MOT Tracking

We tried both Deep-sort and Towards-Realtime-MOT models and submit both results with other modules fixed to the official leader board, the results are provided in Table 2. We find that although the tracklets generated by Mort have much higher Recall, some of the tracklets seem make the whole system confused and lead to worse IDF1. Since IDF1 is the official metric on the leader board, we select Deep-sort as our tracking model and fix it throughout our submissions.

Backbone	Loss	Tracking	IDF1	Recall
res50	$loss_{agg}$	Deep Sort	0.3705	0.3259
res50	$loss_{agg}$	Mort	0.3550	0.4461

Table 2: Results on official leader board with different tracking model

4.4. Vehicle Re-identification

We prepare the training set and validation set based on the provided annotations. In the experiments, we select S01, S03, S04 and S05 as our training set and S02 as our validation set. For each tracklet with a specific tracking id, we select one frame as the query and the rest as galleries for training and testing. Totally we get 666 vehicles, 521 of them appear in training set and the rest 145 appear in validation set. In sum, we extract 3028 queries, 206059 galleries in training set and 20494 galleries in validation set. We select Adagrad as optimizer and train our model with three different loss functions, $loss_{agg}$, $loss_{xe}$ and $loss_{tr}$, the results are provided in Table 3. Apparently, model trained with $loss_{agg}$ shows the best performance with both backbone networks. Meanwhile, models with res101 as backbone also have better performance then those use res50.

Backbone	Loss	Rank-1	Rank-5	Rank-10
res50	$loss_{agg}$	44.7	56.2	63.6
res50	$loss_{xe}$	36.2	53.1	69.3
res50	$loss_{tr}$	10.4	13.1	15.8
res101	$loss_{agg}$	63.1	75.8	81.6
res101	$loss_{xe}$	24.9	36.2	42.9
res101	$loss_{tr}$	54.2	71.3	76.7

Table 3: Comparison of results of multi-camera ReID models on inner validation set.

4.5. Multi-target Cross-camera Tracking

In the experiments, we firstly tried several data augmentation methods, like image flipping, random-erase and color-augmentation. According to the results provided in Table 4, these data augmentation methods does not have great influence on the final official metric. Meanwhile, it greatly increase the time of training. Considering the limit of our GPU resource, we did not apply these data augmentation methods in our later experiments. We then tune the parameters of our system on the inner validation set. In Table 5, we report all the values of the tuned parameters in our system.

Data Augmentation	IDF1
None	0.4161
flip	0.4171
flip+erase+coloraug	0.4195

Table 4: Results comparison on official leader board with and without data augmentation

Parameter Name	Parameter Value
α	1
β	1
$maxThresh$	100
$syncThresh$	2
λ	0.3

Table 5: Tuned parameters set for our submission system on the official leader board

5. Conclusion

In this paper, we propose an efficient multi-camera vehicle tracking system which is accurate and easy to train without using supplementary data set. In the detection and tracking part, weighted inter-class non-maximum suppression and association algorithm are implemented to generate more accurate bounding boxes. In the Re-ID part, aggregation loss is used for training to overcome the appearance variance caused by different viewing perspectives. Finally, given the tracklets and distance matrix, we adopt fast multi-target cross-camera tracking strategies to generate final results. Our model ranks the first with score 0.4585 and outperforms other teams by a large margin. Meanwhile, it is easy to be applied to real world large scale intelligent traffic and city management applications and be upgraded with future models.

Rank	Team ID	Score
1	**Ours**	**0.4585**
2	11	0.4400
3	63	0.3483
4	111	0.3411
5	72	0.1245
6	75	0.0620
7	30	0.0452
8	31	0.0387

Table 6: Official results of City-Scale Multi-Camera Vehicle Tracking on the leader board of AI City Challenge 2020.

6. Acknowledgement

This research is supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00340. This project is funded in part by Carnegie Mellon University's Mobility21 National University Transportation Center, which is sponsored by the US Department of Transportation.

References

[1] Pedro Antonio Marin-Reyes, Andrea Palazzi, Luca Bergamini, Simone Calderara, Javier Lorenzo-Navarro, and Rita Cucchiara. Unsupervised vehicle re-identification using triplet networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 166–171, 2018. 2

[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016. 3

[3] Xiaojun Chang, Wenhe Liu, Po-Yao Huang, Changlin Li, Fengda Zhu, Mingfei Han, Mingjie Li, Mengyuan Ma, Siyi Hu, Guoliang Kang, et al. Mmvg-inf-etrol@ trecvid 2019: Activities in extended video. 2

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[5] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009. 3

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2, 3, 6

[7] Zhiqun He, Yu Lei, Shuai Bai, and Wei Wu. Multi-camera vehicle tracking with powerful visual features and spatial-temporal cue. In *Proc. CVPR Workshops*, pages 203–212, 2019. 2

[8] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017. 3

[9] Yunzhong Hou, Heming Du, and Liang Zheng. A locality aware city-scale multi-camera vehicle tracking system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 167–174, 2019. 2

[10] 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 *AI City Challenge Workshop, IEEE/CVF Computer Vision and Pattern Recognition (CVPR) Conference, Long Beach, California*, 2019. 2

[11] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 3

[12] 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. 3, 5

[13] Hongye Liu, Yonghong Tian, Yaowei Yang, Lu Pang, and Tiejun Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2167–2175, 2016. 3

[14] 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. 2

[15] Wenhe Liu, Guoliang Kang, Po-Yao Huang, Xiaojun Chang, Lijun Yu, Yijun Qian, Junwei Liang, Liangke Gui, Jing Wen, Peng Chen, and Alexander G. Hauptmann. Argus: Efficient activity detection system for extended video analysis. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision Workshops*, pages 126–133, 2020. 2

[16] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. Provid: Progressive and multimodal vehicle reidentification for large-scale urban surveillance. *IEEE Transactions on Multimedia*, 20(3):645–658, 2017. 3

[17] Andrii Maksai and Pascal Fua. Eliminating exposure bias and metric mismatch in multiple object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4639–4648, 2019. 3

[18] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019. 2

[19] Yijun Qian, Lijun Yu, Wenhe Liu, Guoliang Kang, and Alexander G. Hauptmann. Adaptive feature aggregation for video object detection. In *The IEEE Winter Conference on Applications of Computer Vision (WACV) Workshops*, March 2020. 2

[20] 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. 2

[21] 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. 2

[22] Jakub Sochor, Jakub Špaňhel, and Adam Herout. Boxcars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance. *IEEE transactions on intelligent transportation systems*, 20(1):97–108, 2018. 3

[23] Jakub Španhel, Vojtech Bartl, Roman Juránek, and Adam Herout. Vehicle re-identifiation and multi-camera tracking in challenging city-scale environment. In *Proc. CVPR Workshops*, 2019. 2

[24] Xiao Tan, Zhigang Wang, Minyue Jiang, Xipeng Yang, Jian Wang, Yuan Gao, Xiangbo Su, Xiaoqing Ye, Yuchen Yuan, Dongliang He, et al. Multi-camera vehicle tracking and re-identification based on visual and spatial-temporal features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 275–284, 2019. 2

[25] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David Anastasiu, and Jenq-Neng Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[26] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. *arXiv preprint arXiv:1909.12605*, 2019. 4

[27] 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. 3

[28] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 3

[29] Lijun Yu, Peng Chen, Wenhe Liu, Guoliang Kang, and Alexander G Hauptmann. Training-free monocular 3d event

detection system for traffic surveillance. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 3838–3843. IEEE, 2019. 2

[30] Lijun Yu, Qianyu Feng, Yijun Qian, Wenhe Liu, and Alexander G. Hauptmann. Zero-virus: Zero-shot vehicle route understanding system for intelligent transportation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2020. 6

[31] Lijun Yu, Dawei Zhang, Xiangqun Chen, and Alexander Hauptmann. Traffic danger recognition with surveillance cameras without training data. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018. 2