Multi-Target Multi-Camera Vehicle Tracking for City-Scale Traffic Management

Kyujin Shim* Sungjoon Yoon* Kangwook Ko Changick Kim

School of Electrical Engineering, Korea Advanced Institute of Science and Technology
Daejeon, Republic of Korea

{kjshim1028, sungjoony, kokangook623, changick}@kaist.ac.kr

Abstract

Multi-target multi-camera (MTMC) tracking is one of the important fields in computer vision, where multiple objects are tracked across multiple cameras. MTMC tracking can be applied to various tasks such as crowd analysis, city-scale traffic management, and transportation systems analysis for intelligent city planning. However, it is challenging due to the large variety of conditions of each camera, such as perspective and illumination. Furthermore, MTMC tracking for vehicles is more problematic because of the relatively large inter-class similarity and intra-class variability. In this paper, we tackle the MTMC tracking problem for vehicles by dividing it into three main steps: (i) vehicle detection and feature extraction, (ii) multi-target single-camera tracking using the appearance feature of each vehicle, and (iii) multi-camera association of local trajectories from each camera. Our method shows comparable results with other highly-ranked methods in AI City Challenge 2021 and outperforms a recent MTMC tracking method that ranked first place in AI City Challenge 2020.

1. Introduction

Multi-target multi-camera (MTMC) tracking aims to track multiple objects of interest across multiple cameras. It has various range of applications such as crowd analysis [5, 16] and city-scale traffic management [18]. In particular, for city-scale traffic management [18], MTMC tracking can be used for city-scale vehicle tracking, which can play a critical role in traffic analysis and management. MTMC tracking is a difficult problem since the perspective, illumination, and video quality of each camera can largely vary, as shown in Figure 1. Also, they often do not share any overlapping areas. It means that even for the same object, its appearance can be widely different for each camera. Furthermore, vehicle tracking is even more problematic because vehicles tend to have relatively large inter-class similarity and intra-class variability [3].

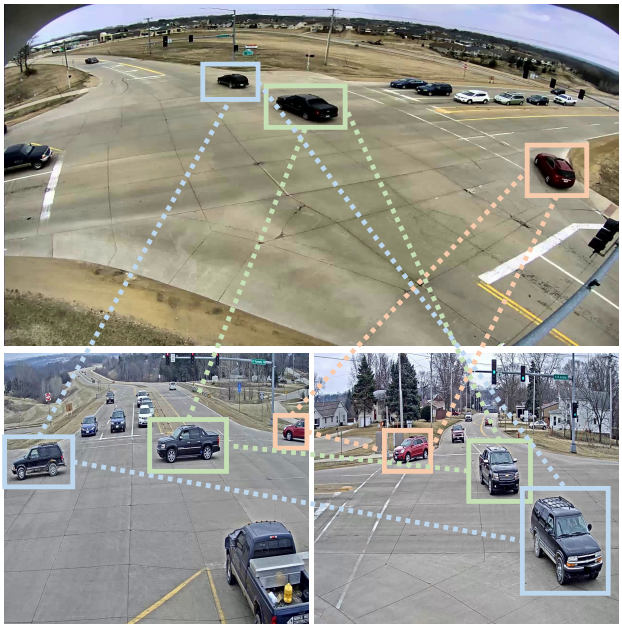


Figure 1. Multi-target multi-camera tracking aims to track multiple objects in multiple cameras. The appearance largely varies due to the different conditions, such as perspective, viewpoint, illumination, and video quality, of each camera. It is particularly more difficult for vehicle tracking because of their large inter-class similarity and intra-class variability.

Most methods divide the problem into the following sub-problems: object detection, multi-target single-camera tracking, and connecting the local trajectories through the cameras using appearance features and spatial-temporal information [5, 16]. As illustrated in Figure 2, our method also follows a similar pipeline. From Mask R-CNN [4] detection results, we extract appearance features. Then we measure the similarity between the detected objects based on the extracted features and associate the detected boxes in the same identity and form trajectories through the Hun-

*Both authors contributed equally to this work.

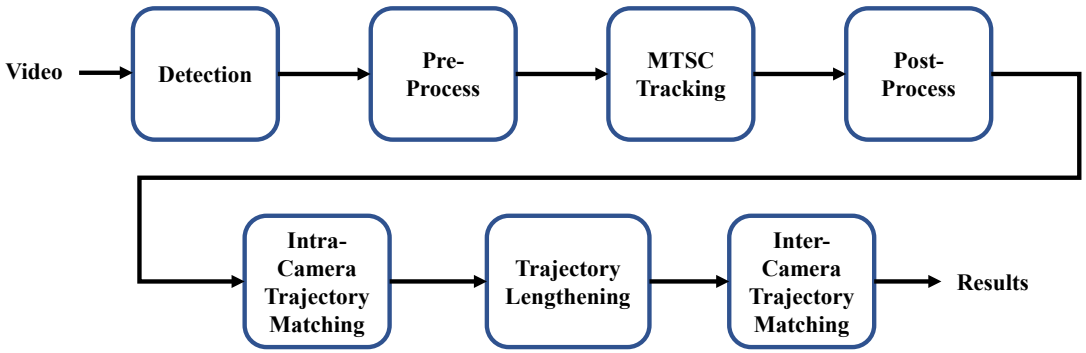


Figure 2. The overall pipeline of our MTMC tracking model.

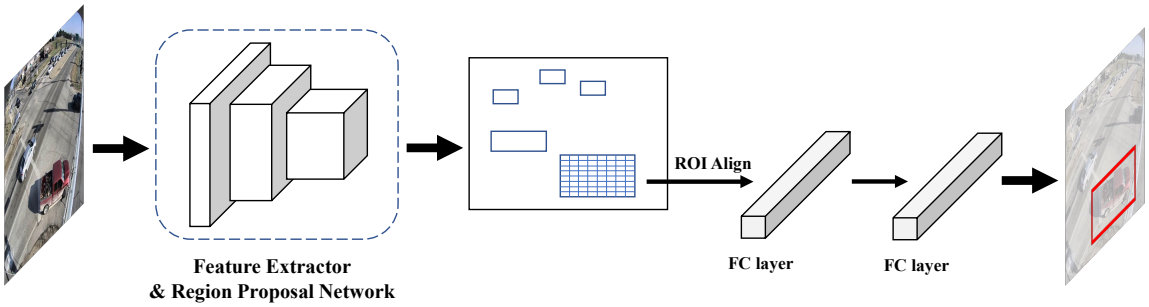


Figure 3. Structure of the Mask R-CNN [4] network.

garian algorithm [12]. Finally, we connect the trajectories across the cameras with the Hungarian algorithm [12] and their pairwise similarity calculated with the appearance features.

2. Related Works

2.1. Object Detection

Object detection is one of the classic and fundamental computer vision tasks, which aims to detect objects in images and videos. More precisely, given an input image, object detection aims to produce bounding boxes and classify each object. Two-stage methods, which handle region proposal and classification separately, have been proposed to deal with both localization and classification. For example, Mask R-CNN [4] employs Region Proposal Network (RPN) and an additional regressor to suggest bounding boxes, and it classifies the objects inside the proposals. On the other hand, one-stage detectors such as YOLO [14], SSD [9], and RetinaNet [8] handle two tasks simultaneously through a single model, making them much simpler and faster than other two-stage detectors.

2.2. Multi-Target Single-Camera Tracking

Multi-target single-camera (MTSC) Tracking, also known as multi-object tracking (MOT), aims to estimate objects' trajectories across the frames in each video. Most of the methods follow the *tracking-by-detection paradigm*, which divides MOT into two separate tasks. First, it detects objects independently for each frame. Then, the detected instances of the same identity are linked to create a single trajectory. In this paradigm, most methods mainly focus on association problems. For example, Xu *et al.* [21] have presented Spatial-Temporal Relation Networks to estimate the similarity scores between existing trajectories and objects in the frame, and they have applied the Hungarian algorithm [12] to match the bipartite graph constructed by the similarity scores.

More recently, Bergmann *et al.* [1] have suggested a single-stage multi-object tracker named Tracktor. It exploits a bounding box regressor of Faster-RCNN [15] to find bounding boxes of objects in the current frame from their bounding boxes in the previous frame. Other methods like Retina-Track [10], JDE [19], and FairMOT [22] are also recently proposed single-stage trackers, which detect objects and extract their appearance features at once. They have adopted various detectors and association algorithms to connect the detected objects using the extracted features and other spatial-temporal information such as distance and intersection over union (IoU).

2.3. Multi-Target Multi-Camera Tracking

Most methods attempt to solve MTMC tracking with the following pipeline: generating the trajectories of de-

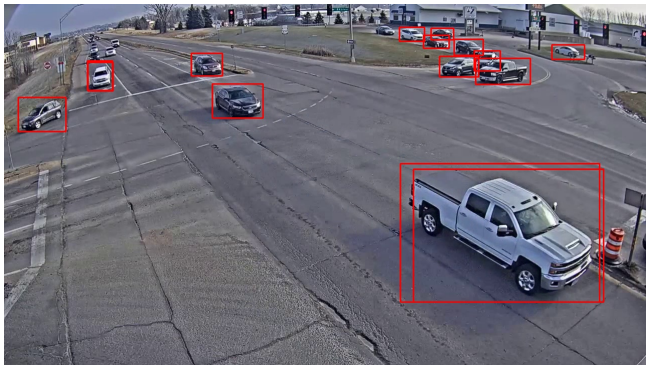
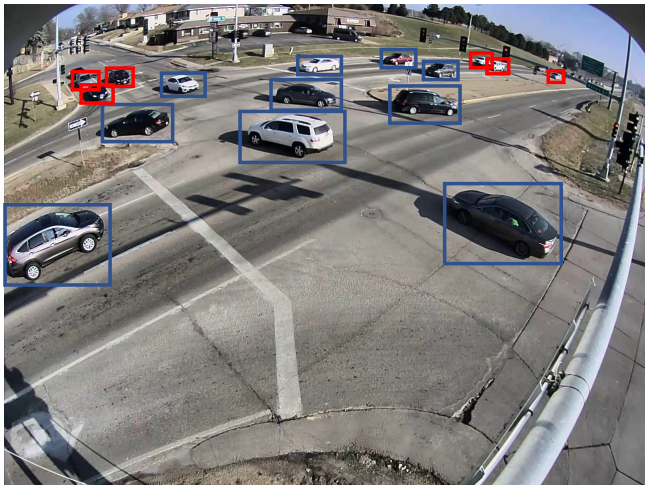


Figure 4. Example detection results with Mask R-CNN [4]. As shown in these examples, very small vehicles are also detected and sometimes more than one box is predicted for a single object.



(a) Pre-process 1

(b) Pre-process 2

Figure 5. There are two pre-processing steps to refine the detection results: (a) we discard object bounding boxes with an objective score lower than 0.2 or size smaller than 660 pixels; (b) If there are two bounding boxes with IoU greater than 0.5, they are regarded as duplicated boxes for the same object, and the box with the lower objectiveness score is discarded.

tected objects for each camera, then connecting those trajectories across multiple cameras to infer complete trajectories. For example, Ristani and Tomasi [16] have proposed the MTMC tracking method that uses adaptive weighted triplet loss to train a feature extraction network and correlation clustering to match the trajectories. Similarly, He *et al.* [5] have modified the trajectory matching problem to the tracklet-to-target assignment problem and used the Restricted Non-negative Matrix Factorization algorithm to solve the problem. For MTMC tracking on vehicles, the CityFlow dataset [18] has provided an important benchmark from city-scale traffic cameras. Facilitated by public datasets and challenges, Hsu *et al.* [6] and Qian *et al.* [13] have shown developments to track each vehicle on multi-camera.

3. Method

Our proposed framework is composed of three main steps: (i) detection and feature extraction, (ii) multi-target single-camera tracking, and (iii) multi-target multi-camera tracking. We explain each of the steps in detail in the following sections.

3.1. Detection

Our method follows the tracking-by-detection paradigm, similar to other state-of-the-art multi-target single camera (MTSC) tracking methods. To detect vehicles and find their bounding boxes, we use Mask R-CNN [4], a well-known object detection method illustrated in Figure 3. From its detection results shown in Figure 4, we pre-process before we apply our multi-target single-camera algorithm as shown in Figure 5. First, we delete detected boxes with an objectiveness score lower than 0.2 an area smaller than 660 pixels.

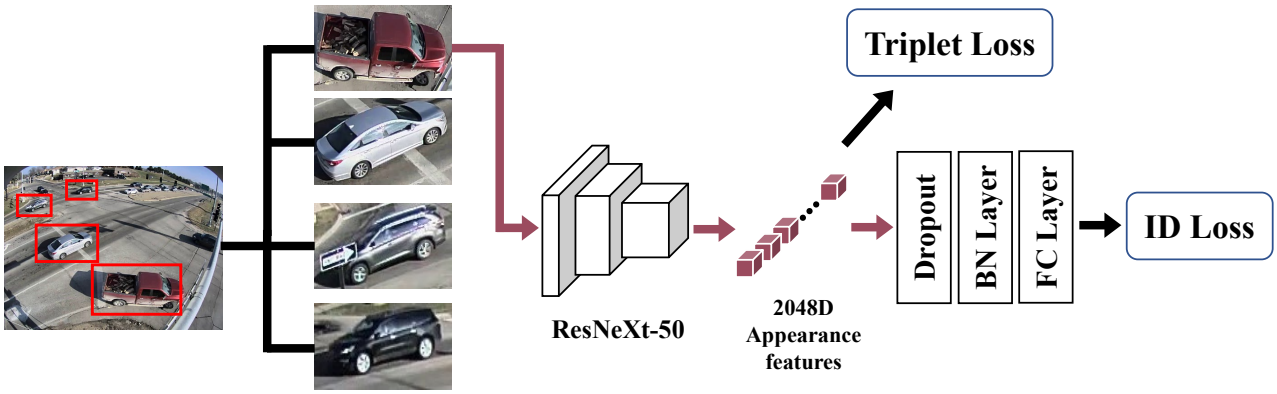


Figure 6. Each detected object embeds to appearance features of the 2048 dimension. With triplet loss, the feature distances between the same objects are minimized, and the distances between different objects are maximized. Also, the features are classified into corresponding object identities after the dropout, batch normalization, and fully connected layer.

Besides, we apply an inter-class non-maximum suppression (NMS) for each frame, which sorts the detection results in descending order with their objectiveness score and delete the detection box of the lowest score until boxes with IoU greater than 0.5 does not exist.

3.2. Feature Extractor

After detecting the vehicles, a feature extractor is required to extract an appearance feature for each detected object. These features are used during the association process of MTSC and MTMC tracking by comparing their pairwise feature distance. For the feature extractor, we adopts ResNeXt-50 [20] as the backbone network and attach the dropout [17], batch normalization [7], and fully connected layer similar to BNNeck [11], as shown in Figure 6. The ResNext-50 [20] network of our feature extractor is initialized using ImageNet [2] pre-trained weight. We train the extractor using SGD with Nesterov momentum for 20 epochs after five epochs of warm-up. In the warm-up stage, the learning rate started at 0.002 and increased by 0.002 for each epoch. Then, the learning rate started at 0.01 and multiplied by 0.1 at 10 and 15 epochs. Every input patch is resized as 320x320 using the letterbox algorithm to maintain the aspect ratio of objects, which is important appearance information for vehicles. During the training, we augment the data by applying random horizontal flip, random color jittering. Also, we generate input patches from randomly adjusted ground truth boxes of objects, since the detection boxes used in the testing process are often bounded with a noisy margin. In particular, the width and height of ground truth boxes are randomly resized in a ratio of 0.8 to 2 and 0.8 to 1.2, respectively.

3.3. Multi-Target Single-Camera Tracking

To perform multi-target single-camera tracking, we connect trajectories tracked until frame $T-1$ and detected bounding boxes from frame T by comparing their appearance. We first calculate pairwise Euclidean distance between extracted appearance features of the last object boxes of the trajectories and newly detected boxes at frame T. Then, we apply the Hungarian algorithm and connect the trajectories with the boxes only if the feature distance is less than a threshold ψ_{mtsc} and their IoU is greater than 0.25. Note that we estimate the bounding box of the object at frame T for each trajectory by the Kalman filter and calculate the IoU between the newly detected boxes with the estimated box. We also connect the trajectories with the remaining bounding boxes if IoU between the estimated boxes and detected boxes is greater than 0.5.

If some boxes are not connected, we try to connect them with trajectories disconnected during the previous tracking process. This time, we also calculate pairwise Euclidean distances and connect them based on the Hungarian algorithm in a similar way. Boxes that still remained are then considered as newly starting trajectories. We finish to track the disconnected trajectories if they are not re-connected for more than ten frames, their estimated object position after three frames is out of the image, or their estimated box resolution after three frames is less than 660 pixels. Also, we delete disconnected trajectories if they were tracked in less than three frames. Finally, we update states of the not finished but still disconnected trajectories with the estimated object boxes at frame T.

After the tracking, we apply several more post-processing steps shown in Figure 7. First, we delete trajectories tracked in less than five frames. Then we classify whether this trajectory is stationary or not by comparing the IoU of its first object box and its last object

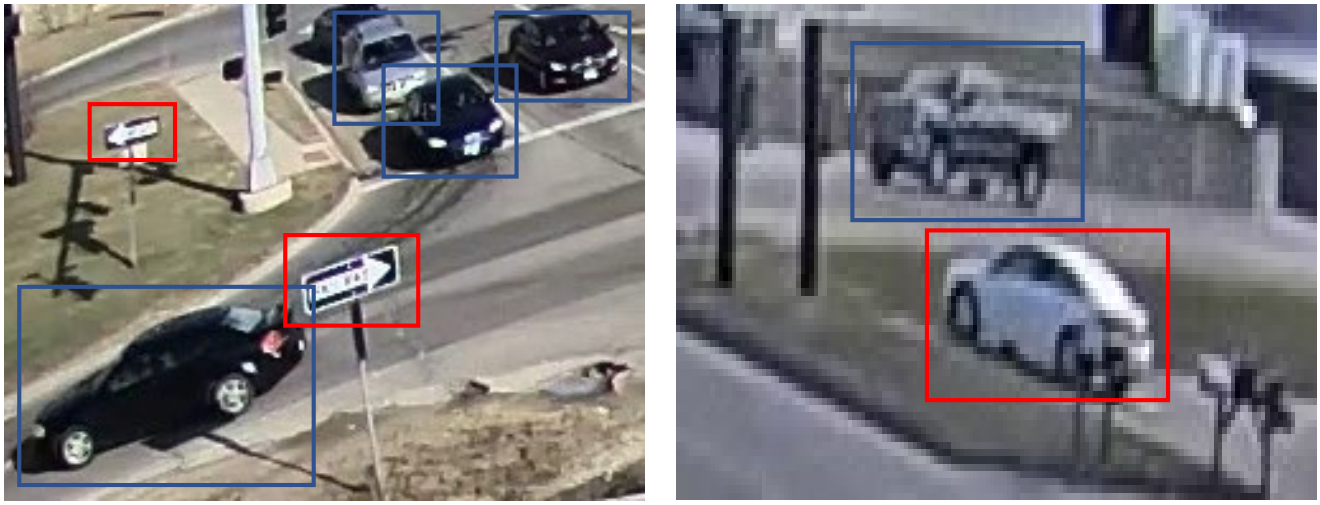


Figure 7. Post-processing is applied to remove trajectories that do not move. Stationary trajectories may track inappropriate objects, such as a traffic sign (left) or a parked vehicle (right). We remove the trajectory if its first and last box's IoU is greater than 0.5.

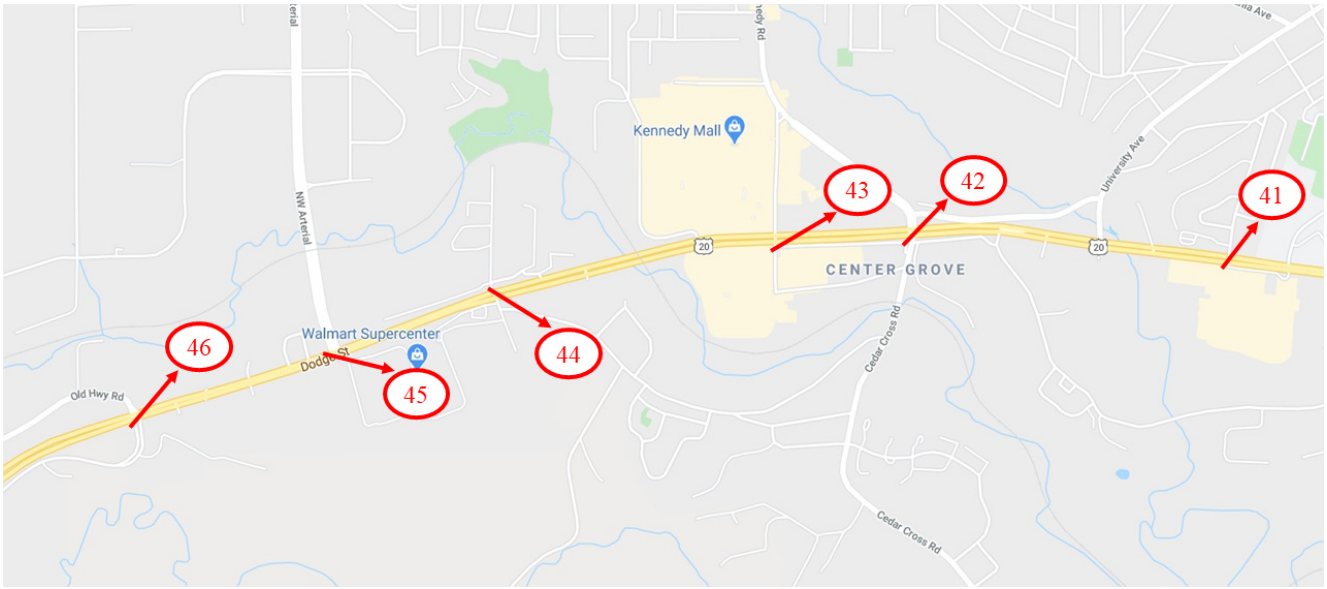


Figure 8. Camera locations of S06, which is for the test.

box. We compare the appearance feature of the stationary trajectories and connect them through the Hungarian algorithm if their feature distance is less than the threshold ψ_{mtsc}, the frame number difference is less than 100, and IoU is more than 0.5. Similarly, we re-connect stationary-moving, moving-stationary, and moving-moving trajectory pairs step-by-step. After that, we delete trajectories, which are still stationary. The next step of post-processing is finding previous and later object boxes with the Kalman filter for each trajectory and comparing their appearance feature with the trajectory. If their distance is less than the threshold ψ_{mtsc}, we append the newly find boxes to the trajectory and lengthen the trajectory. Finally, we only select and leave the trajectories, which pass through the roads connected to the other cameras.

3.4. Multi-Target Multi-Camera Tracking

Since the test scene does not include any overlapping areas and connected in order with a single highway, as shown in Figure 8, we match the trajectories from camera 41 to 46 one-by-one. First, we select five boxes with a top-5 objectiveness score for each trajectory and calculate pairwise euclidean feature distance and take the minimum value as the distance between two trajectories. Then we match them through the Hungarian algorithm if their distance value is less than a threshold ψ_{mtmc}. During the matching, we also

Detector	IDF1	IDP	IDR	Precision	Recall
DLA-34	0.5311	0.5284	0.5339	0.6387	0.6455
Mask R-CNN	0.5452	0.5736	0.5195	0.7112	0.6441

Table 1. Quantitative results of our proposed method with different detectors, DLA-34 [22] and Mask R-CNN [4].

care about the direction of the vehicles, whether it goes to the next camera or previous camera, and the minimum time difference, which represents the minimum time that must require to show up on the adjacent camera after passing the current camera. The minimum time difference is calculated based on the physical distance between the trajectories. If some trajectories are not matched, we stop to match that trajectory with trajectories from other cameras.

The threshold ψ_{mtsc} and ψ_{mtmc} are calculated using the validation data. We first generate image patches for each object with randomly adjusted ground truth boxes and extract their features. Then we randomly sample 25,000 object pairs for each case: the same camera and same identity, the same camera and different identity, the different camera and same identity, and the different camera and different identity. After that, we calculate the feature distance of each pair and select the threshold value that best distinguishes whether it is the same identity or different identity within the same camera and the threshold value that best distinguishes whether the identity is the same or different identity in different cameras. In this paper, ψ_{mtsc} and ψ_{mtmc} are set as 0.837 and 1.175, respectively.

4. Experiments

4.1. Dataset

We used the CityFlow benchmark [18] for our experiments. The dataset includes videos collected from 46 cameras across 16 intersections in a U.S. city. The whole videos are 215.03 minutes long and divided into six scenarios, three for training, two for validation, and one for testing. The dataset contains 313931 bounding boxes with 880 unique vehicle IDs, where only the vehicles featured on more than one camera are annotated. Each video has a resolution of at least 960p, and most of the videos have a frame rate of 10 FPS.

4.2. Quantitative Results

We have tested our algorithm with two different detectors, Mask R-CNN [4] and the advanced version of DLA-34 [22]. As shown in Table 1, using Mask R-CNN [4] shows better performance than using DLA-34 [22]. With this result, we have participated in AI City Challenge 2021 Track 3: City-Scale Multi-Camera Vehicle Tracking, where participants are expected to design an MTMC vehicle tracking algorithm that tracks vehicles across multiple cameras in different intersections across a city. Table 2 shows the leaderboard. Our team has made top-10 IDF1 score of 0.5452 among the total of 20 teams. Note that team #3 ranked 11th shows a 0.2974 IDF1 score, which is significantly lower than our result. Also, our method shows comparable performance with the result of team #27 ranked sixth. We further have compared with AI City Challenge 2020 Track 3 in Table 3. It shows much more advanced performance than the other top-ranked methods.

Rank	Team ID	IDF1	Rank	Team ID	IDF1
1	75	**0.8095**	11	3	0.2974
2	29	0.7787	12	45	0.2908
3	7	0.7651	13	110	0.2568
4	85	0.6910	14	60	0.2526
5	42	0.6238	15	82	0.2285
6	27	0.5763	16	67	0.2038
7	15	0.5654	17	11	0.1924
8	48	0.5534	18	123	0.1343
9	79	0.5458	19	61	0.1157
10	**112 (Ours)**	0.5452	20	129	0.0558

Table 2. AI City Challenge 2021 Track 3 results.

Rank	Team ID	IDF1	Rank	Team ID	IDF1
-	**Ours**	**0.5452**	4	111	0.3411
1	92	0.4616	5	72	0.1245
2	11	0.4400	6	75	0.0620
3	63	0.3483	7	30	0.0452

Table 3. Our result compared to AI City Challenge 2020 Track 3 methods.

4.3. Qualitative Results

In Figure 9, we show four example vehicles tracked across the six cameras in the test scene. Green boxes denote true positive results where each vehicle ID is accurately predicted. Red boxes denote false-negative results where a wrong ID is assigned to the vehicle. Crossed-out boxes indicate that the vehicle does not appear in the corresponding camera. As shown in Figure 1, vehicles #280 and #419 are accurately tracked across multiple cameras. Vehicles #282 and #443 are sometimes assigned wrong IDs due to different perspectives and illumination of the cameras.

5. Conclusion

Multi-target multi-camera vehicle tracking is an essential step towards intelligent city-scale traffic management. In this paper, we proposed an efficient MTMC method for vehicle tracking with the following pipeline. First, the appearance feature of each vehicle is extracted for each detection bounding box predicted by Mask R-CNN [4]. Second,

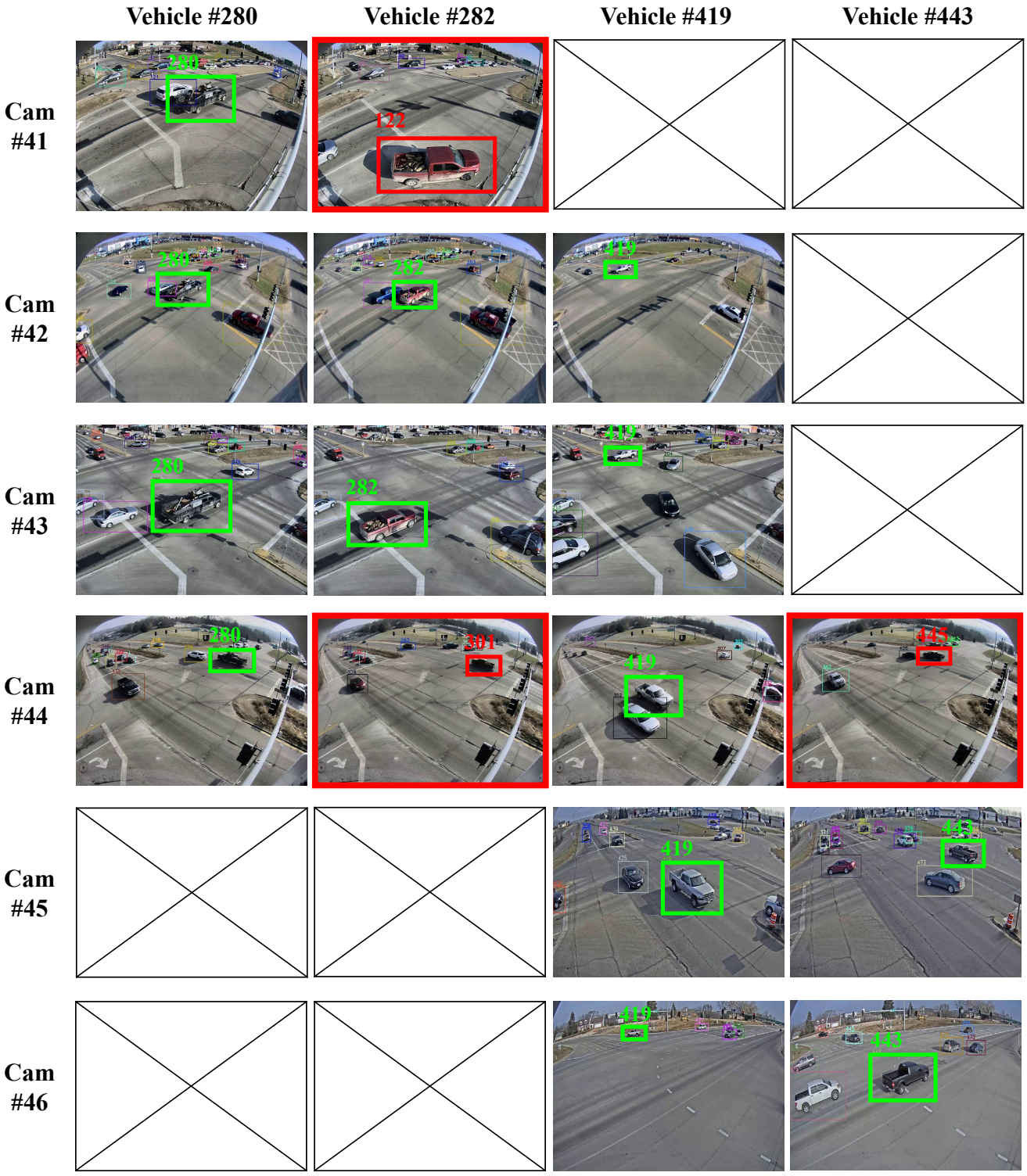


Figure 9. MTMC tracking results on the test scene with 6 cameras, #41, #42, #43, #44, #45 and #46, for four vehicles, #280, #282, #419 and #443. If the vehicle does not appear in the corresponding camera, it is denoted with a crossed-out box. True positive examples where the vehicle ID is accurately predicted are marked with a green box with its ID at the top. False negative examples where a wrong ID is assigned to the vehicle are marked with a red box with the incorrect ID at the top.

trajectories are generated for each vehicle in the same camera by association through feature-based similarity matching followed by the Hungarian algorithm. Lastly, the trajectories are associated across multiple cameras using their pairwise similarity calculated with appearance features and the Hungarian algorithm. We also designed simple preprocessing and post-processing strategies to enhance the performance of our MTMC tracking framework further. Our method outperforms a recent MTMC tracking method that ranked first place in AI City Challenge 2020 by 0.0867 in terms of IDF1.

6. Acknowledgement

This work was supported by LIG Nex1.Co.,Ltd, originally funded by DAPA and ADD(UC190031FD).

References

[1] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[3] Yan Em, Feng Gag, Yihang Lou, Shiqi Wang, Tiejun Huang, and Ling-Yu Duan. Incorporating intra-class variance to fine-grained visual recognition. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1452–1457. IEEE, 2017.

[4] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017.

[5] Yuhang He, Xing Wei, Xiaopeng Hong, Weiwei Shi, and Yihong Gong. Multi-target multi-camera tracking by tracklet-to-target assignment. *IEEE Transactions on Image Processing*, 29:5191–5205, 2020.

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

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of International Conference on International Conference on Machine Learning (ICML)*, 2015.

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

[9] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

[10] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] Hao Luo, Youzhi Gu, Xingyu Liao, Shenqi Lai, and Wei Jiang. Bag of tricks and a strong baseline for deep person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019.

[12] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

[13] Yijun Qian, Lijun Yu, Wenhe Liu, and Alexander G. Hauptmann. Electricity: An efficient multi-camera vehicle tracking system for intelligent city. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020.

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

[15] 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 (NeurIPS)*, 2015.

[16] Ergys Ristani and Carlo Tomasi. Features for multi-target multi-camera tracking and re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[17] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[18] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[19] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[20] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[21] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[22] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *arXiv preprint arXiv:2004.01888*, 2020.