Multi-Camera Vehicle Tracking System for AI City Challenge 2022

Fei Li* Zhen Wang* Ding Nie* Shiyi Zhang Xingqun Jiang Xingxing Zhao Peng Hu
BOE Technology Group, China
{lifei-cto,wangzhen-cto,nied,zhangshiyi,hupeng-hq}@boe.com.cn

Abstract

Multi-Target Multi-Camera tracking is a fundamental task for intelligent traffic systems. The track 1 of AI City Challenge 2022 aims at the city-scale multi-camera vehicle tracking task. In this paper we propose an accurate vehicle tracking system composed of 4 parts, including: (1) State-of-the-art detection and re-identification models for vehicle detection and feature extraction. (2) Single camera tracking, where we introduce augmented tracks prediction and multi-level association method on top of tracking-by-detection paradigm.(3) Zone-based singe-camera tracklet merging strategy. (4) Multi-camera spatial-temporal matching and clustering strategy. The proposed system achieves promising results and ranks the second place in Track 1 of the AI City Challenge 2022 with a IDF1 score of 0.8437.

1. Introduction

Multi-Target Multi-Camera (MTMC) [1, 9, 11, 12, 14, 21, 27, 29] tracking plays an important role in the extraction of actionable insights from the fast-expanding sensors around the world. Among its branches, city scale Multi-Camera Vehicle Tracking (MCVT) is attracting an increasing number of researchers. Its primary goal is to calculate trajectories of vehicles across multiple cameras, as shown in Fig. 1. Following a general approach, MCVT can be broken down to three sub-tasks, including Single-Camera Tracking (SCT), vehicle re-identification (Re-ID), and Multi-Camera Tracklets Matching (MCTM). As the initial condition, SCT can have significant impact on the correctness of successive steps. A few broken tracklets or ID switches in SCT are enough to cause massive chaos and chain reaction in multi-camera tracklet matching, forming an invisible bottleneck on recall and precision scores. We can observe the challenges seen in traditional MCVT with some actual scenarios:

1. Most tracking-by-detection SCT algorithms face the uncertainty caused by detection noises such as false positive or false negative results, which in turn affect the matching process and ultimately result in unstable or broken tracks.

2. Most tracking-by-detection SCT algorithms underestimate broken tracklets caused by long traffic jam, which is shown in Fig. 6. that afterwards

3. Most tracking-by-detection SCT algorithms extract impure features when vehicle is occluded by another object, such that the bounding box may contain many pixels from adjacent object.

4. In multi-camera tracklet matching, many vehicles with similar appearance cannot be distinguished, thus creating ID switches across camera.

Because of these problems, we design a system that labels and handles three types of detection results using confidence score and Intersection Over Union (IOU) ratios, which are low-quality-occluded, low-quality-non-occluded, and high-quality-non-occluded vehicles. Among these,

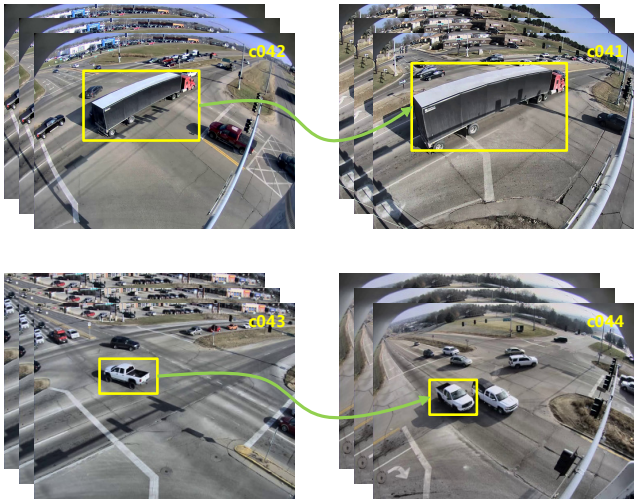


Figure 1. Tasks faced by MCVT, showing the trajectory of vehicles across multiple cameras.

*Equal contribution.

high-quality-non-occluded vehicles have the highest priority when matching, and are used for track initialization. Low-quality ones, on the other hand, have lower priorities when matching. Moreover, the Re-ID features of low-quality-occluded vehicles will be dropped out. By sorting them out, vehicle tracks can be more accurately performed, which enables a high recall rate, and filtered Re-ID features can make the process easier for tracklet matching later on.

For the missing pieces of a vehicle, we introduce two more SOT algorithms other than the Kalman Filter, so that stable vehicle trajectory can be synthesised even if one is in uncertainty. Then, the results will go through multi-level matching and clustering, within and across cameras. As for trajectory disruptions caused by heavy occlusion or appearance changes, we propose a zone-based tracklet merging strategy, piecing together most tracklet fragments within cameras.

As for multi-camera matching, we propose a direction based spatial-temporal strategy that significantly reduces the searching space and an aggregation strategy that solves edge cases like U-turns. The main contribution of this paper are summarized as follows:

- We propose multi-level detection handler, augmented tracks prediction method, and multi-level association to cope with broken tracklets and ID switches.
- We propose zone-based singe-camera tracklet merging strategy, which not only links tracklet fragments, but also enrich vehicle features in order to increase the recall rate.
- We propose spatial-temporal matching and aggregation strategy that significantly reduces the searching space and solves edge cases like U-turns.

2. Related Work

2.1. Vehicle Detection

Object detection is a traditional assignment of computer vision and image processing which deals with detecting instances of semantic objects of some certain classes, such as human, bikes or cars, in digital images and videos.

Object detection algorithms typically leverage machine learning and deep learning to output reasonable results. Commonly, object detection algorithms are classified into two categories, two-stage object detection and one-stage object detection.

Faster R-CNN is a representative two-stage object detection algorithm [5, 25]. Its architecture contains two networks, region proposal network (RPN) and object detection network. RPN firstly generates a series of anchors on the input feature map. Then, Region of interesting pooling (ROI) takes the output of RPN as input and generate a series of fixed-sized feature maps. Finally, softmax and bounding box regression layer flatten the feature maps and output the location of the object in the image.

Single Shot MultiBox Detector (SSD) [18]and You Only Look Once (YOLO) [4, 22–24, 30] are both famous one-stage object detection algorithms. SSD [18] predicts the object classes and locations with a series bounding boxes in various aspect ratio and size. YOLO tries to accomplish the real-time object detection task with anchors derived from Faster R-CNN. It fiendishly turns a object detection task into a classification task. Meanwhile, it balances prediction accuracy and inference performance.

2.2. Multiple Object Tracking

2.2.1 Single-Target Single-Camera Tracking

single object tracking (SOT) is one of the foremost assignments in computer vision that has numerous applications such as intelligent video analysis, robotics, autonomous vehicle tracking, and so on. It generally has three kinds of methods to approach the solutions.

The algorithms related with first method are based on correlation filter, such as Kernelized Correlation Filters (KCF) [10], Continuous Convolution Operators for Visual Tracking(C-COT) [8] and Efficient convolution operators(ECO) [7], which were commonly used on signal processing to describe the correlation or similarity of two signals. The general procedures to apply the correlation filter on SOT have three steps. Firstly, obtain the initialized target position based on first frame of stream. Then, extract the target position features as filter template like Histogram of Oriented Gradients (HOG). Finally, obtain the predicted target position of current frame by executing correlation operation based on the previous filter template.

The second class of algorithms is based on optical flow, such as MedianFlow [13]. This method extracts the target position features with Harris Corner or SIFT in the first frame. Then, it predicts the corresponding positions of the feature points for the following frames by optical flow algorithms. It finally gives the predicted target position based on the predicted feature-point positions. optical flow algorithms mainly focus on local features predictions which leads to robust results in occlusion challenge.

The last class of tracking algorithms is based on CNN Siamese Network, which is a kind of offline object tracking, such as siamFc [2], siamRPN [16], siamRPN++ [15], etc. The main principle is like the second method but replacing the Harris Corner or SIFT feature extractor with CNN. These methods have all achieved satisfactory results in the field of SOT.

2.2.2 Multi-Target Single-Camera Tracking

Multiple Object Tracking (MOT) is a challenging task. It not only involves the difficulties derived from SOT such as occlusion, illumination variation, object deformation, etc. but also concerns about the relative positions of each target to reduce ID switching possibilities.

Tracking-by-detection is the main paradigm to achieve the goal of MOT. Simple Online and Realtime Tracking (SORT) [3] is a typical bare-bone implementation of MOT framework. It predicts the object locations of the following frames by Kalman Filter and then computes the overlaps with the detection. In the end, it uses Hungarian algorithm to assign detection to tracklets. However, it always causes unsatisfied tracking results due to lacking feature extraction especially in crowded and fast motion scenes. To solve the previous problem, Simple online and realtime tracking with a deep association metric (DeepSort) [31] was proposed in 2017 and integrate appearance information to improve the performance of SORT. It effectively reduces the number of ID switches with acceptable computational complexity increasing.

In support of SORT and DeepSort, there are more works that joints the detection and tracking. For example, FairMOT [33] accomplishes integrating object detection and Re-ID in the same backbone to improve inference speed.

2.3. Multi-Camera Vehicle Tracking

Recently, MCVT [19] has become a fascinating research area due to the demands of the city-scale traffic management increasing. With the progression of multi-object tracking techniques, the vehicle Re-ID techniques and the integrated framework, MCVT can be settled in better results. Liu. Et al [17] proposes an integrated framework for MCVT guided by crossroad zones, which achieves the top performance in AI City Challenge 2021.

3. Method

3.1. Overview

The proposed MCVT system is shown in Fig. 2, which includes vehicle detection, Re-ID feature, feature dropout based multi-level detection handler, single-camera multi-level tracks and merging strategy, and multi-camera spatial-temporal matching and aggregation strategy.

3.2. Vehicle Detection

Vehicle detection is the first and essential step in MTMC tracking. As most of the MTMC tracking methods, we follow the tracking-by-detection paradigm, such as using the state-of-the-art network YOLOv5 [30] and more specifically the YOLOv5x6 model, which is pre-trained on the COCO dataset to detect vehicles. We tune our detection classes to only cars, trucks, and buses by setting the $classes$ parameter. The $agnostic$ parameter is used to perform non-maximum suppression (NMS) for all detected vehicles in the inference stage.

3.3. Vehicle Re-ID

Following existing Re-ID work [17], we use ResNet50-IBN-a, ResNet101-IBN-a and ResNeXt101-IBN-a models that are pre-trained on the CityFlow dataset to extract feature of vehicles, without introducing external data. Each Re-ID model outputs a 2048-dimensional feature vector, and the final feature of each detected car is the average output of the three models.

3.4. Single-Camera Vehicle Tracking

For single-camera vehicle tracker, we follow the general framework of Simple Online and Realtime Tracking (SORT) [3]. To deal with the limitations of SORT, we propose further improvements to the tracking method. First, relying on predictions from Kalman Filter often produces ID switches when direction of movement changes. Therefore, we utilize two more SOT, namely Efficient Convolution Operators (ECO) [6] and MedianFlow, and propose an augmented tracks prediction method. Next, inspired by DeepSort, we include vehicle appearance features, which then goes through a feature dropout filter and a multi-level matching process. Finally, in order to make sure the completeness of tracklets, we add another post-process for tracklet merging within a single camera.

3.4.1 Vehicle Track Prediction

To enhance the the limitation of Kalman Filter, we first include MedianFlow, which use the current location of a vehicle to take sample pixels, and then predict the next frame's location based on optical flows. MedianFlow can effectively locate vehicles that are occluded by another vehicle moving in parallel, thus becomes more occlusion resilient. Secondly, when a vehicle moves extremely fast or makes sharp turns, it usually undergoes dramatic appearance changes. In this case, MedianFlow might not work well, so we can adjust our prediction using ECO. Thereafter, every vehicle detection box will have a much better match in our multi-level association method, as shown in Fig. 3.

3.4.2 Multi-Level Detection Handler

For our method, vehicle Re-ID features plays an important role in both single and multiple camera vehicle tracking. Therefore, it is imperative to extract accurate features from Re-ID models, which act as the enabler for the rest of processes down the pipeline. We propose multi-level detection

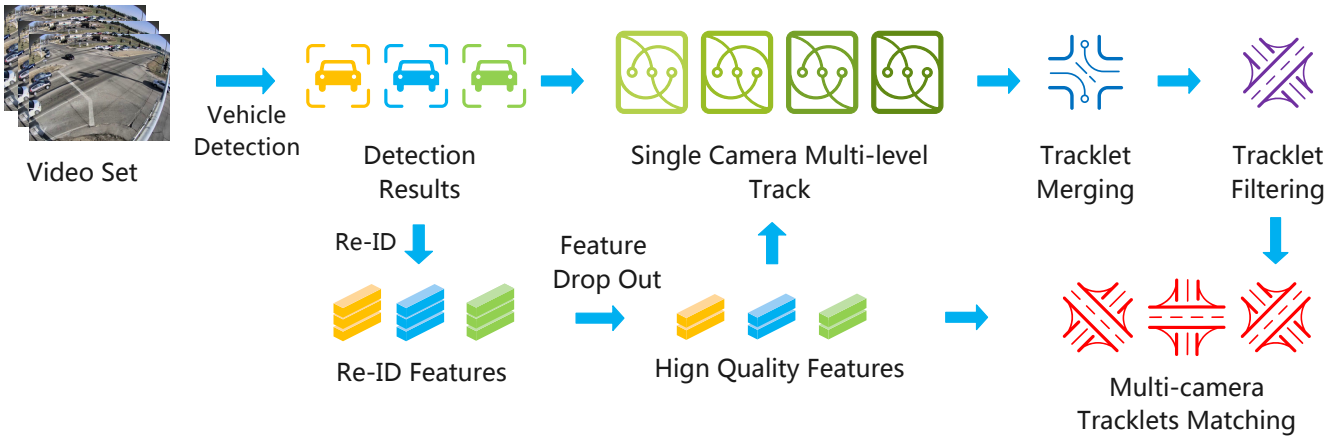
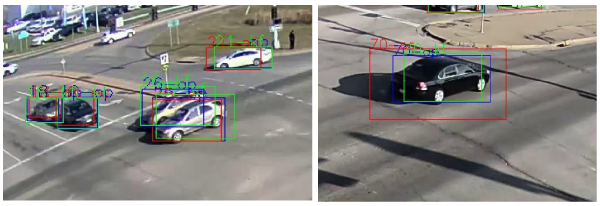


Figure 2. Pipeline of our MCVT system. The system first uses the detector to obtain all bounding boxes of vehicles from video set; then uses Re-ID models to extract features, with some Re-ID features being dropped out; then single-camera algorithms generate tracklets; finally, tracklets across cameras are synchronized by matching and clustering strategies.



(a) vehicles moving in parallel (b) vehicle with shadow

Figure 3. Augmented tracks prediction method in different scenarios, where red represents ECO, green represents Kalman Filter, and blue represents optical flow.

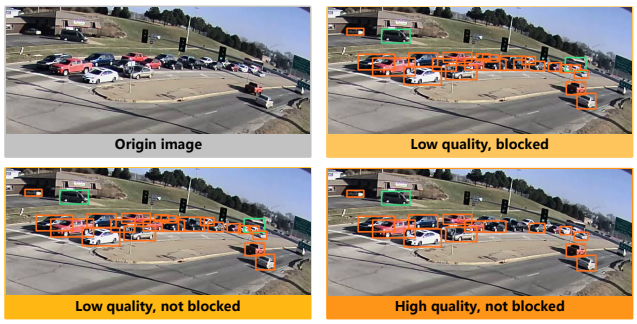


Figure 4. Multi-level detection results, where red has confidence of 0.1 and NMS-IOU of 0.45, green has confidence of 0.1 and NMS-IOU of 0.3, and blue has confidence of 0.3 and NMS-IOU of 0.3.

handler. We start our detection process with a relatively low confidence level of 0.1 and a relatively high NMS-IOU threshold of 0.45. After that we select from the result twice, first with confidence values of 0.1 and NMS-IOU of 0.3, then with confidence value of 0.3 and NMS-IOU of 0.3. At this point, we can split the result into three levels, as shown in Fig. 4.

It follows that the features from low-quality and blocked vehicles should be discarded, leaving only the boxes themselves for the track matching process to get high recalls [32], which is shown as the feature dropout filter in Fig. 5. On the other hand, vehicles that are not blocked will not only participate in tracker matching, but also add their features to the corresponding tracks.

3.4.3 Multi-Target Multi-level Association

To make sure the tracks predicted match adequately with detection results, our method includes four rounds of associations, which can be seen in Fig. 5.

1. select high quality and non-occluded vehicles from Fig. 4 and associate with tracks of age 1, generating the following matrix:

$$M = \alpha_1 * A + \beta_1 * B + \gamma_1 * C \tag{1}$$

where M represents the resulting cost matrix, $\alpha_1, \beta_1, \gamma_1$ represent corresponding weights, A represents feature cosine cost matrix, B represents cost matrix of IOU distance between MedianFlow boxes and detection boxes, and C represents cost matrix of IOU distance between ECO boxes and detection boxes.

2. pair the unmatched tracks and detection, resulting in the following matrix:

$$W = \alpha_2 * P + \beta_2 * B + \gamma_2 * C \tag{2}$$

where W is the new cost matrix, $\alpha_2, \beta_2, \gamma_2$ are corresponding weights, P is cost matrix of IOU distance between Kalman Filter boxes and detection boxes.

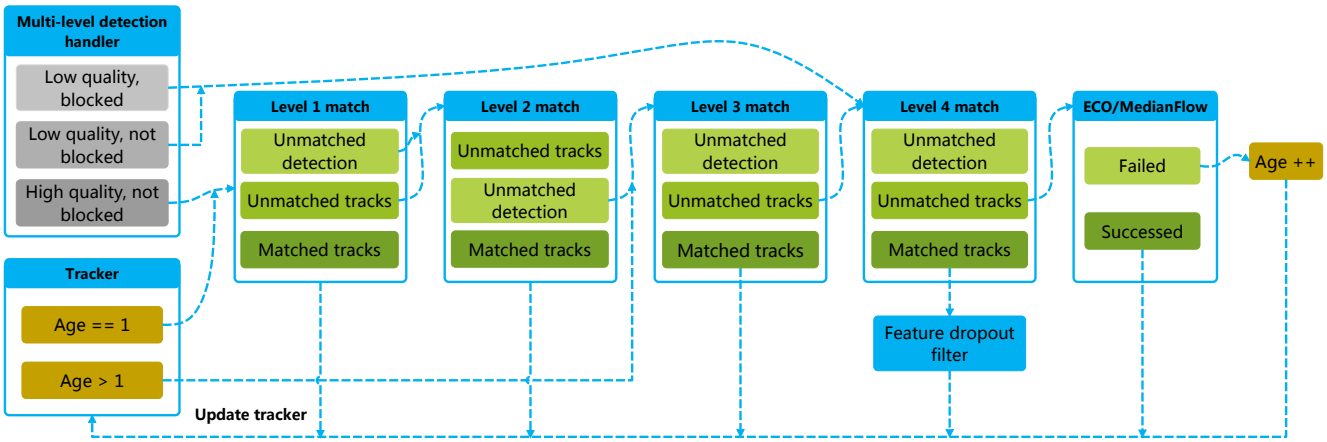


Figure 5. Multi-Target Multi-level Association.

3. associate the detection still unmatched with tracks with age greater than 1, yielding:

$$F = A \tag{3}$$

where A represents feature cosine cost matrix. This step is inspired by DeepSort, assuming that tracks with lower age should have higher priority when matching.

4. match the remaining low quality vehicles (occluded or not) with tracks of age 1, with similar matrix calculation to step 2. This step aims to save the boxes predicted by tracks from the low quality detection, in order to ensure the completeness of our tracklets.

3.4.4 Tracks Life-cycle

After four rounds of associations, if there are still some unmatched high quality detection boxes, then they are thought to be new, and new tracks will be initialized, including Kalman Filter model, ECO tracks, and sampling pixels for optical flow. For matched detection and tracks, the tracks will be updated accordingly. First the type of detection box is determined, and if the vehicle is occluded, then only the Kalman Filter is updated, else if the IOU distance between ECO prediction and matched detection is lower than a threshold, the track is reinitialized with the matched detection box rather than being updated, and optical flow sampling points as well as feature are updated accordingly. For tracks not matched with detection boxes, we try to salvage them by using predictions from MedianFlow or ECO, while updating the Kalman Filter model with those predictions to compensate for the missing parts.

3.4.5 Zone-Based Tracklet Merging

Although we make many attempts to capture fragments of every tracklet to make sure of its completeness, in real-world scenario, there are still split pieces made by unexpected objects such as the traffic light, which can be seen in Fig. 6. Due to these cases, we propose a zone-based tracklet merging method.

Tracklet Selection We divide crossroad images into 9 effective zones and 1 traffic zone, which can be determined by specific cases, as shown in Fig. 7. These 9 zones can be sort into starting zones (1, 3, 5), middle zone (10), and ending zones(2, 4, 6). Before merging, we pick some tracklets under the criteria:

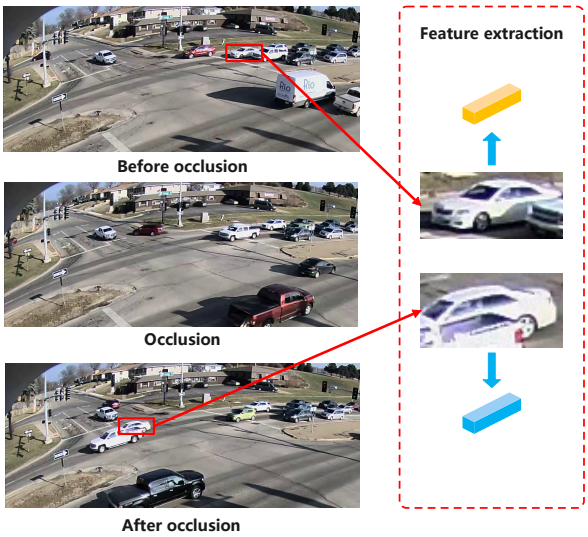
- tracklet that starts normally and end in either the same zone or middle zone.
- tracklet that starts in either middle zone or traffic zone.

These tracklets are thought to be abnormal and will become candidates for merging. Then, they will go through a filter to remove noises such as negligibly short ones (less or equal to 4 frames), stationary ones, and small pixel count ones. From there, these candidate will go to tracklet merging in the next step.

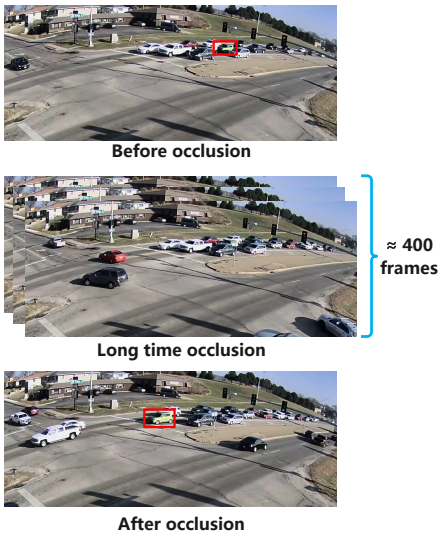
Tracklet Merging Considering the fact that a tracklet might have multiple fragments within one camera, this paper cope with abnormal tracklet fragments using hierarchical clustering. Assuming there are n fragmented tracklets, first the mean feature for each tracklet are calculated, then we have:

$$H_{n\times n} = \begin{bmatrix} cos(T_1, T_1) & \cdots & cos(T_1, T_n) \\ \vdots & \ddots & \vdots \\ cos(T_n, T_1) & \cdots & cos(T_n, T_n) \end{bmatrix} \tag{4}$$

Where T_i represents the tracklets fragments to be merged, and $H_{n\times n}$ represents the cost matrix of the tracklets. Because of one tracklet cannot merge with itself, we set the



(a) Vehicles stop for traffic light, and then moving under occlusion. After the reappearance of the vehicle, because of the part of feature occluded differ from that afterwards, the two features cannot be reliably matched.



(b) The track is deleted due to long time occlusion, when the vehicle finally shows up, a new ID will be assigned.

Figure 6. Example of unmatched tracks.

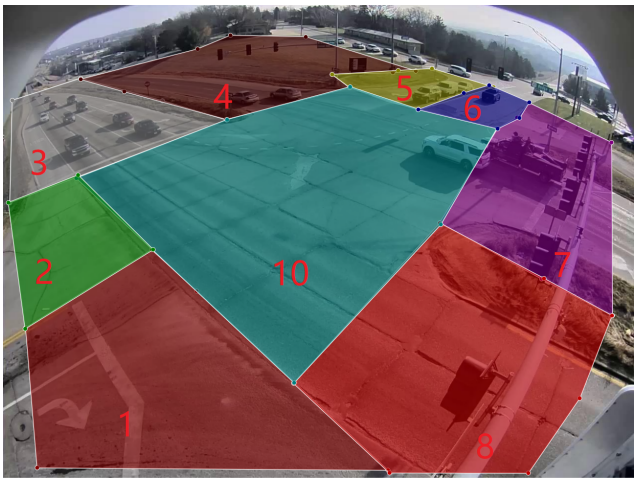


Figure 7. 9 Zones for a single camera.

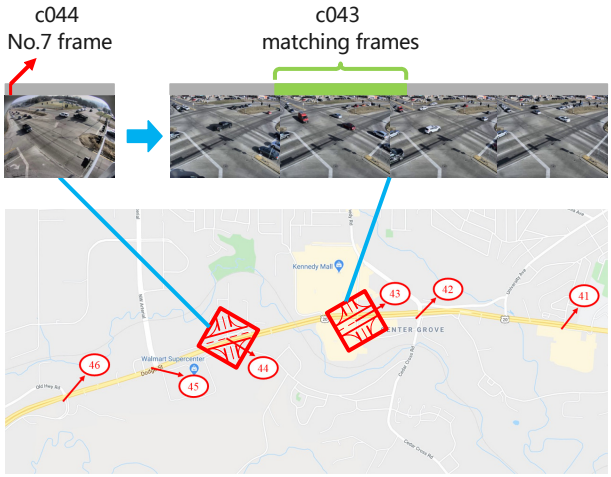


Figure 8. Spatial-temporal matching strategy.

diagonal values with 2, and then do the clustering to get tracklets under the same cluster:

1. sort tracklets by their starting frame in ascending order
2. check if two tracklets agree with space and time:
   (a) if the latter's starting frame is within a range regarding the former's ending frame.
   (b) if the distance between former's and latter's starting location is greater than the distance between former's starting location and ending location.
   (c) if the latter' starting direction is the direction pointed by the former.
   (d) if the latter's ending location is within the ending zones, if not, recursively look for one with ending zone until none matched.

Using the above techniques, tracklets fragment can be selected and merged under the same cluster, yielding more accurate tracklet results for single camera tracking.

3.5. Multi-Camera Tracklets Matching

In multi-camera matching, our approach consists of a selection step, shown in Fig. 8, an aggregation step, and a clustering step.

3.5.1 Tracklet Matching

The cosine similarity matrix used in multi-camera matching is similar to the one used in Sec. 3.4.5, except that now it is comparing tracklets across cameras. For n tracklets across two adjacent cameras, each tracklet with 2048-dimensional averaged features from all frames are compared with each other to form a n by n matrix S, filled by cosine similarity. The similarity matrix enables merging multiple tracklets across different cameras. However, as the number of tracklets increases, the search space within the matrix often become too large that results in many mismatches, which can become a bottleneck that limits the quality of clustering later on. In this case, filtering by edge cases does not seem to be enough. Therefore, lowering the search space within the matrix is a crucial step to improve the matching results.

3.5.2 Tracklet Selection

In order to simplify the process for multi-camera matching, we utilize both directional and temporal information to clamp the scope of the matching process. Based on the GPS location of each camera, we can rotate zones in Sec. 3.4.5 for each cameras to connect the track inlets and outlets. Inspired by [17], our zones can be simplified for multi-camera matching, where (Zone 1, Zone 2) correspond to West, (Zone 3, Zone 4) correspond to North, (Zone 5, Zone 6) correspond to East, and (Zone7, Zone 8) correspond to South. Furthermore, we can calculate the possible time range between each connection using the maximum and minimum speed limit as well as traffic light signals of a given scene. This gives us a complete picture of whether a vehicle can appear in specific zone of a specific camera in the given range of time, which forms a trajectory clamping mask that limits the search scope in space and time down to certain ranges. For example, for a tracklet T_i that end in West at time t_e, a matching tracklet in the next camera down the road T_j must start in East within time range $[t_e+t(i,j)_{min}, t_e+t(i,j)_{max}]$, where $t(i,j)_{min}, t(i,j)_{max}$ are the minimum and maximum time interval for crossing the cameras. Compared to the filtering mask used in [17], applying selecting mask to the similarity matrix narrows down the search space almost by half, which greatly increases IDP while maintaining IDR.

3.5.3 Tracklet Clustering

Inspired by [17], our method contains two rounds of multi-camera tracklet clustering. The first round is directional based. For instance, tracklets going from camera 41 to 42 and the other way could be clustered together, but with reliable single-camera tracking result, we can compare only those tracklets going in the same direction, reducing the clustering space. After the first round of clustering, tracklets

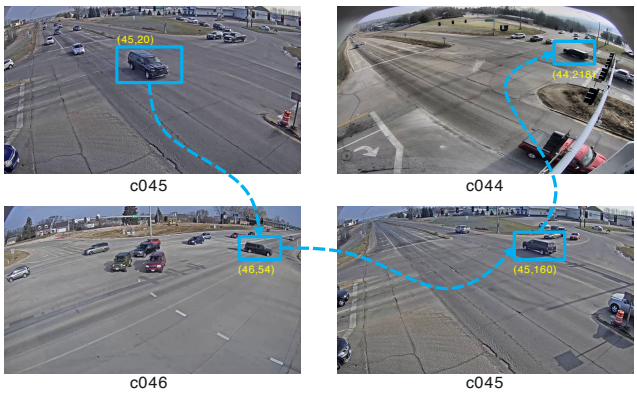


Figure 9. Example of U-turns, where one vehicle enters c045, goes to c046, turns around to c045, and finally appears in c044.

of the same vehicle from adjacent cameras need to be aggregated together, and the order of aggregation matters because the same vehicle can have different IDs across cameras. We propose a iterative searching strategy that effectively solves edge cases like U-turns, as shown in Fig. 9.

That same vehicle that from 45 to 46, and then from 46 to 44, can be express as (camera ID, track ID) pairs like (45,20)(46,54)(45,160)(44,218). One aggregation order can simply be [41,42][42,43][43,44][44,45][45,46], [42,41][43,42][44,43][45,44][46,45], searching for track ID intersections between each camera. If there is intersection, the two sets of different camera can be merged, and if not, new sets are created, and so on. However, U-turns in this case cannot be properly aggregated. For example, in Fig. 9, begin with (45,20),(46,54), when we reach (44,218),(45,160), that same vehicle will not be recognized since the turning point has not been reached, and the intersection is empty. Going forward, we get results like (45,20),(46,54),(45,160), (44,218),(45,160),(46,54). Apparently, there is an intersection between these two sets, yet they are split into two tracklets. Therefore, we propose a aggregation strategy that searches though tracklets sets in each camera, until there are no intersection left between any sets, which effectively solves the U-turn issue.

4. Experiment

4.1. Datasets and Evaluation Metrics

The AIC22 benchmark (CityFlowV2) [20, 28] is captured by 46 cameras in real-world traffic surveillance environment. A total of 880 vehicles are annotated in 6 different scenarios. 3 of the scenarios are used for training. 2 scenarios are for validation. And the last scenario is for testing. There are 215.03 minutes of videos in total. The length of the training videos is 58.43 minutes, that of validation videos is 136.60 minutes, and that of testing videos is 20.00 minutes. For MTMC tracking, we use the IDF1 [26]

Method	IDF1	IDP	IDR
Baseline	0.7522	0.8173	0.6967
+Feature Dropout Filter	0.8192	0.8871	0.7610
+Multi-level matching	0.8312	0.8799	0.7877
+Tracklet Merging	0.8362	0.8732	0.8022
+Tracklet Selection	0.8437	0.8900	0.8020

Table 1. IDF1 score changes after several optimizations.

Team ID	IDF1 score
28	0.8486
59	**0.8437**
37	0.8371
50	0.8348
70	0.8251

Table 2. Comparison of our method with other teams.

score as evaluation indicators. IDF1 measures the ratio of correctly identified detection over the average number of ground-truth and computed detection. The evaluation system of the AI City Challenge will display IDF1,IDP, IDR, Precision (detection) and Recall (detection).

4.2. Results

We use the evaluation opportunity provided by the evaluation system to verify the effect of our algorithm, and optimize our algorithm according to the IDF1,IDP, IDR, Precision and Recall results. We record the changes of IDF1 after several optimizations which are shown in Tab. 1.

In the final ranking of track 1 of AI City Challenge 2022, we rank the second place among all participating teams. The comparison of our method with other teams' on the evaluation system is shown in Tab. 2. Our code will be released later.

5. Conclusion

In this paper, we propose an accurate multi-camera vehicle tracking system. For single-camera tracking, we incorporate three reinforcing methods for augmented tracks prediction, and multi-level association and zone-based singe-camera tracklet merging strategy. For multi-camera tracking, we develop a spatial-temporal strategy that reduces search space when matching, and improve on hierarchical clustering that captures U-turn tracklets. Our results show that these methods improve both IDR and IDP, with a final IDF1 score of 0.8437.

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. 1

[2] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016. 2

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

[4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 2

[5] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6154–6162, 2018. 2

[6] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[7] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6638–6646, 2017. 2

[8] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European conference on computer vision*, pages 472–488. Springer, 2016. 2

[9] Zhiqun He, Yu Lei, Shuai Bai, and Wei Wu. Multi-camera vehicle tracking with powerful visual features and spatial-temporal cue. In *CVPR Workshops*, pages 203–212, 2019. 1

[10] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE transactions on pattern analysis and machine intelligence*, 37(3):583–596, 2014. 2

[11] Yunzhong Hou, Heming Du, and Liang Zheng. A locality aware city-scale multi-camera vehicle tracking system. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 167–174, 2019. 1

[12] 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 *CVPR Workshops*, pages 416–424, 2019. 1

[13] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE transactions on pattern*

analysis and machine intelligence, 34(7):1409–1422, 2011. 2

[14] Young-Gun Lee, Jenq-Neng Hwang, and Zhijun Fang. Combined estimation of camera link models for human tracking across nonoverlapping cameras. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2254–2258. IEEE, 2015. 1

[15] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4282–4291, 2019. 2

[16] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8971–8980, 2018. 2

[17] Chong Liu, Yuqi Zhang, Hao Luo, Jiasheng Tang, Weihua Chen, Xianzhe Xu, Fan Wang, Hao Li, and Yi-Dong Shen. City-scale multi-camera vehicle tracking guided by crossroad zones. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021. 3, 7

[18] 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

[19] Pedro Antonio Marín-Reyes, Luca Bergamini, Javier Lorenzo-Navarro, Andrea Palazzi, Simone Calderara, and Rita Cucchiara. Unsupervised vehicle re-identification using triplet networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 166–1665, 2018. 3

[20] Milind Naphade, Shuo Wang, David C. Anastasiu, Zheng Tang, Ming-Ching Chang, Xiaodong Yang, Yue Yao, Liang Zheng, Pranamesh Chakraborty, Christian E. Lopez, Anuj Sharma, Qi Feng, Vitaly Ablavsky, and Stan Sclaroff. The 5th AI City Challenge. In *Proc. CVPR Workshops*, pages 4263–4273, Virtual, 2021. 7

[21] 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*, pages 588–589, 2020. 1

[22] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 2

[23] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, 2017. 2

[24] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 2

[25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017. 2

[26] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016. 7

[27] Jakub Španhel, Vojtech Bartl, Roman Juránek, and Adam Herout. Vehicle re-identification and multi-camera tracking in challenging city-scale environment. In *Proc. CVPR Workshops*, volume 2, page 1, 2019. 1

[28] 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 *Proc. CVPR*, pages 8797–8806, Long Beach, CA, USA, 2019. 7

[29] Zheng Tang, Gaoang Wang, Hao Xiao, Aotian Zheng, and Jenq-Neng Hwang. Single-camera and inter-camera vehicle tracking and 3d speed estimation based on fusion of visual and semantic features. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 108–115, 2018. 1

[30] Ultralytics. Yolov5. `https://github.com/ultralytics/yolov5`. Accessed in 2022.04. 2, 3

[31] 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, 2017. 3

[32] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. *arXiv preprint arXiv:2110.06864*, 2021. 4

[33] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision*, 129(11):3069–3087, 2021. 3