A Robust MTMC Tracking System for AI-City Challenge 2021

Jin Ye* Xipeng Yang* Shuai Kang Yue He Weiming Zhang Leping Huang
Minyue Jiang Wei Zhang Yifeng Shi Meng Xia Xiao Tan
Department of Computer Vision Technology (VIS), Baidu Inc., China
{yejin03,yangxipeng01,kangshuai,heyue04,zhangweiming,huangleping}@baidu.com
{jiangminyue,zhangwei99,shiyifeng,v_xiameng,tanxiao01}@baidu.com

Abstract

Multi-Target Multi-Camera tracking (MTMC) is an essential task in the intelligent city and traffic analysis. It is a great challenging task due to several problems such as heavy occlusions and appearance variance caused by various camera perspectives and congested vehicles. In this paper, we propose a practical framework for dealing with the MTMC problem. The proposed framework contains three stage. Firstly, in the vehicles detection and Re-ID stage, the proposed system leverages Cascade R-CNN to detect all vehicles and extract appearance features with a Re-ID module for all cameras. Secondly, in the Multi-Target Single-Camera tracking (MTSC) stage, on the basis of the detected boxes and appearance features, it tracks multiple vehicles to generate candidate trajectories within each single camera with Tracklet-Plane Matching (TPM) tracking algorithm. Finally, in the Inter-Camera Association (ICA) stage, it associates all candidate trajectories between two successive cameras using the established distance matrix, and combines all successively matching results for final submission. The established distance matrix is simply computed by the Re-ID features and refined by the constraints of traveling time, road structures, and traffic rules to accelerate matching time as well as reduce search space. Extensive experiments on the public track3 test set of NVIDIA AI CITY 2021 CHALLENGE demonstrate the effectiveness of our method, which achieves IDF1 of 77.87%.

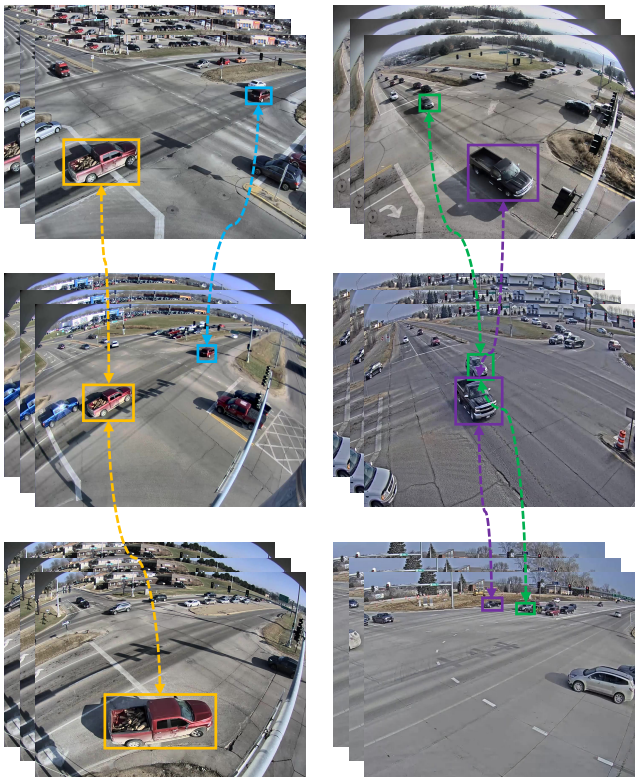


Figure 1. Illustration of Multi-Target Multi-Camera tracking task. Given a variety of vehicles in different cameras, our goal is to match all identical vehicles across all cameras.

1. Introduction

In recent years, with the rapid development of intelligent transportation system, the demand of Multi-Target Multi-Camera tracking (MTMC) is rapidly increased. The purpose of MTMC task is to tracks multiple vehicle targets across multiple cameras as shown in Figure 1, which can help with traffic flow analysis and vehicle routes planning.

Typically, MTMC consists of vehicle detection, Re-Identification (Re-ID), Multi-Target Single-Camera tracking (MTSC) and Inter-Camera Association (ICA) stage. In last few years, there have emerged some successful research achievements for MTMC [1, 13, 15, 16, 17, 25, 30, 34, 36, 37]. A classical pipeline of MTMC in the intelligent transportation scene, based on vehicle detection results and Re-ID features, is to first track vehicles to generate candidate trajectories in single cameras with MTSC module, and then

*Equally-contributed authors.

associate these candidate trajectories along through cameras to find vehicles with the same identity with ICA module. For MTSC module, Many tracking-by-detection methods have been proposed recently because the precision of vehicle detection results has a direct impact on the tracking process. Andreas *et al*. [10] and Zhang *et al*. [44] adopt vehicle features and vehicle locations with detected bounding boxes to identify same objects, which achieve remarkable tracking results. Peng *et al*. [24] propose a novel Tracklet-Plane Matching (TPM) algorithm, which not only combines multiple short trajectories into a single long trajectory but also attaches missing detection boxes for tracked trajectories. For ICA module, Hsu *et al*. [16] firstly propose many intricate traveling strategies to distinguish trajectories with several pre-defined zones for every camera, and then apply the greedy algorithm to generate final matching result. Qian *et al*. [25] propose a rule-based algorithm, which adopts the geometry information to repeatedly update MTSC tracking result until the matching result is merged.

However, there are still several challenges to overcome. For MTSC tracking, it has a higher risk of vehicles loss because Re-ID features may easily affected by severe occlusion, complex light conditions, and serious deformation of a specific vehicle in different areas in a camera. Meanwhile, for the intrinsic property of this task, it is difficult to deal with ID switch problem in MTSC tracking mainly because of serious occlusion of congested vehicles, which is frequently happened while a large number of vehicles are waiting for the traffic lights. For ICA, a vehicle may not appear in the next camera when it drives out the last camera, and vice versa. For example, let us consider those vehicles from camera 41 to 42 as shown in Figure 5, vehicles can leave camera 41 but may not enter camera 42, or they can drive into camera 42 but not exit camera 41. These vehicles may be tracked in MTSC but cannot be found those pairs in MTMC ground-truth. As most matching algorithms (e.g. Hungarian matching algorithm [23]) attempt to find much more pairs as possible, falling to remove those impossible camera-cross-traveling trajectories will have a harmful impact on the final matching result. All these problems described above make MTMC tracking a more challenging task.

In this paper, we propose an effective and accurate system for multi-target multi-camera tracking. The flowchart of our proposed MTMC system is shown in Figure 2. Given a set of videos under different camera views, the system first leverages Cascade R-CNN to detect all vehicles and extract appearance features with Re-ID module for all cameras. On the basis of the detected boxes and appearance features, it tracks multiple vehicles to generate candidate trajectories within each single camera with TPM algorithm. Finally, in the ICA procedure, it associates all candidate trajectories between two successive cameras using the established distance matrix, and combines all matching results for final submission. Particularly, The established distance matrix is simply computed by the Re-ID features and refined by the constraints of traveling time, road structures, and traffic rules to reduce the searching space as well as accelerate the matching time.

The rest of the paper is organized as follows: An overview of related work is described in Section 2. In section 3, our proposed framework is introduced in detail. In Section 4, we demonstrate sufficient experiments of our method on the track3 of CVPR AI City Challenge 2021. Finally, conclusion is concluded in Section 5.

2. Related Work

2.1. Vehicle Detection

Object Detection is one of the most popular tasks in computer vision and image processing, and it locates the existence of objects in an image using bounding boxes and categorizing the objects found. In general, there are two different branches for this task: one stage methods [26, 20] and two stage methods [27, 11, 5]. SSD [20] and YOLO [26] are representative one stage methods. SSD uses a set of pre-defined boxes of different aspect ratios and scales in order to predict the presence of an object in a certain image, and YOLO attempts at building a fast real-time object detector treating the detection task as a regression problem. For two stage methods, the Faster R-CNN [27] firstly generate a set of Region of Interests (RoIs) by Region Proposal Network (RPN), then a binary classifier processes the RoIs to having objects or backgrounds, finally, RoIs with objects are refined to obtain regressed boxes and fine grained classes. Based on Faster R-CNN, He *et al*. [11] propose a new layer named RoIAlign, which can correct the misalignments between the RoIs and the extracted features. However, detection performance by [11] tends to degrade with increasing the IoU thresholds. Cai *et al*. [5] propose a Cascade R-CNN to address above problems by training a sequence of detectors with increasing IoU thresholds. Compared to one-stage methods, two stage methods are usually more accurate and flexible. In this paper, we use Cascade R-CNN to detect vehicles.

2.2. Vehicle Re-identification

Vehicle Re-Identification (Re-ID) is a fundamental task in multi-camera traffic flow for smart cities applications, with the goal of retrieval vehicles that appear in different cameras. Convolution Neural Network based (CNN) visual deep features representation has recently gained a lot of attraction. Several loss functions, sampling strategies and samples generation methods have been proposed to learn discriminative representations. Representative loss functions include cross entropy loss [46], triplet loss [14], circle

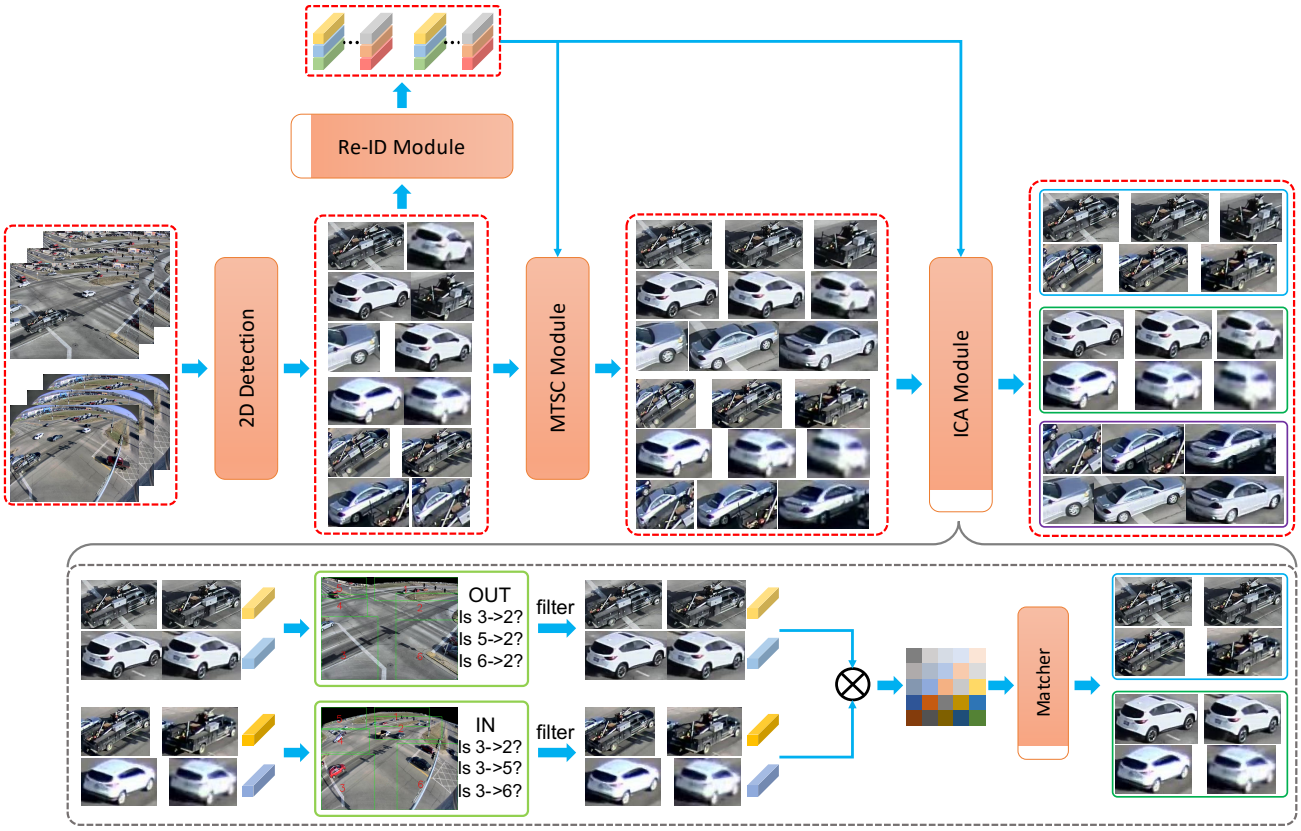


Figure 2. The pipeline of our MTMC system. We first detect all vehicles using Cascade R-CNN and extract their appearance features with Re-ID module. Then, all extracted boxes with their features are fed into MTSC Module, which generates all trajectory candidates for every single camera. Finally, our proposed ICA module matches all trajectory candidates across all cameras as the final result.

loss [32], and other effective loss [8].

For sampling strategies, Chen *et al*. [7] and Rikiya *et al*. [33] mine informative samples in vehicle Re-ID training phase. Tang *et al*. [35] focus on part of vehicle or key-point of vehicle learning strategy, which embeds key-points, heatmaps and segmentation task into vehicle Re-ID training stage, which guides the convolution network to learn the part-related, point-related or pixel-related information. Shen *et al*. [29] apply the spatial-temporal constraints to reduce the sample search space, which eliminates the hard-negative samples. However, due to the large visual appearance changes caused by different cameras, vehicle orientation, illuminations and occlusions, Re-ID is still a challenging task.

In order to learn the robust vehicle representation, many recent works have explored samples generation methods. For instance, Generative Adversarial Network (GAN) [22] and game engine have demonstrated the effectiveness of the synthetic data in training Re-ID networks [40, 47]. GAN transfers the style from a source domain to a target domain, and generates samples conditioned with the specific attributes, such as color, orientation and occlusions. Zhou *et al*. [51] synthesize a multi-view feature by transforming a single-view feature to against the orientation variation.

2.3. Single Camera Tracking

Tracking-by-detection. Most traditional tracking algorithms follow the tracking-by-detection paradigm, which obtains multiple detection boxes in each frame firstly with an effective detector and then associate the detection boxes. Bewley *et al*. [4] introduce a SORT algorithm, which tracks bounding boxes by using a Kalman filter and Hungarian algorithm correctly. Nicolai *et al*. [41] propose a DeepSort algorithm on the base of [4], which uses appearance features from a deep network to enhance the association cost . Yu *et al*. [43] propose a POI algorithm, which designs an efficient detector based on Faster R-CNN and then gets better appearance feature more efficiently. Based on all the above, Peng *et al*. [24] propose a novel TPM algorithm, which not only considers how to combine multiple short sub-trajectories into a long trajectory properly, but also complements the supplement of missing detection boxes based on trajectory context.

Joint detection and tracking. Currently, a more pop-

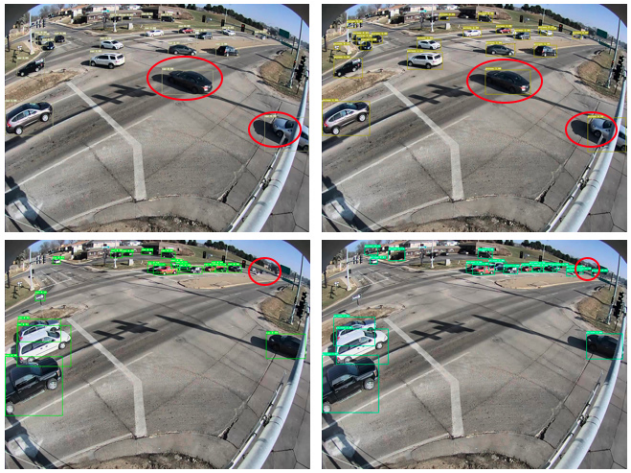


Figure 3. Examples of detected vehicles. The left two images show the box shift of vehicle-shadow and detection loss of tiny vehicle issues. The right two images demonstrate our detection model can solve those problems to same extent.

ular method in the field of multi-target tracking is integrating the detector into the tracker and training both tasks in the same framework. Bergmann *et al*. [2] introduce a novel algorithm named Tracktor, which removes the box association strategy and propagates identities of region proposals directly by bounding box regression strategy. Zhang *et al*. [45] propose FairMOT considering the difference between detection task and Re-ID trask in-depth, which promotes the development of multi-object tracking task. Zhou *et al*. [49] propose CenterTrack on the base of CenterNet [50], which associates different objects between different frames by using the offsets of center points and is simpler, faster and more accurate. Recently, some tracking algorithms based on the TransFormer [38] have attracted the attention of many researchers. Meinhardt *et al*. [21] firstly introduce TransFormer to the multi-object tracking task and proposed the TrackFomer algorithm, which exploits self-attention and encoder-decoder attention mechanisms and reliefs issues caused by occluded, missing or noisy detection. Subsequently Sun *et al*. [31] propose TransTrack, which combines track query and learned object query with DETR [6] and further improves the track performance.

3. Method

This section presents the details of our framework for Multi-Target Multi-Camera tracking (MTMC). Figure 2 shows a brief overview of our system. We describe each key module (vehicle Detection, Re-ID, MTSC, and ICA) in detail in the following sections.

3.1. Vehicle Detection

Vehicle detection is the first and essential step in MTMC tracking. To effectively detect vehicles as much as possible for all images, we adopt the state-of-the-art network Cascade R-CNN [5] as our vehicle detection model for each frame. We use a powerful convolutional neural work ResNet-101 [12], which is equipped with the Feature Pyramid Network (FPN) [19], as its backbone for feature extraction. Then abundant candidate bounding boxes are produced by Region Proposal Network (RPN) and Region of Interest Alignment (RoIAlign). The detected box with its object class and confident score are generated by multiple output heads.

In this traffic scene, we only need to focus on the vehicle object, and the difficulty of the scene is to detect tiny vehicles and box shift of vehicle with shadows, as shown in the top-left of Figure 3. To reliably address these problems, we sample more images that contains vehicle-shadow samples from validation videos to detect more accurate boxes for box shift of vehicle-shadow problem. For tiny vehicles, we not only use FPN but also increase the loss weight for tiny targets to recall more correctly tiny boxes. Furthermore, in order to obtain a more robust and accurate detection model, we train our detection model using all possible data in the AI City Challenge 2021 within the rules, including all validation videos in track3, and all train/val data from track1 and track4 .

3.2. Vehicle Re-identification

Following existing Re-ID works [48, 35], we simply use HRNet [39] and Res2Net [9] as backbone for feature extraction, and we just concatenate all output features of these models as our appearance features. The loss function we used is Cross-Entropy loss and Triplet loss. Given the input image x, the cross-entropy loss is formulated as follow:

$$L_{ce} = -\sum_{i=1}^{C} y_i \log(\hat{y_i}) + (1 - y_i) \log(1 - \hat{y_i}) \quad (1)$$

where C is the number of vehicle identities in the dataset, $\hat{y_i}$ is the i th predicted probability, and y_i is the i th ground truth label of the input x. Note that y_i = 1 if x equals to the ground truth label y_i, else y_i = 0.

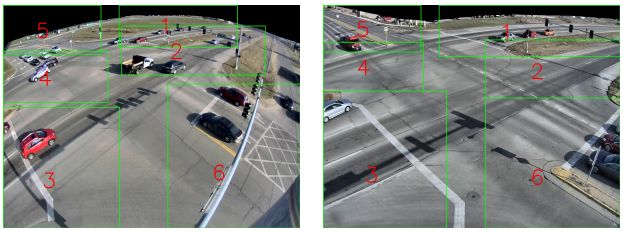
Triplet loss focuses on optimizing the distance between the training samples, which makes the positive vehicles closer to each other and push the negative samples away from each other, the triplet loss with N samples can be formulated as:

$$L_{tri} = \sum_{i=1}^{N} [m + Dis(f_i^a, f_i^p) - Dis(f_i^a, f_i^n)] \quad (2)$$

where m is the margin, $Dis(g_1, g_2) = \|g_1 - g_2\|$ denotes the L2-norm, f_a, f_p, f_n is the triplet pair in the feature

Figure 4. Examples of MTSC tracking in different scenarios.

Figure 5. The camera locations and their surrounding roads.



(a) zones in c042 camera (b) zones in c043 camera

Figure 6. Examples of predefined zones to describe trajectories. According to the traffic rules, each trajectory must be valid. For the two cameras, valid trajectory is [(1, 4), (1, 5), (1, 6), (3, 2), (3, 5), (3, 6), (5, 4), (5, 2), (5, 6), (6, 2), (6, 4)]. For those vehicles from c042 to c043, they must drive out of zone 4 in c042 and must drive in zone 1 in c043. Only a small number of trajectories will be retained with this strict constraint. For c042 to c043, Only trajectories through [(1, 4), (5, 4), (6, 4)] in c042 and [(1, 5), (1, 4), (1, 6)] in c043 will be filtered out as possibly matching candidates.

space, f^a is the anchor sample in the feature space, f^p is the same identities with f^a, while f^n and f^a is different identities.

3.3. Multi-Target Single-Camera Tracking

DeepSort [41] and Tracklet-Plane Matching (TPM) [24] are two popular tracking-by-detection algorithms for MTSC tracking. To link all detected vehicles to several trajectory candidates, we compare the two algorithms and TPM outperforms DeepSort in our experiments. Considering the performance and effectiveness, we choose TPM as our MTSC tracking method. The comparison of two tracking algorithms can be found in Table 2.

The insight of TPM [24] is using sub-trajectories as the basic association units and fusing those sub-trajectories that belong to same planes into long trajectories. Due to a small amount of missing or incorrect detection results, the traditional tracking methods may be disconnected or misconnected temporally. However, TPM first generates numerous sub-trajectories through the traditional tracking algorithms with higher threshold. Then it adopts the sub-trajectory plane module to organize those sub-trajectories that potentially belong to the same long trajectory into same sub-trajectory-plane. Those sub-trajectories that are allocated to the same sub-trajectory-planes are merged into long trajectories in the last step. Some examples of MTSC tracking in different scenarios are shown in Figure 4.

3.4. Inter-Camera Association

As shown in Figure 5, there are only a small number of determined vehicles can through two different cameras with the constraint of traveling time, road structures, and traffic rules. Due to those natural properties, we propose an efficiently fast MTMC strategy that can reduce the matching space and accelerate the matching time to get a more reliable matching result. In this part, we describe the whole proposed procedure in detail.

Identifying enter/exit area for trajectories. Since vehicles must obey the traffic rules and can only pass along special routes due to road structures, how to efficiently filter certain vehicles that are impossible to arrive other cameras is crucial for final matching. To tackle this problem, Hsu *et al*. [16] pre-define numerous zones in each camera and make several intricate rules to distinguish trajectories. Different from the excessively pre-defined zones and complicated rules in [16], we simply pre-define a zone just for each enter/exit area of every camera as shown in Figure 6. We first define "in zone" and "out zone": "in zone" is defined as the zone where the detected vehicles first touched, and "out zone" is defined as the zone where they last touched. The zone construction must be followed two principles: One is to be the first touched zone that cov-

ers the entering vehicles as much as possible for "in zone", and to be the last touched zone that covers the exiting vehicles as much as possible for "out zone". Another is to avoid ambiguity, which means that an entering vehicle should not be assigned to two different "in zone"s, and an exiting vehicle should not be assigned to two different "out zone"s. Particularly, as shown in Figure 6(a), because of the traffic rules, some zones can be designed as only a "in zone" (e.g. zone 1 and zone 3) or only a "out zone" (e.g. zone 2 and zone 4), while others can be defined as both a "in zone" and a "out zone" (e.g. zone 5 and zone 6).

Distinguishing trajectory candidates. After predefined all "in zone"s and "out zone"s for all camera. Our goal is to use those zones to distinguish each trajectory. Specifically, we define a distinguished trajectory within a camera as,

$$Traj_i = [m, n] \tag{3}$$

where m, n is the "in zone" id and "out zone" id of trajectory i, respectively. As shown in Figure 6(a), if one vehicle is just entering the c042 camera and its center point first touches zone 1, the "in zone" m and the "out zone" n are set as 1. Then m is fixed but n is updated every frame. The trajectory is determined as [1, 4] if zone 4 is the final appeared zone of the vehicle. With the elaborately designed algorithm, the in/out zone accuracy for all trajectories is almost up to 100% in our experiments.

Filtering out trajectory candidates. Using the previously obtained trajectories $Traj$, we filter out those trajectories as the matching candidates who directly follow traffic rules, road structures, and traveling time. For example, let us consider the vehicles from camera c042 to camera c043 in Figure 6, only those vehicles leaving zone 4 of the "out" camera c042 will be able to access zone 1 of the "in" camera c043. Therefore, we keep all trajectories whose "out zone" m is 4 in camera c042 and whose "in zone" n is 1 in camera c043 as the matching candidates between c042 and c043.

Multi-Camera trajectory matching. To further reduce the re-identification search space and accelerate the matching time, we consider find matching pairs between two successive cameras and then fuse all matched pairs among all cameras. For matching within two cameras, we first compute the pairwise distance matrix $D \in \mathbb{R}^{N_{out} \times N_{in}}$ with N_{out} and N_{in} filtered tracklet candidates in "out" camera and "in" camera. The distance between two candidates is calculated as,

$$dist(n_{out}, n_{in}) = \frac{1}{k} \sum_{i=1}^{k} sorted(F_{n_{out}} \circ F_{n_{in}})[:k] \tag{4}$$

Where $n_{out} \in N_{out}$, $n_{in} \in N_{in}$. The operation $\circ$ calculates the cosine distance between each pair of two sets. The length of the result is $n1 \times n2$ if the length of set1 is $n1$ and the length of set2 is $n2$. $F_{n_{out}} = [f^1_{n_{out}}, f^2_{n_{out}}, \ldots, f^{m_{out}}_{n_{out}}]$ and $F_{n_{in}} = [f^1_{n_{in}}, f^2_{n_{in}}, \ldots, f^{m_{in}}_{n_{in}}]$ are Re-ID feature sets for n_{out} th candidate in "out" camera and n_{in} th candidate in "in" camera, respectively. $sorted(\cdot)$ arranges the calculated distances in ascending order. k is the parameter that we use to measure the distance between two candidates by averaging the top k minimized distances. In our experiment, we choose $k = 3$. Before matching, we need to refine D with the constraint of traveling time. If the traveling time is out of the valid time window, we multiply a penalty factor to the distance of the pair.

$$\overline{dist}(n_{out}, n_{in}) = \begin{cases} \alpha \times dist(n_{out}, n_{in}), & t \leq T_l \\ dist(n_{out}, n_{in}), & T_l < t < T_u \\ \alpha \times dist(n_{out}, n_{in}), & t \geq T_u. \end{cases} \tag{5}$$

Where T_l and T_u are the thresholds of traveling time window, and α being 2 in the paper is the penalty factor for those pairs outside the time window. After refining D, the final matching pairs between two cameras are found with the Hungarian matching algorithm [23] directly. Following the matching process, we fuse all matched pairs throughout all cameras as our final result using a simple rule that two pairs will be identified with the same global id if they share one tracklet. For example, there are two matching pairs: One is $[Traj1, Traj2]$ between camera 41 and 42. Another is $[Traj2, Traj3]$ between camera 42 and 43. As the two pairs share $Traj2$ in camera 42, we simply merge the two pairs into one $[Traj1, Traj2, Traj3]$ among camera 41, 42, and 43.

4. Experiments

4.1. Datasets

The CityFlowV2 [1] dataset contains 3.58 hours (215.03 minutes) videos collected from 46 cameras spanning 16 intersections in a mid-sized U.S. city. The distance between the two furthest simultaneous cameras is 4 km. The dataset covers a diverse set of location types, including intersections, stretches of roadways, and highways. For city-scale multi-camera vehicle tracking track, the dataset is divided into 6 scenarios. 3 scenarios are used for training, 2 scenarios are for validation, and the remaining scenario is for testing. The dataset contains 313931 bounding boxes for 880 distinct annotated vehicle identities in total. Only vehicles passing through at least 2 cameras have been annotated. The resolution of each video is at least 960p and the majority of the videos have the frame rate of 10 FPS. Additionally, in each scenario, the offset from the start time is available for each video, which can be used for synchronization.

The data of our Re-ID module is the AIC21 benchmark (CityFlowV2-ReID), which is captured by 46 cameras in real-world traffic environment. There are 85058 images in total, and 880 vehicles are annotated and 52717 images (440

[1] https://www.aicitychallenge.org/2021-data-and-evaluation/

	IDF1	IDP	IDR	Precision	Recall
Mid	50.1	69.5	39.2	83.2	46.9
k=3	**56.4**	**82.9**	42.8	**87.6**	45.2
k=10	55.8	77.9	**43.5**	86.2	**48.1**

Table 1. The performance of different distance computing methods. "Mid": we select the feature in the middle of the feature set F as the feature of the tracklet, and we simply compute the cosine distance of the selected two features as the distance between two tracklets. "k=3" and "k=10": we use equation 4 with $k = 3$ and $k = 10$ to compute the distance between two tracklets.

	IDF1	IDP	IDR	Precision	Recall
DeepSort	56.4	**82.9**	42.8	**87.6**	45.2
TPM	**66.8**	74.9	**60.2**	77.7	**62.4**

Table 2. The performance of different single camera tracking schemes.

vehicles) are used for training. There are some synthesis dataset is generated by VehicleX [35, 42], which is a publicly aviablable 3D engine. A total of 1362 vehicles and 192150 images are annotated with detailed labels.

4.2. Implementation Details

Detection. For the Cascade R-CNN ResNet-101 model, Stochastic Gradient Descent (SGD) is adopted for the training process, weight decay and momentum are set as 0.0001 and 0.9, respectively. ResNet-101 model is trained in 50K iterations with the initial learning rate 0.02 and the batch size 2. The learning rate is reduced by the factor of 10 at iteration 30K and 40K, respectively. We have 5 layers (i.e. level 2 to level 6) feature map for FPN. We cluster ground truth boxes in the training dataset same as [26], and the selected anchors for each level are $[16, 32, 64, 128, 256]$. We train our detection model using all train/val data from track1, track3, and track4. All detection experiments are implemented based on PaddlePaddle.

ReID We mainly also use PaddlePaddle to train our global models, and the model is trained using SGD with momentum 0.9. Random crop, random flip and scale jittering are used as data augmentations in training. During training, the cos-decay learning rate scheduler is adopted with base learning rate 0.001.

4.3. Metrics of Evaluation

For MTMC tracking, the IDF1 score [28] is used to rank the performance in the final leader board. IDF1 measures the ratio of correctly identified detections over the average number of ground-truth and computed detections. The evaluation tool provided by the challenge also computes other evaluation measures, which are adopted by the MOTChallenge [3, 18], such as Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP), Mostly Tracked targets (MT), and False Alarm Rate (FAR). However, they are not used for ranking. The measures, which are computed in the evaluation system, are IDF1, IDP, IDR, Precision (detection) and Recall (detection).

Team ID	IDF1 score
75	80.95
29	77.87
7	76.51
85	69.10
42	62.38
27	57.63
15	56.54
48	55.34
79	54.58
112	54.52

Table 3. Comparison of out method with other teams on the CityFlowV2 dataset. Our team ID is marked as bold.

4.4. Experiments results

In this section, we conduct two ablation studies and compare our results to other teams.

Distance between two tracklets. We conduct an ablation study using various different distance computing methods to find the best way to measure the distance between two tracklets. As shown in Table 1, computing distance with multiple features outperforms using only one feature. However, it damages the IDF1 score when k is too large. The result of different k demonstrates that not all features are suitable for representing the tracklet.

Different single camera tracking schemes. We constructed an ablation experiment to select a baseline scheme for single-camera tracking. We compare the performance of two classical tracking-by-detection algorithms: TPM and DeepSort algorithm. As shown in the Table 2, TPM algorithm outperforms the DeepSort algorithm with a large margin in most indicators. Specifically, TPM achieves 66.8% IDF1, which is the most important metrics. As TPM not only combines multiple short trajectories into a single long trajectory but also attaches missing detection boxes for tracked trajectories, it would recall more trajectories but may also introduce several incorrect trajectories. The IDP and IDR metrics have proved this. Since IDF1 is the official metric on the leader board, we select TPM as our tracking algorithm.

Comparison with other teams The proposed system is submitted to the track 3 of *NVIDIA AI City Challenge 2021* for evaluation. We rank second place among over 20+ teams from all over the world. In the final experiment, our system obtains 77.87% IDF1, which outperforms third place by 1.4%. Meanwhile, because of the fast speed and robustness

Figure 7. Visualization of final matching results on CityFlowV2. The same vehicles are marked with same ID. For example, in the first row, the vehicle with ID 170 in every column image is identified as the same vehicle.

of the system, it is easy to apply to real-world Multi-Target Multi-Camera applications.

4.5. Visualization

The final matching results of proposed algorithm on CityFlowV2 are shown in Figure 7. Each row represents the matched trajectories in different cameras with the same ID. Our algorithm can find the right matching pairs even if the trajectories have different angles or occlusion in different cameras. For example, in the second row of Figure 7, the matched id is 76, and the camera ids are c042, c043, c044, and c045 from left to right. As we can see, the vehicles in cameras c044 and c045 have different angles and appearances than those in cameras c042 and c043, but they still match correctly.

5. Conclusion

In this paper, we propose an effective system for MTMC tracking. First, the system uses Cascade R-CNN and Re-ID module to detect vehicles and extract their appearance features in single cameras. In the MTSC stage, TPM algorithm used to associate vehicles in every camera based on the detection results and appearance features. In the ICA stage, it associates all trajectory candidates between two successive cameras using the established distance matrix, and combines all matching results for final submission. Particularly, The established distance matrix is simply computed by the Re-ID features and refined by the constraints of traveling time, road structures, and traffic rules to reduce the searching space as well as accelerate the matching time. The result shows the effectiveness of the system, which achieves 77.87% IDF1 on the track3 test set of NVIDIA AI CITY 2021 CHALLENGE.

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] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019. 4

[3] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 7

[4] 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

[5] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 2, 4

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 4

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 3

[8] Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. Beyond triplet loss: a deep quadruplet network for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 403–412, 2017. 3

[9] Shanghua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip HS Torr. Res2net: A new multi-scale backbone architecture. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 4

[10] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 3d traffic scene understanding from movable platforms. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):1012–1025, 2013. 2

[11] 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

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[13] 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

[14] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017. 2

[15] 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

[16] 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, 2, 5

[17] 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

[18] Yuan Li, Chang Huang, and Ram Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *2009 IEEE conference on computer vision and pattern recognition*, pages 2953–2960. IEEE, 2009. 7

[19] 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. 4

[20] 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

[21] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. *arXiv preprint arXiv:2101.02702*, 2021. 4

[22] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 3

[23] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957. 2, 6

[24] Jinlong Peng, Tao Wang, Weiyao Lin, Jian Wang, John See, Shilei Wen, and Erui Ding. Tpm: Multiple object tracking with tracklet-plane matching. *Pattern Recognition*, 107:107480, 2020. 2, 3, 5

[25] Yijun Qian, Lijun Yu, Wenhe Liu, and Alexander G. Hauptmann. Electricity: An efficient multi-camera vehicle tracking system for intelligent city. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 1, 2

[26] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 2, 7

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015. 2

[28] 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

[29] Yantao Shen, Tong Xiao, Hongsheng Li, Shuai Yi, and Xiaogang Wang. Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1900–1909, 2017. 3

[30] 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, 2019. 1

[31] Peize Sun, Yi Jiang, Rufeng Zhang, Enze Xie, Jinkun Cao, Xinting Hu, Tao Kong, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple-object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020. 4

[32] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6398–6407, 2020. 3

[33] Rikiya Suzuki, Sumio Fujita, and Tetsuya Sakai. Arc loss: Softmax with additive angular margin for answer retrieval. In *Asia Information Retrieval Symposium*, pages 34–40. Springer, 2019. 3

[34] 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 *CVPR Workshops*, pages 275–284, 2019. 1

[35] Zheng Tang, Milind Naphade, Stan Birchfield, Jonathan Tremblay, William Hodge, Ratnesh Kumar, Shuo Wang, and Xiaodong Yang. Pamtri: Pose-aware multi-task learning for vehicle re-identification using highly randomized synthetic data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 211–220, 2019. 3, 4, 7

[36] 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*, pages 8797–8806, 2019. 1

[37] 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

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 4

[39] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 4

[40] Longhui Wei, Shiliang Zhang, Wen Gao, and Qi Tian. Person transfer gan to bridge domain gap for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 79–88, 2018. 3

[41] 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, 5

[42] Yue Yao, Liang Zheng, Xiaodong Yang, Milind Naphade, and Tom Gedeon. Simulating content consistent vehicle datasets with attribute descent. In *ECCV*, 2020. 7

[43] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *European Conference on Computer Vision*, pages 36–42. Springer, 2016. 3

[44] Hongyi Zhang, Andreas Geiger, and Raquel Urtasun. Understanding high-level semantics by modeling traffic patterns. In *Proceedings of the IEEE international conference on computer vision*, pages 3056–3063, 2013. 2

[45] 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. 4

[46] Zhilu Zhang and Mert R Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *arXiv preprint arXiv:1805.07836*, 2018. 2

[47] Zhedong Zheng, Xiaodong Yang, Zhiding Yu, Liang Zheng, Yi Yang, and Jan Kautz. Joint discriminative and generative learning for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2138–2147, 2019. 3

[48] Zhedong Zheng, Liang Zheng, and Yi Yang. A discriminatively learned cnn embedding for person reidentification. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 14(1):1–20, 2017. 4

[49] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020. 4

[50] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 4

[51] Yi Zhou and Ling Shao. Aware attentive multi-view inference for vehicle re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6489–6498, 2018. 3