International Journal of Computer Vision (2021) 129:3255–3278
https://doi.org/10.1007/s11263-021-01527-y

Deep Trajectory Post-Processing and Position Projection for Single & Multiple Camera Multiple Object Tracking

Cong Ma[1,2] · Fan Yang[1] · Yuan Li[1] · Huizhu Jia[1] · Xiaodong Xie[1] · Wen Gao[1]

Received: 12 December 2020 / Accepted: 30 August 2021 / Published online: 15 October 2021

Abstract
Multiple Object Tracking (MOT) has attracted increasing interests in recent years, which plays a significant role in video analysis. MOT aims to track the specific targets as whole trajectories and locate the positions of the trajectory at different times. These trajectories are usually applied in Action Recognition, Anomaly Detection, Crowd Analysis and Multiple-Camera Tracking, etc. However, existing methods are still a challenge in complex scene. Generating false (impure, incomplete) tracklets directly affects the performance of subsequent tasks. Therefore, we propose a novel architecture, Siamese Bi-directional GRU, to construct Cleaving Network and Re-connection Network as trajectory post-processing. Cleaving Network is able to split the impure tracklets as several pure sub-tracklets, and Re-connection Network aims to re-connect the tracklets which belong to same person as whole trajectory. In addition, our methods are extended to Multiple-Camera Tracking, however, current methods rarely consider the spatial-temporal constraint, which increases redundant trajectory matching. Therefore, we present Position Projection Network (PPN) to convert trajectory position from local camera-coordinate to global world-coodrinate, which provides adequate and accurate temporal-spatial information for trajectory association. The proposed technique is evaluated over two widely used datasets MOT16 and Duke-MTMCT, and experiments demonstrate its superior effectiveness as compared with the state-of-the-arts.

Keywords Trajectory post-processing · Position projection · Multiple object tracking · Multiple-camera multiple object tracking

1 Introduction

Multiple Object Tracking (MOT) is an important task in video surveillance analysis, which aims to locate the position of targets and associate specific targets as whole trajectories at all times, whose outputs (trajectories) are commonly utilized for Action Recognition, Anomaly Detection, Human Behavior Analysis, Crowd Analysis and Multiple-Camera Tracking, etc. However, MOT task still has many difficulties, for example, partial or long-term occlusion deteriorates the description of targets, which affects continuity of trajectory, some tracklets are split as several sub-tracklets when the tracklet is occluded by others; Frequent occlusion and cross-motion in crowd scene usually cause the neighboring target occludes the tracked-target, the target is gradually replaced by another target, and then the tracklet contains one more targets which make the tracklet impure. Furthermore, generating false (impure or incomplete) tracklets directly influences the subsequent tracklet-based tasks, such as Action Recognition and Multiple-Camera Tracking.

Communicated by Dong Xu.

✉ Huizhu Jia
hzjia@pku.edu.cn

Cong Ma
macong@sensetime.com

Fan Yang
fyang.eecs@pku.edu.cn

Yuan Li
yuanli@pku.edu.cn

Xiaodong Xie
donxie@pku.edu.cn

Wen Gao
wgao@pku.edu.cn

[1] National Engineering Laboratory for Video Technology, Peking University, Beijing, China

[2] Sensetime Research, Beijing, China

Therefore, we focus on trajectory post-processing strategy to cleave the impure tracklets and re-connect the same tracklets. In addition, our strategy applies not only in Single Camera Tracking, it is also extended to Multiple-Camera Tracking. However, for trajectory matching in multiple-camera tracking, the existing methods take less consideration on the positional relationship between cameras, which causes redundant trajectory matching process. As a result, the redundant matching affects the calculating efficiency and increases mis-matching probability of trajectory. In this paper, we present multiple stages Multiple-Camera Multiple Object Tracking (MCMOT) framework, which are divided into three parts (as shown in Fig. 1): 1. Single Camera Multiple Object Tracking; 2. Trajectory Position Projection; 3. Temporal-Spatial Constraint Trajectory Matching.

Single Camera Multiple Object Tracking (MOT) aims to identify each object and predict their trajectories in a single camera video sequence. MOT based methods address this problem by data association, which jointly optimize the matching process of bounding boxes detected by a detector within the inter-frames of a sequence. The same individual has regular temporal or spatial cues in video. For example, a person has slight appearance, velocity and direction changes. Therefore, MOT usually depends on the combination of multiple cues (e.g. appearance, motion and interactions) to associate the similar bounding boxes. Although the performance is gradually improving at the MOT Challenges Milan et al. (2016), the effectiveness of MOT is still limited by object detection quality, long-term occlusion and scene complexity. To solve this sophisticated problem, we intend to extract discriminative features, and design more effective association metrics for MOT.

Tracking-by-detection is a dominant solution for MOT, which links similar objects into trajectory by associating their feature representation and bounding box position. Tracking-by-detection is to search the optimal assignment from multiple cues within a set of bounding boxes. For example, the appearance and motion of person are discriminative cues for data association. Currently, deep networks have achieved significant performance improvement in MOT Tang et al. (2017); Chu et al. (2017); Sadeghian et al. (2017). However, some difficulties remain unresolved, such as "mis-tracking" and "lost-tracking". As shown in Fig. 6, a tracked person is gradually occluded by another person, which causes mis-tracking. As shown in Fig. 7 , a tracklet is split into several fragments, long-term occlusion leads to lost-tracking. Thus trajectory post-processing becomes particularly important for multiple-camera tracking.

Multiple-Camera Multiple Object Tracking focuses on associating trajectories from different cameras' tracking results (illustrated as Fig. 2). Duke-MTMCT Ristani et al. (2017) is a typical example for MCMOT task, which includes 8 cameras in different locations and viewpoints. The existing

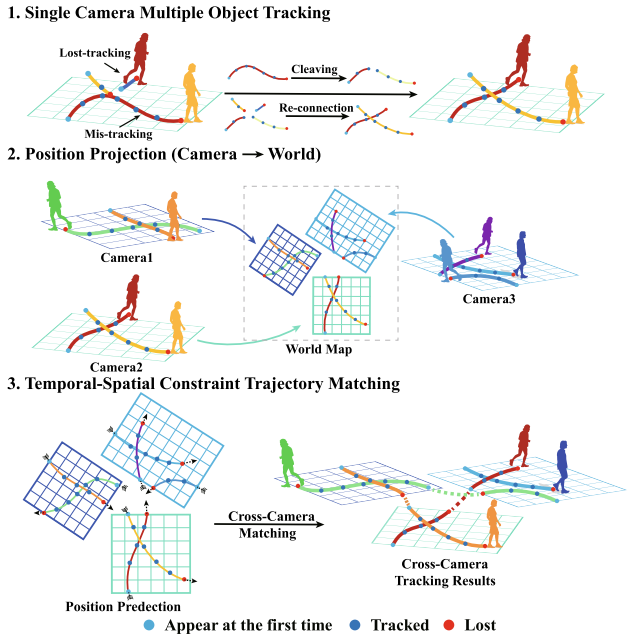

Fig. 1 The Framework of Multiple-Camera Multiple Object Tracking: 1. Single camera multiple object tracking with tracklet cleaving and re-conection. 2. Position Projection for each trajectory. 3. Cross-camera trajectories matching based on temporal-spatial constraint

methods of trajectory matching overly depend on extracting tracklet appearance features, which is regarded as Person Re-identification task in general. However, due to the difference of illumination, the angle of camera and body posture, appearance features are not sufficient to match trajectory. It increases huge amount of operations to match each trajectory one by one and improves the probability of mis-matching. Therefore, incorporating temporal-spatial information eliminates redundant amount of matching operations, and improves matching accuracy.

Temporal-spatial information contains position, velocity, timestamp and camera ID. The information is utilized for predicting the trajectory position in the future and then narrowing the search area. The relative locations between cameras are also conducive to matching trajectories. Due to the differences of camera locations and viewpoints, we have to correlate the trajectory positions between camera and actual scenario. For example, as shown in Fig. 2, both camera No.5 and No.7 are adjacent. When the person disappears from the bottom of the camera No.5, it will most likely appear from the left side of the camera No.7. For previous methods, to correlate cameras, traditional method Jiang et al. (2018) constructed a topological graph based on camera locations and viewpoints, which needs to calibrate each camera. However if some cameras are changed, they have to re-calibrate and re-construct the topological graph. Therefore, it is necessary to convert the trajectory position to the uniform coordinate such as world-coordinate.

Fig. 2 The illustration of Multiple-Camera Multiple Object Tracking (MCMOT) task. The Tracking Dataset is from Duke-MTMCT, which includes 8 cameras in different location and view, the middle bottom image indicates the overall map

In this paper, we construct multi-stage framework to reduce the number of mis-tracking and lost-tracking. Our proposal consists of several independent modules which includes tracking module and post-processing module. We propose a novel architecture, Siamese Bi-directional GRU (SiaBiGRU) for trajectory post-processing. Based on the SiaBiGRU, we design Cleaving Network to check the purity of tracking and split impure tracklets, and address Re-conection Network to link sub-tracklets as trajectory. Additionally, in order to verify whether our post-processing model can improve the effect of subsequent tasks, we select multiple-camera tracking as subsequent task. In trajectory matching phase, we construct a Position Projection Network (PPN) to convert the trajectory location from camera-coordinate to world-coordinate. Finally, we reduce the search range according to the trajectory motion prediction in world-coordinates and associate the trajectories from different cameras by their general appearance features. The proposed method is divided into three steps as illustrated in Fig. 1: (1.Single Camera Multiple Object Tracking, 2.Trajectory Position Projection, 3.Multiple Camera Trajectory Matching), where Single Camera Tracking (the first step) includes three sub-steps: (a.Tracklet Generation, b.Tracklet Cleaving, c.Tracklet Re-connection, as shown in Fig. 3). Our contributions in this paper are shown as follows:

- We design a novel multiple-stages framework for Multiple-Camera Multiple Object Tracking (MCMOT) which includes "trajectory processing" in single camera and temporal-spatial based trajectory association in multiple-camera scene.
- For post-processing, we propose a novel Siamese Bi-directional GRU (SiaBiGRU) to cleave the impure tracklets into sub-tracklets and re-connect these sub-tracklets according to their similarity.
- For cross-camera trajectory matching, we present a Position Projection Network, which effectively leverages temporal-spatial information by converting the trajectories location from camera-coordinate to world-coordinate.
- The proposed model greatly reduces the amount of trajectory matching and further decreases the number of mis-matching. Experiments demonstrate its superior effectiveness and robustness over the state-of-the-arts in MOT Benchmark.

2 Related Work

2.1 Single Camera Multiple Object Tracking

Single Camera Multiple Object Tracking in videos has attracted great attention. Single camera MOT generates trajectories corresponding to each object in a video sequence that is captured by a single camera. The main strategy is to guide object tracking by detection. For example, Tang et al. (2017); Xiang et al. (2016); Choi (2015); Kim et al. (2015); Chen et al. (2017) focus on designing an ingenious data association or multiple hypothesis. Schulter et al. (2017); Levinkov et al. (2017); Maksai et al. (2017) rely on network flow and graph optimization which are powerful approaches for tracking. Bergmann et al. (2019) exploited the bounding

box regression of an object detector to predict the position of an object to convert a detector into a Tracktor. Liu et al. (2020) proposed a novel graph representation that takes both the feature of individual object and the relations among objects into consideration. Xiang et al. (2020) presented an end-to-end conditional random field within a unified deep networks. The inter-relation of targets has multiple cues in a sequence including appearance, motion and interaction, which are summarized by Sadeghian et al. (2017). Some scholars have carried out research on tracking cluster and post-processing to improve the tracking performance. For example, Zhang et al. (2020) constructed motion evaluation network and appearance evaluation network to learn long-term features of tracklets for association. Peng et al. (2018) adopted a constrained clustering to piece tracklets according to appearance characteristic of tracklet, and Peng et al. (2020) utilized Box-Plane matching strategy to achieve association.

The appearance model aims to extract person features. For example, Le et al. (2016); Yang Hoon et al. (2016) adopt the appearance model of some early traditional algorithms such as color histogram to represent the image features, or that Choi (2015); Bae and Yoon (2014); Yang and Jia (2016) utilize covariance matrix or hand-crafted keypoint features. Henschel et al. (2017) uses a novel multi-object tracking formulation to incorporate several detectors into a tracking system. Kim et al. (2015) extended the multiple hypothesis by enhancing detection model. Ma et al. (2019) presented an end-to-end deep learning framework for MOT. With the development of deep learning model Zhuang et al. (2018, ?); Hou et al. (2020); Guo et al. (2020); Li et al. (2020), CNN are gradually utilzed in MOT. Tang et al. (2017); Sadeghian et al. (2017) train the CNN on the basis of person re-identification strategy Liu and Zhang (2020, 2021); Liu et al. (2019) to extract the image features, and Son et al. (2017) utilized the quadruplet loss to enhance the feature expression. Chu et al. (2017) builds the CNN model to generate visibility maps to solve the occlusion problem. Wang et al. (2016); Bae and Yoon (2014) are presented to improve the tracklet association and tracklet confidence to perform the tracklet task. Ma et al. (2021) adopted human-interaction model to improve the representation of targets in crowd scene.

The motion model defines the rule of object movement, which is divided into linear position prediction Son et al. (2017) and non-linear position prediction Dicle et al. (2013). Zhu et al. (2018); Gao and Jiang (2016) proposed spatial and temporal attention mechanisms to enhance the performance of MOT. Following the success of RNN models for sequence prediction tasks, Alahi et al. (2016) proposed social-LSTM to predict the position of each person in the scene. The interaction model described the inter-relationship of different pedestrians in the same scene. Yang Hoon et al. (2016) designed the structural constraint by the location of people to optimize assignment. In addition, Henschel et al. (2018) used a novel multi-object tracking formulation to incorporate several detectors into a tracking system. Ma et al. (2018) addressed a sophisticated model to process trajectories.

2.2 Multiple-Camera Multiple Object Tracking

Given the trajectories generated by the single camera MOT, cross-camera MOT further associates trajectories corresponding to the same object that are captured by different cameras. Ristani and Tomasi (2018); Jiang et al. (2018); Yoon et al. (2016); Tesfaye et al. (2017); Zhang et al. (2017); Maksai et al. (2017) are evaluated at the Duke-MTMCT Ristani et al. (2017) benchmark. Maksai et al. (2017) presented a Non-Markovian method to impose global consistency by using behavioral patterns to guide the tracking algorithms. Ristani and Tomasi (2018) proposed an adaptive weighted triplet loss for training and a new technique for hard-identity mining on extracting appearance feature. Yoon et al. (2016) applied a multiple hypothesis tracking (MHT) to handle the tracking problem with disjoint views. Jiang et al. (2018) addressed an orientation-driven person ReID and an effective camera topology estimation based on appearance feature for online inter-camera trajectory association. Cai and Medioni (2016) operated by comparing entry/exit rates across pairs of cameras. Berdereck et al. (2012) relied on completely overlapping and unobstructed views. Chen et al. (2011) built an adaptive and unsupervised method for a camera network, it can incrementally refine the clustering results of the entry/exit zones and the transition time probability distributions. Tesfaye et al. (2017) proposed a unified three-layer hierarchical approach for solving tracking problems in multiple non-overlapping cameras. [51] designed a system of multiple interacting targets in a camera network which decides the group state of each trajectory.

3 Single Camera Multi-Object Tracking

To generate accurate and robust trajectories from every single camera, we design multiple-stage single camera MOT framework, which is divided into three modules (as illustrated in Fig. 3): A. tracklet generation aims to generate tracklet candidates using bounding boxes of appearance and motion features; B. tracklet cleaving aims to estimate suitable split positions for impure tracklets; C. tracklet re-connection aims to associate the sub-tracklets which belong to the same person. The cleaving and re-connection processes are tracklet-to-tracklet based method. Fig. 5 shows the architecture of cleaving network and re-connection network. In this Section, the data association metric which generates tracklets from relatively sparse scenario as the tracklet candidate is described in Sect. 3.1(A). We present the reason why the algorithm mis-tracks the other people and how to estimate

Fig. 3 Single Camera MOT are divide into three steps: 1.Tracklet Generation: Tracklets are generated by appearance and motion feature 2.Tracklet Cleaving: The impure tracklets are split by cleaving network 3.Tracklet Re-connection: tracklets which belong to the same person are linked by re-connection network

the tracklet reliability and split the impure tracklets in Sect. 3.2(B). Section 3.3(C) gives the traclets re-connection and association strategy, moreover, the training method of our network is also discussed.

3.1 Tracklet Generation

Tracklet Generation is a online sequential process that associates the bounding boxes of high similarity frame by frame to generate tracklet candidates (as described in Fig. 4). The set of nodes (bounding boxes) are composed of detection $\mathcal{D}$ and bounding box candidates $\mathcal{C}$. We denote the set of detection bounding boxes $\mathcal{D}_t$ ($d_t^k \in \mathcal{D}_t$), where d_t^k indicates the k-th detection bounding box in frame t. $\mathcal{C}_t$ ($c_n^k \in \mathcal{C}_t; n \leq t, \mathcal{C}_t = \mathcal{C}_{t-1} \bigcup \mathcal{D}_{t-1}$) denotes the set of tracked object candidates, where c_t^k is the k-th candidate in frame t (all of the red dots which include unassigned dots and previous residual of $\mathcal{D}_{t-1}$ for t-th frame in Fig. 4 are attributable to $\mathcal{C}_t$). To connect the candidates and detection within inter-frames, we match the candidates c_t^k and d_t^k in a bipartite graph with Hungarian algorithm Sahbani and Adiprawita (2017). The bipartite graph $G = (\mathcal{V}, \mathcal{E})$ whose nodes V are divided into left part $\mathcal{C}_t \in \mathcal{V}_L$ and right part $\mathcal{D}_t \in \mathcal{V}_R$, $e_{ij} \in \mathcal{E}$ is the edge of c_t^i and d_t^j. Each node (bounding box) is defined as 10 dimensions $[cid, id, t, x, y, w, h, wx, wy, s]$, which represents the camera id, tracklet id by tracker, the bounding box frame, the left-top position (x, y), width and height of the bounding box, the world-coordinate (wx, wy) and the state of the tracklet, respectively. The state of tracklet includes "tracked", "lost" and "quit", which are similar to Markov Decision Processes Xiang et al. (2016). If the node is associated by another node in the next frame, the node statement is labeled "tracked" (the blue and indigo dots in Fig. 4). On the contrary, due to objects "escaping" the sight, the unassigned nodes are labeled "lost" (the red dots in Fig. 4, lost node generally appears at the tail of tracklet). We define search interval length to be η_s frames, if the "lost" node is found to be associated within η_s frames, its state changes from "lost" to "tracked", otherwise, the nodes from whole tracklet states are labeled "quit". To generate tracklet candidates, the bipartite graph's edge weights between nodes are defined as $e_{ij} = \mathcal{S}(c_t^i, d_t^j)$, where $\mathcal{S}(c_t^i, d_t^j)$ estimates the similarity between two nodes. The overall cost function can thus be determined as follows:

$$\mathcal{S}(c_t^i, d_t^j) = \lambda_a F_a(c_t^i, d_t^j) + \lambda_m F_m(c_t^i, d_t^j) \quad (1)$$

$$F_a(c_t^i, d_t^j) = 1 - cos(f_{c_t^i}, f_{d_t^j}) \quad (2)$$

$$F_m(c_t^i, d_t^j) = \| \hat{p}_{c_t^i} - p_{d_t^j} \|_2^2 \quad (3)$$

where $F_a(c_t^i, d_t^j)$ denotes the appearance similarity between c_t^i and d_t^j. Function $cos(A, B)$ is formulated as $\frac{A \cdot B}{|A| \cdot |B|}$. The features $f_{c_t^i}$, $f_{d_t^j}$ are extracted by appearance model. λ_a, λ_m are the weight coefficients of the function. $F_m(c_t^i, d_t^j)$ estimates the motion distance between the detection position $p_{d_t^j}$ and candidate prediction position $\hat{p}_{c_t^i}$, which is defined in 4 dimensions $[\hat{x}, \hat{y}, \hat{w}, \hat{h}]$ that stand for the prediction of x, y-coordinate, width and height, respectively.

Appearance model extracts the pedestrian appearance features (e.g. color, shape and texture). Coherent understanding of the pedestrian appearance and further discriminative feature representation are essential for node matching in MOT. We adopt the Person Re-identification method as the appearance model Luo et al. (2019). The total loss of appearance model is defined as

$$\mathcal{L}_{app} = \lambda_{id}\mathcal{L}_{id} + \lambda_{tri}\mathcal{L}_{tri} + \lambda_c\mathcal{L}_c \quad (4)$$

which combines three types of loss (ID Loss: $\mathcal{L}_{id}$ Zheng et al. (2018), Triplet Loss: $\mathcal{L}_{tri}$ Hermans et al. (2017) and Center Loss: $\mathcal{L}_c$ Wen et al. (2016)) together to train the appearance model, where the $\lambda_{id}, \lambda_{tri}$ and λ_c are the loss weights to adjust training effectiveness. Appearance Extraction is treated as the multi classification task, which aims to classify the embedding feature of person image in the hyperspace.

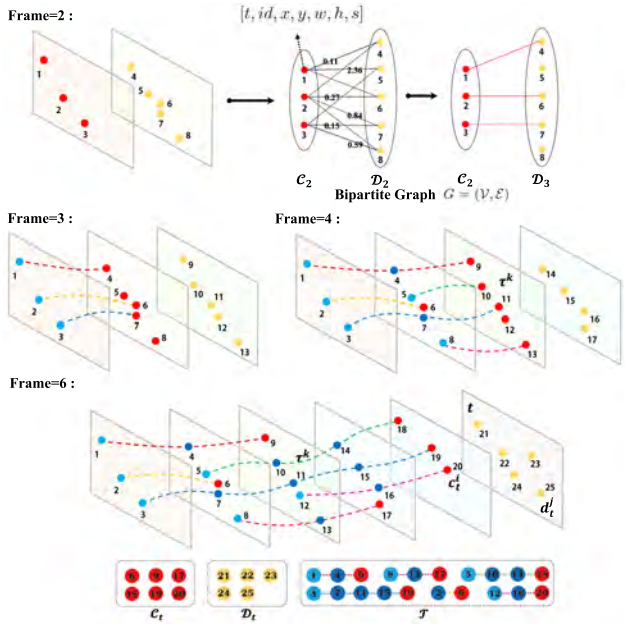

Fig. 4 The process of Tracklet Generation, the tracker aims to associate the bounding box frame by frame. The yellow and red dots for t-th frame belong to $\mathcal{D}_t$ and $\mathcal{C}_t$, respectively. The blue and indigo dots indicate the 'tracked' nodes, where blue dot is the initial position of tracklet. For each step, we construct the Bipartite Graph G_t by c_t^i, d_t^j and optimize the graph, and then corresponding c_t^i, d_t^j are updated to c_{t+1}^i, d_{t+1}^j for next step. We show the processing result of the first six frames (omitting the 5-th frame). The last frame corresponding C_6, D_6 and $\mathcal{T}_6$ are shown at bottom of figure

The ID loss is formulated as:

$$\mathcal{L}_{id} = \sum_{v=1}^{N} -q_v log(\hat{q_v}), \quad \hat{q_v} = softmax(f_{cls,v}) \tag{5}$$

where $\mathcal{L}_{id}(f_{cls,v})$ indicates the cross-entropy loss, $f_{cls,v}$ is the classification feature after the output of the CNN f_v by fully-connected layer, where v represents the node from C_t^i or D_t^i $\hat{q_v}$ denotes the predict probability of classification, which is the output of the softmax. N and q_v indicate the number of class and ground truth label, respectively.

The triplet loss is a metric learning, given an anchor node v_a, and the corresponding embedding feature f_{v_a}, and we select the same class node v_p and different class node v_n as the positive and negative nodes, respectively. The triplet loss is defined as:

$$\begin{aligned} &\mathcal{L}_{tri} = [d_p - d_n + \alpha]_+ \\ &d_p = \| f_{v_a} - f_{v_p} \|_2^2, \quad d_n = \| f_{v_a} - f_{v_n} \|_2^2 \end{aligned} \tag{6}$$

where the d_p and d_n indicate the Euclidean distance of positive pair and negative pair, α is the distance threshold and $[*]_+$ is equivalent to function $max(*, 0)$.

The center loss is defined as:

$$\mathcal{L}_c = \frac{1}{2} \sum_{i=1}^{m} \| f_v - c_{y_v} \|_2^2 \tag{7}$$

where m is the number of batch size, and y_v indicates the label of the v-th image in a mini-batch, c_{y_v} denotes the y_v-th class center of deep features.

After training appearance model, the output of CNN model f_v is L_2 normalized and utilized for Eq. 2 to calculate the similarity of appearance cue.

Motion model analyzes the pedestrian movement rule and predict the position in the future. The inputs of motion model include historical location of tracklet and its corresponding timestamp. The architecture of motion model is a LSTM, which is able to learn sequential data. We construct tracklets ground truth position as the LSTM training set. The inputs of LSTM are the tracklet historical position $[p_t^u, p_{t+1}^u, p_{t+2}^u, ...]$, where p_t^u means the u-th tracklet posotion in frame t. The outputs of LSTM are the predicted positions in the next frame, $[\hat{p}_{t+1}^u, \hat{p}_{t+2}^u, \hat{p}_{t+3}^u, ...]$. We use the actual position of tracklet to supervise the LSTM. The position loss is described as

$$\mathcal{L}_{mot} = \sum_{i=1}^{L^u - 1} \| \hat{p}_i^u - p_i^u \|_2^2 \tag{8}$$

where L_u is the length of u-th tracklet. we compute the distance between the predicted and the actual position. After training motion model, the position of each frame is input into the LSTM to generate the position for future frames.

After bipartite graph construction, we adopt Hungarian Algorithm Sahbani and Adiprawita (2017) to optimize the bipartite graph and obtain association results. To evaluate the performance of optimized results, we define a target function F_G to calculate the differences between the ground truth graph G_t^{gt} and the optimized graph result $\hat{G}_t$, which is given by

$$\begin{aligned} &F_G(\hat{G}_t, G_t^{gt}) = \\ &\sum_{e_{ij}^{gt} \in \mathcal{E}_t^{gt}} (e_{ij}^{gt} - \hat{e}_{ij}) + \sum_{e_{ij}^{gt} \notin \mathcal{E}_t^{gt}} \sigma * (\hat{e}_{ij} - e_{ij}^{gt}) \end{aligned} \tag{9}$$

e_{ij}^{gt} indicate the ground truth connection between node i and j in G_t^{gt}, the Eq.9 is divided into two parts by the plus ("+"). The left part $e_{ij}^{gt} \in \mathcal{E}_t^{gt}$ means that c_t^i and d_t^j are associated, $e_{ij}^{gt} \equiv 1$, and vice versa, $e_{ij}^{gt} \equiv 0$ at the right part. $\hat{e_{ij}} = \{0, 1\}$ is the optimized edge of G_t. If the $\hat{e_{ij}} = 0$ at the left part, but $e_{ij}^{gt} = 1$ in ground truth, the same targets are not connected, the tracked target will be lost (lost-tracking). If the $\hat{e_{ij}} = 1$ at the right part, but $e_{ij}^{gt} = 0$ in ground truth, the different

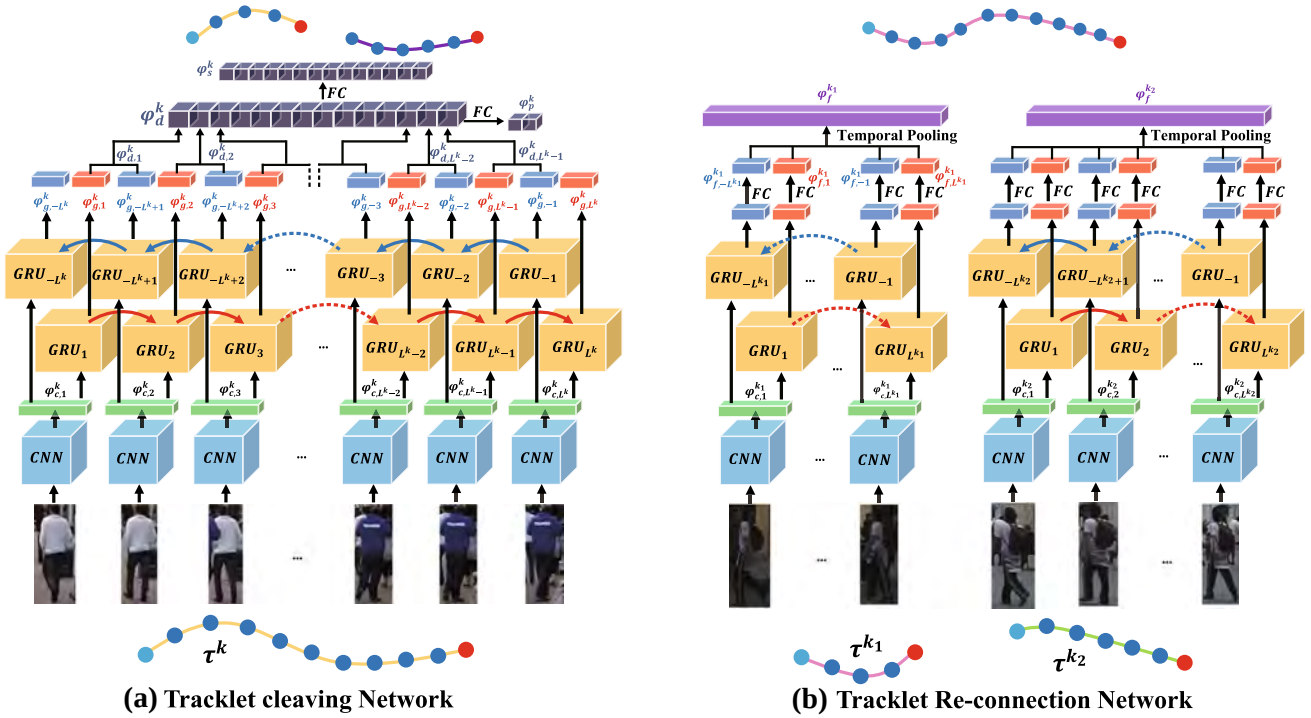

Fig. 5 The architecture of tracklet cleaving and re-connection network, **a** Cleaving the tracklets by bidirectional outputs of GRU, **b** Re-connecting the tracklets by the features of siamese GRU.

targets are connected, the tracked target will be switched to other target (mis-tracking). Since the negative effects of mis-tracking is greater than lost-tracking, we define a weight σ to focus more on mis-tracking, σ is set to 2. Finally, we select the optimal models with minimum score (output of Eq.9) from all of models as Tracklet Generation model.

The generated tracklets $\tau^k \in \mathcal{T}$ have the following attributions: τ^k: $[\tau^k[id], \tau^k[t_s], \tau^k[t_e], \tau^k[\mathbf{v}_s], \tau^k[\mathbf{v}_e], \tau^k[l], \tau^k[r_t^k]_{l\times 10}, \tau^k[\mathbf{v}_t^k]_{l\times 1}, \tau^k[s]]$, which are tracklet id, start frame, end frame, start velocity, end velocity, tracklet length and all of the nodes, where nodes $\tau^k[r_t^k]$ is a $l \times 10$ matrix, which contains all of the node information (each row stands for a node), $\tau^k[\mathbf{v}_t^k]_{l\times 1}$ records the velocity at all times. $\tau^k[s]$ is state of the tracklet, which includes "tracked", "lost" and "quit".

3.2 Tracklet Cleaving

After Tracklet Generation, we have a set of tracklet $\mathcal{T}$ in sequence. However, the Tracklet Generation may mis-track the wrong person when two persons have cross-motion or occlude each other, which degrades the generated tracklet purity. Fig. 6 shows an example of an impure tracklet, when another person (blue shirt) gradually occludes the target person (white shirt). Even though the two pedestrians present different apperance, when a large area of the target is occluded by the occluder, the bounding boxes related

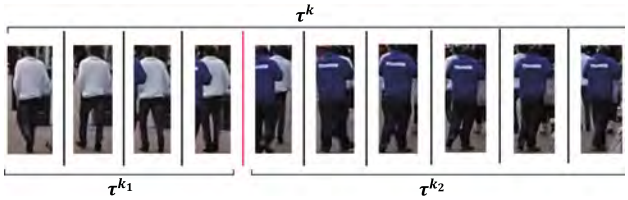

Fig. 6 The example of impure tracklet (mis-tracking), cleaving network aims to split the impure tracklet as two pure tracklet. The pink line is the most suitable split position for the example

to target and the occluder are indistinguishable. Traditional methods only considers the bounding box in short-term adjacent frames. As the result, the target is replaced by occluder. Meanwhile, due to the two pedestrians being that the same position, the appearance and motion model on tracklet generation are not able to check whether tracker is mis-tracking.

To guarantee the tracklet being the same person, we design a bidirectional output Gated Recurrent Unit to estimate the tracklet purity and cleave the false tracklets. Tracklet Cleaving is an end-to-end training network to check whether the tracklet is pure and search the suitable split position of impure tracklet. We define the pure tracklets $\mathcal{T}^+$ and impure tracklets $\mathcal{T}^-$ as:

$$\begin{cases} \tau^k \in \mathcal{T}^+ \ \forall i, j, r_i^k, r_j^k \in \tau^k, r_i^k(id) \equiv r_j^k(id) \\ \tau^k \in \mathcal{T}^- \ \exists i, j, r_i^k, r_j^k \in \tau^k, r_i^k(id) \neq r_j^k(id) \end{cases} \tag{10}$$

where r_i^k, r_j^k are the i-th, j-th elements on tracklet τ^k. All of the tracklets $\tau^k \in \mathcal{T}, k \in K$ are fed into the Cleaving Network to check purity of the tracklets, and search the best split position of impure tracklets. The tracklet Cleaving Network is shown in Fig. 6. First of all, we utilize the CNN to extract the image features $\varphi_{c,i}^k,\ i \in [1, L^k]$ from the tracklet, L^k is the k-th length of tracklet. Secondly, all the features $\varphi_{c,i}^k$ are input into the forward-GRU and backward-GRU, respectively. Both GRUs have the shared weights, and the output is $\varphi_{g,i}^k,\ i \in [-L^k, -1] \cup [1, L^k]$, the positive and negative superscript values stand for forward and backward features from GRU. And then, we calculate the adjacent vectors distance between the features from the forward and the backward(e.g. length=10, $\{\varphi_{g,1}^k, \varphi_{g,-9}^k\}, \{\varphi_{g,2}^k, \varphi_{g,-8}^k\}, \ldots$) as a series of feature distance to concatenate as an $1 \times (L^k - 1)$ vector φ_d^k:

$$\begin{aligned} \varphi_{d,i}^k &= \| \varphi_{g,i}^k - \varphi_{g,i-L^k}^k \|_2^2,\ i \in [1, L^k - 1] \\ \varphi_d^k &= ([\varphi_{d,1}^k, \varphi_{d,2}^k, ..., \varphi_{d,L^k}^k]) \end{aligned} \tag{11}$$

The algorithm calculates the distance $\varphi_{d,i}^k$ between the features from the left to current position i and the right to corresponding position $i - L^k$. $\varphi_{d,i}^k$ is fed into two fully-connected layers separately after normalization. First output of the FC layer φ_s^k is used for searching the most suitable split position, which generally appears at the maximum disparity from these distances. The output of other FC layer φ_p^k is utilized for checking whether the tracklet is pure. The advantage of GRU is to be able to summarize the general characteristics with the same person and eliminate occlusion in order to obtain preferable feature expression. For training Cleaving Network, the cleaving loss $\mathcal{L}_{clv}$ is defined as:

$$\mathcal{L}_{clv} = \lambda_f \mathcal{L}_{c,ftr} + \lambda_s \mathcal{L}_{c,srh} + \lambda_p \mathcal{L}_{c,pur} \tag{12}$$

The feature loss $\mathcal{L}_{c,ftr}$ calculates the difference between each output of GRU $\varphi_{f,i}^k,\ i \in [-L^k, -1] \cup [1, L^k]$ and the ground truth. The searching loss $\mathcal{L}_{c,srh}$ measures distance between the predict split position and real split position. The purity loss is a Binary task to differentiate whether the tracket is pure. Eq.12 is expanded as:

$$\begin{aligned} \mathcal{L}_{clv} = \sum_{k \in K} \Big(\frac{\lambda_f}{2L^k} \Big(\sum_{i=-L^k}^{-1} F_\xi(\varphi_{g,i}^k) + \sum_{j=1}^{L^k} F_\xi(\varphi_{g,j}^k) \Big) \\ + \lambda_s F_\iota(\varphi_s^k) + \lambda_p F_2(\varphi_d^k) \Big) \end{aligned} \tag{13}$$

where λ_* is the loss weight coefficient, K indicates the number of the tracklets in training set, ξ and ι denote the number of trakclet class and length of k-th tracklets, respectively. F_N ($N = \{\xi,\ \iota,\ 2\}$) indicates Cross-Entropy loss:

Fig. 7 The example of fragmented tracklets (lost-tracking), re-connection network aims to connect the fragmented tracklets as a whole tracklet

$$F_N = \sum_{i=1}^{N} -q_i log(\hat{q}_i), \quad \hat{q}_i = softmax(\varphi) \tag{14}$$

Pure Loss $\mathcal{L}_{c,pur}$ is a binary classification task, which supervises φ_p^k to judge the purity of a tracklet. Search Loss $\mathcal{L}_{c,srh}$ is a multiple classification task, which is utilized for optimizing φ_s^k to search the best splitting position. Search Loss $\mathcal{L}_{c,srh}$ can also be defined by L1-Loss to calculate the differences between prediction split position and ground truth.

When the training cleaving network is completed, the network is used for checking the tracklets $\tau^k \in \mathcal{T}$ generated from tracklet generation part. In general, the most suitable split position occurs on the maximum feature distance between left-side and right-side for the position. If the τ^k belongs to impure tracklet, it will be split into two tracklets τ^{k_1} and τ^{k_2}. To guarantee the tracklet purity, the tracklet cleaving step is vital for tracklet description and cross-camera trajectory matching.

3.3 Tracklet Re-connection

After Tracklet Cleaving step, we obtain some pure tracklets. However, long-term occlusion usually breaks the whole tracklet, frequent occlusion in crowd produces the lost-tracking and id-switch for tracklets, and illumination variation influences the appearance feature description and then affects the tracking performance. Fig. 7 is an example of tracklet fragments, which belong to the same person.

Re-connection focuses on extracting the general features of tracklets and calculating similarity between tracklets and connecting fragmented tracklets as a whole tracklet. The architecture of re-connection network is shown in Fig. 5. For matching tracklets, each tracklet is extracted the general feature to describe the tracklet appearance characteristics. The tracklets with similar general feature will be re-connected. We combine various losses to reduce the within-class distance and enlarge the between-class distance, simultaneously. Our network is designed with the verification loss and identification loss at each GRU output. The re-connection loss is defined as:

$$\mathcal{L}_{rcn} = \mathcal{L}_{glo} + \mathcal{L}_{loc} \tag{15}$$

Where the $\mathcal{L}_{glo}$ and $\mathcal{L}_{loc}$ indicate the global loss and local loss of the network, respectively. We use the contrastive loss by Euclidean distance for the verification and the cross-entropy losses in the multi-classification task for the identification. The identification loss $F(*)$ is the same as the Eq.14. The details of the contrastive loss $E(*)$ are shown as:

$$E(\varphi_f^{k_1}, \varphi_f^{k_2}) = y \parallel \varphi_f^{k_1} - \varphi_f^{k_2} \parallel_2^2 + (1-y)max\{0, (\eta - \parallel \varphi_f^{k_1} - \varphi_f^{k_2} \parallel_2^2)\} \tag{16}$$

where $\varphi_f^{k_*}$ indicates the output feature of GRU $\varphi_g^{k_*}$ after fully-connected (FC) and ReLU. $E(\varphi_i, \varphi_j)$ is contrastive function, $y \in \{0, 1\}$ is the label indicator, η is a margin constant. The representation of loss can be formulated as follows:

$$\begin{aligned}\mathcal{L}_{glo} &= \lambda_v \mathcal{L}_v + \lambda_{id}(\mathcal{L}_{id1} + \mathcal{L}_{id2}) \\ &= \lambda_v E(\varphi_f^{k_1}, \varphi_f^{k_2}) + \lambda_{id}(F(\varphi_f^{k_1}) + F(\varphi_f^{k_2}))\end{aligned} \tag{17}$$

$$\varphi_f^k = \frac{1}{2L^k}\Big(\sum_{i=-L^k}^{-1} \varphi_{f,i}^k + \sum_{j=1}^{L^k} \varphi_{f,j}^k\Big) \tag{18}$$

where φ_f^k is the temporal pooling McLaughlin et al. (2016) of each output of GRU.

$$\mathcal{L}_{loc} = \lambda_{loc\_v}\mathcal{L}_{loc\_v} + \lambda_{loc\_id}\mathcal{L}_{loc\_id} \tag{19}$$

$$\begin{aligned}\mathcal{L}_{loc\_v} =& \parallel \varphi_{f,1}^{k_1} - \varphi_{f,L^{k_1}}^{k_1} \parallel_2^2 + \parallel \varphi_{f,1}^{k_2} - \varphi_{f,L^{k_2}}^{k_2} \parallel_2^2 \\ &- \parallel \varphi_{f,1}^{k_1} - \varphi_{f,1}^{k_2} \parallel_2^2 - \parallel \varphi_{f,L^{k_1}}^{k_1} - \varphi_{f,L^{k_2}}^{k_2} \parallel_2^2 + \delta\end{aligned} \tag{20}$$

$$\mathcal{L}_{loc\_id} = \sum_{k \in k_1, k_2} \Big(\sum_{i=-L^k}^{-1} F(\varphi_{f,i}^k) + \sum_{j=1}^{L^k} F(\varphi_{f,j}^k)\Big) \tag{21}$$

λ is the loss weight coefficient. $\mathcal{L}_v$, $\mathcal{L}_{id*}$, $\mathcal{L}_{loc\_v}$ and $\mathcal{L}_{loc\_id}$ denote the verification and identification loss of global and local, respectively. $\mathcal{L}_{loc\_v}$ is similar to triplet loss (refer to Son et al. (2017)), including the disparity of head and tail of the tracklet, head between different tracklets and tail between different tracklets. δ is the threshold of margin. $\mathcal{L}_{loc\_id}$ is the multi-classification task for each output.

When the training of re-connection network is completed, the network is utilized for extracting tracklet general feature, and the tracklet re-connection step compares several tracklets and connects tracklets $\{\tau^{k_1}, \tau^{k_2}, \tau^{k_3}, ...\}$ which belong to the same person as the whole tracklet τ^k.

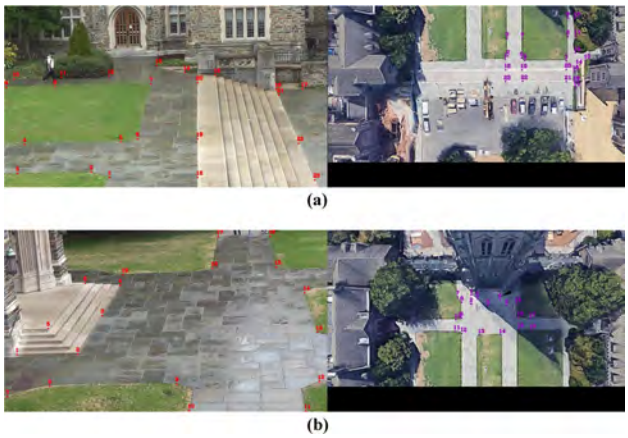

Fig. 8 Annotated point-pairs for training Position Projection Network , the images are from Duke-MTMCT, where (a) is camera 2 and (b) is camera 5

4 Multiple-Camera Tracking

For Multiple-Camera Multiple Object Tracking, we aim to associate the trajectories from different cameras. Each trajectory is generated from the single camera tracking. Section 4.1(A) introduces how to train Position Projection Network (PPN) and position conversion of the trajectories from camera-coordinate to world-coordinate. Section 4.2(B) discusses how to eliminate redundant matching operation by temporal-spatial constraint. Section 4.3(C) describes how to associate the trajectories by similarity of general appearance feature.

4.1 Position Projection

Position Projection focuses on transferring each tracklet position from camera-coordinate to world-coordinate. We propose a deep learning method called Position Projection Network (PPN) to project each point position. Given the input position (camera-coordinate) (σ_x, σ_y), the target (output) of the network is the world-coordinate (ω_x, ω_y). We treat the position projection as a fitting task. Compared with the traditional method such as Geometric Camera Calibration Hartley and Zisserman (2003), our method only annotates a few point-pairs between camera image and world map to train PPN instead of calculating the translation matrix and rotation matrix.

Figure 8 shows how to annotate and generate more point-pairs for the training set. The left and right images denote the camera view image and world map. The same point number between camera image and world map indicates the same position. For example Fig. 8a, we only mark 23 points on camera image and world map, respectively, and we choose the representative locations as the point-pairs such as corner-points. Duke-MTMCT dataset has 8-camera sequences, we

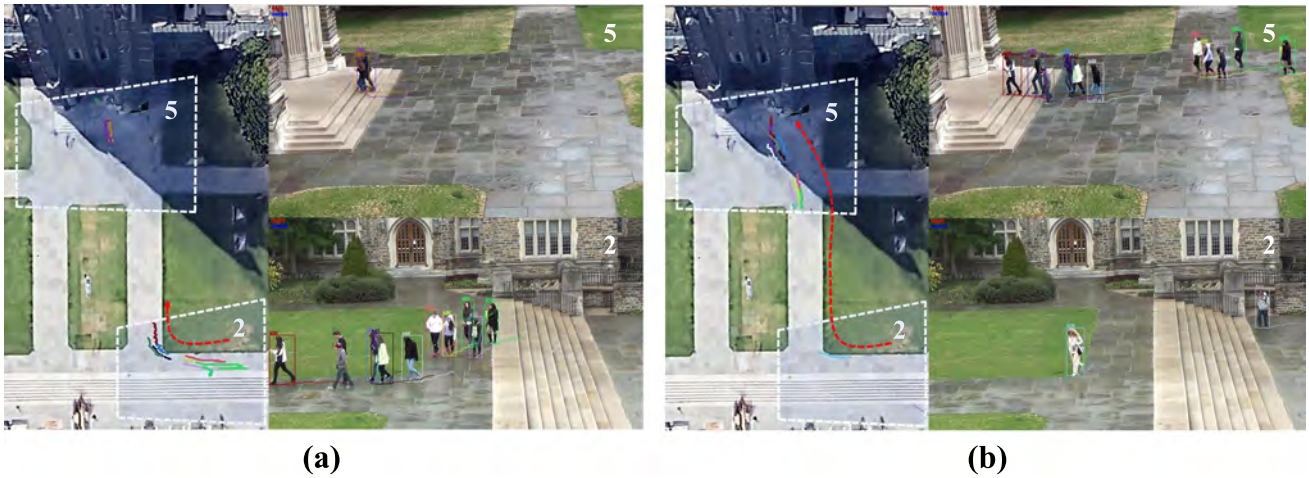

Fig. 9 The example of Position Projection, the figure shows the tracklets position from camera2 (bottom-right) and camera5 (top-right) transferred to world map (left), **a** and **b** are the tracking and projection results at the 246025-th and 247216-th frame (The calibration-frame) respectively

annotate 8 groups of point-pairs respectively, 8-camera coordinates are projected to the same world map. After point-pair annotation, we get several point-pairs, of which is a vector ϱ_i: $[cid, i, \sigma_x^i, \sigma_y^i, \omega_x^i, \omega_y^i]$, where cid indicates the camera number and i denotes the point number.

In order to improve the precision of projection, we present an interpolation method to enlarge the point-pairs. We divide the scene into several areas according to plane, for example Fig. 8a, points $\varrho_a \in \mathcal{A}_1^0$, a: $[1 \rightarrow 14, 18, 19, 20]$ belong to the same plane, points $\varrho_b \in \mathcal{A}_2^0$, b: $[15, 16, 18 \rightarrow 23]$ belong to the other plane, points $\varrho_c \in \mathcal{A}_3^0$, c: $[16, 17, 21, 22, 23]$ belong to the third plane. For each Area $\mathcal{A}_*$, we interpolate a new point from the midpoint of neighboring points on camera image and world map, respectively, which is defined as:

$$\begin{aligned} \mathcal{A}_*^1 &= \mathcal{A}_*^0 \cup \mathcal{Q}_*^1 \\ \mathcal{Q}_*^1 = \{\varrho_{ij} \mid i, j \in \mathcal{A}_*^0\}, &\quad \varrho_{ij} = midpoint(\varrho_i, \varrho_j) \end{aligned} \tag{22}$$

where $\mathcal{Q}_*^1$ indicates the generated point-pair set from the midpoint ϱ_{ij}: $[cid, ij, \sigma_x^{ij}, \sigma_y^{ij}, \omega_x^{ij}, \omega_y^{ij}]$ of neighboring points ϱ_i, ϱ_j, the point number ij is numbered in order. $\mathcal{Q}_*^1$ is the combination of original point-pair set and generated point-pair set. $*$ denotes the area number. Point-pair enlargement is a recursive process, which is computed as:

$$\mathcal{A}_*^{s+1} = \mathcal{A}_*^s \cup \mathcal{Q}_*^{s+1}, \quad \mathcal{Q}_*^{s+1} = \{\varrho_{ij} \mid i, j \in \mathcal{A}_*^s\} \tag{23}$$

For Duke-MTMCT dataset, we have five iterations to generate 30k point-pair for training PPN. The architecture of PPN contains 3 fully-connected (fc) layers for each camera projection, the camera position $[\sigma_x, \sigma_y]$ are divided by the length and width of the image, respectively, for normalizing the value to 0-1 $[\bar{\sigma_x}, \bar{\sigma_y}]$ as the PPN input (1×2 vector). The normalized vector is fed into fc_1 $(2 \rightarrow 128) \rightarrow$ fc_2 $(128 \rightarrow 128) \rightarrow$ fc_3 $(128 \rightarrow 2)$, each layer contains fully-connected, batch normalization and ReLU. The output of fc_3 is a 1×2 vector $[\hat{\bar{\omega_x}}, \hat{\bar{\omega_y}}]$. We utilize the generated world position $[\omega_x, \omega_y]$ to supervise PPN. The loss is defined as:

$$\mathcal{L}_p = \left|\bar{\omega_x} - \hat{\bar{\omega}}_x\right| + \left|\bar{\omega_y} - \hat{\bar{\omega}}_y\right| \tag{24}$$

where $\mathcal{L}_p$ is a L1-loss, $[\bar{\omega_x}, \bar{\omega_y}]$ are normalized position by world position. The PPN is an one-to-one mapping between camera coordinate and world coordinate, so $\mathcal{L}_p$ aims to narrow the distance between the predicted position and the real position.

4.2 Temporal-Spatial Constraint

We adopt the temporal-spatial constraint to reduce the amount of matching operation. The tracklet positions are projected on the uniform world map. Figure 9 shows the camera2 and camera5 tracking results and projection results at the 289752-th and 290160-th frames. A throng in camera2(a) exits the field of view from the right. For each disappearing tracklet τ (introduced in Sect. 3.1(A)), we compute the velocity $\tau[v_e]$ at the last frames before it disappears, which is defined as:

$$\tau[\mathbf{v}_e] = \tau[\mathbf{v}_L] = \frac{1}{2}(\tau[\mathbf{v}_{L-1}] + \frac{1}{\eta_v}\sum_{t=1}^{\eta_v}(p_L - p_{L-t})) \tag{25}$$

which is a recursion equation, where $\tau[\mathbf{v}_L]$ is the velocity of the last frame for τ, L is the current length of tracklet τ^k. $\tau[\mathbf{v}_{L-1}]$ is the previous frame velocity, and η_v indicates the velocity computing width parameter, p_L denotes the position of the last frame for τ. The velocity of the current frame is

based on the previous velocity and the average of the vectors between the current position and the previous few positions.

Tracklet association are restricted previously by temporal-spatial constraints. Our proposal eliminates lots of irrelevant trajectories that appear too early or too late as compared with the target duration time, meanwhile, it removes trajectories which are too far away from the target prediction position . The constraint of matching is shown as:

$$\left\| \tau^{k_j}[p_1] - \tau^{k_i}[\hat{p}_{L^{k_i}+\Delta t_{i,j}}] \right\|_2^2 < \eta_c \tag{26}$$

where

$$\begin{aligned} \tau^{k_i}[\hat{p}_{L^{k_i}+\Delta t_{i,j}}] &= \tau^{k_i}[\hat{p}_{L^{k_i}}] + \Delta t_{i,j} * \tau^{k_i}[\mathbf{v}_e], \\ \Delta t_{i,j} &= \tau^{k_j}[t_s] - \tau^{k_i}[t_e], \\ s.t.\ 0 < \tau^{k_j}[t_s] - \tau^{k_i}[t_e] &< \eta_t;\ \tau^{k^i}[s], \tau^{k^j}[s] \neq quitted \end{aligned} \tag{27}$$

$\tau^{k_j}[p_1]$ indicates the position of τ^{k_j} at the first frame of tracklet. $\tau^{k_i}[\hat{p}_{L^{k_i}+\Delta t_{i,j}}]$ is the predicted position of the τ^{k_j} in the future. η_c is the spatial constraint parameter. $\Delta t_{i,j}$ is the interval between start frame τ^{k_j} and end frame τ^{k_i}. η_t is the temporal constraint parameter. $\tau[s]$ are the states of the tracklet, which contains "tracked", "lost" and "quit"(same as the node states in Section 3.1(A)).

4.3 Trajectory Association

Trajectory matching candidates are filtered by temporal-spatial constraint. Trajectory association is based on calculating the general appearance feature distance between Trajectories. The trajectory general feature is extracted by Re-connection Network (Sec.3.3(C)). Given a trajectory $\tau^k \in \mathcal{T}$, and its corresponding trajectory candidate set with constraint is $\mathcal{T}_k : \{\tau_k^1, \tau_k^2, \tau_k^3, ...\}$. We calculate the Cosine distance between τ^k and each trajectory candidate τ_k^*. We define an association threshold, θ_R, and associate the trajectories of which the distance is less than θ_R. At last, we utilize "Union-Find-Set" to classify the connected subset. Each subset indicates a whole trajectory.

5 Experiments

5.1 Datasets and Benchmarks

We evaluate our proposal on MOT16 Milan et al. (2016) and Duke-MTMCT Ristani et al. (2017) datasets, which contain large-scale video sequences from different cameras and scenes. Both datasets' tracking results are evaluated on MOT Challenge Leal-Taix et al. (2015). We additionally use the Person ReID dataset Market-1501 Zheng et al. (2015) and DukeMTMC-reID Ristani et al. (2017) to train Appearance Model, utilize tracking dataset PathTrack Manen et al. (2017), Duke-MTMCT Ristani et al. (2017) and video re-identification dataset MARS Zheng et al. (2016) to train Cleaving and Re-connection Network. The Motion Model and Position Projection Networks are trained on Duke-MTMCT.

Duke-MTMCT is a large-scale dataset for multiple target multiple-camera tracking with the videos captured by 8 surveillance cameras at different viewing angles including 2800 identities (persons) in Duke University. The video duration of each camera is 86 minutes, which is split into training set (0–50 min) and testing set (50-86 min). In addition, the dataset provides DPM Felzenszwalb et al. (2010) and Openpose Cao et al. (2018) detection results for each frame as the tracker input. However, currently the dataset has been removed from the MOT benchmark.

MOT16 is a classical evaluation dataset comparing several tracking methods on MOT Challenge, which includes 14 sequences captured from surveillance, hand-held shooting and driving recorder by static cameras and moving cameras. The length of each video is about 500-1500 frames. And the dataset also provides the detection DPM.

5.2 Implementation Details

In our experiments, our networks consist of CNN, LSTM and GRU, where GRU Cho et al. (2014) is a type of RNN with gates and hidden units. For tracklet generation, we train the CNN network of appearance model with SeResNet50 Hu et al. (2018), the images are resized to 128*256 from ReID training set and the output of CNN $f_{c_t^i}$, $f_{d_t^j}$ produces a 2048-dimensional vector to describe the image. In addition, the inputs of the LSTM network for motion model is a series of 2-dimensional vector $p_{c_t^i}$: $[x, y]$ with a tracklet, which are divided by image width and height to have the input normalized, and the LSTM output is the prediction of the position and size $\hat{p}_{c_t^i}$:$[\hat{x}, \hat{y}]$. For tracklet cleaving and re-connection, the model is a deep Siamese Bi-GRU, which includes four hidden-layers and the maximum length of GRU is 120 frames. The input of the GRU is a series of appearance features by CNN. The outputs of the GRU $\varphi_{g,i}^k, i \in [-L^k, -1] \cup [1, L^k]$ are 128-dimensional vectors, which are fed to FC network for verification loss and for comparison of the corresponding features for classification loss. At single-camera tracking, association threshold θ_G=0.7, tracklet matching threshold θ_R=0.5. The searching interval length η_s is 100 frames.

For cross-camera, the spatial constraint parameter η_c=300 pixels on world map, the temporal constraint parameter η_t=3000 frames to retrieve tracklets. We associate the tracklets which satisfy the constraint by the Re-connection

Table 1 The performance with different steps on MOT16 validation set

Tracker	MOTA↑	IDF1↑	IDP↑	IDR↑	MT↑	ML↓	FP↓	FN↓	IDSw.↓	Frag↓
TPA(baseline)	52.5	54.4	74.4	42.9	12	27	2634	19805	90	536
+Cleaving	52.5	56.2	75.7	44.7	12	27	2634	19805	78	533
+Re-connection	52.5	59.1	77.9	47.7	12	27	2634	19805	60	533
+Gap Insertion	54.6	59.7	74.3	49.9	18	27	3576	17973	40	145
+Smoothing	54.6	59.7	74.3	49.9	18	27	3570	17993	37	86

Network output φ_f^k. The tracklet association is the tracklet similarity matching task, which compares the distance of feature pair by pair. The closer the distance of feature, the more similar the image. In addition, the temporal constraint makes sure the re-connection candidates appear at different times and the target doesn't appear twice at the same time. The spatial constraint checks whether two tracklet candidates meet normal movement rule. If the distance of tracklets is less than θ_R, the tracklets are connected. However, the matching of tracklets between different cameras might affect the labeling of ID, e.g. τ^i linked by τ^j; τ^j connected by τ^k; τ^i also associated by τ^k. As the result, τ^i, τ^j, τ^k need to be labeled the same ID number. We adopt "Union-Find-Set" algorithm to solve the tracklet association task, which searches the connected sub-graph as the same trajectory from global matching graph. We use the AdamOptimizer Kingma and Ba (2015) as the training optimizer, our experiment is implemented with Python 3.6 and Pytorch 0.4.1 framework and on Nvidia Tesla K40 GPUs.

5.3 Ablation Study

For Single-Camera Tracking, Table 1 describes our performance for each evaluation parameter at different steps. The first row indicates the tracklet generation step, which reaches 52.2 MOTA and 54.4 IDF1. The cleaving step aims to check the tracklet purity and split impure tracklets, and the re-connection step focuses on linking the tracklet fragments. Both Step 2 and Step 3 improve ID types of measures such as IDF1 from 54.4 to 59.1, but they don't basically affect CLEAR-MOT metrics except ID switch. The last two steps are explained in Fig. 10, the Gap Insertion method fills the disappearing bounding boxes due to occlusion according to the existing bounding boxes, this step can increase the True Positive (TP) bounding boxes, meanwhile the False Positive is (FP) also increased. On the whole, most of the performance measures are improved, especially MOTA, IDF1 and Frag. The last step, tracklet smoothing method adjusts the boxes' size to optimize the boxes' "waggle" between frames, which can decrease the frag and slightly reduce ID switch.

For Cross-Camera Tracking, we present a Position Projection matching strategy to associate trajectories from different

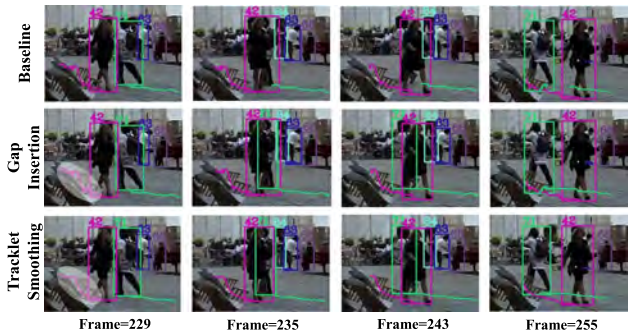

Fig. 10 The explanation of Gap Insertion and Tracklet Smoothing

Table 2 The comparison from different matching strategy on DukeMTMCT training set

Matching strategy	Matching number↓	IDF1↑
Global retrieval	3042695	54.4
Temporal	1577665	57.3
Strong temporal	99870	66.2
Temporal-topology	44133	71.1
Spatial-temporal	**18107**	**78.5**

Bold indicates the highest score (performance) for each column or group in the table

cameras. Table 2 shows the influence of using temporal and spatial constraints on the quality of trajectory matching and performance. The Global Retrieval indicates traversal search for each trajectory one-by-one in the history trajectory gallery. The Temporal gives the constraint for the start-frame and end-frame of trajectory. We filter out the trajectories where their start-frame is earlier than target trajectory end-frame. Temporal constraint can reduce by half the unavailable matching pairs. The strong temporal constraint additionally sets the upper limit of frame interval, which can decrease the amount of trajectory matching operation from 1577665 to 99870. Topology strategy aims to construct the connected relation between cameras on actual scenario, which only matches the trajectories from neighboring cameras, it removes half of the irrelevant trajectories. Spatial-temporal constraint is our method, which combines strong temporal and position projection, and the constraint details are described in Eq.27. The spatial-temporal can

Table 3 The performance of Multiple-Camera Tracking with different models in Duke-MTMCT validation set

Tracker	Multi-Camera			Single-Camera	
	IDF1↑	IDP↑	IDR↑	MOTA↑	IDF1↑
B+R	66.3	70.0	62.4	83.1	80.5
B+C+R	68.3	73.1	64.4	86.4	83.7
B+R+P	73.4	77.9	69.9	83.1	80.5
B+C+R+P	78.5	80.0	77.2	86.4	83.7

(B: Baseline; R: Re-connection; C: Cleaving; P: Position Projection)

extremely reduce the number of matching-pairs, the remaining pairs meet motion rule between appearing position of candidate trajectory and predicted position of target trajectory on the real scenario. It can largely eliminate redundant amount of matching operation and decrease the probability of mis-matching.

Table 3 shows the performance of single and multiple camera tracking with different models in Duke-MTMCT validation set. "R" includes Re-connection, Gap Insertion and Smoothing processing. From the first and second rows, using the Cleaving Network can improve 3.3% MOTA and 3.2% IDF1 in single camera, and 2.0% IDF1 in multiple-camera, respectively. The third row adopts Position Projection, thus IDF1 in multiple camera is greatly improved. From the last row of the table, the IDR is raised a lot by using Cleaving Network and Position Projection. Hence, the IDF1 is obviously higher than the others in Table 3.

5.4 Networks Analysis

5.4.1 Cleaving Network

To train the Cleaving Network, we construct a training dataset from Duke-MTMCT Ristani et al. (2017), PathTrack Manen et al. (2017) and MOT16 Milan et al. (2016). The datasets includes more than 2500 pedestrians' tracklet from different cameras and scenes, we divide each tracklet into several sub-tracklet of uniform-length (120 frames). Ultimately, we have more than 10K sub-tracklets (1.2M images) for training Cleaving Network.

Before training the Cleaving Network, we randomly select two tracklets τ^a and τ^b. We generate a random parameter l_p from 0 or 1. If l_p=0, we input an impure tracklet (we constructed) into Cleaving Network, so we generate another random parameter l_s from 0 to 120, which indicates the splitting position. We combine the two tracklets as a joint tracklet, which is formulated as: $l_{id} = [l_{id1}[0 : l_s], [l_{id2}[l_s : 120]]$. If l_p=1, we input a pure tracklet into Cleaving Network, we set $l_s = 120$.

In training step, we randomly shuffle the order of the impure and the pure tracklets to feed into Cleaving Network. We define three types of losses (Feature Loss $\mathcal{L}_{c,ftr}$, Pure Loss $\mathcal{L}_{c,pur}$, Search Loss $\mathcal{L}_{c,srh}$) to supervise the network. Feature Loss $\mathcal{L}_{c,ftr}$ is treated as a multiple classification task, which supervises the outputs of each step $\varphi^k_{g,i}$ to generate discriminative features. Pure Loss $\mathcal{L}_{c,pur}$ is a binary classification task, which supervises φ^k_p to judge the purity of a tracklet. Search Loss $\mathcal{L}_{c,srh}$ is utilized for optimizing φ^k_s to search the best splitting position of impure tracklet. We utilize two kinds of loss (cross-entropy and L1-loss) to train cleaving network, respectively.

To evaluate the performance of cleaving network in training, we define three evaluation indexes: 1.Accuracy of ID classification; 2. Accuracy of purity; 3. Average distance of best splitting position. If we define $\mathcal{L}_{c,src}$ as cross-entropy, the task initially is treated as multiple classification task (121 classes), we fix the tracklet length of input (120 frames), each tracklet has 121 gaps between frames. At the 21st epoch, the accuracy of ID: 98.6%; Accuracy of purity: 95.3%; Average distance of best splitting position: 2.57 gaps. If we define $\mathcal{L}_{c,src}$ as L1-loss, which is to minimize the distance between the output φ^k_s and the ground truth l_s. At the 13rd epoch, the Accuracy of ID: 98.6%; Accuracy of purity: 95.2%; Average distance of best splitting position: 2.43 gaps. Although the performances (φ^k_p and φ^k_s) with two kinds of loss are the same, using L1-loss can make earlier convergence than using cross-entropy.

In testing step, we fix the length of Cleaving Network input (120 frames), therefore, we equably sample the over-length tracklets and padding the under-length tracklets to being of a uniform length.

For over-length tracklet τ^1:

$$\tau^1 = [I_1, I_2, I_3, ..., I_{238}, I_{239}, I_{240}]_{1*240}$$

where I_* indicates the i-th croped image of the tracklet τ.

We convert τ^1 to $\tau^{1'}$:

$$\tau^{1'} = [I_1, I_3, I_5, ..., I_{235}, I_{237}, I_{239}]_{1*120}$$

For under-length tracklet τ^2:

$$\tau^2 = [I_1, I_2, I_3, ..., I_{58}, I_{59}, I_{60}]_{1*60}$$

We convert τ^2 to $\tau^{2'}$:

$$\tau^{2'} = [I_1, I_1, I_2, ..., I_{59}, I_{60}, I_{60}]_{1*120}$$

For under-length tracklet, we can easily get the best splitting position in the original tracklet:

$$\tau^{2'} = [I_1, ..., I_{30}, I_{30}, |I_{31}, I_{31}, ..., I_{60}]_{1*120}$$

The splitting position of original tracklet is:

$$\tau^2 = [I_1, ..., I_{29}, I_{30}, |I_{31}, I_{32}, ..., I_{60}]_{1*60}$$

However, for over-length tracklets, we cannot directly get the best splitting positions in the original tracklet. Thus, we select ±60 frames around the splitting position to form a tracklet to input to the network again.

For example:

$$\tau^{1'} = [I_1, ..., I_{169}, |I_{171}, ..., I_{239}]_{1*120}$$

Select the sub-tracklet and re-feed to Cleaving Network:

$$\tau^{1''} = [I_{110}, ..., I_{169}, I_{170}, I_{171}, ..., I_{229}]_{1*120}$$

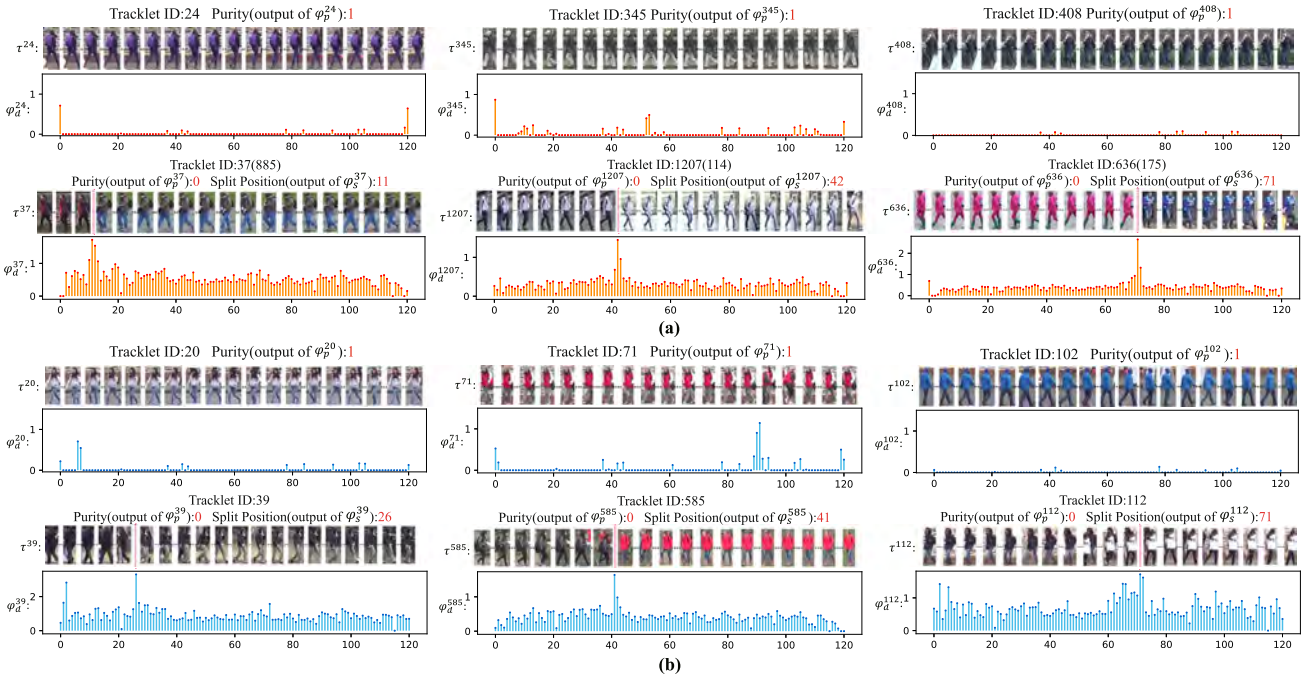

Fig. 11 The purity and split results of Cleaving Network for pure & impure tracklets on training and testing dataset. **a**: In training set **b**: In testing set

The splitting position of original tracklet is:

$$\tau^1 = [I_1, ..., I_{169}, |I_{170}, I_{171}, ..., I_{240}]_{1*240}$$

An example of searching splitting position recursively for over-length tracklet τ^{585} is illustrated in Fig. 12.

Figure 11 illustrates tracklets τ^{k_*} and the corresponding outputs of Cleaving Network, which includes purity results: $\varphi_p^{k_*}$, splitting results: $\varphi_s^{k_*}$, and feature visualization results: $\varphi_s^{k_*}$. If the tracklet is pure, since the change of the appearance of the target being tracked in a short time of period is very little (because of the frame rate assumption), the change of feature of the the tracklet appearance is also slight. Thus, most of the elements of feature $\varphi_{d,i}^*$ tend to be zero. On the contrary, if the tracklet is impure, since the target is replaced by the other, the difference of the appearances is obvious before and after the replacement. Thus, most of the elements of feature $\varphi_{d,i}^*$ are not zero. The position of the maximum value of the feature (pink dotted line of each impure tracklet in the Fig. 11) indicates that the features at the two sides have the largest difference.

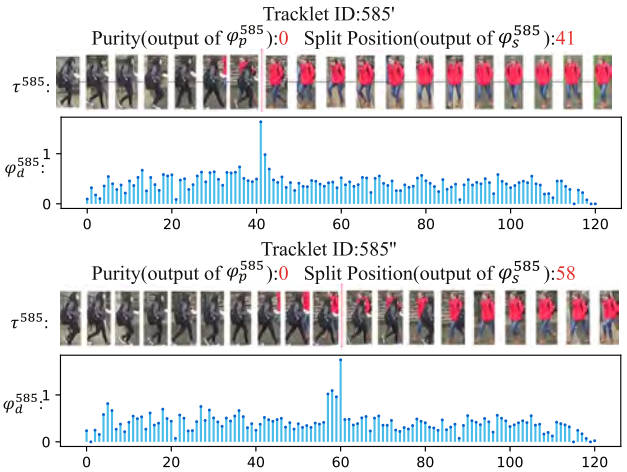

Fig. 12 The illustration of searching splitting position recursively for over-length tracklet τ^{585} by Cleaving Network

5.4.2 Re-connection Network

The training dataset of Re-connection Network is composed of DukeMTMCT Ristani et al. (2017), MOT16 Milan et al. (2016) and MARS Zheng et al. (2016), which includes more than 2800 persons, and each person has several tracklets from different cameras, viewpoint, pose, illumination, etc. The Re-connection Network is finally convergent to the 90th epoch in training. The re-connection network is regarded as a general feature extractor, in post-processing step, re-connection strategy determines whether the tracklets belong to the same person based on appearance similarity between the tracklets and spatial-temporal constraint.

Fig. 13 illustrates the performance of Re-connection Network on Duke-MTMCT testing set, where Fig. 13a shows the tracklets from multiple cameras, each tracklet is marked with their tracklet ID, camera ID and timestamps at the top of the images. Fig. 13b illustrates the similarity between the tracklets. The similarity is calculated by $\lambda_a F_a(c_t^i, d_t^j)$ (described

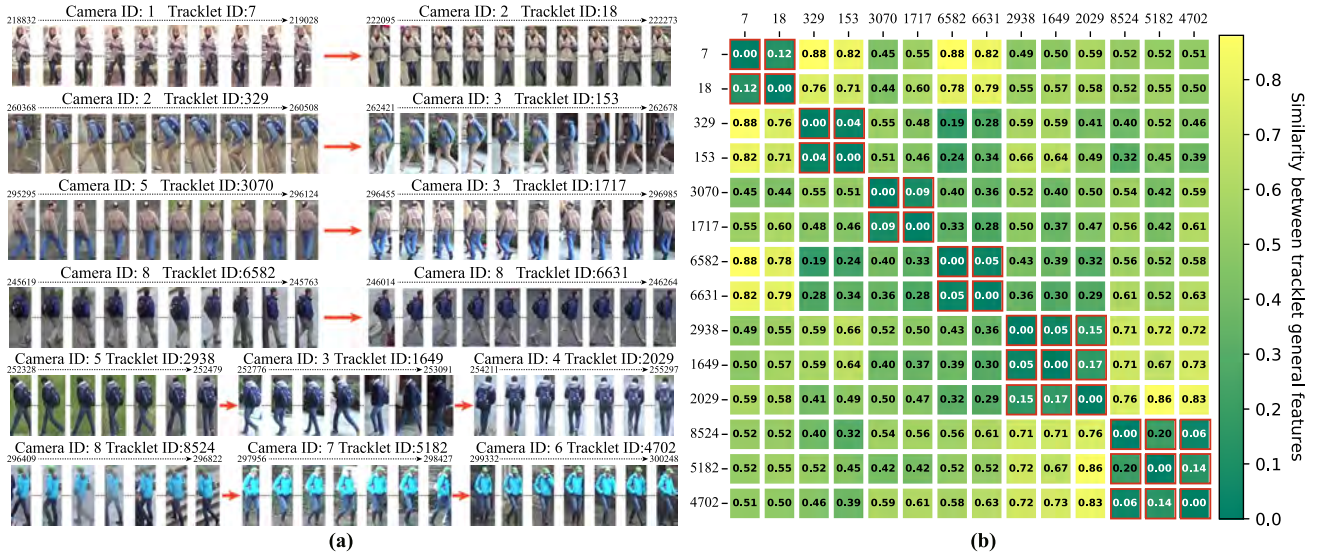

Fig. 13 Experimental results of Re-connecction Network (**a**): tracklets from multiple cameras (**b**): Similarity matrix between tracklets

in Eq.(1)). Generally λ_a is set 0.5, the value of $\lambda_a F_a(c_t^i, d_t^j)$ is between [0, 1]. The more similar the features are, the smaller the value is. In the Fig. 13b, the similarity value between the tracklets of the same person is generally not more than 0.15, and most of the similarity value between the tracklets of different persons is not less than 0.3. However, for two tracklets which are of highly similar appearance and don't belong to the same person, their values are less than 0.3, such as No.329 and No.6582. Actually, at the matching step, No.329 is removed initially from the No.6582 matching candidates by the temporal-spatial constraint. Thus, matching tracklets by threshold is able to re-connect the tracklets effectively.

5.4.3 Position Projection Network

The proposal of position projection strategy is normalizing the positions from different cameras to the same map, which is utilized for tracklet motion prediction on world map and tracklet matching between cameras. The position projection task is treated as regression task: we assume that the pedestrians move at the surface of the ground, given a camera-coordinate (σ_x, σ_y), the output of PPN is a unique position (ω_x, ω_y) on world-coordinate.

To improve the performance of position projection, we propose a data expansion method (described in Sect. 4.1(A)). Extending the point-pairs by generating midpoints is implemented within each plane. The function of Position Projection Network (PPN) is the same as Projective Transformations of 2D. As Hartley and Zisserman (2003) describes in Sect. 2.3, Definition 2.11: "A planar projective transformation is a linear transformation on homogeneous 3-vectors represented by a non-singular 3*3 matrix", which is formulated as:

$$\begin{pmatrix} x_1^{'} \\ x_2^{'} \\ x_3^{'} \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_1 \\ x_3 \end{pmatrix} \tag{28}$$

where $x_*, x_*^{'}$ indicate the coordinates of original and transformed plane, respectively. The position transformation between camera image and world map belongs to an affine transformation, which is a non-singular linear transformation as following:

$$\begin{pmatrix} \sigma_x \\ \sigma_y \\ 1 \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ 1 \end{pmatrix} \tag{29}$$

Affine transformation ensures that a point (σ_x, σ_y) in camera image has a unique point (ω_x, ω_y) in world map. Given two points ϱ_1, ϱ_2 in a plane, their midpoint ϱ_{12} must be also in the plane. Therefore, generated midpoint between points in camera image also corresponds to a midpoint between the corresponding points in the world map.

Compared with affine transformation, PPN learns the projection relation from labeled and generated data by deep learning. The geometry-based method Jiang et al. (2018) matches the trajectory by manually creating a topology association according to the path between cameras. However, if the number of cameras are increased, the manual operation will become more complicated. Additionally, since the view of cameras are different, the velocity and direction calculated by position in camera-coordinate exist errors. On the contrary, PPN needn't previously make the geometric calibration, such as vanishing point camera calibration. The prediction position based on world map is much easier, because all of trajectories from different cameras are in the

Table 4 Results on the MOT16 testing set

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSw.↓	Frag↓
EDMT Chen et al. (2017)	45.3	47.9	17.0	39.9	11122	87890	639	946
MHT_DAM Kim et al. (2015)	45.8	46.1	16.2	43.2	6412	91758	590	7812
STAM16 Chu et al. (2017)	46.0	50.0	14.6	43.6	6895	91117	473	1422
NOMT Choi (2015)	46.4	53.3	18.3	41.4	9753	87565	359	504
AMIR Sadeghian et al. (2017)	47.2	46.3	14.0	41.6	**2681**	92856	774	1675
NLLMPa Wang et al. (2016)	47.6	50.9	15.2	38.3	9253	85431	792	1858
FWT Henschel et al. (2017)	47.8	44.3	19.1	38.2	8886	85487	852	1534
TSN Peng et al. (2018)	48.2	45.7	**19.9**	38.9	8447	85315	665	829
LMP Tang et al. (2017)	48.8	51.3	18.2	40.1	6654	86245	481	**595**
eTC Wang et al. (2019)	49.2	56.1	17.3	40.3	8400	83702	606	882
CRF_TRACK Xiang et al. (2020)	50.3	54.4	18.3	**35.7**	7148	82746	702	1387
TPM Peng et al. (2020)	51.3	47.9	18.7	40.8	2701	85504	569	707
MLT Zhang et al. (2020)	52.8	**62.6**	21.1	42.4	5362	80444	**299**	702
Tracktor Bergmann et al. (2019)	54.4	52.5	19.0	36.9	3280	79149	682	1480
Baseline(Ours)	48.6	49.3	13.2	43.5	5854	87260	994	1660
TG_CR(Ours)	**55.0**	52.9	19.1	37.2	3590	**77829**	673	865

Bold indicates the highest score (performance) for each column or group in the table

same plane, the velocity and direction cannot be affected by the view of each camera.In experimental results, the average distance error $\mathcal{L}_p$ between output and labeled position is 0.0032 pixels/point for each camera.

5.5 MOT Evaluation Metrics

The MOT Challenge Benchmark adopted the standard CLEAR-MOT mapping Bernardin and Stiefelhagen (2008) and ID measures Ristani et al. (2017) for evaluating MOT performance. The main metrics for MOT are MOTA and IDF1. MOTA (Multiple Object Tracking Accuracy) measures the effect of tracking for each tracklet, which depends on True Positives (TP) ,False Positives (FP), False Negatives (FN) and Id Switches (IDSw), $MOTA = 1 - \frac{FP+FN+IDS}{TP}$. Total number of Fragment (Frag), Mostly tracked targets (MT), Mostly lost targets (ML) are used for evaluating tracklet integrity as the reference indexes. The IDF1 (ID F1 Score) is the ratio of correctly identified detection over the average number of true and computed detection, which depends on ID True Positives (IDTP), ID False Positives (IDFP) and ID False Negatives (IDFN), $IDF1 = \frac{2*IDTP}{2*IDTP+IDFN+IDFP}$. IDP (ID Precision) indicates fraction of computed detections that are correctly identified, $IDP = \frac{IDTP}{IDTP+IDFP}$. IDR (ID Recall) means fraction of ground-truth detection that are correctly identified, $IDR = \frac{IDTP}{IDTP+IDFN}$.

5.6 Method Comparison

We evaluate our proposal against the other traditional state-of-the-art methods on two public Tracking Benchmark datasets (MOT16, MOT17 and Duke-MTMCT). Table 4 compares the performance of our method with the existing methods on MOT16 testing set. Compared with the others methods, we achieve the higher performance of 55.0% on MOTA, 19.1% on ML and 77829 on FN. Our method outperforms most of the methods on the others metrics. Table 5 shows the performance of different state-of-the-art methods on MOT17 testing set. Trackers are divided into two types: Tracking-by-Detection (TBD) and Joint Detection and Tracking (JDT). TBD-based methods follow the traditional framework to associate bounding boxes according to the given detection results; JDT-based methods combine both of models (detection and tracking) as an end-to-end network Zhou et al. (2020) or tracking module interacts with the detection module in feature layer Peng et al. (2020). Therefore, these methods are slightly higher than TBD-based trackers. Our proposal achieves excellent performance compared to other TBD-based trackers. For multiple-camera tracking, Duke-MTMCT datasets are captured from surveillance on static camera (shown in Table 6). Our proposal effectively computes the trajectory motion rules and accurately associates tracklets from different cameras. Due to cleaving and re-connection step, for Single-Camera we reach 85.6% on MOTA and accomplish the highest scores of MT, ML, FN. For cross-camera, most of the traditional methods Maksai et al. (2017); Ristani et al. (2017); Liang and Zhou (2017);

Table 5 Results on the MOT17 testing set

Process	Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSw.↓	Frag↓
JDT	FAMNet Chu and Ling (2016)	52.0	48.7	19.1	33.4	**14138**	253616	3072	5318
JDT	JBNOT Henschel et al. (2019)	52.6	50.8	19.7	35.8	31572	232659	3050	3792
JDT	Tracktor+CTdet Bergmann et al. (2019)	54.4	56.1	25.7	29.8	44109	210774	**2574**	**2763**
JDT	CTracker Peng et al. (2020)	66.6	57.4	32.2	**24.2**	22284	160491	5529	9114
JDT	CTTrack17 Zhou et al. (2020)	**67.8**	**64.7**	**34.6**	24.6	18498	**160332**	3039	6102
TBD	eHAF17 Sheng et al. (2018)	51.8	54.7	23.4	37.9	33212	236772	1834	2739
TBD	eTC17 Wang et al. (2019)	51.9	58.1	23.1	35.5	36164	232783	2288	3071
TBD	CRF_TRA Xiang et al. (2020)	53.1	53.7	24.2	30.7	27194	234991	2518	4918
TBD	HDTR Babaee et al. (2018)	54.1	48.4	23.3	34.8	18002	238818	1895	2693
TBD	TPM Peng et al. (2020)	54.2	52.6	22.8	37.5	13739	242730	1824	2472
TBD	TT17 Zhang et al. (2020)	54.9	**63.1**	24.4	38.1	20236	233295	**1088**	2392
TBD	Tracktor Bergmann et al. (2019)	56.3	55.1	21.1	35.3	**8866**	235449	1987	3763
TBD	GSM_Tracktor Liu et al. (2020)	56.4	57.8	22.2	**34.5**	14379	230174	1485	2763
TBD	Baseline(Ours)	50.9	54.3	22.4	37.0	31945	242820	2220	3413
TBD	TG_CR(Ours)	**57.1**	59.3	**23.5**	34.9	15216	**224841**	1766	**2349**

Bold indicates the highest score (performance) for each column or group in the table

Tesfaye et al. (2017); Yoon et al. (2016); Zhang et al. (2017) neglect the temporal-spatial information, and Jiang et al. (2018) only constructs topological graph for matching trajectory. Our position projection strategy can correctly describes the real tracklet position, we can eliminate irrelevant trajectories by trajectory appearing time and predicted position, therefore we greatly enhance the ID Recall (IDR), which is up to 73.5%, meanwhile our IDF1 gets the highest score on Duke-MTMCT.

6 Conclusion

In this paper, for single camera tracking, we propose cleaving and re-connection networks to process the tracklets on crowd or long-term occlusion by Deep Siamese Bi-GRU. The method examines each output of bidirectional GRU to search the suitable split position and match the tracklets to reconnect the same person. For training, we extract the tracklet dataset from existing MOT datasets for training our frameworks. Our proposal has better performance for static camera such as surveillance. The algorithm achieves 55.0%, 57.1% and 85.6% in MOTA that approach to the state-of-the-art methods on MOT16, MOT17 and Duke-MTMCT benchmark dataset, respectively, where the visualization tracking results at different phase is shown in Fig. 14, and single camera tracking result is shown in Fig. 15. For multiple-camera tracking, we present a position projection strategy to convert the tracklet position from camera-coordinate to world-coordinate. We only annotate few point-pairs to train the Position Projection Network. Figure 16 illustrates the position projection and multiple-camera tracking results on Duke-MTMCT, where the left of the figure is the world map, and the right of figure shows each camera view. The same numbers between world map and camera image indicate projection areas. For trajectory matching, we predict the tracklet position in the future on the world map, and extract the tracklet general features by the re-connection network, meanwhile we add the temporal-spatial constraint to reduce the unavailable pairs. The final cross-camera tracking results are shown in Fig. 17. The crowd orderly appear on the camera No.1, No.2 and No.5, where the bottom of the images illustrates the person tracklet in different cameras.

In the future, we will further research on the cleaving phase. To completely cleave the impure tracklets which contain more than two sub-tracklets, we will utilize the "Transformer" to calculate the similarity of any two moments of impure tracking to replace the GRU. In addition, we will explore how to optimize the occlusion problem, i.e., we intend to combine the head-detector and body-detector as collaborative track of pedestrians. To describe the general features effectively, we will consult to the video-based person ReID methods such as 3D CNN model, non-local, and attention strategy. To extract the interaction feature, we will explore the movement relationships between pedestrians. We will also increase the processing efficiency. Lastly, we attempt to utilize transfer learning to improve the model robustness. For the overlapping regions cross-camera tracking task, we firstly determine the overlapping regions between the cameras. We match the tracklets according to the similarity between tracklets' appearance and calculating the distance between the tracklets' positions in the world map.

Table 6 Results on the Duke-MTMCT easy testing dataset Benchmark

Tracker	Multi-camera			Single-amera							
	IDF1↑	IDP↑	IDR↑	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSw.↓	Frag↓
PT_BIPCC Maksai et al. (2017)	34.9	41.6	30.1	59.3	71.2	666	234	71381	361673	298	799
BIPCC Ristani et al. (2017)	56.2	67.0	48.4	59.4	70.1	665	234	68634	361589	**290**	**783**
lx_b Liang and Zhou (2017)	58.0	72.6	48.2	61.3	70.3	–	–	–	–	–	–
MTMC_CDSC Tesfaye et al. (2017)	60.0	68.3	53.5	70.9	77.0	740	110	38655	268398	693	4717
MYTRACKER Yoon et al. (2016)	65.4	71.1	60.6	73.8	80.3	914	72	**35580**	193253	406	1116
TAREIDMTMC Jiang et al. (2018)	68.8	71.8	66.0	83.3	**83.8**	1051	17	44691	131220	383	2428
MTMC_ReIDp Zhang et al. (2017)	74.4	**84.4**	66.4	70.7	79.2	726	143	52408	277762	449	1060
TG_CRP(Ours)	**75.3**	77.2	**73.5**	**85.6**	81.6	**1114**	**9**	66552	**85207**	332	1206

Bold indicates the highest score (performance) for each column or group in the table

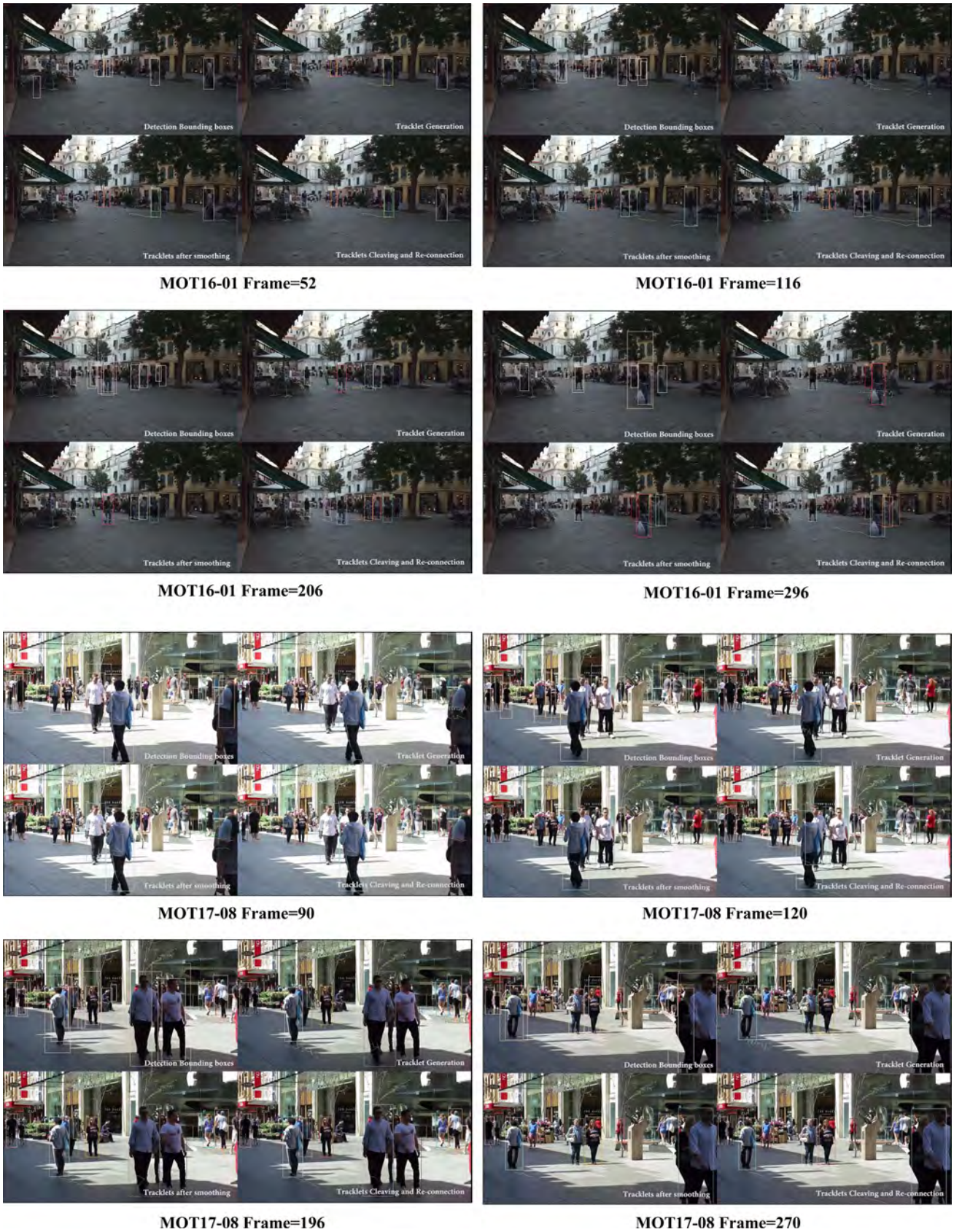

Fig. 14 The visualization tracking results at different phase

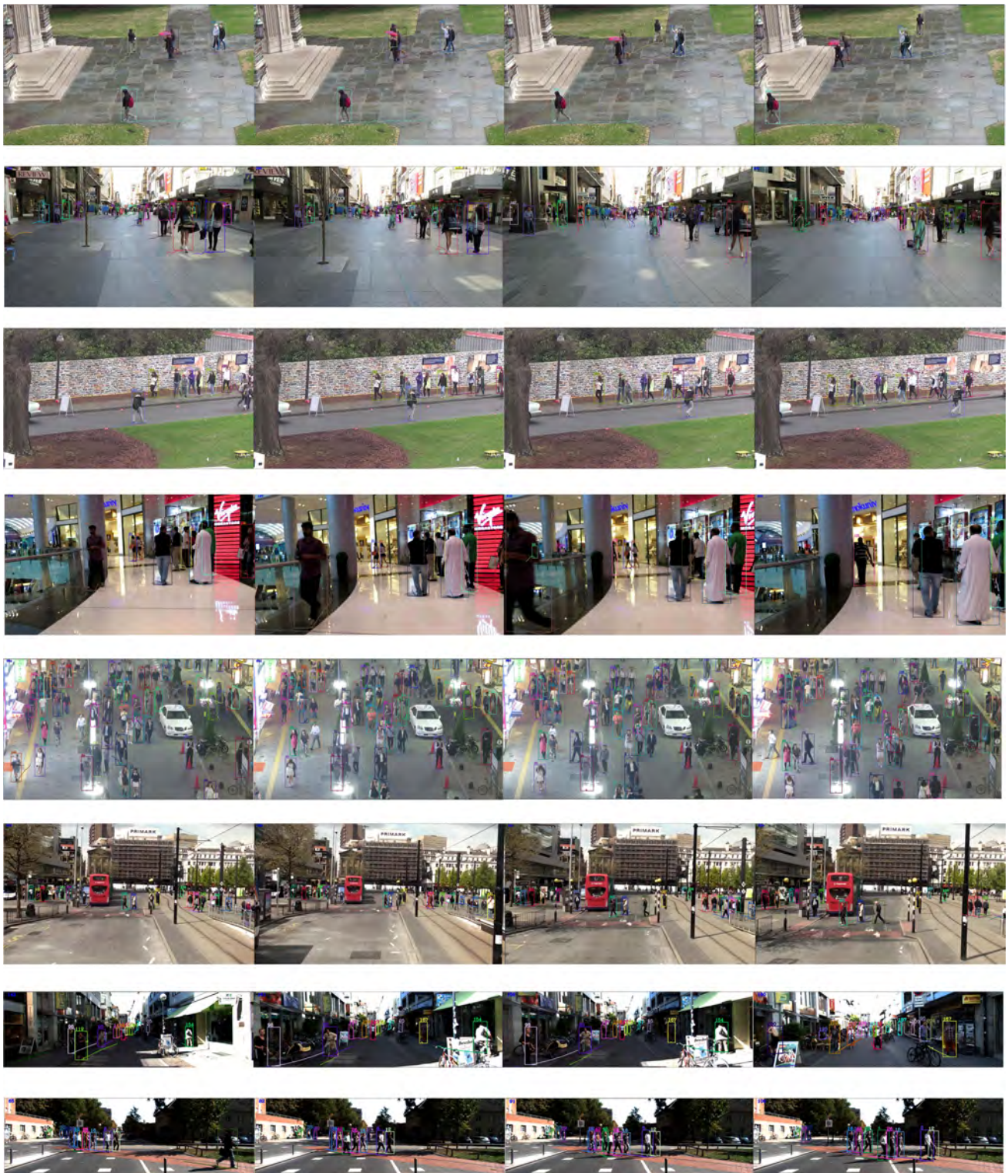

Fig. 15 Single-camera Tracking results on testing set

Fig. 16 Multiple-camera Tracking and Position Projection results on Duke-MTMCT. Left: world map Right: 8-cameras tracking results (In order to illustrate the tracking and position projection results more clearly, we mark the ID of corresponding camera in the world map respectively)

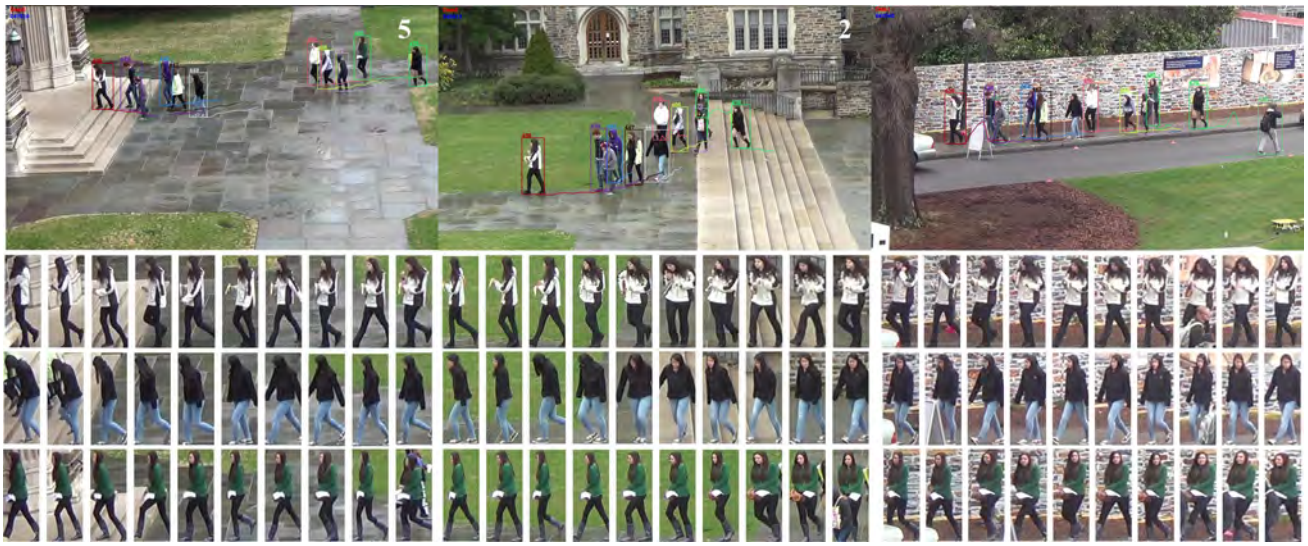

Fig. 17 Cross-camera Tracking results on Duke-MTMCT testing set. Top-right: The 243345-th frame on Camera 1; Top-middle: The 245913-rd frame on Camera 2; Top-left: The 247216-th frame on Camera 5. The bottom 3 lines: No.436, No.449 and No.485 tracklets on different cameras

And then, we set a threshold to judge whether the tracklets belong to the same person. As for the person making turn, we can properly enlarge the radius η_c to cover this case. In addition, we need to improve the appearance model in the future to further enhance the matching accuracy.

References

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., & Savarese, S. (2016). Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 961-971).

Babaee, M., Athar, A., & Rigoll, G. (2018). Multiple people tracking using hierarchical deep tracklet re-identification. arXiv:1811.04091

Bae, S.H., & Yoon, K.J. (2014). Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1218-1225).

Bergmann, P., Meinhardt, T., & Leal-Taixe, L. (2019). Tracking without bells and whistles. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 941-951).

Bernardin, K., & Stiefelhagen, R. (2008). Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing (1)* 246309.

Bredereck, M., Jiang, X., Körner, M., & Denzler, J. (2012). Data association for multi-object tracking-by-detection in multi-camera networks. In: *2012 Sixth International Conference on Distributed Smart Cameras (ICDSC)*, IEEE, (pp. 1-6).

Cai, Y., & Medioni, G. (2014). Exploring context information for inter-camera multiple target tracking. In: *IEEE Winter Conference on Applications of Computer Vision*, IEEE, (pp. 761–768).

Cao, Z., Hidalgo, G., Simon, T., Wei, S.E., & Sheikh, Y. (2018). Openpose: realtime multi-person 2d pose estimation using part affinity fields. arXiv:1812.08008

Chen, J., Sheng, H., Zhang, Y., & Xiong, Z. (2017). Enhancing detection model for multiple hypothesis tracking. *In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 18–27).

Chen, K. W., Lai, C. C., Lee, P. J., Chen, C. S., & Hung, Y. P. (2011). Adaptive learning for target tracking and true linking discovering across multiple non-overlapping cameras. *IEEE Transactions on Multimedia, 13*(4), 625–638.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., & Schwenk, H., Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv:1406.1078

Choi, W. (2015). Near-online multi-target tracking with aggregated local flow descriptor. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 3029-3037).

Chu, P., & Ling, H. (2019). Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 6172–6181).

Chu, Q., Ouyang, W., Li, H., Wang, X., Liu, B., & Yu, N. (2017). Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 4836-4845).

Dicle, C., Camps, O.I., & Sznaier, M. (2013). The way they move: Tracking multiple targets with similar appearance. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 2304-2311).

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE TPAMI, 32*(9), 1627–1645.

Gao, X., & Jiang, T. (2018). Osmo: Online specific models for occlusion in multiple object tracking under surveillance scene. In: *26th ACM international conference on Multimedia* (pp. 201–210).

Guo, M., Chen, M., Ma, C., Li, Y., Li, X., & Xie, X. (2020). High-level task-driven single image deraining: Segmentation in rainy days. In: *International Conference on Neural Information Processing*, Springer, (pp. 350–362).

Hartley, R., & Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge University Press.

Henschel, R., Leal-Taixé, L., Cremers, D., & Rosenhahn, B. A. (2017). Novel multi-detector fusion framework for multi-object tracking. In: arXiv:1705.08314

Henschel, R., Leal-Taixé, L., Cremers, D., & Rosenhahn, B. (2018). Fusion of head and full-body detectors for multi-object tracking. In: *Computer Vision and Pattern Recognition Workshops (CVPRW)*

Henschel, R., Zou, Y., & Rosenhahn, B. (2019). Multiple people tracking using body and joint detections. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*.

Hermans, A., Beyer, L., & Leibe, B. (2017). *In defense of the triplet loss for person re-identification*. arXiv:1703.07737.

Hong Yoon, J., Lee, C.R., Yang, M.H., & Yoon, K.J. (2016). Online multi-object tracking via structural constraint event aggregation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1392-1400).

Hou, Y., Li, C., Yang, F., Ma, C., Zhu, L., Li, Y., Jia, H., & Xie, X. (2020). Bba-net: A bi-branch attention network for crowd counting. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE* (pp. 4072–4076).

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7132–7141).

Jiang, N., Bai, S., Xu, Y., Xing, C., Zhou, Z., & Wu, W. (2018). Online inter-camera trajectory association exploiting person re-identification and camera topology. In: *Proceedings of the 26th ACM International Conference on Multimedia* (pp. 1457–1465).

Kim, C., Li, F., Ciptadi, A., & Rehg, J.M. (2015). Multiple hypothesis tracking revisited. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 4696–4704).

Kingma, D., & Ba, J. (2015). Adam: A method for stochastic optimization. *ICLR*

Le, N., Heili, A., & Odobez, J. M. (2016). Long-term time-sensitive costs for crf-based tracking by detection. In: *European Conference on Computer Vision* (pp. 43-51).

Leal-Taixé, L., Milan, A., Reid, I., Roth, S., & Schindler, K. (2015). Motchallenge 2015: Towards a benchmark for multi-target tracking. arXiv:1504.01942

Levinkov, E., Uhrig, J., Tang, S., Omran, M., Insafutdinov, E., Kirillov, A., Rother, C., Brox, T., Schiele, B., & Andres, B. (2017). Joint graph decomposition & node labeling: Problem, algorithms, applications. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6012-6020).

Li, Y., Chen, F., Yang, F., Ma, C., Li, Y., Jia, H., & Xie, X. (2020). Optical flow-guided mask generation network for video segmentation. In: *IEEE International Symposium on Circuits and Systems (ISCAS). IEEE*, (pp. 1–5).

Liang, Y., & Zhou, Y. (2017). Multi-camera tracking exploiting person re-id technique. In: *International Conference on Neural Information Processing*, Springer, (pp. 397–404).

Liu, X., & Zhang, S. (2020). Domain adaptive person re-identification via coupling optimization. In: *Proceedings of the 28th ACM International Conference on Multimedia* (pp. 547–555).

Liu, X., & Zhang, S. (2021). Graph consistency based mean-teaching for unsupervised domain adaptive person re-identification. In: *IJCAI*.

Liu, Q., Chu, Q., Liu, B., & Yu, N. (2020). Gsm: Graph similarity model for multi-object tracking. In: *International Joint Conferences on Artificial Intelligence (IJCAI)*

Liu, X., Zhang, S., Wang, X., Hong, R., & Tian, Q. (2019). Group-group loss-based global-regional feature learning for vehicle re-identification. *IEEE Transactions on Image Processing, 29*, 2638–2652.

Luo, H., Gu, Y., Liao, X., Lai, S., & Jiang, W. (2019). 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.*

Ma, C., Li, Y., Yang, F., Zhang, Z., Zhuang, Y., Jia, H., & Xie, X. (2019). Deep association: End-to-end graph-based learning for multiple object tracking with conv-graph neural network. *Proceedings of the 2019 on International Conference on Multimedia Retrieval* (pp. 253-261).

Ma, C., Yang, C., Yang, F., Zhuang, Y., Zhang, Z., Jia, H., & Xie, X. (2018). Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-gru for multiple object tracking. In: *2018 IEEE International Conference on Multimedia and Expo (ICME)*

Maksai, A., Wang, X., Fleuret, F., & Fua, P. (2017). Globally consistent multi-people tracking using motion patterns. In: *Proceedings of the IEEE International Conference on Computer Vision*

Maksai, A., Wang, X., Fleuret, F., & Fua, P. (2017). Non-markovian globally consistent multi-object tracking. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, IEEE, (pp. 2563–2573).

Manen, S., Gygli, M., Dai, D., & Van Gool, L. (2017). Pathtrack: Fast trajectory annotation with path supervision. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 290-299).

Ma, C., Yang, F., Li, Y., Jia, H., Xie, X., & Gao, W. (2021). Deep human-interaction and association by graph-based learning for multiple object tracking in the wild. *International Journal of Computer Vision, 129*(6), 1993–2010.

McLaughlin, N., Martinez del Rincon, J., & Miller, P. (2016). Recurrent convolutional network for video-based person re-identification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1325-1334).

Milan, A., Leal-Taixé, L., Reid, I. D., Roth, S., & Schindler, K. (2016). MOT16: A benchmark for multi-object tracking. *CoRR* arXiv:1603.00831

Peng, J., Gu, Y., Wang, Y., Wang, C., Li, J., & Huang, F. (2020). Dense scene multiple object tracking with box-plane matching. In: *Proceedings of the 28th ACM International Conference on Multimedia* 4615–4619.

Peng, J., Qiu, F., See, J., Guo, Q., Huang, S., Duan, L. Y., & Lin, W. (2018). Tracklet siamese network with constrained clustering for multiple object tracking. In: *IEEE Visual Communications and Image Processing (VCIP)* (pp. 1–4).

Peng, J., Wang, T., Lin, W., Wang, J., See, J., Wen, S., & Ding, E. (2020). Tpm: Multiple object tracking with tracklet-plane matching. *Pattern Recognition,107480*

Peng, J., Wang, C., Wan, F., Wu, Y., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., & Fu, Y. (2020). Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In: *European Conference on Computer Vision*, (pp. 145–161).

Ristani, E., & Tomasi, C. (2018). Features for multi-target multi-camera tracking and re-identification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6036-6046).

Ristani, E., Solera, F., Zou, R., Cucchiara, R., & Tomasi, C. (2016). Performance measures and a data set for multi-target, multi-camera tracking. In: *European Conference on Computer Vision* (pp. 17-35).

Sadeghian, A., Alahi, A., & Savarese, S. (2017). Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 300-311).

Sahbani, B., & Adiprawita, W. (2017). Kalman filter and iterative-hungarian algorithm implementation for low complexity point tracking as part of fast multiple object tracking system. In: *2016 6th International Conference on System Engineering and Technology* (pp. 109–115).

Schulter, S., Vernaza, P., Choi, W., & Chandraker, M. (2017). Deep network flow for multi-object tracking. *In: CVPR.* (pp. 6951–6960).

Sheng, H., Zhang, Y., Chen, J., Xiong, Z., & Zhang, J. (2018). Heterogeneous association graph fusion for target association in multiple object tracking. *IEEE Transactions on Circuits and Systems for Video Technology, 29*(11), 3269–3280.

Son, J., Baek, M., Cho, M., & Han, B. (2017). Multi-object tracking with quadruplet convolutional neural networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5620-5629).

Tang, S., Andriluka, M., Andres, B., & Schiele, B. (2017). Multiple people tracking by lifted multicut and person reidentification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern* Recognition (pp. 3539–3548).

Tesfaye, Y.T., Zemene, E., Prati, A., Pelillo, M., & Shah, M. (2017). Multi-target tracking in multiple non-overlapping cameras using constrained dominant sets. arXiv preprint arXiv:1706.06196

Wang, B., Wang, L., Shuai, B., Zuo, Z., Liu, T., Luk Chan, K., & Wang, G. (2016). Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association. *In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 1-8).

Wang, G., Wang, Y., Zhang, H., Gu, R., & Hwang, J. N. (2019). Exploit the connectivity: Multi-object tracking with trackletnet. In: *Proceedings of the 27th ACM International Conference on Multimedia, ACM* (pp. 482–490).

Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In: *European Conference on Computer Vision*, Springer (pp. 499–515).

Xiang, Y., Alahi, A., & Savarese, S. (2015). Learning to track: Online multi-object tracking by decision making. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 4705-4713).

Xiang, J., Xu, G., Ma, C., & Hou, J. (2020). End-to-end learning deep crf models for multi-object tracking. *IEEE Transactions on Circuits and Systems for Video Technology*

Yang, M., & Jia, Y. (2016). Temporal dynamic appearance modeling for online multi-person tracking. *Computer Vision and Image Understanding,153*, 16–28.

Yoon, K., Song, Y. M., & Jeon, M. (2018). Multiple hypothesis tracking algorithm for multi-target multi-camera tracking with disjoint views. *IET Image Processing*, *12*(7), 1175–1184.

Zhang, Z., Wu, J., Zhang, X., & Zhang, C. (2017). Multi-target, multi-camera tracking by hierarchical clustering: Recent progress on dukemtmc project. arXiv preprint arXiv:1712.09531

Zhang, S., Zhu, Y., & Roy-Chowdhury, A. (2015). Tracking multiple interacting targets in a camera network. *Computer Vision and Image Understanding*, (pp. 64–73).

Zhang, Y., Sheng, H., Wu, Y., Wang, S., Ke, W., & Xiong, Z. (2020). Multiplex labeling graph for near-online tracking in crowded scenes. *IEEE Internet of Things Journal*, *7*(9), 7892–7902.

Zhang, Y., Sheng, H., Wu, Y., Wang, S., Lyu, W., Ke, W., & Xiong, Z. (2020). Long-term tracking with deep tracklet association. *IEEE Transactions on Image Processing, 29*, 6694–6706.

Zheng, L., Bie, Z., Sun, Y., Wang, J., Su, C., Wang, S., & Tian, Q. (2016). Mars: A video benchmark for large-scale person re-identification. In: *European Conference on Computer Vision* (pp. 868–884).

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., & Tian, Q. (2015). Scalable person re-identification: A benchmark. In: *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1116-1124).

Zheng, Z., Zheng, L., & Yang, Y. (2018). A discriminatively learned cnn embedding for person reidentification. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 14*(1), 1–20.

Zhou, X., Koltun, V., & Krähenbühl, P. (2020). Tracking objects as points. In: *European Conference on Computer Vision*, Springer, (pp. 474–490).

Zhu, J., Yang, H., Liu, N., Kim, M., Zhang, W., & Yang, M.H. (2018). Online multi-object tracking with dual matching attention networks. In: *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 366-382).

Zhuang, Y., Tao, L., Yang, F., Ma, C., Zhang, Z., Jia, H., & Xie, X. (2018). Relationnet: Learning deep-aligned representation for semantic image segmentation. In: *2018 24th International Conference on Pattern Recognition (ICPR), IEEE* (pp. 1506–1511).

Zhuang, Y., Yang, F., Tao, L., Ma, C., Zhang, Z., Li, Y., Jia, H., Xie, X., & Gao, W. (2018). Dense relation network: Learning consistent and context-aware representation for semantic image segmentation. In: *25th IEEE International Conference on Image Processing (ICIP). IEEE,2018*, 3698–3702.