Dynamic Gallery for Real-time Multi-target Multi-camera Tracking

Yu-Sheng Chou[1], Chien-Yao Wang[2], Ming-Chiao Chen[3], Shou-De Lin[1], Hong-Yuan Mark Liao[2]
[1]Graduate Institute of Networking and Multimedia, National Taiwan University
[2]Institute of Information Science, Academia Sinica
[3]Department of Computer Science and Information Engineering, National Taitung University
d03944008@ntu.edu.tw, kinyiu@iis.sinica.edu.tw, joechen@nttu.edu.tw,
sdlin@csie.ntu.edu.tw, and liao@iis.sinica.edu.tw

Abstract

For multi-target multi-camera recognition tasks, tracking of objects of interest is one of the essential yet challenging issues due to the fact that the task requires re-identifying identical targets across distinct views. Multi-target multi-camera tracking (MTMCT) applications span a wide range of variety (e.g. crowd behavior analysis, anomaly individual tracking and sport player tracking), so how to make the system perform real-time tracking becomes a crucial research issue. In this paper, we propose an online hierarchical algorithm for extreme clustering based MTMCT framework. The system can automatically create a dynamic gallery with real-time fashion by collecting appearance information of multi-object tracking in single-camera view. We evaluate the effectiveness and efficiency of our framework, and compare the state-of-the-art methods on MOT16 as well as DukeMTMC for single and multiple camera tracking. The high-frame-rate performance and promising tracking results confirm our system can be used in real-world applications.

1. Introduction

Among all computer vision research topics, the problem that a person re-ID system would like to solve is the closest to what an MTMCT system likes to tackle. In a person re-ID system, a pedestrian image grabbed by an arbitrary camera will be broadcasted to other cameras in the network, and then compare it with the pedestrian image gallery obtained by other cameras. For each pedestrian image gallery corresponding to each camera, the system will sort the gallery images based on the similarity between each gallery image and the unknown input. The purpose of the above move is to perform person re-ID in a non-overlapping camera network. Although the purpose of an MTMCT system is similar to a person re-ID system, the former has to face several challenges. First, an MTMCT system does not have a gallery like a person re-ID system. The only available information that an MTMCT system can use is those that have appeared and been tracked target pedestrian images. The tracking performance may underlie this restriction. Second, an MTMCT system is basically a classification problem. It needs to use a systematic method to determine a threshold, and then use this threshold to decide on whether the object being tracked is the same one. The problem cannot be solved with a person re-ID system because the object that was retrieved with the top rank may be wrong. In particular, a tracked target that appears in a new frame may not exist in the gallery at all. Due to the fact that a gallery needs to be prepared, a person re-ID system cannot completely solve the problem that an MTMCT system has to deal with. Recently there are more and more datasets prepared for different person re-ID systems. In cases that these datasets have enough training data, they can provide an MTMCT system more robust appearance descriptors and assist single camera to track targets more accurately.

The recent development of deep learning has promoted object detection to make rapid progress. Because of this trend, the already existing tracking-by-detection technique becomes more viable. The concept of tracking-by-detection is to use object detectors to detect objects of interest in a video, and merge the detected bounding boxes of adjacent frames into tracklets. Then, the tracklets in different frames are merged into trajectories. For people using tracking-by-detection technique, it is important to develop a good object detector. In the past few years, Bewley et al. [1] proposed the simple online and realtime tracking (SORT) technique. They use the Faster R-CNN based object detector [16] together with Kalman filter to predict object trajectory. To assure better matching accuracy, they introduce Hungarian algorithm to achieve the goal. A main drawback of SORT is that it cannot handle identity switches well. The reason is the lack of an appearance descriptor which makes the tracklet vulnerable when the bounding box contents are occluded

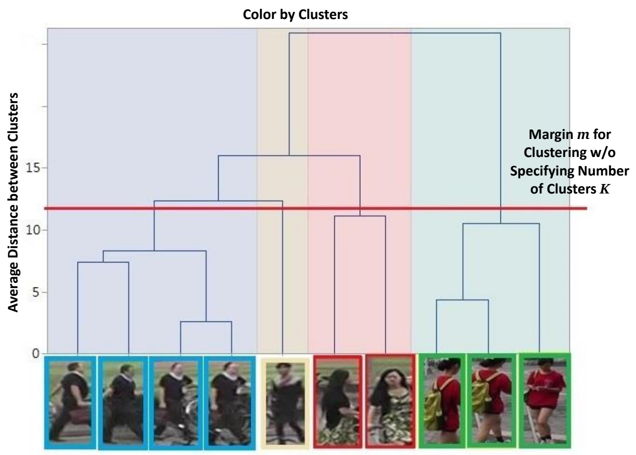

Figure 1: Dynamic gallery construction by a balanced binary tree. We collect the tracked pedestrians as retrievable appearance information and solve the multi-camera tracking problem with real-time fashion.

by each other. Under these circumstances, Bewley et al. [21] further proposed an extension model: deep SORT, in which they use the deep metric learning model optimized by person re-ID objective, to effectively compare the appearance information. In this way, the number of ID switches can be successfully reduced. Nowadays, there are some single view tracking methods that introduce a re-ID model to improve the tracking accuracy by the appearance of a target. However, under the original MTMCT system architecture, there is no gallery like re-ID system. If we directly apply single-camera tracking technique to an MTMCT system, it will lead to easy failure when performing cross-camera tracking, because the appearance information that can assist the comparison task is not available.

In this paper, we propose an online-hierachical-algorithm-for-extreme-clustering-based MTMCT framework. The system can automatically create a dynamic gallery with real-time fashion by collecting appearance information of multi-object tracking in single-camera view. The main idea is to enable an MTMCT system to achieve real-time performance when tracking across cameras by systematically storing the appearance information that has been tracked by previous cameras. We have three main contributions: First, we leverage the online hierarchical extreme clustering method to systematically cluster the appearance information. Second, with the dynamic gallery design, our system can solve the problem of identity switches caused by long-term occlusion. Third, our system outperforms existing real-time state-of-the-art in performing single-camera multi-object tracking and performs competitively with state-of-the-art results of multi-target multi-camera tracking task, which cannot be real-time.

2. Approach

In this section, we show the details of the proposed real-time MTMCT system. In Section 2.1, we report how object detection and appearance feature learning are implemented. Single camera tracking and dynamic gallery construction are detailed in Section 2.2. In Section 2.3, we elaborate how multi-camera tracking is realized.

2.1. Object Detection and Appearance Feature Learning

To make the proposed MTMCT system have as close to real-time performance as possible, the proposed strategy has to maintain the detection rate simultaneously. According to recent literature [16, 5, 10, 13], state-of-the-art object detection methods are all CNN-based. CNN-based object detection methods can be categorized into two kinds. They are either region proposal based methods [19, 4, 6, 3, 16, 2, 9, 5] or one-step method [10, 13, 14, 15]. A region proposal based method is basically a two-step process. By nature, it is usually R-CNN based. This kind of approach will first detect the bounding boxes of objects of interest. Then, the objects detected by those bounding boxes will be classified based on their features. Although the detection accuracy of a region proposal based method is better than that of a one-step method, its time-consuming nature does not fit in any system that requires real-time computation. In order to keep the real-time detection requirement, we thus adopt one-step object detection framework YOLO [15]. We regress each input image frame into several bounding box coordinates and the classification probabilities indicate existence of object categories. Object detection of this kind can significantly cut down the time needed to generate object proposals and then classifying each proposal into different object categories.

Recently, some CNN-based person re-ID system [7, 22, 23, 24, 20] use a large number of labelled pedestrian images to conduct the training task, and the accuracy of these systems is close to human performance. In our framework, we train an appearance feature embedder to discriminate pedestrians. In the mean while, we follow Ristani et al. [18] and adopt the adaptive weighted triplet loss to train our network. This triplet loss can be expressed as follows:

$$L_3 = \left[m + \sum_{x \in P(a)} w_p d(x_a, x_P) - w_n d(x_a, x_n) \right]_+ . \quad (1)$$

For an anchor sample x_a, the above defined triplet loss can assure its distance to the projection of a positive point x_p is closer to that of a negative point x_n, and this distance is greater than m. In addition, the introduction of two adaptive weights into the equation may provide more flexibility when facing hard samples. For example, when there is imbalance existing between positive and negative samples, we are able to dynamically adjust the weights based on the distributions of samples. The adaptive weights can be calculated by the softmax function, which is defined as follows:

$$w_p = \frac{e^d(x_a, x_p)}{\sum_{x \in P(a)} e^d(x_a, x)}, w_n = \frac{e^{-d}(x_a, x_p)}{\sum_{x \in N(a)} e^{-d}(x_a, x)}. \quad (2)$$

The philosophy of adaptive weighting is as follows: easy samples will be assigned with low weights, while hard samples will be assigned with high weights. The main reason we can perform dynamic weighting is that we know the sum of the distance between each anchor and the positive and negative samples. If we add up the sums corresponding to all distances of anchors, it is easy to assign weights based on the proportion of each anchor distance to all anchor distances. For each training batch, we also adopt the hard sample mining strategy. We select the positive samples with the largest distances from the anchor in the batch, and the negative sample with the smallest samples with the smallest distances from the anchor. This selection rule can prevent the training process from being trapped by the easy sample in the poor local minimum. In order to achieve good system immediacy and grab appropriate pedestrian appearance, we use a highly efficient YOLO detector to obtain a sensible bounding box, and use adaptive weighted triplet loss in the training task. In this way, the neural network can embed the pedestrian content bounded by the bounding box and we can use this information to measure the degree of similarity of pedestrian appearance. This degree similarity can be used as prior information for tracking.

2.2. Single Camera Tracking and Dynamic Gallery Construction

The task of tracking-by-detection is mainly divided into three steps: (1) detecting the position of the sensible object, (2) predicting the position of the corresponding object in the current frame based on the detection result of the previous frame, and (3) associating the detection and prediction results to complete object tracking.

In the subsection, we elaborate how to predict the position of objects and how to combine the detection and prediction results so that the object tracking task can be completed. For predicting the position of objects, we use the standard Kalman filter approach. It includes constant velocity motion model and linear observation model. For the observation of object state, we directly use the coordinates of the bounding box (x, y, w, h). When detection result and target match, the detected bounding box is used to update the target object state. As to the velocity components, they will be optimally calculated by Kalman filter. Conversely, if detection and target have no match, the target object state can be predicted with a simple linear velocity model. In the previous subsection, we have described how object detection is accomplished. Here we propose a concept of dynamic gallery. The dynamic gallery is constructed in an online clustering process, and it is also part of the tracking system that performs object association. We consider object association as an assignment problem, its main purpose is to assign detections to the tracked targets. In our tracking-by-detection system, there are two factors that will affect the assignment results. The first factor is appearance similarity and the second is position prediction through Kalman filter. We can use the above two information to track pedestrians and construct dynamic gallery. Here, we reformulate the problem of pedestrian matching into a clustering problem. We count the number of pedestrians detected per frame in a video stream as the number of data points on the embedding space. We cluster all the data according to their correlation into k disjoint groups, and the union G formed by these disjoint groups will become a gallery. But in order to achieve real-time goal, we must incrementally include new data points based on the number of frames detected.

We apply online hierarchical extreme clustering to simultaneously solve the problem of single camera tracking and dynamic gallery construction. The reason for using this clustering algorithm is because it can construct a binary cluster tree structure and this structure is easy to understand and manage. The input to this structure is done incrementally, and the total number of clusters does not need to be specified in advance. The clustering structure is very suitable for handling single camera tracking. When we track a specific person, its collection of continuous images can be regarded as a cluster. It is very common to have dozens of pedestrians in a video stream at the same time. Binary tree construction for online clustering is based on the following assumption. *A dynamic gallery union G is separable with respect to a cluster C^* if*

$$\max_{(x,y) \in C^*} ||x - y|| < \min_{(x',y') \notin C^*} ||x' - y'||. \quad (3)$$

Under this assumption, within-cluster distance for all clusters will be less than any between-cluster distance. Although this assumption does not apply in all cases, our appearance feature descriptor objective function (Equation 1) can make pedestrian information close to such a distribution in embedding space by large number of pedestrian images.

We follow the method adopted by PERCH [8], using masking-based rotations, balance-based rotations, bounding box approximations, and collapsed mode to quickly search and grow the tree ($O(\log(k))$). We use PERCH's rotation operations to recover errors that occur during greedy incremental clustering. The `Masked` function is used to prevent the situation when the inserted data is clustered to a wrong group. It functions as follows: *A node v with sibling v' and aunt a in a tree T is masked if there exists a point $x \in lvs(v)$ (the set of leaves for any internal node v)*

Algorithm 1 `Insert` (x_i, T)

```
1: $t$ = NearestNeighbor($x_i$)
2: $l$ = Split($t$)
3: for $a$ in Ancestors ($l$) do
4:     $a$.AddPoints($x_i$)
5: end for
6: $T$ = $T$.Rotate($l$.Sibling(),CheckMasked)
7: $T$ = $T$.Rotate($l$.Sibling(),CheckBalanced)
8: CollapsedMode()
9: return $T$
```

such that

$$\max_{y \in lvs(v')} ||x - y|| > \min_{z \in lvs(a)} ||x - z||. \quad (4)$$

In this case, v contains a data point x, which is closer to a point in aunt a than a point in sibling v, which is judged to be masked. When a point i' is inserted and its sibling i is detected as `Masked`, rotation function is triggered and we will swap i' and its tree position aunt a. This function will be recursively executed until no `Masked` condition is detected.

The above rotation algorithm guarantees the best dendrogram purity under the separable assumption, but there is no limit to the depth of the tree, and it affects the search time. To assure the search and insertion function can be executed within logarithmic time, we have to make the binary cluster tree become a balanced binary tree. *The balance of a cluster tree* T*, denoted* $bal(T)$, is the average local balance of all nodes in T, where the local balance of a node v with children vl, vr is

$$bal(v) = \frac{\min\{|lvs(v_l)|, |lvs(v_r)|\}}{\max\{|lvs(v_l)|, |lvs(v_r)|\}}. \quad (5)$$

According to the above definition, we will do the same actions as masking-based rotation. This process will keep going until the balance-rotated tree T' with $bal(T') > bal(T)$ or when there is any data point in T that meets the condition of `Masked`.

In conventional point-by-point search algorithms, the computation time is usually intensive. Thus, using the bounding box approximation to represent all the data points of the subtree will greatly reduce the computation complexity. *In the embedding space of dimension* d*, all leaves of internal node are in the interval of* j *of each dimension, i.e.,* $[v_-(j), v_+(j)]$. Therefore, the squared maximum distance and the squared minimum distance of each internal node v and the newly inserted point x can be represented by the following two expressions:

$$d_+(x, v)^2 = \sum_{j=1}^{d} \max\{(x(j) - v_-(j))^2, (x(j) - v_+(j))^2\} \quad (6)$$

$$d_-(x, v)^2 = \sum_{j=1}^{d} \begin{cases} (x(j) - v_-(j))^2 & \text{if } x(j) \leq v_-(j) \\ (x(j) - v_+(j))^2 & \text{if } x(j) \geq v_+(j) \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

If we want to insert a point x, and two children v and v' are chosen to conduct search, then when $d_+(x, v') < d_-(x, v)$, we only search the v'-based subtree. For x, the v'-based subtree is closer to it. Using this search strategy, we are able to significantly cut down the number of searches and thus save much time.

To save the memory of a dynamic gallery, the last strategy, `Collapsed` mode, is very important. The reason of having this mode is to prevent the binary cluster tree from growing too big and thus hinder the possibility of efficient search. The philosophy behind this strategy is to delete the children below LCA (lowest common ancestor) in the same cluster so that the search space can be significantly reduced. On the other hand, the pedestrian information contained in bounding boxes will be stored in node v. The `Collapsed` mode is to define the maximum number of leaves allowed in a cluster. With this restriction, the number of leaves in a cluster will not grow unlimitedly.

When we collect the data of pedestrians in a frame and construct the corresponding binary cluster tree, the average distance between clusters according to the depth of the tree can be calculated. We use the margin m set when we train the appearance feature embedder, and use it as the threshold of between-cluster distance and then systematically determine the number of cluster k (as illustrated in Figure 1). Assuming that a cluster already exists in the previous frame, the newly added data point must satisfy the condition that the IOU (intersection over union) of bounding box and the prediction is greater than 0.5.

2.3. Multi-camera Tracking

When we get the dynamic gallery through the first camera, the multi-camera tracking problem has been transformed into the task of re-identifying the current pedestrian in the camera through searching the dynamic gallery tree. When performing tracking, each camera needs to establish its own binary cluster tree. We merge the subtrees of the same cluster into the dynamic gallery tree built by all cameras. This allows us to synchronize the information gathered by all camcorders.

2.4. Tree Visualization

To prove that dynamic gallery is really effective for long-term occlusion and person re-identification in multi-camera environment, in this section we try to visualize the change of dynamic gallery tree under the conditions of long-term occlusion and view change among multiple cameras. Fig-

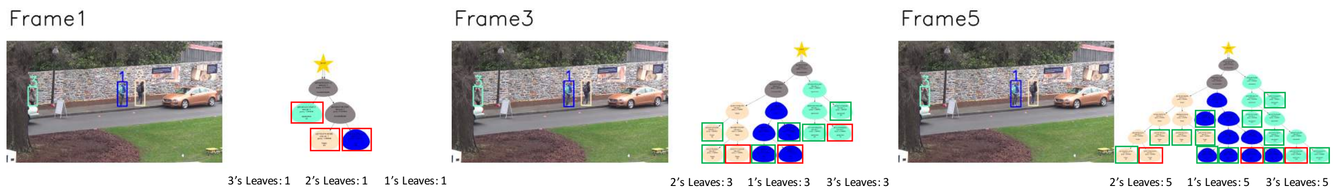

Figure 2: Tree visualization for tree growth case.

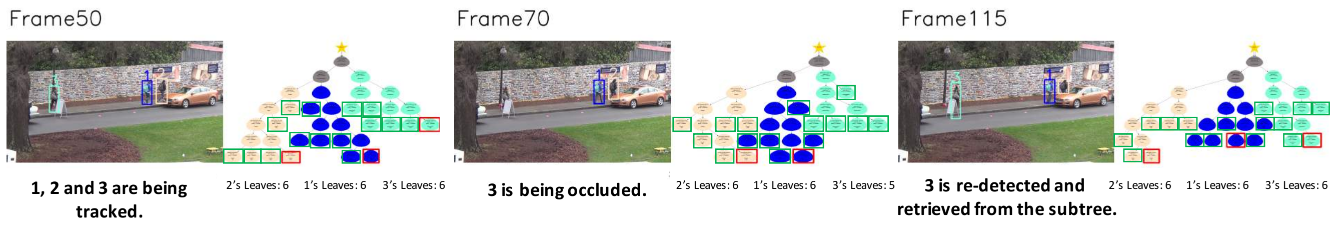

Figure 3: Tree visualization for long-term occlusion case (the number of max leaves is 5).

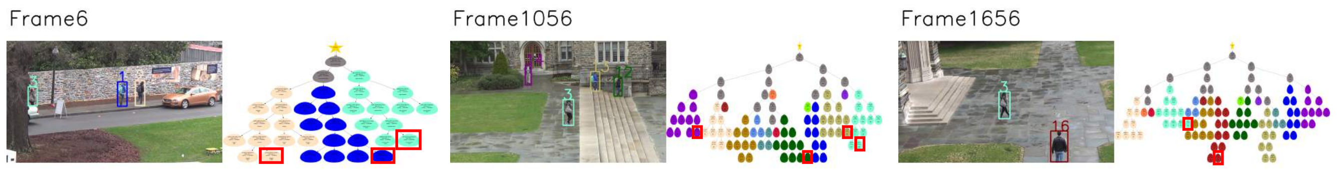

Figure 4: Tree visualization for view changing case (the number of max leaves is 5).

ure 3 shows the tree growth process when tracking three pedestrians labeled number 1, 2, and 3. The trees shown on the right hand side of Frame 1, 2, and 5 are the corresponding trees. Our system first detects pedestrian and these detected pedestrians will be bounded by bounding boxes. These detected bounding boxes of pedestrians are then sequentially inserted into a tree, and we cluster these information based on the algorithm mentioned in Section 2.2. There will be different clusters in a tree, and each color represents a specific tracked pedestrian (e.g. the leaves corresponding to the pedestrians marked 3 are aquamarine). Whenever a new instance is inserted into the same cluster, the number of leaves is increased. Therefore, from the number of leaves of the tree corresponding to frame 3 and 5 of Figure 3, it is apparent that when the number of frames increases, the dynamic gallery will grow larger. In Figure 3, the red frames are newly added leaves, while the green frames are the previously inserted leaves. Figure 4 illustrates how a tracked object is re-tracked through the dynamic gallery when it encounters a long-term occlusion. In the 50th frame shown in Figure 4, pedestrians labeled 1, 2, and 3 are tracked. The corresponding dynamic gallery tree is shown on the right hand side of this frame. In the 70th frame, the pedestrian labeled 3 is blocked by a brand on the road and no bounding box is detected. Under these circumstances, the pedestrian information to be tracked cannot be obtained by tracking-by-detection. However, the originally obscured pedestrian 3 reappeared in the 115th frame. The reason why this labeled pedestrian can be re-detected is through the help of dynamic gallery tree shown on the right hand side of frame 115. Many traditional tracking methods will discard the original tracking number after obscuring for a period of time, and identify re-appearing pedestrians as new tracking targets. Conversely, if a dynamic gallery does exist, the system can still retrieve a tracked target no matter how long it is obscured. Therefore, in the 115th frame, the system can re-assign the number 3 pedestrian into the corresponding cluster subtree and complete the tracking process. Figure 5 shows that when multi-camera tracking is performed, the dynamic gallery of each camera can be used to help track the same target that travels between cameras. In the 1056th frame and the 1650th frame, the pedestrians labeled 3 appear in two other different cameras. Through the assistance of their corresponding dynamic gallery, even if a target being tracked goes within the scope of other cameras, the newly formed instance can still be inserted into the corresponding cluster tree.

3. Experiment

To evaluate the effectiveness and efficiency of our framework, we split our experiments into three parts. Single camera tracking will be discussed in Section 3.1, runtime analysis will be elaborated in Section 3.2, and multiple camera tracking will be detailed in Section 3.3. To make a fair comparison with existing tracking systems, all the de-

Table 1: **Single camera tracking results on MOT16.**

Method	MOTA↑	MOTP↑	IDs↓	FM↓	Runtime↑
DeepSORT	32.5	75.4	513	1357	**40Hz**
MOTDT	36.3	**75.9**	**192**	**594**	20.6Hz
Ours(DG)	**36.7**	75.6	237	723	34.3Hz

Table 2: Runtime analysis on MOT16.

	MOT16-02	MOT16-04	MOT16-05	MOT16-09	MOT16-10	MOT16-11	MOT16-13
Avg. BBoxes	8.7	18.4	5.7	6.7	9.1	6.7	5.1
Runtime(Hz) w/o YOLO	35.2	19.2	38.4	36.8	34.8	36.7	38.8
Runtime(Hz) w/ YOLO	20.2	4.2	23.4	21.8	19.8	21.7	23.8

Table 3: Multi-camera tracking results on DukeMTMC.

Method	IDF1	IDP	IDR
DeepCC	79.56	79.68	79.44
Ours(DG)	73.82	74.26	71.32

tected bounding boxes are generated by YOLOV3 [15]. Besides, our appearance feature embedder is pre-trained by DukeMTMC dataset [17]. When conducting triplet loss training, the margin m used for hyperparameter setting is determined by soft margin. After training, this margin value m can be used as the threshold for subsequent clustering tasks.

3.1. Single Camera Tracking Results

We use MOT16 benchmark [12] to evaluate the performance of our dynamic-gallery-based tracking method (as shown in Table 1). We simultaneously use YOLOV3 detector as well as the seven training video sequences used in MOT16 as testing tools. When we compare our method with deep SORT (also adopt YOLOV3 as detector), our method outperforms deep SORT in some important evaluation metrics. Among these metrics, our method makes ID switches (IDs) and Fragmentation (FM) drop significantly. This is mainly due to the fact that dynamic gallery can help correct the mistakes made by the tracklet. We also compare our method with deep SORT when long-term occlusion occurs, as shown in Figure 5. As to MOTDT [11], since its architecture is mainly a two-step detector, the results of multi-object tracking precision (MOTP) is better than that of our method. As to ID switches and fragmentation, MOTDT also slightly defeats us, but our method is much better than deep Sort, which is also a one-step detector.

3.2. Runtime Analysis

In this subsection, we compute runtime under different situations. We did a runtime performance analysis for the

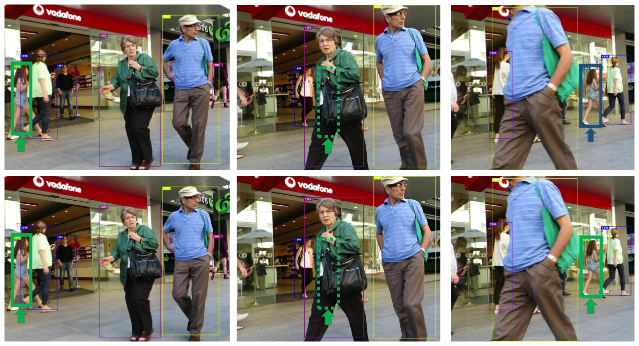

Figure 5: Qualitative comparison of long-term occlusion (Top row: DeepSORT. Bottom row: Ours(DG).). This example shows our method can retrieve dynamic gallery after the tracklet is not detected for T_{Lost} frames (green dashed rectangle) such that the tracked target (assigned by green arrow) can tolerate long-term occlusion and prevent ID switches.

number of people in different videos. We would like to confirm that the proposed framework still has real-time performance under the high number of pedestrians in each frame. From Table 2, it is clear that except for MOT16-04, other videos all reach 30 fps results. This means our tracker can perform tracking in real-time for most videos. To compute the runtime of our complete framework, we also add the time spent by YOLOV3 detector. Therefore, the average runtime of our framework is about 19.2 fps, which is close to real-time.

3.3. Multi-camera Tracking Results

In the final part of experiments, we test our framework with DukeMTMC's trainval-mini dataset. We compare our system with a state-of-the-art tracker DeepCC [18] (as shown in Table 3), and DeepCC are better than our results 5-10% for all evaluation metrics. However, the core technique used by DeepCC is to apply post-processing to match each camera's tracked trajectories, which is very time-consuming and basically cannot perform real-time tracking work. As for our system, it still performs competitively under high frame rate, which means it can be used in real-world applications.

3.4. Advanced Application and Ablation Study

In this subsection, we will show two advanced applications of the dynamic gallery and examine their feasibility through the assistance of tree visualization. In Figure 6, we built separate dynamic gallery trees in the field of view seen by Camera1 and Camera2, respectively. Then, tree synchronization is performed every 600 frames (the 600th frame in Figure 6 is for tree synchronization, which can be seen by the change of its tree). Since the benchmark videos adopted

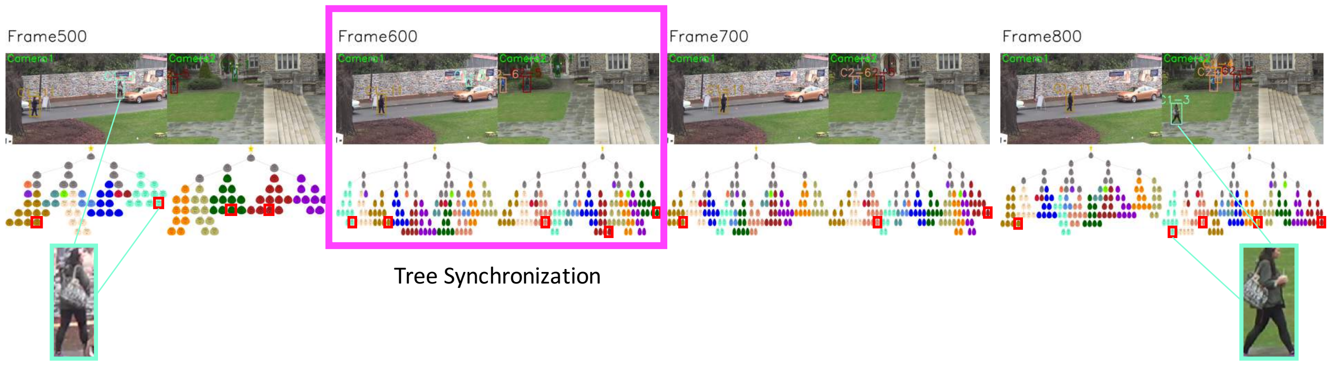

Figure 6: Tree visualization for tree synchronization.

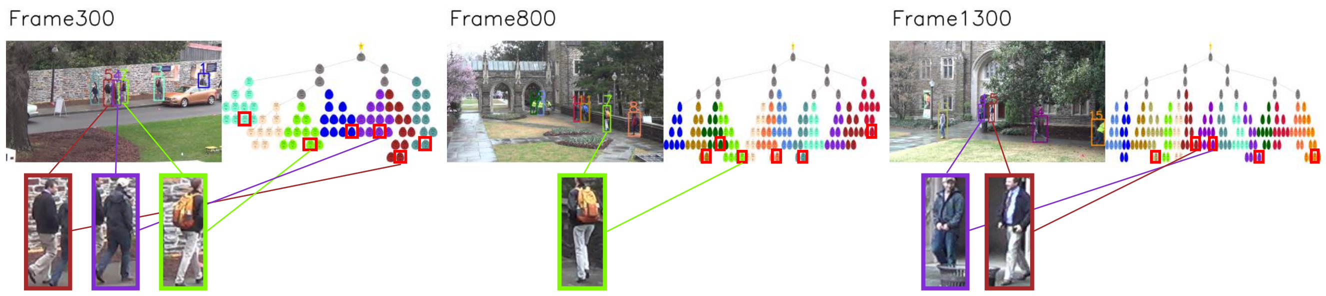

Figure 7: Tree visualization for group tracking.

in this experiment have 60 fps frame rate, every 600 frames equals to 10 seconds. We will use centralized servers for tree synchronization, because this application is a specific field of monitoring system (campus), the scope is not too large. Thus, the amount of information covered is not too much, so the use of a centralized control sever is appropriate. For the actual implementation of tree synchronization, we directly use the insert algorithm in Section 2.2 to insert the root of the subtree into another tree. After rotation, the synchronized tree will be optimized by all the inserted information for the balanced tree. When the target to be tracked moves from Camera1 to Camera2 (the 800th frame in Figure 6), after the tree synchronization, the dynamic gallery tree corresponding to Camera2 can retrieve the correct target. In Figure 7, we use group tracking as our second advanced application. In the 300th frame in Figure 7, we tracked down to three people who might be in the same group (framed with dark red, purple, and chartreuse colors, respectively). After tracking through our MTMCT system, frame 800 and frame 1300 are images taken by two different cameras. At this point, the two people respectively framed by dark red and purple appear in these two separate frames at the same time. As to the people who is framed by chartreuse appears in another independent camera view. Therefore, we can say the two people who are framed by dark red and purple belong to the same group. In the multi-camera group tracking application, since it can be used as a part of security system, the real-time requirements are very important in operation. The dynamic gallery-based system we proposed is able to operate in real-time, and this is indeed very useful in real-world applications.

4. Conclusion

In this paper, we have formulated the object association problem of MTMCT as an online extreme clustering problem. We then propose a concept of dynamic gallery. The dynamic gallery is constructed in an online clustering process, and it is also part of the tracking system that performs object association. Detailed techniques that we adopted for performing high-frame-rate are presented. In addition, the three parts of experiments validate our effectiveness and efficiency. Our method outperforms the state-of-the-art real-time single camera trackers, and we harvest the competitive results with state-of-the-art multi-camera tracking method. The improved performance of our method indicates that it is capable of being applied to real-time MTMCT applications.

Acknowledgment: This work was supported in part by MOST grants 108-2634-F-001-003 and 107-2218-E-155-011-MY2.

References

[1] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *Image Processing (ICIP),*

2016 IEEE International Conference on, pages 3464–3468. IEEE, 2016.

[2] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[3] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer, 2014.

[7] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[8] A. Kobren, N. Monath, A. Krishnamurthy, and A. McCallum. A hierarchical algorithm for extreme clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 255–264. ACM, 2017.

[9] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[11] C. Long, A. Haizhou, Z. Zijie, and S. Chong. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *ICME*, 2018.

[12] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[14] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint*, 2017.

[15] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[16] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[17] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision*, pages 17–35. Springer, 2016.

[18] E. Ristani and C. Tomasi. Features for multi-target multi-camera tracking and re-identification. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[19] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[20] Y. Sun, L. Zheng, W. Deng, and S. Wang. Svdnet for pedestrian retrieval. *arXiv preprint*, 1(6), 2017.

[21] N. Wojke and A. Bewley. Deep cosine metric learning for person re-identification. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 748–756. IEEE, 2018.

[22] X. Zhang, H. Luo, X. Fan, W. Xiang, Y. Sun, Q. Xiao, W. Jiang, C. Zhang, and J. Sun. Alignedreid: Surpassing human-level performance in person re-identification. *arXiv preprint arXiv:1711.08184*, 2017.

[23] Z. Zheng, L. Zheng, and Y. Yang. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. *arXiv preprint arXiv:1701.07717*, 3, 2017.

[24] Z. Zhong, L. Zheng, D. Cao, and S. Li. Re-ranking person re-identification with k-reciprocal encoding. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 3652–3661. IEEE, 2017.