Multiple Target Tracking Based on Undirected Hierarchical Relation Hypergraph

Longyin Wen Wenbo Li Junjie Yan Zhen Lei Dong Yi* Stan Z. Li
Center for Biometrics and Security Research & National Laboratory of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences, China
{lywen,wbli,jjyan,zlei,dyi,szli}@cbsr.ia.ac.cn

Abstract

Multi-target tracking is an interesting but challenging task in computer vision field. Most previous data association based methods merely consider the relationships (e.g. appearance and motion pattern similarities) between detections in local limited temporal domain, leading to their difficulties in handling long-term occlusion and distinguishing the spatially close targets with similar appearance in crowded scenes. In this paper, a novel data association approach based on undirected hierarchical relation hypergraph is proposed, which formulates the tracking task as a hierarchical dense neighborhoods searching problem on the dynamically constructed undirected affinity graph. The relationships between different detections across the spatio-temporal domain are considered in a high-order way, which makes the tracker robust to the spatially close targets with similar appearance. Meanwhile, the hierarchical design of the optimization process fuels our tracker to long-term occlusion with more robustness. Extensive experiments on various challenging datasets (i.e. PETS2009 dataset, ParkingLot), including both low and high density sequences, demonstrate that the proposed method performs favorably against the state-of-the-art methods.

1. Introduction

Multi-target tracking is an interesting but difficult problem. Although numerous tracking methods have been proposed in literatures, their performance are unsatisfactory in practical applications. As shown in Fig. 1 (a), most of the previous methods focus on the pairwise relationships (e.g. appearance and motion pattern similarities) of the tracklets in the local limited temporal domain, rather than among multiple tracklets across the whole video temporal domain in a global view. When the targets walk closely with similar appearance or motion patterns, as denoted by the circles

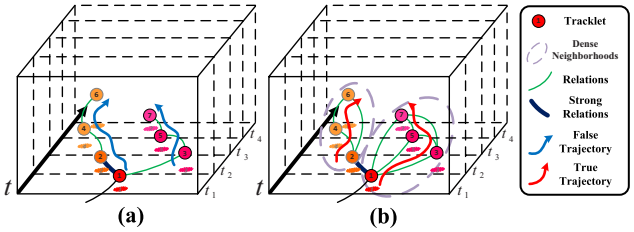

Figure 1. (a) describes the previous methods fail to handle the challenge that the targets walk closely with similar appearance or motion patterns, and (b) describes our undirected hierarchical relation hypergraph based tracker successfully handle this challenge. The circles represent different tracklets and their colors represent the inherent patterns (e.g. appearance and motion patterns). Similar colors represent similar patterns of the tracklets. Previous methods, which focus on the pairwise similarities of spatial-temporal neighboring tracklets in local limited temporal domain, generate wrong trajectories (blue splines). On the contrary, the proposed method searching the dense neighborhoods in the tracklet relation graph/hypergraph, which considers the similarities among multiple tracklets across the temporal domain, generate correct trajectories (red splines).

♯1 and ♯2 in Fig. 1, the identity switches will follow the previous trackers [24, 11, 26, 12, 18, 6, 4, 19, 10, 9]. To alleviate the issues, method based on minimum clique graphs optimization has been developed [25], which considers the relationships between different detections across the temporal domain. However, it is hard to handle the non-linear motion of the targets in crowded scenes, especially when the occlusion happens, mainly due to the unreliable hypothetical nodes generation in optimization process.

In this paper, an undirected Hierarchical relation Hypergraph based Tracker (H^2T) is proposed, which formulates the tracking task as searching multiple dense neighborhoods on the constructed undirected relation affinity graph/hypergraph, as described in Fig. 1 (b). Different from the previous methods, we consider the relationships between different detections across the temporal domain globally. Meanwhile, a local-to-global strategy is exploited

*Corresponding author.

1063-6919/14 $31.00 © 2014 IEEE
DOI 10.1109/CVPR.2014.167

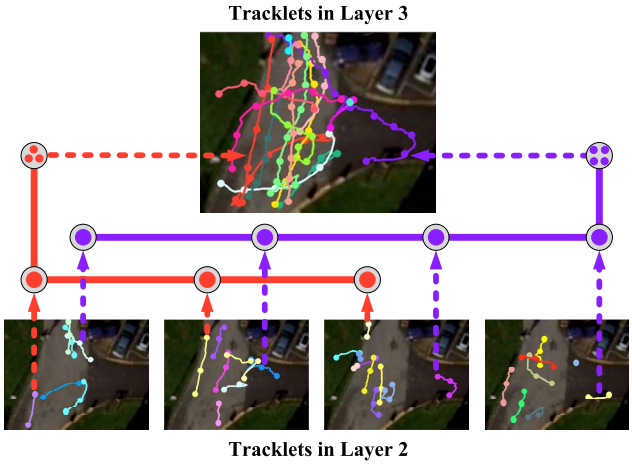

Figure 2. The overlooking of the tracklets in optimization process of the sequence PETS2009-S2L1. The tracklets in four segments of the second layer are combined to generate the target tracklets in the third layer. Two examples of the optimized tracklets are presented as the red and purple pipelines in the figure.

to cluster the detections hierarchically, which greatly reduces the computational complexity in dense neighborhoods searching, while handles the sudden variations of the target's appearance and motion effectively. Firstly, the tracking video is divided into a few segments in the temporal domain and the dense neighborhoods are searched in each segment to construct multiple longer tracklets. Then the nearby segments are merged to construct the new segment division for the next layer. These two steps are iterated until only one segment exists in the layer, i.e. the whole video span, and the dense neighborhoods searching is carried out in that segment to obtain the final target trajectories.

The main contributions of this paper are summarized as follows. (1) The multi-target tracking is first modeled as a dense neighborhoods searching problem on the hierarchically constructed tracklet undirected affinity graph/hypergraph. (2) The motion properties and appearance information of the targets are fully exploited in the optimization process by considering the high-order relationships between multiple tracklets in the constructed hypergraph. (3) Tracking experiments on various publicly available challenging sequences demonstrate the proposed method achieves impressive performance, especially when the long-term occlusion and similar appearance challenges happen in the crowded scene.

2. Related Work

Recently, tracking-by-detection methods [24, 11, 1, 26, 12, 2, 18, 6, 4, 3, 25, 19, 10, 9, 22, 23] become popular, which associate multiple input detection responses in different frames to generate the trajectory of targets. Some researchers formulate the association task as a matching problem, which match the detections with similar appearance and motion patterns in consecutive frames, e.g. bi-partite matching [24] and multiple frames matching [19]. Unfortunately, the limited-temporal-locality degrades their performance when long-term occlusion, complex motion, or spatially close targets with similar appearance challenges happen.

Other researchers construct a k-partite graph to describe the relationships between different detections and use some optimization methods to complete the association task (e.g. Network flow [26, 18], K-Shortest Path (KSP) [4], Maximum Weight Independent Set [6], Linear Programming [11]). These methods, generally termed *global*, make an effort to reduce or remove the limited-temporal-locality assumption in optimization, which can overcome the first two aforementioned challenges to some degree. However, they also only consider the relationships of detections in local consecutive frames, so that they have difficulties in discriminating the spatially close targets with similar appearance. Recently, in [2], a continuous energy minimization based method is proposed, in which the local minimization of a nonconvex energy function is well found by the standard conjugate gradient optimization. The subsequent work [3] designs a discrete-continuous optimization framework, which decomposes the tracking task as two iteratively optimization steps, i.e. one is the data association of the tracklets and the other one is the trajectories fitting. In addition, some methods model the association task as a linking problem and use the hierarchical Hungarian algorithm [9, 22, 23] to complete the tracking task.

Another work related to this paper is [8], which simultaneously handles the multi-view reconstruction and multi-target tracking tasks in the 3D world coordinates. In [8], a directed graph is constructed to describe the association relationships between candidate couplings, which are constituted by the detections appeared in different camera-view at the same time. The edges in the graph describe the association probability between the temporal consecutive pairwise couplings. Different from [8], we construct an undirected hypergraph, in which the nodes represent the tracklets, and the hyperedges are constituted by multiple tracklets across the temporal domain. Meanwhile, since the hypergraph construction and task objective are different, the optimization strategies in these two methods are totally different.

3. Hierarchical Relation HyperGraph based Tracker

Given the frame-by-frame detections, the tracking task is modeled as a hierarchical dense neighborhoods searching problem on the tracklet affinity relation graph/hypergraph, which is constructed dynamically to describe the relationships between the tracklets generated in the previous layer.

Notably, a detection is regarded as a degenerate tracklet, containing only one detection response.

The tracking video is firstly divided into u segments in the temporal domain. Let Δ_r^l represent the time interval of the r-th segment in layer l and T be the total frame length of the video. The time interval set of these u segments is represented as $\{\Delta_1^l, \cdots, \Delta_u^l\}$ such that $\Delta_i^l \bigcap \Delta_j^l = 0$ and $\bigcup_{i=1}^{u} \Delta_i^l = T$. A relation affinity graph is constructed in each segment at the current layer. Let $\mathcal{G}_r^l = (V_r^l, E_r^l)$ represent the constructed graph in the r-th segment of l-th layer, where V_r^l and E_r^l are the node and edge set of the graph respectively. The graph node set is further represented as $V_r^l = \{v_{i,r}^l\}_{i=1}^n$, where $v_{i,r}^l$ is the i-th node and n is the number of nodes in the graph. Each node in $\mathcal{G}_r^l$ corresponds to a tracklet in the current segment. The corresponding tracklet set is defined as $\boldsymbol{\mathcal{T}}_r^l = \{\mathcal{T}_{i,r}^l\}_{i=1}^n$. The edges of the graph describe the relationships between different tracklets and their weight indicate the probability of the tracklets belonging to the same target. Without ambiguity, we use $v_{i,r}^l$ and $\mathcal{T}_{i,r}^l$ interchangeably to represent the i-th tracklet in the r-th segment at layer l in the rest of this paper.

Different from previous works where only the pairwise relationships are considered, in this method, the high-order relationships among multiple tracklets are integrated in the relation graph for the first a few layers, i.e. the edges in the graph involve more than just two vertices. In this way, the motion and trajectory smoothness constraints of the target can be fully used to ensure the tracking performance. We extend the dense neighborhoods searching algorithm [14] to handle the dense neighborhoods revealing problem in both graph and hypergraph, and generate the longer tracklets by stitching the tracklets in each revealed dense neighborhoods. Notably, the graphs/hypergraphs in all segments are processed in the same fashion.

After completing the dense neighborhoods searching problem in all segments of current layer, the nearby segments are merged to generate a new segment division Δ^{l+1} for the next layer. Then, we also construct the graph/hyerpgrah in each segment of Δ^{l+1} and reveals the dense neighborhoods on each graph/hypergraph to generate the longer tracklets. These two steps are iterated until only one segment remains in the layer. Finally, the dense neighborhoods searching is performed again on the constructed relation affinity graph of that segment to obtain the final target trajectories. As an example, the optimization process from layer 2 to 3 of PETS2009-S2L1 is presented in Fig. 2. Without ambiguity, we omit the segment index r and the layer index l is the following sections.

4. Undirected Affinity Graph Construction

For the current segment, we construct a global relation affinity graph $\mathcal{G} = (V, E)$, which is a *complete graph* describing the relationships between different tracklets. $V = \{v_1, \cdots, v_n\}$ is the graph node set. E is the graph edge/hyperedge set, i.e. $E \subset \overbrace{V \times \cdots \times V}^{m}$, where m is the number of vertices involving in each edge/hyperedge. We use the bold symbol $\boldsymbol{e}^m = (v_1, \cdots, v_m)$ to represent the m-tuple vertices involving in the edges/hyperedges of the graph. Obviously, the constructed graph is a hypergraph when $m > 2$ and degenerate to a general graph when $m = 2$. The affinity array $\mathcal{A}$ of the graph $\mathcal{G}$ is a $\overbrace{n \times \cdots \times n}^{m}$ super-symmetry array, in which the elements reflect the probability of the tracklets in $\boldsymbol{e}^m$ belong to the same target. The affinity value in the array $\mathcal{A}(\boldsymbol{e}^m) = 0$, if $\boldsymbol{e}^m \notin E$; otherwise $\mathcal{A}(\boldsymbol{e}^m) \geq 0$.

The design of calculating the affinity values in the undirected relation affinity graph plays a central role in H^{2}T, which indicates how probably the tracklets belong to the same target. We calculate the affinity value $\mathcal{A}(\boldsymbol{e}^m)$ of the edge/hyperedge $\boldsymbol{e}^m$ by three factors: appearance, motion and trajectory smoothness. The appearance factor indicates the appearance similarity between the tracklets in $\boldsymbol{e}_m$; the motion factor indicates the motion consistence between the tracklets in $\boldsymbol{e}^m$; and the smoothness factor indicates the physical smoothness of the merged tracklets constituted by the tracklets in $\boldsymbol{e}_m$. Therefore, the edge affinity value is calculated as:

$$\mathcal{A}(\boldsymbol{e}^m) = \omega_1 \mathcal{A}_a(\boldsymbol{e}^m) + \omega_2 \mathcal{A}_m(\boldsymbol{e}^m) + \omega_3 \mathcal{A}_s(\boldsymbol{e}^m), \quad (1)$$

where $\mathcal{A}_a(\cdot)$, $\mathcal{A}_m(\cdot)$ and $\mathcal{A}_s(\cdot)$ are the appearance, motion and smoothness affinity, respectively. ω_1, ω_2 and ω_3 are the preset balance parameters of these three factors. Obviously, if the appearing temporal domain of the tracklets v_i and v_j in $\boldsymbol{e}^m$ overlap each other, they should not belong to the same target, so we set $\mathcal{A}(\boldsymbol{e}^m) = 0$. Thus, we just need to consider the case that neither two tracklets in $\boldsymbol{e}^m$ overlap each other of temporal domain, when we calculate the appearance, motion and smoothness affinities.

4.1. Appearance Affinity

The appearance of an object is a crucial part to disambiguate it from the background and other objects. For our H^{2}T, two kinds of features in the first and last frames of the tracklet are adopted to describe its appearance, i.e. color histogram feature and shape gradient histogram feature. We use 8 bins for each channel of RGB color histogram and 36 dimensions for shape gradient histogram. Without loss of generality, the tracklet v_i is assumed to appear before v_j in temporal domain. Then the appearance affinity is calculated as follows:

$$\mathcal{A}_a(\boldsymbol{e}^m) = \sum_{v_i, v_j \in \boldsymbol{e}^m} \sum_{k=1,2} \lambda_k \mathcal{H}(\tilde{f}_k(v_i), \hat{f}_k(v_j)), \quad (2)$$

where $\tilde{f}_1(v_i)$ and $\tilde{f}_2(v_i)$ are the color and shape histograms of v_i's last frame, $\hat{f}_1(v_i)$ and $\hat{f}_2(v_i)$ are the color and shape histograms of v_i's first frame, $\mathcal{H}(\cdot,\cdot)$ represents the cosine distance between two feature vectors and λ_1, λ_2 are the preset balance parameters.

4.2. Motion Affinity

A defining property of tracking is that the targets move slowly relative to the frame rate, leading to a fact that the velocity of a target can be set as a constant in a small temporal domain, which is powerful enough to constrain the trajectories of the targets. The motion affinity reflects the motion consistency of the tracklets in $\boldsymbol{e}^m$, which is defined as:

$$\mathcal{A}_m(\boldsymbol{e}^m) = \sum_{v_i, v_j \in \boldsymbol{e}^m} \mathcal{S}_m(v_i, v_j), \tag{3}$$

where $\mathcal{S}_m(v_i, v_j)$ is the motion similarity between the tracklets v_i and v_j. Specifically, there is only one detection response in each tracklet of the first layer, leading to no motion information contained in each tracklet. So we set the motion affinity value $\mathcal{A}_m(\boldsymbol{e}^m) = 0$ at the first layer.

We assume the tracklet v_i appears before v_j. Let $D_{v_i} = \{d_s^{v_i}\}_{s=1}^{|\mathcal{T}_{v_i}|}$ be the detection set of the tracklet v_i in ascending order of temporal domain, where $d_s^{v_i}$ is the s-th detection in the tracklet and $|\mathcal{T}_{v_i}|$ is the number of detections in v_i. We define a linear motion function $\mathcal{P}(\ell(\cdot), \Delta t, \vec{v})$, which generates a predicted position starting from $\ell(\cdot)$, i.e. $\mathcal{P}(\ell(\cdot), \Delta t, \vec{v}) = \ell(\cdot) + \Delta t \cdot \vec{v}$, where $\ell(\cdot)$ is the position of the detection, Δt is the time gap and $\vec{v}$ is the constant velocity in the temporal domain. Let $\mathrm{t}(d_s^{v_i} - d_s^{v_j})$ be the time gap between the detections $d_s^{v_i}$ and $d_s^{v_j}$. Due to fact that the strong correlations between nearby frames and the weak correlations between distant frames, we only use τ detections in the last a few frames of v_i and the first a few frames of v_j to calculate the motion similarity between them. To reduce the influence of noise, the deviations of both v_i backward prediction and v_j forward prediction are used to measure the motion consistency between the two tracklets. Let $\ell_{p,q}^{v_i}(d_{l_i}^{v_i}) = \mathcal{P}\big(\ell(d_{l_i}^{v_i}), \mathrm{t}(d_{l_i}^{v_i} - d_1^{v_j}), \frac{\ell(d_p^{v_i}) - \ell(d_q^{v_i})}{\mathrm{t}(d_p^{v_i} - d_q^{v_i})}\big)$ be the predicted position starting from the detection $d_{l_i}^{v_i}$ with the average velocity between the positions of $d_p^{v_i}$ and $d_q^{v_i}$, $l_i = |\mathcal{T}_{v_i}|$ is the number of detection responses in the tracklet $\mathcal{T}_{v_i}$. Then, the motion similarity between two tracklets v_i and v_j is calculated as

$$\begin{aligned}\mathcal{S}_m(v_i, v_j) = &\sum_{p=2}^{\tau}\sum_{q=1}^{p-1} \exp\big(-\|\ell(d_{l_i}^{v_i}) - \ell_{p,q}^{v_j}(d_1^{v_j})\|^2/\sigma_m^2\big) \\ + &\sum_{p=l_i-\tau}^{l_i-1}\sum_{q=p+1}^{l_i} \exp\big(-\|\ell(d_1^{v_j}) - \ell_{p,q}^{v_i}(d_{l_i}^{v_i})\|^2/\sigma_m^2\big). \end{aligned} \tag{4}$$

4.3. Trajectory Smoothness Affinity

The target trajectories should be continuous and smooth in spatio-temporal domain, which provides us with effective information to measure the confidence of the tracklets in $\boldsymbol{e}^m$ belonging to the same target. We firstly merge the tracklets in the hyperedge $\boldsymbol{e}^m$ according to their appearing time to get the hypothetical tracklet $\mathcal{T}_{\boldsymbol{e}^m}$. Let $D_{\boldsymbol{e}^m} = \bigcup_{i=1}^{m}\{d_j^{v_i}\}_{j=1}^{l_i}$ be the sorted detection response set in the tracklet $\mathcal{T}_{\boldsymbol{e}^m}$ according to the ascending order of temporal domain, where $l_i = |\mathcal{T}_{v_i}|$ represents the number of detection responses in the tracklet. We sample some detections in the tracklet $\mathcal{T}_{\boldsymbol{e}^m}$ with equal step δ to get the fitting detection point set $D_{\boldsymbol{e}^m}^f = \{d_i\}_{i=1+k\cdot\delta, k\in\mathbb{N}}^{1\le i\le\sum|\mathcal{T}_{v_i}|}$. The remained detection point set is $D_{\boldsymbol{e}^m}^r = D_{\boldsymbol{e}^m} \setminus D_{\boldsymbol{e}^m}^f$. We use $D_{\boldsymbol{e}^m}^f$ to get the fitted trajectory $\widehat{\mathcal{T}}_{\boldsymbol{e}^m}$ of the merged hypothetical tracklet using the cubic spline fitting algorithm, and calculate the deviations between the detection points in the set $D_{\boldsymbol{e}^m}^r$ and the points in the fitted trajectory $\widehat{\mathcal{T}}_{\boldsymbol{e}^m}$ with the same time index. Thus, the smooth affinity of hyperedge $\boldsymbol{e}^m$ is calculated as:

$$\mathcal{A}_s(\boldsymbol{e}^m) = \exp\big(-\sum_{d_i \in D_{\boldsymbol{e}^m}^r} \|\ell(d_i) - \ell(\widehat{\mathcal{T}}_{\boldsymbol{e}^m}(\mathrm{t}(d_i)))\|^2/\sigma_s^2\big), \tag{5}$$

where $\ell(d_i)$ is the position of the detection response d_i, $\mathrm{t}(d_i)$ is the appearing time of the detection response d_i, and $\widehat{\mathcal{T}}_{\boldsymbol{e}^m}(\mathrm{t}(d_i))$ represents the detection response in the trajectory $\widehat{\mathcal{T}}_{\boldsymbol{e}^m}$ at time $\mathrm{t}(d_i)$.

5. Tracklets Dense Neighborhoods Searching

After constructing the tracklet relation graph, we reveal the dense neighborhoods on it. The core problem in dense neighborhoods revealing is how to get the number of neighborhoods and the number of nodes in each dense neighborhood. Here, the number of neighborhoods and the number of nodes in each neighborhood are regarded as the hidden variables in optimization process, which are inferred by maximizing the average affinity value of the neighborhoods. In this way, multiple dense neighborhoods can be successfully discovered. Then, the final tracklets in each segment can be obtained by stitching the tracklets in each neighborhood correctly. In this section, we detail the dense neighborhoods searching in each segment.

To ensure all dense neighborhoods are revealed, we set each node in the graph as a starting point and search its dense neighborhood. If the tracklet belongs to a real dense neighborhood, the affinities between it and other nodes in this real dense neighborhood are usually large. Thus, its the dense neighborhood will be found out. On the contrary, if the tracklet has weak relationships with other tracklets, the affinities between them will be low, which indicates that it does not belong to any coherent dense neighborhoods. Thus, the tracklet will be treated as a false positive and ig-

nored in the final clusters. For a starting point v_p, we aim to find out its dense neighborhood $\mathcal{N}(v_p)$, which contains the maximal average affinity value. The optimization problem is formulated as

$$\begin{aligned} &\mathcal{N}^*(v_p) = \arg\max_{\mathcal{N}(v_p)} \mathcal{C}(v_p \cup \mathcal{N}(v_p)) \\ &\text{s.t.} \quad \mathcal{N}(v_p) \subset V, v_p \notin \mathcal{N}(v_p), |\mathcal{N}(v_p)| = k. \end{aligned} \tag{6}$$

where $\mathcal{C}(v_p \cup \mathcal{N}(v_p))$ is the affinity measure function, that reflects the affinity distribution in the graph. Let $U = \{v_p\} \cup \mathcal{N}(v_p)$ represent the set containing the vertex v_p and its k neighborhood vertices. Thus, the subset $U \subset V$ contains $k+1$ vertices. Let $y \in \mathbb{R}^n$ be the indicator vector of the subset U, i.e. $y_i = 1$, if $v_i \in U$; otherwise, $y_i = 0$. Then, the subjects in (6) are converted to $\sum_{i=1}^n y_i = k+1$, $y_i \in \{0,1\}$ and $y_p = 1$. The first two constraints reflect that there exists $k+1$ tracklets belonging to the same target, which is indicated by the solution y, and the last one reflects the solution must contain the tracklet v_p.

Let E_U be the edge set corresponding to the vertex set U. If the tracklets in U belong to the same target, most of the edges in E_U should have large affinity values. Naturally, the total affinity value of the edge set E_U is calculated as $\widetilde{\mathcal{C}}(U) = \sum_{\boldsymbol{e}^m \in E_U} \mathcal{A}(\boldsymbol{e}^m)$. In our tracking task, the affinity values in the graph/hypergraph are all non-negative, i.e. $\mathcal{A}(\boldsymbol{e}_m) \geq 0$. Obviously, $\widetilde{\mathcal{C}}(U)$ usually increases as the number of vertices in subset U increases. Thus, it is more reasonable to use the average affinity value to describe the confidence of the dense neighborhoods than the total affinity value, which can successfully handle the dimension diversity between different dense neighborhoods. Since $\sum_{i=1}^n y_i = k+1$, there are $(k+1)^m$ summands in $\widetilde{\mathcal{C}}(U)$. The average value $\mathcal{C}(U)$ is taken as the objective to indicate the true dense neighborhood, that is

$$\mathcal{C}(U) = \frac{1}{(k+1)^m}\widetilde{\mathcal{C}}(U) = \sum_{\boldsymbol{e}^m \in E_U} \mathcal{A}(\boldsymbol{e}^m) \overbrace{\frac{y_1}{k+1} \cdots \frac{y_m}{k+1}}^{m}.$$

Then the optimization problem in (6) can be further simplified as:

$$\begin{aligned} &\max_x g(x) = \sum_{\boldsymbol{e}^m \in E_U} \mathcal{A}(\boldsymbol{e}^m) \overbrace{x_1 \cdots x_m}^{m} \\ &\text{s.t.} \quad \sum_{i=1}^n x_i = 1, x_i \in \{0, \epsilon\}, x_p = \epsilon. \end{aligned} \tag{7}$$

where $x_i = \frac{y_i}{k+1}$ and $\epsilon = \frac{1}{k+1}$. Essentially, this is a combinatorial optimization problem, which is NP-hard. To reduce its complexity, the subjects in (7) are relaxed to $x_i \in [0, \epsilon]$, i.e. $0 \leq x_i \leq \epsilon$. Then, the pairwise updating algorithm [13] is used to solve the problem in (7) effectively. Please refer to [13] for more details about the optimization strategy of the problem in (7).

6. Post-Processing

As discussed in section 5, we set each node in the relation graph as a starting point to get the optimal clusters (dense neighborhoods) $\Psi = \{\psi_i\}_{i=1}^n$ and the corresponding average affinity values, where n is the total number of starting points. The average affinity value reflects the reliability of the neighborhood to be correct. Ψ is sorted to get the processed clusters $\widetilde{\Psi} = \{\tilde{\psi}_i\}_{i=1}^n$ according to the average affinity values in the descending order. Let Ψ^* be the optimal clusters after post-processing. We set $\Psi^* = \emptyset$ at first and add the clusters in $\widetilde{\Psi}$ sequentially. For the i-th component $\tilde{\psi}_i \in \widetilde{\Psi}$, we check whether it intersects with the clusters in Ψ^*. If there is no intersection between $\tilde{\psi}_i$ and all clusters in Ψ^*, we add $\tilde{\psi}_i$ directly to Ψ^*, i.e. $\Psi^* \leftarrow \Psi^* \bigcup \{\tilde{\psi}_i\}$. Otherwise, we use the designed *Conservative Strategy* or *Radical Strategy* to add the cluster $\tilde{\psi}_i$ in different layers. Suppose the cluster $\tilde{\psi}_i$ intersects with ψ_k^*. In the first a few layers in optimization, the tracklets are so short that contain limited evidence to determine which target it belongs to. To avoid identity switches, a *Conservative Strategy* is designed as removing the intersection part from the cluster $\tilde{\psi}_i$ and then adding it to Ψ^*, i.e. $\tilde{\psi}_i \leftarrow \tilde{\psi}_i / \psi_k^*$ and $\Psi^* \leftarrow \Psi^* \bigcup \{\tilde{\psi}_i\}$. On the other hand, the tracklets in the last a few layers contain enough evidence to determine which cluster it belongs to. Thus, in order to reduce the fragmentations, a *Radical Strategy* is designed as directly merging the clusters $\tilde{\psi}_i$ and ψ_k^*, i.e. $\psi_k^* \leftarrow \psi_k^* \cup \tilde{\psi}_i$. In this way, the post-processed dense neighborhood set Ψ^* is obtained. According to the dense neighborhood set Ψ^*, the optimal tracklets in the segment are acquired by stitching the tracklets in each cluster.

7. Experiments

We evaluate H^{2}T on six challenging publicly available video sequences, including both high-density and low-density sequences. Five of them are part of the PETS2009 database [15] and the rest one is the ParkingLot sequence from [25]. *Notably, we track all the targets in the 2D image plane*. For the quantitative evaluation, we rely on the widely used CLEAR MOT metrics [5]. The Multi-Object Tracking Accuracy (MOTA) combines all errors (False Negatives (FN), False Positives (FP), Identity Switches (IDs)) into a single number. The Multi-Object Tracking Precision (MOTP) averages the bounding box overlap over all tracked targets as a measure of localization accuracy. Mostly Lost (ML) and Mostly Tracked (MT) scores are computed on the entire trajectories and measure how many Ground Truth trajectories (GT) are lost (tracked for less than 20% of their life span) and tracked successfully (tracked for at least 80%). Other metrics include Recall (Rcll), Precision (Prcsn), Fragmentations of the target trajectories (FM) and False Alarms per Frame (Fa/F).

As discussed in [17], the input detection responses and

manually annotated evaluation groundtruth greatly influence the quantitative results of the tracking performance. For fair and comprehensive comparison, we use the same input detection responses and manually annotated evaluation groundtruth for all trackers in each sequence and take some tracking results directly from the published papers. Since some trackers complete the tracking task in 3D world coordinates, similar as [16], we evaluate the tracking performance in 3D world coordinates for the sequences from PETS2009. 2D evaluation is exploited on the ParkingLot sequence because of the lack of camera parameters. For the 3D evaluation, the hit/miss threshold of the distance between the output trajectories and the groundtruth on the ground plane is set to 1m. For the 2D evaluation, the hit/miss threshold of the intersection-over-union of the bounding boxes between the output trajectories and the groundtruth is set to 50%. Table 1 presents the quantitative comparison results of our H^{2}T and seven other state-of-the-art trackers [18, 2, 4, 3, 25, 19, 16]. Some qualitative tracking results of H^{2}T are presented in Fig. 3. Although MOTA reveals the comprehensive performance, IDs, FM and MT still play important roles in determining performance of the tracker. As expected, our H^{2}T outperforms the state-of-the-art trackers in IDs, FM and MT metrics on most of the six sequences and gives comparable or even better performances in MOTA simultaneously. Specifically, due to the different settings of the tracking area, depicted by the highlighted region in PETS2009 sequences in Fig. 3, the tracking results of [8, 9] in PETS2009 sequences are not listed here for comparison.

7.1. Implementation Details

We implement H^{2}T in C++ without any code optimization. Given the detection responses, the proposed method runs about 11-fps in the PETS2009 sequences and 6-fps in the ParkingLot sequence. The experiments are carried out on a Intel 3.2GHz PC platform with 16 GB memory.

The reference parameters used in this paper are presented as follows. The weight parameters in the affinity value calculation of (1) are, $\omega_1 = 0.6$, $\omega_2 = 0.2$, and $\omega_3 = 0.2$. The balance parameters between the RGB color histogram and shape gradient histogram of (2) are, $\lambda_1 = 0.1$ and $\lambda_2 = 0.04$. The sigma in motion and trajectory smoothness affinity values calculation of (4) and (5) are, $\sigma_m^2 = 5.0$ and $\sigma_s^2 = 4.0$. We use 4 and 3 order hypergraph in the first and second layers, respectively. The traditional graph is used for the remained layers. Every 6-8 frames are combined to generate the segments for the 1st layer and 3-5 segments are combined for the remained ones. For the post-processing strategy of H^{2}T, we use the *Conservative Strategy* for the first two layers and the *Radical Strategy* for the remaining ones.

7.2. Low-Density Sequences

PETS2009-S2L1. S2L1 is the most widely used sequence in multi-target tracking task. It consists of 795 frames and includes the non-linear motion, targets in close proximity and a scene occluder challenges. The detector may fail when the targets walk behind the scene occluder, which greatly challenges the performance of the trackers. Although the results presented in Table 1 seem to saturate on MOTA metric, our tracker achieves the lowest IDs (5) and highest MT (22).

ParkingLot. The sequence consists of 1000 frames of a relatively crowded scene. There are up to 14 pedestrians walking in parallel in a parking lot. It includes frequent occlusions, missed detections, and parallel motion with similar appearances challenges. As shown in Table 1, our tracker outperforms other trackers in nearly all evaluation metrics. Our MOTA, MT, IDs and FM are 88.4%, 11, 21 and 23 respectively. Notably, H^{2}T almost tracks all the targets successfully and achieves the lowest IDs (less than the second lowest one by 31).

Discussion. As presented in Table 1, in these two sequences, H^{2}T outperforms other trackers by reliably high MOTA and MT as well as stably low IDs and FM. Just considering the local similarities of detections, it is hard for other trackers [18, 4, 2, 3] to achieve robust tracking performance, especially when two targets with similar appearance walk closely. Note that our H^{2}T performs well in this challenge by considering the similarities among multiple different tracklets in a global view.

7.3. High-Density Sequences

PETS2009-S2L2. S2L2 consists of 436 frames with high dense crowd. Note that it contains 74 pedestrians moving non-linearly. The severe occlusion happens frequently in this sequence. Our tracker has the best performance in MOTA, ML, FN, Rcll, Prcsn and has comparable performance in other metrics.

PETS2009-S2L3. S2L3 is a challenging sequence with high crowd density. It consists of 240 frames with up to 44 pedestrians moving non-linearly. In addition, this sequence also includes frequent occlusions, missed detections and illumination variation challenges. H^{2}T presents the persuasive tracking performance with the highest MOTA, MT, ML, Rcll and Prcsn.

PETS2009-S1L1. PETS2009-S1L1-1 and PETS2009-S1L1-2 are two dense sequences including 221 and 241 frames respectively and both of them include the targets with linear motion. These two sequences are originally intended for person counting and density estimation. H^{2}T not only gives the impressive MOTA, but also stands out with the highest MT as well as lowest IDs.

Discussion. Compared to the low-density sequences, the superiority of H^{2}T on high-density sequences is more

Table 1. Quantitative comparison results of our H^{2}T with other state-of-the-art trackers. The input detection responses and evaluation groundtruth used in each sequence are presented. The tracking results of the methods marked with the asterisk are taken directly from the published papers and the others are obtained by running the publicly available codes with the same input detection responses and evaluation groundtruth used in our tracker. The symbol ↑ means higher scores indicate better performance while ↓ means lower scores indicate better performance. The red and blue color indicate the best and the second best performance of the tracker on that metric.

Sequence	Method	MOTA ↑	MOTP ↑	GT	MT ↑	ML ↓	FP ↓	FN ↓	IDs ↓	FM ↓	Rcll ↑	Prcsn ↑	Fa/F ↓
PETS-S2L1	*Anton *et al.* [16]	90.6%	**80.2%**	23	21	1	59	302	11	**6**	92.4%	98.4%	0.07
(Detection [21])	*Berclaz *et al.* [4]	80.3%	72.0%	23	17	2	126	641	13	22	83.8%	96.3%	0.16
(Groundtruth [15])	Anton *et al.* [2]	86.3%	78.7%	23	18	1	88	417	38	21	89.5%	97.6%	0.11
(795 frames)	Anton *et al.* [3]	88.3%	79.6%	23	19	**0**	**47**	396	18	14	90.0%	**98.7%**	**0.06**
(up to 8 targets)	Pirsiavash *et al.* [18]	77.4%	74.3%	23	14	1	93	742	57	62	81.2%	97.2%	0.12
	H^{2}T	**92.7%**	72.9%	23	**22**	**0**	62	**222**	**5**	10	**94.4%**	98.4%	0.08
PETS-S2L2	*Anton *et al.* [16]	56.9%	59.4%	74	**28**	12	622	2881	99	73	65.5%	89.8%	1.43
(Detection [20])	*Berclaz *et al.* [4]	24.2%	60.9%	74	7	40	**193**	6117	**22**	**38**	26.8%	92.1%	**0.44**
(Groundtruth [15])	Anton *et al.* [2]	48.5%	62.0%	74	15	14	301	3850	152	128	53.9%	93.7%	0.69
(436 frames)	Anton *et al.* [3]	48.0%	61.6%	74	15	11	245	3957	143	125	52.6%	94.7%	0.56
(up to 33 targets)	Pirsiavash *et al.* [18]	45.0%	**64.1%**	74	7	17	199	4257	137	216	49.0%	**95.4%**	0.46
	H^{2}T	**62.1%**	52.7%	74	27	**3**	640	**2402**	125	175	**71.2%**	90.3%	1.47
PETS-S2L3	*Anton *et al.* [16]	45.4%	**64.6%**	44	9	18	169	1572	38	27	51.8%	90.9%	0.70
(Detection [20])	*Berclaz *et al.* [4]	28.8%	61.8%	44	5	31	**45**	2269	**7**	**12**	30.4%	95.7%	**0.19**
(Groundtruth [15])	Anton *et al.* [2]	51.2%	54.2%	44	7	10	144	1366	82	64	58.1%	92.9%	0.60
(240 frames)	Anton *et al.* [3]	46.9%	57.8%	44	7	18	68	1589	73	57	51.3%	96.1%	0.28
(up to 42 targets)	Pirsiavash *et al.* [18]	43.0%	63.0%	44	5	18	46	1760	52	72	46.0%	**97.0%**	**0.19**
	H^{2}T	**55.3%**	53.2%	44	**12**	**9**	149	**1272**	36	40	**61.0%**	93.0%	0.62
PETS-S1L1-2	*Anton *et al.* [16]	**57.9%**	59.7%	36	**19**	11	148	**918**	21	13	**64.5%**	91.8%	0.61
(Detection [15])	*Berclaz *et al.* [4]	51.5%	64.8%	36	16	14	98	1151	**4**	8	55.5%	93.6%	0.41
(Groundtruth [15])	Anton *et al.* [2]	48.0%	64.5%	36	9	12	35	1292	17	12	50.0%	97.4%	0.15
(241 frames)	Anton *et al.* [3]	54.4%	64.3%	36	15	11	54	1102	24	17	57.4%	96.5%	0.22
(up to 20 targets)	Pirsiavash *et al.* [18]	45.4%	**66.8%**	36	9	14	**6**	1367	38	32	47.1%	**99.5%**	**0.02**
	H^{2}T	57.1%	54.8%	36	18	**8**	34	1071	**4**	**7**	58.6%	97.8%	0.14
PETS-S1L1-1													
(Detection [15])	Anton *et al.* [2]	40.0%	69.4%	46	9	20	34	**2236**	25	18	**41.5%**	97.9%	0.15
(Groundtruth [15])	Anton *et al.* [3]	37.6%	65.8%	46	9	**19**	50	2291	44	36	40.1%	96.8%	0.23
(221 frames)	Pirsiavash *et al.* [18]	32.8%	**76.5%**	46	7	22	30	2502	35	42	34.5%	97.8%	0.14
(up to 42 targets)	H^{2}T	**41.1%**	71.9%	46	**11**	**19**	**5**	2237	**11**	**10**	**41.5%**	**99.7%**	**0.02**
ParkingLot	*Zamir *et al.* [25]	**90.4%**	74.1%	14	-	-	-	-	-	-	85.3%	98.2%	-
(Detection [7])	*Shu *et al.* [19]	74.1%	79.3%	14	-	-	-	-	-	-	81.7%	91.3%	-
(Groundtruth [7])	Anton *et al.* [2]	60.0%	70.7%	14	3	1	162	756	68	97	69.3%	91.3%	0.65
(1000 frames)	Anton *et al.* [3]	73.1%	76.5%	14	**11**	**0**	253	326	83	70	86.8%	89.4%	1.01
(up to 14 targets)	Pirsiavash *et al.* [18]	65.7%	75.3%	14	1	1	**39**	754	52	60	69.4%	97.8%	**0.16**
	H^{2}T	88.4%	**81.9%**	14	**11**	**0**	**39**	**227**	**21**	**23**	**90.8%**	**98.3%**	**0.16**

obvious. In the crowded scene, e.g. PETS2009-S2L2, PETS2009-S2L3, PETS2009-S1L1-1, and PETS2009-S1L1-2, the appearance of the targets are similar with each other, and the occlusion happens frequently among the targets, which greatly challenges the robustness of the trackers. As shown in Table 1, H^{2}T outperforms other trackers in high-density sequences mainly due to the dense neighborhoods searching on tracklet relation hypergraph and the local-global hierarchical structure in optimization, which considers the relationships among multiple tracklets globally. However, our tracker obtains relative worse performance in MOTP metric, especially in the sequences PETS2009-S2L2 and PETS2009-S2L3, mainly due to the non-linear motion of the targets when it is occluded and our linear interpolation based trajectory recover mechanism makes it hard for our tracker to recover the precise target states in the occluded frames. On the other hand, other methods achieve higher MOTP, e.g. [18] and [3] always fail to identify the targets when the occlusion happens and miss the targets completely, reflected by the MT and FN metrics. The targets state in each frame of these trackers are generated by the input detection responses, which are precise enough to obtain the higher MOTP. Since the best performance are achieved in the most important metrics for the multi-target tracking task, i.e. MOTA, MT, IDs and FM, we can conclude that our H^{2}T works best.

8. Conclusion

In this paper, a hierarchical dense neighborhoods searching based multi-target tracker is proposed. The multi-target tracking is formulated as a dense neighborhoods searching problem on the multiple relation affinity graphs/hypergraphs constructed hierarchically, which considers the relationships between different tracklets across the temporal domain to restrain the IDs and Fragmentations. The appearance, motion and trajectory smoothness properties are naturally integrated in the graph affinity values. Then, the dense neighborhoods searching is solved by the pairwise updating algorithm effectively. Experimental comparison with the state-of-the-art tracking methods demonstrate the superiority of our tracker. In future work we plan to make our tracker reach real-time performance by more efficient implementation.

Figure 3. Tracking results of our tracker in sequences ParkingLot, PETS2009-S1L1-1, PETS2009-S2L2, PETS2009-S2L3, PETS2009-S1L1-2, and PETS2009-S2L1. The highlight area of PETS2009 sequences is the tracking area, which is set to be the same as [16].

Acknowledgment

This work was supported by the Chinese National Natural Science Foundation Projects 61105023, 61103156, 61105037, 61203267, 61375037, National Science and Technology Support Program Project 2013BAK02B01, Chinese Academy of Sciences Project KGZD-EW-102-2, and AuthenMetric Research and Development Funds.

References

[1] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *CVPR*, 2008.

[2] A. Andriyenko and K. Schindler. Multi-target tracking by continuous energy minimization. In *CVPR*, pages 1265–1272, 2011.

[3] A. Andriyenko, K. Schindler, and S. Roth. Discrete-continuous optimization for multi-target tracking. In *CVPR*, pages 1926–1933, 2012.

[4] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *PAMI*, 33(9):1806–1819, 2011.

[5] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP J. Image and Video Processing*, 2008, 2008.

[6] W. Brendel, M. R. Amer, and S. Todorovic. Multiobject tracking as maximum weight independent set. In *CVPR*, pages 1273–1280, 2011.

[7] Dehghan, O. Oreifej, E. Hand, and M. Shah. http://crcv.ucf.edu/data/ParkingLOT.

[8] M. Hofmann, D. Wolf, and G. Rigoll. Hypergraphs for joint multi-view reconstruction and multi-object tracking. In *CVPR*, pages 3650–3657, 2013.

[9] C. Huang, Y. Li, and R. Nevatia. Multiple target tracking by learning-based hierarchical association of detection responses. *TPAMI*, 35(4):898–910, 2013.

[10] H. Izadinia, I. Saleemi, W. Li, and M. Shah. $(MP)^2T$: Multiple people multiple parts tracker. In *ECCV*, pages 100–114, 2012.

[11] H. Jiang, S. Fels, and J. J. Little. A linear programming approach for multiple object tracking. In *CVPR*, 2007.

[12] C.-H. Kuo, C. Huang, and R. Nevatia. Multi-target tracking by online learned discriminative appearance models. In *CVPR*, pages 685–692, 2010.

[13] H. Liu, L. J. Latecki, and S. Yan. Robust clustering as ensembles of affinity relations. In *NIPS*, pages 1414–1422, 2010.

[14] H. Liu, X. Yang, L. J. Latecki, and S. Yan. Dense neighborhoods on affinity graph. *IJCV*, 98(1):65–82, 2012.

[15] A. Milan. Continuous energy minimization tracker website. http://www.cvg.rdg.ac.uk/PETS2009/a.html.

[16] A. Milan, S. Roth, and K. Schindler. Continuous energy minimization for multitarget tracking. *TPAMI*, 36(1):58–72, 2014.

[17] A. Milan, K. Schindler, and S. Roth. Challenges of ground truth evaluation of multi-target tracking. In *CVPR Workshops*, pages 735–742, 2013.

[18] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, pages 1201–1208, 2011.

[19] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah. Part-based multiple-person tracking with partial occlusion handling. In *CVPR*, pages 1815–1821, 2012.

[20] J. Yan, Z. Lei, D. Yi, and S. Z. Li. Multi-pedestrian detection in crowded scenes: A global view. In *CVPR*, pages 3124–3129, 2012.

[21] B. Yang. http://iris.usc.edu/people/yangbo/downloads.html.

[22] B. Yang and R. Nevatia. Multi-target tracking by online learning of non-linear motion patterns and robust appearance models. In *CVPR*, pages 1918–1925, 2012.

[23] B. Yang and R. Nevatia. An online learned CRF model for multi-target tracking. In *CVPR*, pages 2034–2041, 2012.

[24] T. Yang, S. Z. Li, Q. Pan, and J. Li. Real-time multiple objects tracking with occlusion handling in dynamic scenes. In *CVPR*, pages 970–975, 2005.

[25] A. R. Zamir, A. Dehghan, and M. Shah. GMCP-tracker: Global multi-object tracking using generalized minimum clique graphs. In *ECCV*, pages 343–356, 2012.

[26] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.