Exploiting Hierarchical Dense Structures on Hypergraphs for Multi-Object Tracking

Longyin Wen, Zhen Lei, Siwei Lyu, Stan Z. Li and Ming-Hsuan Yang

Abstract—Most multi-object tracking algorithms are developed within the tracking-by-detection framework that consider the pairwise appearance similarities between detection responses or tracklets within a limited temporal window, and thus less effective in handling long-term occlusions or distinguishing spatially close targets with similar appearance in crowded scenes. In this work, we propose an algorithm that formulates the multi-object tracking task as one to exploit hierarchical dense structures on an undirected hypergraph constructed based on tracklet affinity. The dense structures indicate a group of vertices that are inter-connected with a set of hyperedges with high affinity values. The appearance and motion similarities among multiple tracklets across the spatio-temporal domain are considered globally by exploiting high-order similarities rather than pairwise ones, thereby facilitating distinguish spatially close targets with similar appearance. In addition, the hierarchical design of the optimization process helps the proposed tracking algorithm handle long-term occlusions robustly. Extensive experiments on various challenging datasets of both multi-pedestrian and multi-face tracking tasks, demonstrate that the proposed algorithm performs favorably against the state-of-the-art methods.

Index Terms—Multi-object tracking, tracklet, hierarchical, undirected affinity hypergraph, dense structures.

1 Introduction

Multi-target tracking in unconstrained environments is an important and challenging problem with numerous applications including video surveillance, activity analysis, and abnormal detection, to name a few. Recent multi-object tracking algorithms have been developed within the tracking-by-detection framework where targets are usually detected by pre-trained object detectors or background subtraction methods, and matched throughout video sequences. Such approaches are attractive and effective for handling visual drifts and recovering from tracking failure. Specifically, the task of correctly matching target objects over time is known as the data association problem. Although numerous methods have been proposed to tackle the target association problem, less effort has been made to exploit high-order information (i.e., beyond pairwise relations between objects) contained among multiple objects in the temporal domain.

Most existing multi-object tracking methods exploit similarities between pairwise detection responses or tracklets for data association (e.g., MCMC data association [1], [2], [3], detection matching [4], [5], network flow [6], [7], [8], [9], k-shortest path (KSP) [10], maximum weight independent set [11], linear programming [12], tensor power iteration [13], [14], and Hungarian algorithm [15], [4], [16], [17], [18]), rather than among multiple tracklets in the temporal domain within a global view. For example, when several objects with similar appearance or motion patterns appear in close proximity as denoted by the circles in Fig. 1, identity switches are likely to occur. To alleviate this problem, a method based on minimum clique graph optimization [19] has been developed recently which considers the relationships between different detections across the temporal domain. Dehghan et. al [20] pose the data association problem as a generalized maximum multi-clique task and present an integer program algorithm for multi-target tracking. However, these two methods are less effective in handling non-linear object motion in crowded scenes when occlusions happen frequently, mainly due to only the pairwise relation-

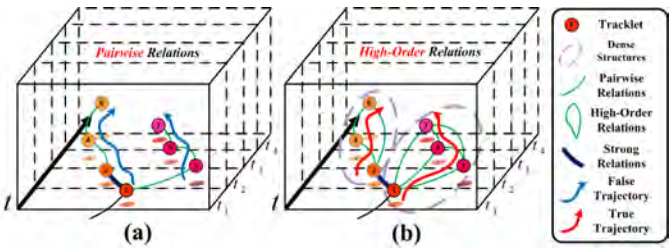

Fig. 1: (a) Existing methods often fail when multiple objects with similar appearance or motion patterns appear in proximity. (b) The proposed tracking algorithm based on an undirected affinity hypergraph effectively handles such cases. The circles denote different tracklets and the colors represent the corresponding appearance or motion patterns. Existing methods, which focus on the pairwise similarities of spatial-temporal neighboring tracklets in short and local temporal span, are likely to generate incorrect trajectories (blue splines). In contrast, the proposed algorithm searches for dense structures on the affinity hypergraph of tracklets which consider similarities among multiple tracklets across the temporal domain (i.e., high-oder information), and generates correct trajectories (red splines).

- *Longyin Wen, Zhen Lei and Stan Z. Li are with the Center for Biometrics and Security Research and National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China.*
- *Siwei Lyu is with the Computer Science Department, University at Albany, SUNY, NY.*
- *Ming-Hsuan Yang is with the school of Engineering, University of California at Merced, CA.*

ships of the targets are considered.

In this paper, an undirected hierarchical affinity hypergraph based tracker (H^2T) is proposed, which formulates the tracking task as exploiting multiple dense structures on a constructed tracklet affinity hypergraph as depicted in Fig. 1(b). Different from existing methods, the similarities (e.g., appearance or motion similarities) among tracklets across the temporal domain are considered globally by considering the high-order connections and exploiting motion constraints to distinguish spatially close target objects with similar appearance. Meanwhile, a local-to-global strategy is developed to generate target trajectories hierarchically, which significantly reduces the computational complexity in exploiting dense structures and handles large appearance variations as well as sudden motion changes effectively. The main contributions of this paper are summarized as follows:

- We propose a novel multi-object tracking algorithm by exploiting dense structures from hierarchically constructed undirected affinity hypergraphs of tracklets.
- The motion and appearance patterns are analyzed in the optimization process by considering high-order similarities among multiple tracklets globally in the constructed hypergraph.
- We use a RANSAC-style approach to convert the hypergraph to an approximate common graph which retains all the significant structures and facilitates extraction of dense structures efficiently.
- We evaluate on both multi-pedestrian and multi-face tracking tasks, and demonstrate the proposed algorithm performs favorably against the state-of-the-art methods, especially in crowded scenes.

2 Related Work

We review the most relevant multi-object tracking methods for generic objects and specifically for faces.

Multi-Object Tracking. Numerous approaches based on Kalman or particle filters and detection results for state prediction have been proposed in recent years [21], [22], [23], [24], [25], [26], [27]. These methods typically predict states effectively for short durations but do not perform well in complex scenes. Data association approaches based on the joint probabilistic data association filter (JPDAF) [28] and multiple hypotheses tracking (MHT) [29] have been developed to address these problems. The JPDAF method estimates the best assignment at each time step by considering all possible associations between targets and detections. In contrast to frame-by-frame predictions by the JPDAF method, the MHT approach evaluates the likelihoods of the hypothesized assignments over several time steps. Since the search space grows exponentially with the number of frames, both methods are less effective for handling long-term association. To alleviate this issue, Yu et al. [2] present a data driven Markov Chain Monte Carlo method to estimate target trajectories using a batch of observations. However, this sampling-based method entails proper settings of numerous parameters that restricts its application domains.

Greedy approaches have been developed for multi-object tracking [4], [5] in which detection results with similar appearance and motion patterns are matched in multiple consecutive frames. As limited temporal locality is used in matching multiple objects, these methods do not perform well in sequences with long-term occlusions, complex motions, or clutter backgrounds.

Recent algorithms consider associations of detection pairs as an optimization task based on network flows [6], [8], [30], [7], K-shortest paths (KSP) [10], maximum weight independent sets [11], tensor power iterations [13], [14], linear programs [12], and high-order motion constraints [31], [32]. Meanwhile, several methods have been developed in which short tracklets are constructed using detections in consecutive frames based on spatial-temporal proximity, and connected to generate long trajectories. The tracklet association problem has been posed as a continuous energy minimization problem [33], or a discrete-continuous optimization task [34] involving data association of the tracklets and trajectory fitting. In addition, the tracklets assignment problem has also been tackled by the hierarchical Hungarian algorithm [15], [17], [18] to generate trajectories of target objects. By exploiting global information, the aforementioned methods are more effective in dealing with partial occlusions and complex motions. However, as only associations of detection results in consecutive frames are considered, these algorithms do not perform well when multiple similar objects appear in proximity.

In [9], a directed graph is constructed to describe pairwise associations between candidate couplings, which are constructed by the detections from different cameras at the same time step. In contrast, we construct an undirected hypergraph, in which each vertex represents one tracklet, and the hyperedges are constituted by multiple tracklets across the temporal domain. The optimization processes for these two approaches are not similar since the hypergraph construction and the objectives are significantly different.

Multi-Face Tracking. Robust multi-face tracking in unconstrained surveillance scenes involves challenging factors (e.g., large pose and illumination changes) different from generic multi-object tracking, and existing approaches mainly focus on constructing robust appearance models. Kim et al. [35] introduce visual constraints using a combination of generative and discriminative models in the particle filtering framework for specific face tracking. Wang et al. [36] combine an offline trained generic face model and an online appearance model specific to a target in a dynamic Bayesian network. However, both methods do not handle drift problems well when abrupt pose variations occur. Consequently, methods based on active appearance models [37], [38] have been developed to handle pose variations.

To better handle recovery from tracking failure, Kalal et al. [39] propose an algorithm in which an offline trained detector is used to localize frontal faces and an online trained validation module is employed for matching the detected results. Cai et al. [40] present a method that integrates an offline detector, an online learned recognizer, and an online learned face tracker for person-specific face tracking. Both methods perform well when the goal is to track one specific face, but fail to handle multiple faces in surveillance scenarios.

Roth et al. [41] associate face detection responses from

two stages using the Hungarian algorithm for multi-face tracking. On the other hand, Duffner et al. [42] propose an approach to address the tracklet management problem (i.e., deciding when to add a new target or stop a tracklet) within a Bayesian filtering framework where the states are estimated by a Markov Chain Monte Carlo sampling method. More recently, an approach [43] that simultaneously clusters and associates face tracklets (based on detection responses) using a hidden Markov Random Field (MRF) model to represent the joint dependencies of cluster labels and tracklet associations. This method focuses on multi-face tracking in movie clips with relative fixed camera view angles in indoor scenes. Although the aforementioned methods perform well in constrained indoor scenes, they are not effective for unconstrained surveillance scenes which contain similar targets with frequent occlusions.

3 Algorithmic Overview

In this section, we give an overview of the proposed multi-object tracking algorithm based on a hierarchical undirected affinity hypergraph. The notations used in this paper are listed in Table 1. After detection responses are obtained in each frame, we construct an undirected affinity hypergraph where the vertices are the tracklets (i.e., we treat each detection response as a degenerated tracklet of unit length), and the hyperedges describe the high-order relationships among them. The affinity value of a hyperedge indicates the probability of the tracklets corresponding to the hyperedge associated with the same object.

The multi-object tracking problem is solved by extracting dense structures on the constructed hypergraph of degree k. As it is computationally expensive to operate directly on a large number of hyperedges, we construct a Random Consensus Hypergraph (RCH) in a way similar to [44] by sampling reliable Minimal Size Samples (MSSs) which are vertex sets of $k-1$ vertices. To further reduce computational complexity, we convert a RCH to a common graph in which the significant dense structures are retained. Consequently, the underlying dense structures in a hypergraph can efficiently be extracted on a common graph using the search algorithm for searching dense neighborhoods on an affinity graph [45]. We further process the extracted results resolve conflicting dense structures, and then connect the tracklets in each dense structure to generate target trajectories.

As it entails significant amount of run time and memory to directly process all image frames, we propose a hierarchical approach for a multi-object tracking that consists of the following main steps:

1. An image sequence is first divided into multiple non-overlapping segments where each one consists of δ_1 frames.
2. For each segment, an undirected hypergraph is constructed where the graph vertices correspond to the tracklets. Dense structures on a graph are extracted to generate longer tracklets. The tracklets in all segments are processed in this manner.
3. Furthermore, temporally δ_l apart segments are merged to generate a new segment division for the next layer.

The above step 2 and 3 are repeated until only one segment remains in the last layer (i.e., the whole image sequence).

TABLE 1: Notation

δ_l	Number of temporal adjacent segments in the l-th layer used to generate the new segment division in the $l+1$-th layer.
k	k is the degree of an edge
v_i	i-th vertex (tracklet) in the affinity hypergraph.
G	Undirected affinity hypergraph, i.e. $G=(V,E)$.
V	Vertex set of a hypergraph G, i.e. $V=\{v_1,\ldots,v_n\}$, where n is the number of vertices.
E	Hyperedge set of the hypergraph, i.e., $E \subset \overbrace{V\times\cdots\times V}^{k}$.
$\boldsymbol{e}$	k-tuple vertices involved in a hyperedge, i.e., $\boldsymbol{e}=\{v_{e_1},\ldots,v_{e_k}\}$.
$\boldsymbol{h}_i$	i-th minimal size samples involving $k-1$ vertices.

Finally, dense structures are extracted on the constructed hypergraph of the segment in the last layer to generate the final target trajectories.

4 Extracting Dense Structures

The multi-object tracking problem is solved by exploiting dense structures on an undirected affinity hypergraph. Here a dense structure indicates a group of vertices that are inter-connected by a set of hyperedges with high affinity values. The core problem for extracting dense structures of a tracklet on an undirected affinity hypergraph is to estimate the number of vertices in each dense structure. This number is treated as the hidden variable in the optimization process and estimated by maximizing the affinity value of each structure. Multiple dense structures can thus be extracted.

4.1 Problem Formulation

We denote the collection of detected tracklets in a segment as $\{\mathcal{T}_1,\cdots,\mathcal{T}_n\}$ and use $\mathbf{t}_i=\{t_1^i,\cdots,t_{r_i}^i\}$ to denote the set of all frame indices of the corresponding tracklet $\mathcal{T}_i$ [1], where r_i is the length of tracklet $\mathcal{T}_i$. In a segment, we construct a global tracklet affinity hypergraph $G=(V,E,\mathcal{A})$ to describe the relationships among multiple tracklets, where the i-th vertex $v_i \in V$ in the hypergraph corresponds to the i-th tracklet $\mathcal{T}_i$ (i.e., $v_i \sim \mathcal{T}_i$), $i=1,\cdots,n$, and E is the hyperedge set, i.e., $E=\{\boldsymbol{e}\} \subset \overbrace{V\times\cdots\times V}^{k}$, where $\boldsymbol{e}=\{v_{e_1},\cdots,v_{e_k}\}$ and k is the degree of the hyperedge $\boldsymbol{e}$. In one hypergraph, $\mathcal{A}: E\rightarrow\mathbb{R}$ is the affinity values of hyperedges in E which reflects the probability of the tracklets in $\boldsymbol{e}$ being associated with one tracked target. The constructed affinity graph is a hypergraph when $k>2$ and degenerates to a common graph when $k=2$.

In the multi-object tracking context, certain prior knowledge can be exploited for better results, e.g., removing edges between tracklets that are far apart even when they have similar appearance. Thus, we construct a vertex constraint function $\mathcal{P}$ indicating whether two vertices in G can belong to the same hyperedge to guide the construction of hyperedges in E.

1. Note that our definition of tracklets generalizes cases for single detection response (i.e., $|\mathbf{t}_i|=1$) or continuous sequence of detection responses (i.e., $\mathbf{t}_i=\{a-1,a,\cdots,b,b+1\}$ where $a<b$ are two integers).

After constructing a graph G, we exploit the dense structures to determine the target trajectories. Intuitively, if some vertices belong to a dense structure, they should be interconnected by a set of hyperedges with high affinity values. Based on this aspect, for the vertex v_p, we aim to determine its dense structure $\mathcal{N}(v_p)$, which has the maximum value based on a predefined affinity measure function $\Gamma(\cdot)$ of the vertex set:

$$\begin{aligned} &\mathcal{N}^*(v_p) = \arg\max_{\mathcal{N}(v_p)} \Gamma\big(v_p \cup \mathcal{N}(v_p)\big) \\ &\text{s.t.} \quad \mathcal{N}(v_p) \subset V, v_p \notin \mathcal{N}(v_p), |\mathcal{N}(v_p)| = \phi, \end{aligned} \tag{1}$$

where ϕ represents the number of vertices in a dense structure which can be inferred automatically. However, for multi-object tracking in crowded scenes, it requires high computational load as the number of hyperedges in G is large. For example, suppose there are 40 vertices in G, and the degree $k = 5$, there exists $\binom{40}{5} \approx 6.6 \times 10^5$ hyperedges. Nevertheless, when $k = 2$, G degenerates to a common graph in which dense structures can be extracted directly [46]. For computational efficiency, we approximate a hypergraph G ($k \geq 3$) with a common graph $G^\star$ through a RCH G' rather than traverses all the hyperedges of G in [46]. The detail algorithm will be discussed in the following sections.

4.2 Enforcing Hyperedge Constraints

In most multi-object tracking applications, the target objects are assumed to move with a terminal velocity. We set the maximal velocity of the tracked targets in a scene to α^* based on prior knowledge of object velocity, and introduce the vertex constraint function $\mathcal{P}$ to guide the construction of hyperedges in G, i.e., $\mathcal{P} : V \times V \to \{0, 1\}$ is a function indicating whether two vertices can be included in the same hyperedge of G. Clearly, if the two vertices v_i and v_j overlap in time, they cannot be associated with the same target since a target cannot occupy two different positions at a time. We set $\mathcal{P}(v_i, v_j) = 0$ if that is the case. Thus, we only consider the vertices with non-overlapping frame indices. Without loss of generality, the vertex v_i is assumed to precede v_j in the temporal domain. If the ℓ_2 distance between the last frame of v_i and the first frame of v_j is larger than the maximal distance the target object can reach with the maximal velocity α^* in the corresponding time lapse, we set $\mathcal{P}(v_i, v_j) = 0$ to indicate v_i and v_j cannot be associated with one target, and otherwise $\mathcal{P}(v_i, v_j) = 1$.

4.3 Constructing Random Consensus Hypergraph

To convert a hypergraph G to a common graph, we first approximate it with a RCH G', which preserves important dense structures. We gradually sample multiple hypothetical MSSs $\{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_i, \ldots\}$, where the i-th sampled MSS $\boldsymbol{h}_i$ is a vertex set involving $k-1$ vertices. To ensure all important structures of a hypergraph are included in a RCH, we traverse all hypergraph vertices to generate the hypothetical MSSs. For the i-th vertex v_i, we randomly select the other $k-2$ vertices $\{v_{s_1}, \ldots, v_{s_{k-2}}\}$ satisfying the hyperedge constraint $\mathcal{P}$. That is, for each s_j, $\mathcal{P}(v_i, v_{s_j}) = 1$, and we have the hypothetical MSS $\boldsymbol{h}_i = \{v_i\} \cup \{v_{s_1}, \ldots, v_{s_{k-2}}\}$. The hyperedge constraints of target objects are thus integrated in the process of generating MSSs to remove unreliable hyperedges introduced in a RCH. To further remove unreliable MSSs, we use three affinity functions to quantify the hypothetical MSS $\boldsymbol{h}$ based on appearance affinity $\mathcal{R}_a(\boldsymbol{h})$, motion affinity $\mathcal{R}_m(\boldsymbol{h})$, and smoothness affinity $\mathcal{R}_s(\boldsymbol{h})$. A hypothetical MSS $\boldsymbol{h}$ is retained if the corresponding affinity values are all above the predefined thresholds, θ_a, θ_m, and θ_s.

We compute the confidence scores of all vertices for each generated MSS to indicate the reliability of all vertices and MSSs for constructing the hyperedges. The confidence scores of $\boldsymbol{h}_i$ are defined as

$$\boldsymbol{\mathcal{C}}(\boldsymbol{h}_i) = \{\mathcal{C}_1(\boldsymbol{h}_i), \ldots, \mathcal{C}_n(\boldsymbol{h}_i)\}. \tag{2}$$

If the j-th vertex v_j is included in $\boldsymbol{h}_i$ (i.e., $v_j \in \boldsymbol{h}_i$), we set $\mathcal{C}_j(\boldsymbol{h}_i) = \mu$, where μ is the predefined confidence threshold. Otherwise, we have

$$\mathcal{C}_j(\boldsymbol{h}_i) = \omega_1 \cdot \mathcal{R}_a(\bar{\boldsymbol{v}}_{i,j}) + \omega_2 \cdot \mathcal{R}_m(\bar{\boldsymbol{v}}_{i,j}) + \omega_3 \cdot \mathcal{R}_s(\bar{\boldsymbol{v}}_{i,j}), \tag{3}$$

where $\mathcal{R}_a(\bar{\boldsymbol{v}}_{i,j})$, $\mathcal{R}_m(\bar{\boldsymbol{v}}_{i,j})$, and $\mathcal{R}_s(\bar{\boldsymbol{v}}_{i,j})$, are the appearance, motion, and trajectory smoothness affinities of the vertex set $\bar{\boldsymbol{v}}_{i,j} = \boldsymbol{h}_i \cup \{v_j\}$, and ω_1, ω_2, and ω_3 are the weight parameters such that $\sum_{i=1}^{3} \omega_i = 1$.

The sampling process of the MSSs is a progressive refinement procedure of a RCH G' to approximate a hypergraph G. For each hyperedge $\boldsymbol{e} \in E$, we define its instantaneous affinity value after sampling the i-th MSS $\boldsymbol{h}_i$,

$$\mathcal{C}_{\boldsymbol{e}}(\boldsymbol{h}_i) = \min_{j=\{1,\ldots,k\}} \mathcal{C}_{e_j}(\boldsymbol{h}_i). \tag{4}$$

If $\mathcal{C}_{\boldsymbol{e}}(\boldsymbol{h}_i)$ is larger than the current affinity value, we set the hyperedge affinity value to be $\mathcal{C}_{\boldsymbol{e}}(\boldsymbol{h}_i)$. The obtained hypergraph is called a Random Consensus Hypergraph G'. As the number of hypothetical MSSs increases, we obtain a series of better approximation of hypergraph G, $\{G'_1, \ldots, G'_m\}$, with $G' = G'_m$. For any hyperedge $\boldsymbol{e}$ in G'_i, we have

$$\mathcal{A}_{\boldsymbol{e}}^{(i)} = \max_{j=1}^{i} \mathcal{C}_{\boldsymbol{e}}(\boldsymbol{h}_j), \tag{5}$$

where $\mathcal{A}_{\boldsymbol{e}}^{(i)}$ is the current affinity value of the hyperedge $\boldsymbol{e}$ after sampling the i-th MSS $\boldsymbol{h}_i$. We have $\mathcal{A}_{\boldsymbol{e}}^{(1)} \leq \ldots \leq \mathcal{A}_{\boldsymbol{e}}^{(m)}$, where m is the number of generated hypothetical MSSs. Clearly, $\mathcal{A}_{\boldsymbol{e}}^{(i)}$ is non-negative with an upper bound. As a number of MSSs are generated, $\mathcal{A}_{\boldsymbol{e}}^{(m)}$ approximates the hyperedge affinity value $\mathcal{A}_{\boldsymbol{e}}$ of the original hypergraph G with non-decreasing affinity values. Thus, we approximate a hypergraph G using a RCH G' more accurately by sampling MSSs gradually. Similar to [44], we do not need to compute or store the affinity value of each hypergraph. Instead, we store all confident vectors $\{\boldsymbol{\mathcal{C}}(\boldsymbol{h}_1), \cdots, \boldsymbol{\mathcal{C}}(\boldsymbol{h}_m)\}$ generated by the sampled MSSs and the graph vertices, which retains all information of a RCH.

As a crucial part in constructing a RCH, three vertex set affinity functions are described with details. As tracklets that overlap in time cannot be associated with one target (i.e., one target cannot occupy two different positions at a time), we set all affinity measures to zero if that is the case. Thus, in the subsequent discussion of affinity functions, we consider only all tracklets with no overlapping frame indices.

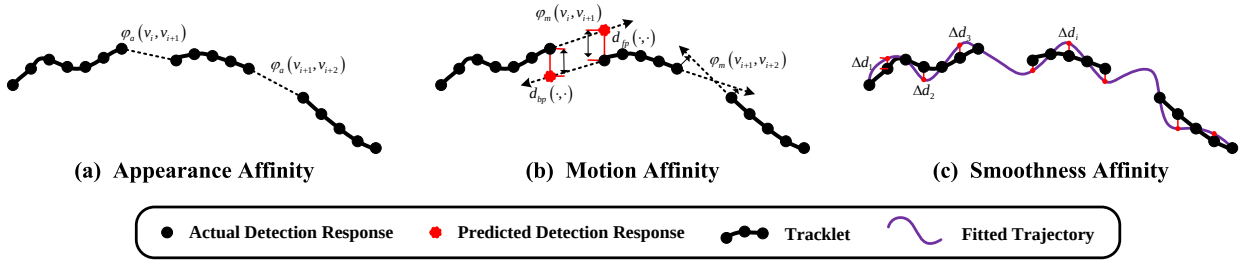

Fig. 2: (a) appearance affinity of a set of tracklets. (b) motion affinity of a set of tracklets. (c) smoothness affinity of a set of tracklets.

Appearance Affinity. We use object appearance to measure the affinity of the tracklets sorted in time $\boldsymbol{v} = \{v_1, \cdots, v_u\}$ associated to one target based on three histograms of color, gradient and local binary patterns (LBPs) [47], respectively. Specifically, we use 8 bins for each channel of the RGB space for the color histogram, and 36 dimensions for the gradient histogram. As depicted in Fig. 2(a), for a pair of tracklets v_i and v_{i+1} with v_i preceding v_{i+1}, the appearance affinity is computed based on the Bhattacharyya similarities between the color histograms $\chi_c(v_i, v_{i+1})$, gradient histograms $\chi_s(v_i, v_{i+1})$, and LBP histograms $\chi_b(v_i, v_{i+1})$ in the last frame detection of v_i and the first frame detection of v_{i+1}, that is

$$\varphi_a(v_i, v_{i+1}) = e^{\lambda_1 \chi_c(v_i, v_{i+1}) + \lambda_2 \chi_s(v_i, v_{i+1}) + \lambda_3 \chi_b(v_i, v_{i+1})}, \quad (6)$$

where λ_1, λ_2 and λ_3 are the predefined weight parameters. Thus, the appearance affinity of a set of tracklets is computed by the sum of affinity values between consecutive tracklet pairs, i.e.,

$$\mathcal{R}_a(\boldsymbol{v}) = \sum_{i=1}^{u} \varphi_a(v_i, v_{i+1}). \quad (7)$$

Motion Affinity. We use the forward-backward prediction strategy to measure the motion affinity of the tracklets sorted in time $\boldsymbol{v} = \{v_1, \cdots, v_u\}$ associated with one target. Before describing the motion affinity of a set of tracklets $\boldsymbol{v}$, we first discuss the motion affinity of a pair of non-overlapping tracklets v_i and v_{i+1} with v_i preceding v_{i+1}, which is computed based on the forward-backward prediction between the last frame detection of v_i and the first frame detection of v_{i+1}. As depicted in Fig. 2(b), the trailing velocity of v_i is first estimated by dividing the position difference of its last two frame detections with their corresponding time lapse. The predicted position for the first frame of v_{i+1} is obtained by projecting the position of the last frame detection of v_i with the estimated trailing velocity, multiplied by the time lapse between the last frame of v_i and that of the first frame of v_{i+1}. After that, the ℓ_2 distance $d_{fp}(v_i, v_{i+1})$ between the actual position of the first frame of v_{i+1} and its forward prediction from v_i is computed. Similarly, the backward prediction is computed as the ℓ_2 distance between the actual position of the last frame of v_i and its backward prediction from v_{i+1}, donated as $d_{bp}(v_i, v_{i+1})$. Thus, the motion affinity between the tracklet v_i and v_{i+1} is computed by

$$\varphi_m(v_i, v_{i+1}) = e^{-\lambda_4 \left(d_{fp}(v_i, v_{i+1}) + d_{bp}(v_i, v_{i+1}) \right)}, \quad (8)$$

where λ_4 is a weight parameter controlling the sensitivity of the affinity to the forward and backward distances. After computing the motion affinity between a pair of tracklets, we define the motion affinity of the tracklets $\boldsymbol{v}$ as

$$\mathcal{R}_m(\boldsymbol{v}) = \sum_{i=1}^{u} \varphi_m(v_i, v_{i+1}). \quad (9)$$

Smoothness Affinity. We assume the tracked target objects to have continuous and smooth motion patterns. The smoothness affinity is used to evaluate the spatio-temporal coherence of a longer trajectory formed by a set of non-overlapping short tracklets $\boldsymbol{v} = \{v_1, \cdots, v_u\}$. Specifically, we fit a piecewise second order smooth parametric trajectory with cubic spline interpolation to a subset of the detection responses sampled with equally interval of these tracklets $\boldsymbol{v}$ as shown in Fig. 2(c). Then, the ℓ_2 distance $d_{smo}(\boldsymbol{v})$ of the remaining detection responses on these tracklets $\boldsymbol{v}$ with their predictions based on the fitted smooth curve is computed. Note that smaller values of this quantity indicate more coherent of the tracklets in $\boldsymbol{v}$ associated with the same tracked target object. The smoothness affinity of these tracklets is computed from $d_{smo}(\boldsymbol{v})$ by

$$\mathcal{R}_s(\boldsymbol{v}) = e^{-\lambda_5 d_{smo}(v)}, \quad (10)$$

where λ_5 is a weight parameter controlling the sensitivity of affinity deviation to smooth trajectories.

4.4 Converting Hypergraph to Common Graph

As there usually exist a large number of hyperedges in a RCH G', we convert it to a common graph $G^\star = (V^\star, E^\star, W^\star)$ by analyzing the consensus information contained in G' to retain most significant structures effectively. The consensus information of vertices describes the mutual supporting evidence of them belonging to the same target object. In a common graph $G^\star$, $V^\star$ is the vertex set containing the same vertices as G', $E^\star = V^\star \times V^\star$ is the edge set describing the consensus information between different vertices, and $W^\star$ is the weight array of the edges. Intuitively, if the two vertices are in the same structure (i.e., they are part of a target trajectory), they are expected to appear in several hyperedges in a RCH. Thus we construct $G^\star$ by counting the number of hyperedges with large affinity values for each vertex pair according to the corresponding confidence scores $\{\boldsymbol{\mathcal{C}}(\boldsymbol{h}_1), \cdots, \boldsymbol{\mathcal{C}}(\boldsymbol{h}_m)\}$.

To remove unreliable hyperedges counted in the edge weight $W^\star$, we construct a binary neighboring graph $\Theta = (\Lambda, \mathcal{E})$ in which $\Lambda = \{v_j | \mathcal{C}_j(\boldsymbol{h}_s) \geq \mu, j = 1, \cdots, n\}$ is a

Algorithm 1 Constructing a Common Graph

Input: Confidence scores of MSSs $\{\boldsymbol{\mathcal{C}}(\boldsymbol{h}_1), \cdots, \boldsymbol{\mathcal{C}}(\boldsymbol{h}_m)\}$ and confidence threshold μ.

Set all elements in $W^\star$ to zeros.
for $s = 1$ to m **do**
  Construct a binary neighboring graph $\Theta = (\Lambda, \mathcal{E})$, where $\Lambda = \{v_j | \mathcal{C}_j(\boldsymbol{h}_s) \geq \mu, j = 1, \cdots, n\}$ and $\mathcal{E}$ is the edge set describing whether two vertices in Λ satisfying $\mathcal{P}$.
  Find all the cliques $\{\mathcal{Y}_1, \cdots, \mathcal{Y}_{\beta_s}\}$ in Θ using [48].
  for $i = 1$ to β_s **do**
    Compute $\kappa_i = \binom{|\mathcal{Y}_i| - 3}{k - 3}$.
    for each vertex pair $\{p, q\}$, $p < q$, $v_p, v_q, v_j \in \mathcal{Y}_i$, $j \neq p$, and $j \neq q$ **do**
      $W^\star(p, q) = W^\star(p, q) + \kappa_i \cdot \mathcal{C}_j(\boldsymbol{h}_s)$.
    end for
  end for
end for
$E^\star(p, q)$ is added, iff $W^\star(p, q) > 0$.
$W^\star(q, p) = W^\star(p, q)$.

Output: Common graph with edge $E^\star$ and weight $W^\star$.

vertex set with the confidence larger than μ corresponding to the MSS $\boldsymbol{h}_s$ (μ is the predefined confidence threshold) and $\mathcal{E}$ is the edge set describing whether two vertices in Λ satisfying $\mathcal{P}$. The neighboring relationship between vertices v_i and v_j is constructed if and only if $\mathcal{P}(v_i, v_j) = 1$. We partition a binary neighboring graph Θ into a few cliques $\{\mathcal{Y}_1, \ldots, \mathcal{Y}_{\beta_s}\}$ using [48], where $\mathcal{Y}_i$ is the i-th clique in Θ, and β_s is the number of cliques in Θ. Different from [44], which treats all vertices equally, we take the confidence scores between the vertices and MSSs into account. Hence, the edge weight between a vertex pair v_p and v_q in $G^\star$ is defined by

$$W^\star(p, q) = \sum_{s=1}^{m} \sum_{i=1}^{\beta_s} \sum_{\substack{v_p, v_q, v_j \in \mathcal{Y}_i \\ j \neq p, j \neq q}} \kappa_i \cdot \mathcal{C}_j(\boldsymbol{h}_s), \tag{11}$$

where $\kappa_i = \binom{|\mathcal{Y}_i| - 3}{k - 3}$ is the number of hyperedges involving vertices v_p and v_q[2]. We note that $W^\star$ is a symmetrical matrix and thus the computational load can be further reduced. Algorithm 1 shows the main steps to construct a common graph from a hypergraph.

4.5 Extracting Structures on Common Graph

As discussed above, the problem of recovering longer trajectories of target objects is formulated as searching for dense structures on a constructed common graph $G^\star$ based on [45]. Starting from each vertex v_p, we aim to find ϕ vertices in $V^\star$ with the maximum value according to a predefined affinity measure function $\Gamma^\star(v_p \bigcup \mathcal{N}(v_p))$ corresponding to a common graph $G^\star$, such that the dense structure contains $\phi + 1$ vertices (as described by (1) in Section 4.1). To avoid the degeneracy problem, we require the minimal size of the dense structure to be a fixed number, i.e., $\phi^* \leq \min_{v_p \in V^\star} |\mathcal{N}(v_p)|$. Let $U = \{v_p\} \bigcup \mathcal{N}(v_p) \subset V^\star$ be the vertex set including v_p and the ϕ exploited vertices.

2. The number of hyperedges contained in a RCH, including vertices v_p, v_q and v_j ($j \neq p$ and $j \neq q$) is a combinational problem, i.e., selecting $k - 3$ vertices from $\mathcal{Y}_i$ that exclude the vertices v_p, v_q and v_j.

We use $\boldsymbol{y} \in \mathbb{R}^n$ as the indicator vector of the subset U where $y_i = 1$ if vertex v_i is included in the dense structure, $v_i \in U$, and $y_i = 0$ otherwise. Thus, we have the following constraints:

$$\sum_{i}^{n} y_i = \phi + 1, \quad y_i \in \{0, 1\}, \quad y_p = 1. \tag{12}$$

The first two terms requires that $\phi + 1$ vertices are included in the dense structure, and the last term requires that a solution must contain vertex v_p.

The key issue of exploiting dense structures is the definition of the affinity measure function $\Gamma^\star(v_p \bigcup \mathcal{N}(v_p))$. For ease of presentation, we define E_U as the edge set corresponding to the vertex set U. Intuitively, if the vertices in U are associated to the same target, most of the edges in E_U should have large weights. Thus, we use the total weight value of the edge set E_U as the affinity measure function,

$$\widetilde{\Gamma}^\star(U) = \sum_{v_i, v_j \in U} W^\star(i, j). \tag{13}$$

However, in the multi-object tracking context, the weight of the edges in $G^\star$ are all non-negative and $\widetilde{\Gamma}^\star(U)$ usually increases as the number of vertices in U increases, which makes it hard to handle dense structures of different size[3]. Thus, we use the average weight values for the affinity measure function describing the confidence of dense structures. Since $\sum_i^n y_i = \phi + 1$, there are $(\phi + 1)^2$ summations in $\widetilde{\Gamma}^\star(U)$, we have

$$\Gamma^\star(U) = \frac{1}{(\phi + 1)^2} \widetilde{\Gamma}^\star(U) = \sum_{v_i, v_j \in U} W^\star(i, j) \frac{y_i}{\phi + 1} \frac{y_j}{\phi + 1}.$$

The problem of exploiting dense structures on a common graph is then formulated as

$$\begin{aligned} \max_{\boldsymbol{x}} g(\boldsymbol{x}) &= \sum_{v_i, v_j \in U} W^\star(i, j) \cdot x_i \cdot x_j \\ \text{s.t.} \quad & \sum_{i}^{n} x_i = 1, x_i \in \{0, \epsilon\}, x_p = \epsilon, \end{aligned} \tag{14}$$

where $x_i = \frac{y_i}{\phi + 1}$ and $\epsilon = \frac{1}{\phi + 1} \leq \frac{1}{\phi^* + 1}$. To reduce the complexity of this NP-hard problem, the conditions in (14) are relaxed to $x_i \in [0, \frac{1}{\phi^* + 1}]$, i.e., $x_i \geq 0$, and $x_i \leq \frac{1}{\phi^* + 1}$. The pairwise update algorithm [49] is used to solve (14) effectively. More details regarding the optimization process for (14) can be found in [49].

4.6 Post-processing

The dense structures directly exploited from a common graph may not obey the physical constraints, e.g., one vertex included in more than two dense structures violates the constraint that an object can only occupy one position at a time. Thus, we propose two post-processing procedures to remove the conflicts among dense structures. As described above, the average affinity value of each dense structure reflects its reliability. Thus, we first obtain the sorted dense structures $\{\psi_1, \cdots, \psi_n\}$ according to the corresponding average affinity values in descending order, where n is the

3. Large size dense structures are always preferable according to this kind of affinity measure function.

number of dense structures. Let Ψ^* be the dense structure set after post-processing. We have $\Psi^* = \emptyset$ first and add the sorted dense structures sequentially. For the i-th dense structure ψ_i, if $\psi_i \cap \psi_j^* = \emptyset, \forall j, \psi_j^* \in \Psi^*$, we add ψ_i to Ψ^*, i.e., $\Psi^* \leftarrow \Psi^* \bigcup \{\psi_i\}$. Otherwise, if $\psi_i \cap \psi_j^* \neq \emptyset$, we use the following conservative or greedy approach to process the dense structure ψ_i.

Clearly, in the first layer of the hierarchical optimization, the tracklets are short that contains relative unreliable evidence (due to limited motion and appearance information) to ensure the accuracy of dense structures. To avoid identity switches, a conservative approach is used to remove the intersecting part from ψ_i and then add it to Ψ^*, i.e., $\psi_i \leftarrow \psi_i / \psi_j^*$ and $\Psi^* \leftarrow \Psi^* \bigcup \{\psi_i\}$. On the other hand, the tracklets are much longer in the remaining layers and contain enough evidence (with much richer motion and appearance information). To reduce the fragmentation errors of tracklets, a greedy approach is designed by directly adding the cluster ψ_i to ψ_j^*, i.e., $\psi_j^* \leftarrow \psi_j^* \bigcup \psi_i$. With these procedures, the post-processed dense structure set Ψ^* is extracted. According to the post-processed dense structure set Ψ^*, the target trajectories in the segment are obtained by threading the tracklets in the same dense structure based on their temporal ordering.

5 Experiments

We evaluate the proposed algorithm against the state-of-the-art methods on both multi-pedestrian and multi-face tracking tasks. We denote the method of our prior work [46] as H^{2}T_woAcc which directly exploits dense structures on a hypergraph using the pairwise update algorithm [49], and denote the proposed RANSAC-style algorithm that exploits dense structures as H^{2}T_Acc. The source code and datasets will be make available to the public.

5.1 Datasets

Multi-Pedestrian Datasets. For multi-pedestrian tracking experiments, we use five sequences from the PETS2009 dataset [50]: S2L1, S2L2, S2L3, S1L1-1, and S1L1-2 sequences and the ParkingLot dataset [19]. These sequences can be roughly categorized as low-density (S2L1 and ParkingLot), or high-density (S2L2, S2L3, S1L1-1, and S1L1-2). The S2L1 sequence is widely used for multi-object tracking which consists of 795 frames with non-linear motion patterns, multiple similar objects appearing at proximity with frequent occlusions. The ParkingLot sequence contains 1000 frames of a few pedestrians. The S2L2 and S2L3 sequences are high-density datasets with long-term occlusions and various motion patterns, which contain 436 and 240 frames, respectively. The S1L1-1 and S1L1-2 sequences are more challenging datasets which contain a large crowd of pedestrians with 221 and 241 frames, respectively.

Multi-Face Datasets. In addition to pedestrians, we also evaluate multi-object tracking methods on the SubwayFaces dataset, which is captured from surveillance videos with ground truth annotations. The dataset consists of the S001, S002, S003, and S004 sequences with 1199, 1000, 1600, and 1001 frames, respectively. The S001, S002, and S004 sequences contain a crowd of people with frequent occlusions, whereas the S003 sequence is composed of a few fast moving people with blurry appearance.

5.2 Evaluation Metrics

We use two CLEAR Multi-Object Tracking (MOT) metrics [51] for evaluation, i.e., the Multi-Object Tracking Accuracy (MOTA) and Multi-Object Tracking Precision (MOTP) metrics. The MOTA metric integrates False Negatives (FN), False Positives (FP), and Identity Switches (IDS) to evaluate the overall performance of a tracker. On the other hand, the MOTP metric computes the total error of estimated positions for matched object-hypothesis pairs over all frames, with normalization to the hit/miss threshold value. In addition, we report the Mostly Lost (ML), the Mostly Tracked (MT), the Ground truth Trajectories (GT), the Identity Switches (IDS), and the Fragmentations of the target trajectories (FM) scores. The ML and MT metrics measure the percentage of tracked trajectories less than 20% of the time span based on the GT, and the targets successfully tracked (where objects are tracked at least 80% of the time span). The IDS metric summarizes the number of times that the matched identity of a tracked trajectory changes, while the FM measure is the number of times that trajectories are disconnected. The IDS and FM metrics reflect the accuracy of tracked trajectories. Other metrics including Recall (Rcll), Precision (Prcsn), and False Alarms per Frame (Fa/F) are also presented.

5.3 Multi-Pedestrian Tracking

It is well known in the MOT literature [52] that detection results and ground truth annotations are important for performance evaluation. For fair and comprehensive comparisons, we use the original source codes [10], [33], [8], [34] with the same detection results and ground truth annotations in each sequence for all methods. In addition, some reported results (marked by asterisk) are also listed for comparisons. Table 3 shows quantitative multi-pedestrian tracking results of the H^{2}T_woAcc and H^{2}T_Acc algorithms, as well as eight state-of-the-art trackers [8], [33], [10], [34], [19], [5], [41], [53]. Furthermore, we also report the average performance of the trackers across all image sequences. Some qualitative tracking results of the H^{2}T_Acc method are shown in Fig. 5, and more tracking results are available at www.youtube.com/watch?v=bZYbVRF7Jfw&feature=youtu.be/.

It is worth noting that the H^{2}T_woAcc and H^{2}T_Acc methods track all targets in the 2D image plane. Since most trackers track targets of the PETS2009 sequences in the 3D space, we evaluate the tracking results similar to [53]. For 3D evaluations, the hit/miss threshold of the distance between an output trajectory and the corresponding ground truth on the ground plane is set to be 1 meter. In addition, 2D evaluations are carried out on the ParkingLot sequence as the camera parameters are not known. For 2D evaluations, the hit/miss threshold of the bounding box overlap between an output trajectory and the ground truth is set to be 50%.

5.3.1 Parameter Settings

The parameters for the H^{2}T_Acc method are detailed as follows. The maximal velocity α^* of an tracked target is

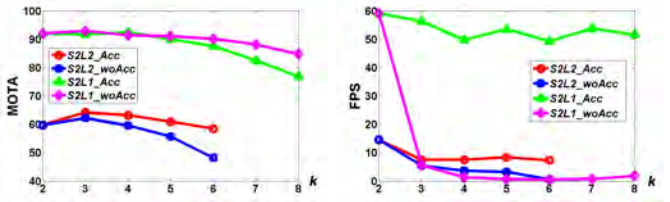

Fig. 3: Effect of the hypergraph degree k on tracking performance and running speed for the H^{2}T_Acc (Denoted by the suffix _Acc) and the H^{2}T_woAcc (Denoted by the suffix _woAcc) methods. The left and right figures show the tracking performance in terms of accuracy (measured by MOTA) and speed (measured by frame-per-second (fps)) by changing the hypergraph degree k.

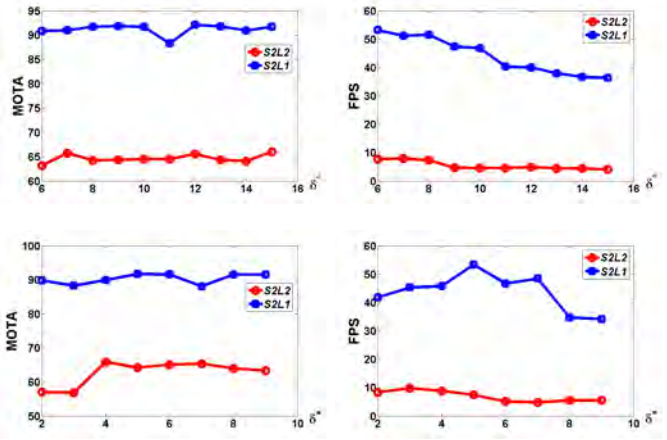

Fig. 4: Effect of the number of temporal adjacent segments used to generate the new segment division for both the first layer and the remaining ones. The first and second columns present the tracking performance (measured by MOTA) and speed (measured by frame-per-second (fps)) by changing δ° and δ^*.

empirically set to be 50 pixels per frame. The affinity thresholds θ_a, θ_m, and θ_s for MSS generation are 0.5, 0.35, and 0.2, respectively. The weight parameters for computing the confidence between the vertices and MSSs in (3) are $\omega_1 = 0.6$, $\omega_2 = 0.2$, and $\omega_3 = 0.2$. The number of generated MSSs m is set to be 60 for each vertex, and the preset confidence threshold μ indicating whether the vertex belongs to the model is set to be 0.1. The values for parameters $\lambda_1, \cdots, \lambda_5$ for computing affinity based on appearance, motion and smoothness affinities are: $\lambda_1 = 0.7$, $\lambda_2 = 0.1$, $\lambda_3 = 0.2$ in (7); $\lambda_4 = 0.02$ in (9); $\lambda_5 = 0.01$ in (10). The minimal size of the dense structure parameter ϕ^* is 3.

The hypergraph degree k and the number of temporal adjacent segments δ_l are two critical parameters for the H^{2}T_Acc method. To examine the effect of the hypergraph degree k, we change the value of k from 2 to 8 while keeping other parameters fixed. Since only the common graph is used to describe the pairwise similarities between the tracklets for k is 2, the H^{2}T_woAcc method and the H^{2}T_Acc method achieve the same tracking performance. We present the results of both the H^{2}T_woAcc and H^{2}T_Acc methods for $k = 2, \ldots, 6$ on the S2L2 sequence, due to high computational cost of the H^{2}T_woAcc method when k is larger than 6. As depicted in Fig. 3, the performance for both the H^{2}T_woAcc and H^{2}T_Acc methods decrease as k increases when k is larger than 5. Taking both accuracy and speed into account, we use the hypergraph of degree 3 in our algorithm for multi-pedestrian tracking.

For the number of temporal adjacent segments δ_l, we set δ_l to be δ^* for all layers $l > 1$, and δ_1 to be δ° in all the experiments. To show the effect of δ_l, we carry out two experiments: 1) change δ° from 6 to 15 while keeping other parameters fixed; 2) change δ^* from 2 to 8 while keeping other parameters fixed. The results of the first experiment, presented in the first column of Fig. 4, show that δ° has limited effect on the performance, and the speed is decreased as the segment size increases in the first layer. We set δ° to be 8 in the following experiments. The results of the second experiment, presented by the curves in the second column of Fig. 4, demonstrate that δ^* has little effect on both accuracy and speed of the H^{2}T_Acc method. In the following evaluations, we set δ^* to be 4.

5.3.2 Discussion

We first demonstrate the contributions of different components of the proposed algorithm in Table 2 using the CLEAR MOT metrics.

Effectiveness of Dense Structures. We construct two trackers to demonstrate the effect of dense structures exploited in the proposed algorithm. One is the undirected Hierarchical affinity Graph based Tracker (HGT) which considers pairwise similarities between tracklets, and the other one is based on the widely used Hungarian algorithm. Since similarities between different tracklets of both methods are the same as the H^{2}T_Acc method, we evaluate these two algorithms to demonstrate the importance of exploiting dense structures for MOT.

Table 2 shows that the HGT algorithm outperforms the Hungarian algorithm. The Hungarian algorithm uses the similarities between pairwise tracklets greedily instead of considering multiple similarities of pairwise tracklets jointly used in the HGT algorithm. Consequently, the Hungarian algorithm generates a considerable number of incorrect associations as shown by the high IDS and FM scores. The IDS, FM, and MOTA scores on average performance of both methods show that exploiting dense structures on the affinity graph significantly helps improve tracking performance.

Effectiveness of Hypergraph. To demonstrate the importance of hypergraph in the proposed algorithm, we compare the H^{2}T_woAcc and HGT methods. The HGT algorithm merely considers pairwise similarities between tracklets instead of the high-order similarities among multiple tracklets in a hypergraph of the H^{2}T_woAcc method.

As shown in Table 2, the H^{2}T_woAcc algorithm performs well with higher MOTA scores in three sequences, and better results on IDS and FM metrics in all sequences than the HGT method. These results can be attributed to the use of hypergraphs in which high-order similarities among multiple tracklets instead of simple pairwise ones

TABLE 2: Effect of different components in the proposed method. The symbol ↑ means higher scores indicate better performance while ↓ means lower scores indicate better performance. The red and blue colors indicate the best and 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	Hungarian	51.0%	73.5%	23	**95.7%**	**0.0%**	1502	287	149	66	92.7%	70.9%	1.89
Detection [54]	HGT	92.0%	**82.7%**	23	**95.7%**	**0.0%**	**31**	267	20	19	93.3%	**99.2%**	**0.04**
Groundtruth [50]	H^{2}T_woAcc	**92.7%**	72.9%	23	**95.7%**	**0.0%**	62	**222**	**5**	**10**	**94.4%**	98.4%	0.08
PETS-S1L1-1	Hungarian	34.8%	63.8%	46	**34.8%**	**23.9%**	586	**1656**	249	137	**56.7%**	78.7%	2.65
Detection [50]	HGT	**43.8%**	**75.2%**	46	**34.8%**	45.7%	59	2063	25	31	46.0%	96.8%	0.27
Groundtruth [50]	H^{2}T_woAcc	41.1%	71.9%	46	23.9%	41.3%	**5**	2237	**11**	**10**	41.5%	**99.7%**	**0.02**
PETS-S2L2	Hungarian	43.3%	**61.8%**	74	**43.2%**	**2.7%**	2173	**1909**	653	428	**77.1%**	74.8%	4.98
Detection [55]	HGT	59.7%	56.8%	74	29.7%	5.4%	**551**	2725	**91**	206	67.4%	**91.1%**	**1.26**
Groundtruth [50]	H^{2}T_woAcc	**62.1%**	52.7%	74	36.5%	4.1%	640	2402	125	**175**	71.2%	90.3%	1.47
PETS-S2L3	Hungarian	32.9%	**64.1%**	44	20.5%	40.9%	406	1613	169	119	50.5%	80.2%	1.69
Detection [55]	HGT	53.2%	55.8%	44	**31.8%**	**20.5%**	261	**1210**	56	63	**62.9%**	88.7%	1.09
Groundtruth [50]	H^{2}T_woAcc	**55.3%**	53.2%	44	27.3%	**20.5%**	**149**	1272	**36**	**40**	61.0%	**93.0%**	**0.62**
Average	Hungarian	40.5%	65.8%	-	**48.6%**	16.9%	1166.8	**1366.3**	305.0	187.5	**69.3%**	76.2%	2.80
Performance	HGT	62.2%	**67.6%**	-	48.0%	17.9%	225.5	1566.3	48.0	79.8	67.4%	94.0%	0.67
	H^{2}T_woAcc	**62.8%**	62.7%	-	45.9%	**16.5%**	**214.0**	1533.3	**44.3**	**58.8**	67.0%	**95.4%**	**0.55**

are used such that full motion information is exploited in the presence of appearance ambiguities, thereby reducing IDS by 8% and FM by 26% on average. Although we use an approximate algorithm to convert the hypergraph to a common graph, such that the dense subgraph can be searched efficiently, the approximate common graph retains the high-order information to ensure tracking performance.

5.3.3 Quantitative Evaluation

As presented in Table 3, the H^{2}T_woAcc and H^{2}T_Acc methods perform favorably against the state-of-the-art trackers [10], [33], [34], [8] with more than 8% and 9% gain in MOTA and MT scores, while on average reducing more than 7% trajectories that are not correctly matched (ML score) by other methods.

Low-Density Sequences. The S2L1 sequence is one of the most widely used videos in multi-pedestrian tracking which contains non-linear motion, targets in close proximity with similar appearance and frequent occlusions. As shown in Table 3, the H^{2}T_woAcc and H^{2}T_Acc methods perform well against the state-of-the-art trackers based on the MOTA metric. The ParkingLot sequence which contains 14 pedestrians with frequent occlusions and similar appearance. Table 3 shows that the proposed H^{2}T_woAcc and H^{2}T_Acc algorithms outperform the state-of-the-art trackers [8], [10], [33], [34] in nearly all metrics.

Table 3 shows that in the two low-density sequences the H^{2}T_Acc method outperforms other trackers with high MOTA and MT scores as well as low IDS and FM scores. When only local similarities of detection results are considered, it is difficult for other methods [8], [10], [33], [34] to robustly track multiple objects, especially when two similar targets appear in close proximity. Note that the H^{2}T_Acc algorithm performs well by considering similarities among multiple tracklets across the temporal domains in a global view, thereby achieving lower IDS and FM than other methods.

Compared to the H^{2}T_woAcc tracker, the H^{2}T_Acc method achieves comparable performance in most metrics (i.e., MOTA, MT and FM). Fig. 3 also shows that the H^{2}T_Acc method performs well against the H^{2}T_woAcc tracker when the degree of hypergraph changes in both S2L1 and S2L2 sequences with much faster execution speed. This can be explained by hypothesize-and-test approaches which construct a common graph that approximates a hypergraph to efficiently exploit dense structures. Specifically, as shown in Fig. 3, we note that both the H^{2}T_woAcc and H^{2}T_Acc methods perform well for the hypergraph with the degree between 3 and 5. That is, the approaches using excessive high-degree hypergraphs to exploit dense structures are not effective in dealing with scenes where objects move in drastically different direction and speed.

High-Density Sequences. The S2L2 sequence contains 74 pedestrian with different motions and frequent occlusions, while the S2L3 sequence contains up to 44 pedestrians with frequent occlusions and illumination changes. The proposed H^{2}T_woAcc and H^{2}T_Acc algorithms perform well in both sequences in terms of MOTA, ML, FN and Rcll metrics.

The S1L1-1 and S1L1-2 sequences are two dense sequences containing target objects with linear motion patterns. Overall, the H^{2}T_Acc method performs well with high MOTA, MT, FN, and Rcll scores and low ML scores. The results also demonstrate that the H^{2}T_Acc method performs more effectively on the high-density sequences. As presented in Table 3, the H^{2}T_Acc method outperforms the state-of-the-art trackers [8], [10], [33], [34] in the high-density sequences with the highest MOTA scores, mainly by exploiting the hierarchical dense structures which consider similarities among multiple tracklets globally.

Although the H^{2}T_Acc algorithm performs better than the H^{2}T_woAcc method in MOTA, it has higher IDS and FM scores. This can be explained by two factors. First, in the crowded scenes, the hypothesize-and-test process in the H^{2}T_Acc method facilitates exploiting dense structures in the presence of noise. Second, some true positives for each dense structure are inevitably removed in the sampling process, thereby resulting in higher IDS and FM scores.

The H^{2}T_Acc and H^{2}T_woAcc methods perform worse in terms of MOTP for the crowded scenes (e.g., S2L2, and S2L3) containing non-linear motion patterns. As the linear interpolation based trajectory recover mechanism is used in the proposed methods, it is unlikely to handle scenes

TABLE 3: Quantitative comparison results of the proposed trackers with other state-of-the-art trackers in the multi-pedestrian tracking sequences (results marked with the asterisk are taken directly from the literature). The symbol ↑ means higher scores indicate better performance while ↓ means lower scores indicate better performance. The red and blue colors 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 Detection [54] Groundtruth [50] 795 frames up to 8 targets	*Milan et al. [53]	90.6%	**80.2%**	23	91.3%	4.4%	59	302	11	**6**	92.4%	98.4%	0.07
	*Berclaz et al. [10]	80.3%	72.0%	23	73.9%	8.7%	126	641	13	22	83.8%	96.3%	0.16
	Andriyenko et al. [33]	86.3%	78.7%	23	78.3%	4.4%	88	417	38	21	89.5%	97.6%	0.11
	Andriyenko et al. [34]	88.3%	79.6%	23	82.6%	**0.0%**	47	396	18	14	90.0%	98.7%	0.06
	Pirsiavash et al. [8]	77.4%	74.3%	23	60.9%	4.4%	93	742	57	62	81.2%	97.2%	0.12
	H²T_woAcc	**92.7%**	72.9%	23	**95.7%**	**0.0%**	62	**222**	**5**	10	**94.4%**	98.4%	0.08
	H²T_Acc	91.7%	78.4%	23	**95.7%**	**0.0%**	**41**	266	16	20	93.3%	**98.7%**	**0.06**
PETS-S2L2 Detection [55] Groundtruth [50] 436 frames up to 33 targets	*Milan et al. [53]	56.9%	59.4%	74	37.8%	16.2%	622	2881	99	73	65.5%	89.8%	1.43
	*Berclaz et al. [10]	24.2%	60.9%	74	9.5%	54.1%	**193**	6117	**22**	**38**	26.8%	92.1%	**0.44**
	Andriyenko et al. [33]	48.5%	62.0%	74	20.3%	18.9%	301	3850	152	128	53.9%	93.7%	0.69
	Andriyenko et al. [34]	48.0%	61.6%	74	20.3%	14.9%	245	3957	143	125	52.6%	94.7%	0.56
	Pirsiavash et al. [8]	45.0%	**64.1%**	74	9.5%	23.0%	199	4257	137	216	49.0%	**95.4%**	0.46
	H²T_woAcc	62.1%	52.7%	74	36.5%	4.1%	640	2402	125	175	71.2%	90.3%	1.47
	H²T_Acc	**64.2%**	57.3%	74	**40.5%**	**2.7%**	708	**2141**	136	235	**74.4%**	89.8%	1.62
PETS-S2L3 Detection [55] Groundtruth [50] 240 frames up to 42 targets	*Milan et al. [53]	45.4%	**64.6%**	44	20.5%	40.9%	169	1572	38	27	51.8%	90.9%	0.70
	*Berclaz et al. [10]	28.8%	61.8%	44	11.4%	70.5%	**45**	2269	**7**	**12**	30.4%	95.7%	**0.19**
	Andriyenko et al. [33]	51.2%	54.2%	44	15.9%	22.7%	144	1366	82	64	58.1%	92.9%	0.60
	Andriyenko et al. [34]	46.9%	57.8%	44	15.9%	40.9%	68	1589	73	57	51.3%	96.1%	0.28
	Pirsiavash et al. [8]	43.0%	63.0%	44	11.4%	40.9%	46	1760	52	72	46.0%	**97.0%**	**0.19**
	H²T_woAcc	55.3%	53.2%	44	27.3%	**20.5%**	149	1272	36	40	61.0%	93.0%	0.62
	H²T_Acc	**59.9%**	57.0%	44	**34.1%**	25.0%	91	**1180**	37	50	**63.8%**	95.8%	0.38
PETS-S1L1-1 Detection [50] Groundtruth [50] 221 frames up to 42 targets	Berclaz et al. [10]	41.8%	67.0%	46	**32.6%**	**41.3%**	80	2109	35	53	44.8%	95.5%	0.36
	Andriyenko et al. [33]	40.0%	69.4%	46	19.6%	43.5%	34	2236	25	18	41.5%	97.9%	0.15
	Andriyenko et al. [34]	37.6%	65.8%	46	19.6%	**41.3%**	50	2291	44	36	40.1%	96.8%	0.23
	Pirsiavash et al. [8]	32.8%	**76.5%**	46	15.2%	47.8%	30	2502	35	42	34.5%	97.8%	0.14
	H²T_woAcc	41.1%	71.9%	46	23.9%	**41.3%**	**5**	2237	**11**	**10**	41.5%	**99.7%**	**0.02**
	H²T_Acc	**44.9%**	70.5%	46	**32.6%**	**41.3%**	59	**2016**	31	30	**47.3%**	96.8%	0.27
PETS-S1L1-2 Detection [50] Groundtruth [50] 241 frames up to 20 targets	*Milan et al. [53]	**57.9%**	59.7%	36	**52.8%**	30.6%	148	**918**	21	13	**64.5%**	91.8%	0.61
	*Berclaz et al. [10]	51.5%	64.8%	36	44.4%	38.9%	98	1151	**4**	**8**	55.5%	93.6%	0.41
	Andriyenko et al. [33]	48.0%	64.5%	36	25.0%	33.3%	35	1292	17	12	50.0%	97.4%	0.15
	Andriyenko et al. [34]	54.4%	64.3%	36	41.7%	30.6%	54	1102	24	17	57.4%	96.5%	0.22
	Pirsiavash et al. [8]	45.4%	**66.8%**	36	25.0%	38.9%	**6**	1367	38	32	47.1%	**99.5%**	**0.02**
	H²T_woAcc	57.1%	54.8%	36	50.0%	22.2%	34	1071	**4**	**7**	58.6%	97.8%	0.14
	H²T_Acc	56.6%	56.3%	36	44.4%	**19.4%**	72	1023	28	26	60.4%	95.6%	0.30
ParkingLot Detection [56] Groundtruth [56] 1000 frames up to 14 targets	*Zamir et al. [19]	**90.4%**	74.1%	14	-	-	-	-	-	-	85.3%	98.2%	-
	*Shu et al. [5]	74.1%	79.3%	14	-	-	-	-	-	-	81.7%	91.3%	-
	*Izadinia et al. [7]	88.9%	77.5%	14	-	-	-	-	-	-	**96.5%**	93.6%	-
	Berclaz et al. [10]	46.1%	76.2%	14	28.6%	**0.0%**	105	1052	172	166	57.3%	93.1%	0.42
	Andriyenko et al. [33]	60.0%	70.7%	14	21.4%	7.1%	162	756	68	97	69.3%	91.3%	0.65
	Andriyenko et al. [34]	73.1%	76.5%	14	**78.6%**	**0.0%**	253	326	83	70	86.8%	89.4%	1.01
	Pirsiavash et al. [8]	65.7%	75.3%	14	7.1%	7.1%	**39**	754	52	60	69.4%	97.8%	**0.16**
	H²T_woAcc	88.4%	**81.9%**	14	**78.6%**	**0.0%**	**39**	**227**	**21**	**23**	90.8%	**98.3%**	**0.16**
	H²T_Acc	79.8%	69.7%	14	**78.6%**	**0.0%**	196	279	23	81	88.7%	91.8%	0.78
Average Performance	Berclaz et al. [10]	45.5%	67.1%	-	33.4%	35.6%	107.8	2223.2	42.2	49.8	49.8%	94.4%	0.33
	Andriyenko et al. [33]	55.7%	66.6%	-	30.1%	21.7%	127.3	1652.8	63.7	56.7	60.4%	95.1%	0.39
	Andriyenko et al. [34]	58.1%	67.6%	-	43.1%	21.3%	119.5	1610.2	64.2	53.2	63.0%	95.4%	0.39
	Pirsiavash et al. [8]	51.6%	**70.0%**	-	21.5%	27.0%	**68.8**	1897.0	61.8	80.7	54.5%	**97.5%**	**0.18**
	H²T_woAcc	66.1%	64.6%	-	52.0%	**14.7%**	154.8	1238.5	**33.7**	**44.2**	69.6%	96.3%	0.42
	H²T_Acc	**66.2%**	64.9%	-	**54.3%**	**14.7%**	194.5	**1150.8**	45.2	73.7	**71.3%**	94.8%	0.57

with non-linear target motions precisely. On the other hand, methods that perform well with higher MOTP [8], [34] fail to re-identify the targets when the occlusions occur and miss the targets completely, as shown by the FN scores.

5.4 Multi-Face Tracking

We evaluate the proposed algorithms on four challenging multi-face tracking sequences, which are collected by ourselves in the crowded subway scenes. For fair comparisons, we use publicly available source codes [10], [33], [8], [34] with the same detection results and annotated ground truth as our methods. Both the H²T_woAcc and H²T_Acc methods track multi-face in the 2D image plane. Similar to multi-pedestrian tracking, the hit/miss threshold is set to be 50%. Quantitative evaluation results using different metrics are shown in Table 4, and some tracking screenshots are presented in Fig. 5.

For multi-face tracking, we consider both face and upper torso regions for object representations. We include $\zeta\%$ of a whole face (20% in all our experiments), and extract color, gradient, and LBP histograms for representation. Most parameters for multi-face tracking are set the same as the multi-pedestrian tracking, as described in Section 5.3.1, except the number of temporal adjacent segments used to generate the new segment division is set as $\delta_1 = \delta^\circ = 10$, $\delta_l = \delta^* = 5$, $l > 1$, and the maximal velocity of the tracked objects in the scene α^* is set to be 60 pixels per frame.

TABLE 4: Quantitative results of the evaluated MOT algorithms in the subway surveillance sequences. The symbol ↑ denotes higher scores indicate better performance while ↓ means lower scores indicate better performance. The red and blue colors indicate the best and second best performing method using one metric.

Sequence	Method	MOTA ↑	MOTP ↑	GT	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓	FM ↓	Rcll ↑	Prcsn ↑	Fa/F ↓
SubwayFaces-S001 Detection [57] 1199 frames	Berclaz et al. [10]	37.6%	**76.6%**	29	20.7%	27.6%	628	2724	**22**	40	49.7%	81.1%	0.52
	Andriyenko et al. [33]	23.0%	73.7%	29	27.6%	13.8%	1633	2499	35	57	53.8%	64.1%	1.36
	Andriyenko et al. [34]	40.9%	75.4%	29	**34.5%**	**6.9%**	1190	**1967**	39	36	**63.6%**	74.3%	0.99
	Pirsiavash et al. [8]	46.8%	75.6%	29	31.0%	10.3%	709	2139	33	44	60.5%	82.2%	0.59
	H^{2}T_woAcc	**52.0%**	75.9%	29	31.0%	10.3%	**335**	2224	39	42	58.9%	**90.5%**	**0.28**
	H^{2}T_Acc	50.8%	75.2%	29	31.0%	**6.9%**	675	**1967**	**22**	**29**	**63.6%**	83.6%	0.56
SubwayFaces-S002 Detection [57] 1000 frames	Berclaz et al. [10]	45.8%	75.5%	28	28.6%	10.7%	498	2670	64	73	55.2%	86.9%	0.50
	Andriyenko et al. [33]	23.2%	72.9%	28	28.6%	35.7%	1126	3406	45	55	42.9%	69.4%	1.13
	Andriyenko et al. [34]	46.9%	75.5%	28	**35.7%**	**7.1%**	852	2258	55	50	62.1%	81.3%	0.85
	Pirsiavash et al. [8]	51.5%	**75.7%**	28	**35.7%**	10.7%	431	2424	**36**	**43**	59.3%	89.1%	0.43
	H^{2}T_woAcc	51.2%	75.5%	28	32.1%	10.7%	**396**	2465	47	50	58.7%	**89.8%**	**0.40**
	H^{2}T_Acc	**53.3%**	75.1%	28	**35.7%**	**7.1%**	560	**2182**	41	55	**63.4%**	87.1%	0.56
SubwayFaces-S003 Detection [57] 1600 frames	Berclaz et al. [10]	18.1%	67.9%	24	4.2%	62.5%	**223**	3065	**7**	55	23.8%	81.1%	**0.14**
	Andriyenko et al. [33]	20.1%	**68.8%**	24	**16.7%**	51.2%	379	2818	18	**22**	30.0%	76.1%	0.24
	Andriyenko et al. [34]	26.0%	67.7%	24	8.3%	29.2%	537	2402	40	45	40.3%	75.1%	0.34
	Pirsiavash et al. [8]	26.7%	67.7%	24	8.3%	29.2%	312	2566	69	92	36.2%	82.4%	0.20
	H^{2}T_woAcc	26.7%	68.5%	24	8.3%	29.2%	259	2648	41	47	34.2%	**84.1%**	0.16
	H^{2}T_Acc	**31.8%**	68.1%	24	12.5%	**25.0%**	355	**2358**	31	60	**41.1%**	82.4%	0.22
SubwayFaces-S004 Detection [57] 1001 frames	Berclaz et al. [10]	29.6%	**76.0%**	43	7.0%	27.9%	1245	5898	187	161	43.3%	78.4%	1.24
	Andriyenko et al. [33]	9.3%	70.0%	43	2.3%	48.8%	1603	7658	181	267	26.4%	63.1%	1.60
	Andriyenko et al. [34]	36.4%	76.0%	43	23.3%	**7.0%**	2361	4137	131	106	60.3%	72.7%	2.36
	Pirsiavash et al. [8]	45.4%	75.9%	43	23.3%	**7.0%**	1264	4304	113	117	58.6%	82.8%	1.26
	H^{2}T_woAcc	44.5%	75.7%	43	23.3%	11.6%	**1227**	4448	104	103	57.3%	**82.9%**	**1.23**
	H^{2}T_Acc	**47.3%**	75.2%	43	**30.2%**	**7.0%**	1379	**4029**	**78**	**85**	**61.3%**	82.2%	1.38
Average Performance	Berclaz et al. [10]	32.8%	**74.0%**	-	15.1%	32.2%	648.5	3589.3	70.0	82.3	43.0%	81.9%	0.60
	Andriyenko et al. [33]	18.9%	71.4%	-	18.8%	37.4%	1185.3	4095.3	69.8	100.3	38.3%	68.2%	1.08
	Andriyenko et al. [34]	37.6%	73.7%	-	25.5%	12.6%	1235.0	2691.0	66.3	59.3	56.6%	75.9%	1.14
	Pirsiavash et al. [8]	42.6%	73.7%	-	24.6%	14.3%	679.0	2858.3	62.8	74.0	53.7%	84.1%	0.62
	H^{2}T_woAcc	43.6%	73.9%	-	23.7%	15.5%	**554.3**	2946.3	57.8	60.5	52.3%	**86.8%**	**0.52**
	H^{2}T_Acc	**45.8%**	73.4%	-	**27.4%**	**11.5%**	742.3	**2634.0**	**43.0**	**57.3**	**57.4%**	83.8%	0.68

5.4.1 *Quantitative Evaluation*

Compared to the state-of-the-art trackers [10], [33], [8], [34], Table 4 shows the proposed H^{2}T_woAcc and H^{2}T_Acc algorithms achieve more than 3.2% and 1.9% improvements on MOTA and MT metrics, while reduce more than 24% IDS of average performance.

Low-Density Sequence. The S003 sequence contains a few faces in the scenes. Due to large illumination variations and motion blurs, some faces are not detected in all the frames, which in turn affects the performance of the tracking methods. In contrast to the state-of-the-art trackers [8], [10], [33], [34], the H^{2}T_Acc method performs well since it considers similarities among multiple tracklets across the temporal domain and associates the tracklets with long-term temporal interval. Meanwhile, the H^{2}T_Acc algorithm performs better than the H^{2}T_woAcc method with higher MOTA, MT and Rcll, and lower IDS and ML scores, mainly due to the RANSAC-style optimization process for exploiting dense structure that helps distinguishes noisy observations.

High-Density Sequences. The S001, S002, and S004 sequences contain multiple faces with fast motion and frequently occlusions in the unconstrained scenes. The H^{2}T_Acc method performs well against the state-of-the-art trackers [8], [10], [33], [34], which can be explained by high-order similarities among multiple tracklets for distinguishing similar targets in close proximity. In contrast to multi-pedestrian tracking, the H^{2}T_Acc algorithm outperforms the H^{2}T_woAcc method with higher MOTA, MT, ML, IDS, FM, FN, Rcll scores in most of the sequences. Overall, the H^{2}T_Acc method is more effective for handling multi-object tracking in that scenes (e.g., the crowded scenes) where the targets are similar but with different motion patterns.

TABLE 5: Run time performance of evaluated methods. Frame-per-second (fps) is used to measure the speed of the tracker. The red and blue colors indicate the top performing and second methods in each sequence.

Method (fps)	[10]	[8]	[33]	[34]	H^{2}T_woAcc	H^{2}T_Acc
PETS-S2L1	18.15	**802.8**	10.00	5.880	5.510	56.31
PETS-S2L2	7.730	**517.9**	1.960	0.490	5.430	7.500
PETS-S2L3	19.93	**877.4**	1.450	2.470	6.400	10.58
PETS-S1L1-1	23.67	**1692**	2.700	2.650	10.11	16.82
PETS-S1L1-2	20.62	**1803**	2.940	3.390	11.57	22.02
ParkingLot	88.93	**1426**	3.030	20.96	13.43	123.5
Average Speed	29.84	**1187**	3.680	5.973	8.742	39.46
SubwayFaces-S001	129.3	**3218**	5.560	12.39	45.43	257.4
SubwayFaces-S002	104.8	**2678**	4.350	12.39	45.46	191.0
SubwayFaces-S003	154.3	**9374**	33.33	53.30	215.7	1086
SubwayFaces-S004	69.47	**1474**	1.100	6.94	17.74	151.2
Average Speed	114.5	**4186**	11.09	21.26	81.08	421.4

5.5 Run Time Performance

We implement the H^{2}T_woAcc and H^{2}T_Acc methods in C++ without any code optimization. We run all the evaluated tracking methods 5 times in a single thread for all the sequences of both the multi-pedestrian and multi-face tracking on a laptop with a 3.2 GHz Intel processor and 16 GB memory. Given the detection responses, we present the average execution speed for all these trackers in Table 5.

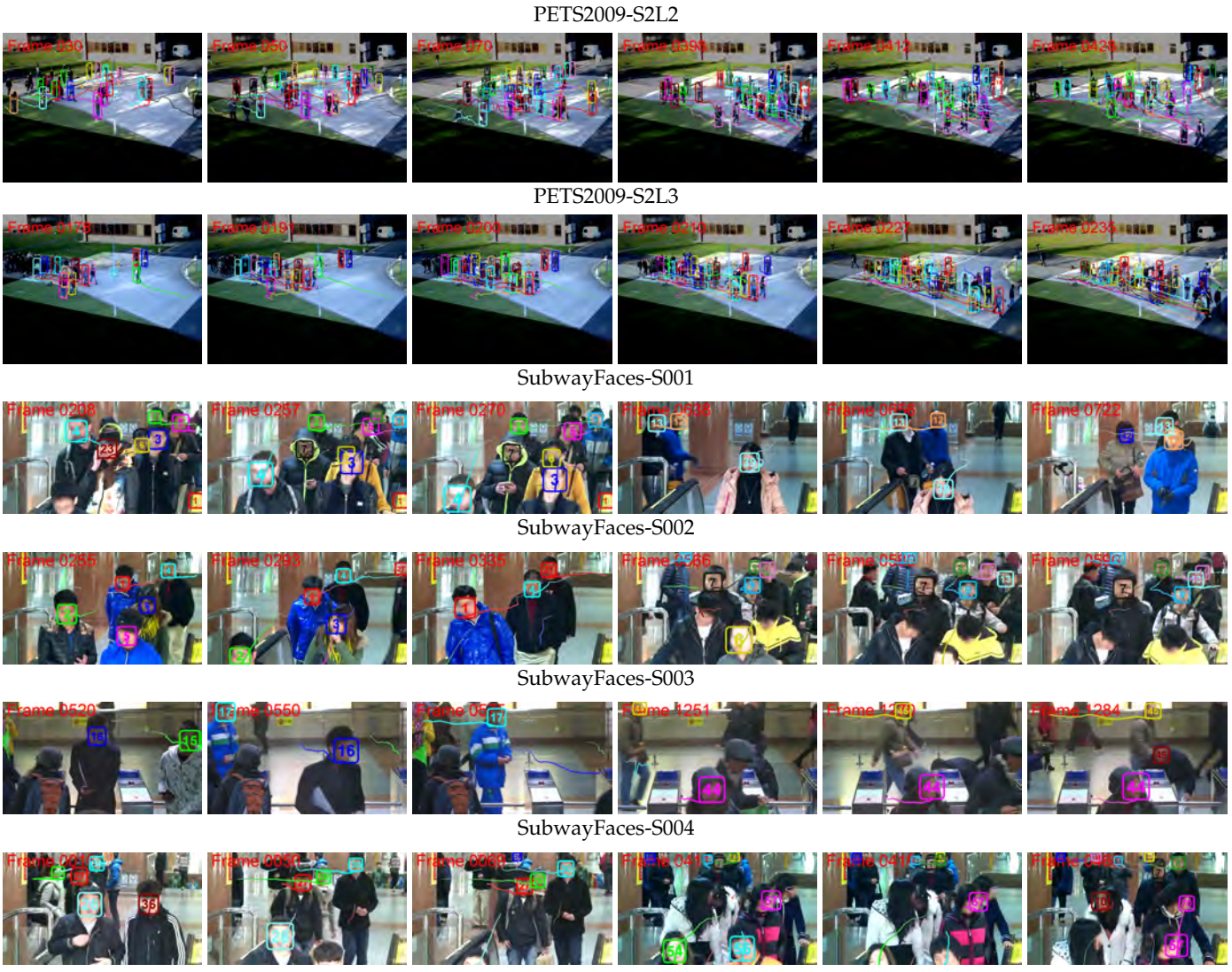

Fig. 5: Tracking results of the proposed tracking algorithm in multi-pedestrian tracking sequences (PETS2009-S2L2 and PETS2009-S2L3) and multi-face sequences (SubwayFaces-S001 SubwayFaces-S002, SubwayFaces-S003 and SubwayFaces-S004). The highlighted area in the PETS2009 sequences is the tracking region which is set as [53].

Frame-per-second (fps) is used to measure the speed of the tracker. Compared with other state-of-the-art methods [8], [10], [33], [34], the proposed H^{2}T_Acc algorithm performs well in run time for both multi-pedestrian and multi-face tracking sequences. Meanwhile, the H^{2}T_Acc method generates better tracking results against [8], [10], [33], and [34] with 14.6%, 20.7%, 10.5% and 8.1% higher MOTA scores in multi-pedestrian tracking (Table 3) and 3.2%, 13.0%, 26.9% and 8.2% higher MOTA scores in multi-face tracking (Table 4). In addition, the H^{2}T_Acc algorithm runs about 4.5 times faster in the multi-pedestrian tracking sequences and 5.2 times faster in the multi-face tracking sequences than the H^{2}T_woAcc method on average, respectively. Overall, the proposed H^{2}T_Acc method is more effective for multi-object tracking task for real-world applications.

6 Conclusion

In this paper, a multi-object tracking algorithm that exploits dense structures is proposed. The multi-object tracking task is carried out by exploiting dense structures on multiple affinity hypergraphs constructed hierarchically, which consider similarities among different tracklets across the temporal domain for better association in terms of identity and trajectory. Visual cues including appearance, motion and trajectory smoothness are used for measuring affinity. For computational efficiency, we propose a hypothesize-and-test algorithm to approximate a hypergraph with a common graph from which the dense structures are exploited. Experimental evaluations on both multi-pedestrian and multi-face tracking demonstrate the proposed algorithms perform favorably against the state-of-the-art tracking methods.

Acknowledgements

Longyin Wen, Zhen Lei and Stan Z. Li are supported by the National Natural Science Foundation of China Projects (No.61375037, No.61473291, No.61572501, No.61572536), National Science and Technology Support Program Project (No.2013BAK02B01), Chinese Academy of Sciences Project

(No. KGZD-EW-102-2), and AuthenMetric R&D Funds. Siwei Lyu is supported by US NSF CAREER Award IIS-0953373 and US NSF Research Grant CCF-1319800. Ming-Hsuan Yang is supported in part by NSF CAREER Grant (No. 1149783) and NSF IIS Grant (No. 1152576). Zhen Lei is the corresponding author.

REFERENCES

[1] W. Ge and R. T. Collins, "Multi-target data association by tracklets with unsupervised parameter estimation," in *Proceedings of British Machine Vision Conference*, 2008, pp. 1–10.

[2] Q. Yu and G. G. Medioni, "Multiple-target tracking by spatiotemporal monte carlo markov chain data association," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2196–2210, 2009.

[3] B. Benfold and I. Reid, "Stable multi-target tracking in real-time surveillance video," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3457–3464.

[4] T. Yang, S. Z. Li, Q. Pan, and J. Li, "Real-time multiple objects tracking with occlusion handling in dynamic scenes," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 970–975.

[5] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1815–1821.

[6] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[7] H. Izadinia, I. Saleemi, W. Li, and M. Shah, "$(MP)^2T$: Multiple people multiple parts tracker," in *Proceedings of European Conference on Computer Vision*, 2012, pp. 100–114.

[8] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1201–1208.

[9] M. Hofmann, D. Wolf, and G. Rigoll, "Hypergraphs for joint multi-view reconstruction and multi-object tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3650–3657.

[10] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua, "Multiple object tracking using k-shortest paths optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1806–1819, 2011.

[11] W. Brendel, M. R. Amer, and S. Todorovic, "Multiobject tracking as maximum weight independent set," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1273–1280.

[12] H. Jiang, S. Fels, and J. J. Little, "A linear programming approach for multiple object tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[13] X. Shi, H. Ling, J. Xing, and W. Hu, "Multi-target tracking by rank-1 tensor approximation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2387–2394.

[14] X. Shi, H. Ling, W. Hu, C. Yuan, and J. Xing, "Multi-target tracking with motion context in tensor power iteration," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3518–3525.

[15] C. Huang, Y. Li, and R. Nevatia, "Multiple target tracking by learning-based hierarchical association of detection responses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 898–910, 2013.

[16] B. Yang, C. Huang, and R. Nevatia, "Learning affinities and dependencies for multi-target tracking using a CRF model," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1233–1240.

[17] B. Yang and R. Nevatia, "Multi-target tracking by online learning of non-linear motion patterns and robust appearance models," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1918–1925.

[18] ——, "An online learned CRF model for multi-target tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 2034–2041.

[19] A. R. Zamir, A. Dehghan, and M. Shah, "GMCP-tracker: Global multi-object tracking using generalized minimum clique graphs," in *Proceedings of European Conference on Computer Vision*, 2012, pp. 343–356.

[20] A. Dehghan, S. M. Assari, and M. Shah, "GMMCP-Tracker:globally optimal generalized maximum multi clique problem for multiple object tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4091–4099.

[21] L. Marchesotti, L. Marcenaro, G. Ferrari, and C. S. Regazzoni, "Multiple object tracking under heavy occlusions by using kalman filters based on shape matching," in *Proceedings of IEEE International Conference on Image Processing*, 2002, pp. 341–344.

[22] D. R. Magee, "Tracking multiple vehicles using foreground, background and motion models," *Image and Vision Computing*, vol. 22, no. 2, pp. 143–155, 2004.

[23] W. F. Leven and A. D. Lanterman, "Unscented kalman filters for multiple target tracking with symmetric measurement equations," *IEEE Trans. Automat. Contr.*, vol. 54, no. 2, pp. 370–375, 2009.

[24] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[25] K. Smith, D. Gatica-Perez, and J.-M. Odobez, "Using particles to track varying numbers of interacting people," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 962–969.

[26] Z. Khan, T. R. Balch, and F. Dellaert, "MCMC-based particle filtering for tracking a variable number of interacting targets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1805–1918, 2005.

[27] M. Yang, Y. Liu, L. Wen, Z. You, and S. Z. Li, "A probabilistic framework for multitarget tracking with mutual occlusions," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1298–1305.

[28] T. Fortmann, Y. B. Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE J. Oceanic Engineering*, vol. 8, no. 3, pp. 173–184, 1983.

[29] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, pp. 843–854, 1979.

[30] Z. Wu, T. H. Kunz, and M. Betke, "Efficient track linking methods for track graphs using network-flow and set-cover techniques," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1185–1192.

[31] R. T. Collins, "Multitarget data association with higher-order motion models," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1744–1751.

[32] A. A. Butt and R. T. Collins, "Multi-target tracking by lagrangian relaxation to min-cost network flow," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1846–1853.

[33] A. Andriyenko and K. Schindler, "Multi-target tracking by continuous energy minimization," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1265–1272.

[34] A. Andriyenko, K. Schindler, and S. Roth, "Discrete-continuous optimization for multi-target tracking," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1926–1933.

[35] M. Kim, S. Kumar, V. Pavlovic, and H. A. Rowley, "Face tracking and recognition with visual constraints in real-world videos," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[36] P. Wang and Q. Ji, "Robust face tracking via collaboration of generic and specific models," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1189–1199, 2008.

[37] J. Sung, T. Kanade, and D. Kim, "Pose robust face tracking by combining active appearance models and cylinder head models," *International Journal of Computer Vision*, vol. 80, no. 2, pp. 260–274, 2008.

[38] M. Zhou, L. Liang, J. Sun, and Y. Wang, "AAM based face tracking with temporal matching and face segmentation," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 701–708.

[39] Z. Kalal, K. Mikolajczyk, and J. Matas, "Face-tld: Tracking-learning-detection applied to faces," in *Proceedings of IEEE International Conference on Image Processing*, 2010, pp. 3789–3792.

[40] Z. Cai, L. Wen, D. Cao, Z. Lei, D. Yi, and S. Z. Li, "Person-specific face tracking with online recognition," in *Proceedings of IEEE*

International Conference on Automatic Face and Gesture Recognition, 2013, pp. 1–6.
[41] M. Roth, M. Bäuml, R. Nevatia, and R. Stiefelhagen, "Robust multi-pose face tracking by multi-stage tracklet association," in *Proceedings of International Conference on Pattern Recognition*, 2012, pp. 1012–1016.
[42] S. Duffner and J.-M. Odobez, "Track creation and deletion framework for long-term online multiface tracking," *IEEE Transactions on Image Processing*, vol. 22, no. 1, pp. 272–285, 2013.
[43] B. Wu, S. Lyu, B. Hu, and Q. Ji, "Simultaneous clustering and tracklet linking for multi-face tracking in videos," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2856–2863.
[44] H. Liu and S. Yan, "Efficient structure detection via random consensus graph," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 574–581.
[45] H. Liu, X. Yang, L. J. Latecki, and S. Yan, "Dense neighborhoods on affinity graph," *International Journal of Computer Vision*, vol. 98, no. 1, pp. 65–82, 2012.
[46] L. Wen, W. Li, Z. Lei, D. Yi, and S. Z. Li, "Multiple target tracking based on undirected hierarchical relation hypergraph," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1282–1289.
[47] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.
[48] C. Bron and J. Kerbosch, "Finding all cliques of an undirected graph (algorithm 457)," *Commun. ACM*, vol. 16, no. 9, pp. 575–576, 1973.
[49] H. Liu, L. J. Latecki, and S. Yan, "Robust clustering as ensembles of affinity relations," in *Advances in Neural Information Processing Systems*, 2010, pp. 1414–1422.
[50] A. Milan, "Continuous energy minimization tracker website," http://www.cvg.rdg.ac.uk/PETS2009/a.html.
[51] R. Stiefelhagen, K. Bernardin, R. Bowers, J. S. Garofolo, D. Mostefa, and P. Soundararajan, "The clear 2006 evaluation," in *CLEAR*, 2006, pp. 1–44.
[52] A. Milan, K. Schindler, and S. Roth, "Challenges of ground truth evaluation of multi-target tracking," in *Workshops in Conjunction with IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 735–742.
[53] A. Milan, S. Roth, and K. Schindler, "Continuous energy minimization for multitarget tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 58–72, 2014.
[54] B. Yang, http://iris.usc.edu/people/yangbo/downloads.html.
[55] J. Yan, Z. Lei, D. Yi, and S. Z. Li, "Multi-pedestrian detection in crowded scenes: A global view," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3124–3129.
[56] Dehghan, O. Oreifej, E. Hand, and M. Shah, http://crcv.ucf.edu/data/ParkingLOT.
[57] S. Z. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum, "Statistical learning of multi-view face detection," in *Proceedings of European Conference on Computer Vision*, 2002, pp. 67–81.

Longyin Wen received the B.Eng. degree in Automation from University of Electronic Science and Technology of China (UESTC) in 2010, and the PhD degree from Institute of Automation, Chinese Academy of Sciences (CASIA) in 2015. He now moves to the University at Albany, State University of New York, for postdoc research. His research interests are computer vision, pattern recognition and object tracking in particular.

Zhen Lei received the B.S. degree in automation from the University of Science and Technology of China, in 2005, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, in 2010, where he is currently an Associate Professor. He has published over 100 papers in international journals and conferences. His research interests are in computer vision, pattern recognition, image processing, and face recognition in particular. He served as an Area Chair of the International Joint Conference on Biometrics in 2014, the IAPR/IEEE International Conference on Biometric in 2015 and 2016, and the IEEE International Conference on Automatic Face and Gesture Recognition in 2015.

Siwei Lyu is an associate professor in Computer Science at University of Albany, State University of New York. He obtained his B.S. degree in Information Science and M.S. degree in Computer Science, both from Peking University, China, in 1997 and 2000, respectively. From 2000 to 2001, he was an assistant researcher at Microsoft Research Asia. He obtained Ph.D. degree in computer science from Dartmouth College. From 2005 to 2008 he was a post-doctoral research associate at New York University. His research interests include image processing and forensics, machine learning and computer vision.

Stan Z. Li received the B.Eng. degree from Hunan University, China, the M.Eng. degree from National University of Defense Technology, China, and the Ph.D. degree from Surrey University, U.K. He is currently a professor at the National Laboratory of Pattern Recognition and the director of the Center for Biometrics and Security Research (CBSR), Institute of Automation (CASIA), and the director of the Center for Visual Internet of Things Research (VIOT), Chinese Academy of Sciences. He worked at Microsoft Research Asia as a researcher from 2000 to 2004. Prior to that, he was an associate professor at Nanyang Technological University, Singapore. He was elevated to IEEE Fellow for his contributions to the fields of face recognition, pattern recognition, and computer vision. His research interest includes pattern recognition and machine learning, image and vision processing, face recognition, biometrics, and intelligent video surveillance. He has published over 200 papers in international journals and conferences, and authored and edited eight books. He was an associate editor of IEEE Transactions on Pattern Analysis and Machine Intelligence and was acting as the Editor-in-Chief for the Encyclopedia of Biometrics. He serves / served as a program co-chair for the International Conference on Biometrics 2007, 2009, 2015 and 2016, a general chair for the 9th IEEE Conference on Automatic Face and Gesture Recognition, and has been involved in organizing other international conferences and workshops in the fields of his research interest.

Ming-Hsuan Yang is an Associate Professor of electrical engineering and computer science with the University of California, Merced, CA, USA. He received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 2000. Prior to joining UC Merced in 2008, he was a Senior Research Scientist with the Honda Research Institute, Mountain View, CA, USA, on vision problems related to humanoid robots. He served as an Associate Editor of the IEEE Transactions on Pattern Analysis and Machine Intelligence from 2007 to 2011, and is an Associate Editor of the International Journal of Computer Vision, Image and Vision Computing, and Journal of Artificial Intelligence Research. He received the NSF CAREER Award in 2012, and the Google Faculty Award in 2009. He is a senior member of the ACM.