Noname manuscript No.
(will be inserted by the editor)

Multi-Camera Multi-Target Tracking with Space-Time-View Hypergraph

Longyin Wen · Zhen Lei · Ming-Ching Chang · Hongang Qi · Siwei Lyu

Received: date / Accepted: date

Abstract Incorporating multiple cameras is an effective solution to improve the performance and robustness of multi-target tracking to occlusion and appearance ambiguities. In this paper, we propose a new multi-camera multi-target tracking method based on a space-time-view hypergraph that encodes higher-order constraints (*i.e.*, beyond pairwise relations) on 3D geometry, appearance, motion continuity, and trajectory smoothness among 2D tracklets within and across different camera views. We solve tracking in each single view and reconstruction of tracked trajectories in 3D environment simultaneously by formulating the problem as an efficient search of dense subhypergraphs on the space-time-view hypergraph using a sampling based approach. Experimental results on the PETS 2009 benchmark dataset demonstrate that our method outperforms state-of-the-art methods in both single-view and multi-camera multi-target tracking, while achieving close to real-time running efficiency. We also provide experimental analysis of the influence of various aspects of our method to the final tracking performance.

Keywords multi-camera multi-target tracking, single-camera multi-target tracking, space-time-view hypergraph, dense sub-hypergraph search

Longyin Wen and Siwei Lyu
Computer Science Department, University at Albany, State University of New York.

Zhen Lei
National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences.

Ming-Ching Chang
GE Global Research Center.

Honggang Qi
School of Computer and Control Engineering, University of the Chinese Academy of Sciences.

1 Introduction

As an important problem in computer vision, multi-target tracking finds wide applications in video surveillance, traffic monitoring and crowd analysis. With the maturity of detection algorithms [9], the current state-of-the-art performance in multi-target tracking is attained with the *tracking-by-detection* methodology, in which reliable detection of short sequence of moving objects or *tracklets* are linked based on their affinities in appearance and motion to form long tracks. Albeit these successes, the majority of existing multi-target tracking algorithms use a single camera view. As such, their performance succumbs to false/miss detections due to target occlusions and ambiguous appearances. Using detections gleaned from different but overlapping camera views, these problems can be effectively solved and the accuracy of multi-target tracking can be significantly improved.

Many previous methods on multi-camera multi-target tracking [12,46,4,47,32,3] entail two consecutive steps: (i) repeated single view *tracking* that finds tracklets in each individual camera view, and (ii) cross-view *reconstruction* of 2D tracklets using 3D geometric constraints. These simple approaches do not take advantage of the fact that single view tracking and cross-view reconstruction provide mutually bootstrapping information: 3D geometric constraints can rule out false detection and improve tracklet linking in each view, while reliable linking of tracklets in individual views can compensate the effect of noise and outliers that often plague the reconstruction step.

An alternative strategy of multi-camera tracking is to jointly solve the tracking and reconstruction problems in a single optimization framework. However, as will be detailed in § 2, two existing methods [26,13] us-

ing this strategy rely on pairwise association of tracklets (Figure 1(a,b)), and do not take full advantage of the strong higher order correlations among the tracklets across time and space. Such higher order correlations are particularly useful to handle multi-camera tracking scenarios with severe occlusions.

In this work, we describe a new multi-camera multi-target tracking method based on a weighted hypergraph that represents higher-order affinities of 2D tracklets, which characterize their consistencies in 3D geometry, appearance, motion continuity and trajectory smoothness. We term this hypergraph as *space-time-view hypergraph* (STV hypergraph). The nodes of STV hypergraph correspond to potential *3D couplings* of 2D tracklets, which are reconstructed 3D tracklets from 2D tracklets across different views that are potentially associated with the trajectory of the same tracked target (see Figure 1(c) for an illustrative example). Geometric consistency of these 2D tracklets in forming a coupling is encoded with the weight of each node. Hyperedges of STV hypergraph with their associated weights reflect affinities among the couplings. In order to correctly associate tracklets across multiple views and eventually reconstruct the 3D trajectory, we further perform a search of dense subhypergraphs on STV hypergraph, which correspond to subhypergraphs with higher weights over including nodes and hyperedges, and then accumulatively link couplings in such subhypergraphs.

Contributions Our work has several contributions. First, we introduce STV hypergraph constructed from input videos of multiple views as a flexible and compact representation for the inference of higher order correlations among tracklets across space, time, and camera views, which is a generalization of the hypergraph based single view multi-target tracking method presented in [45]. Second, we formulate the multi-camera multi-target tracking problem as searching for dense subhypergraphs on STV hypergraph, which is solved efficiently by the proposed sampling based approximation method. Third, we perform extensive experiments on the PETS2009 benchmark dataset to compare with the state-of-the-art methods, and show improved effectiveness and running efficiency of our method.

The rest of the paper is organized as follows. In § 2 we review relevant related works. In § 3 we describe our method in detail. Experimental results are presented in § 4 and § 5 concludes the paper with discussion of future works.

2 Related Works

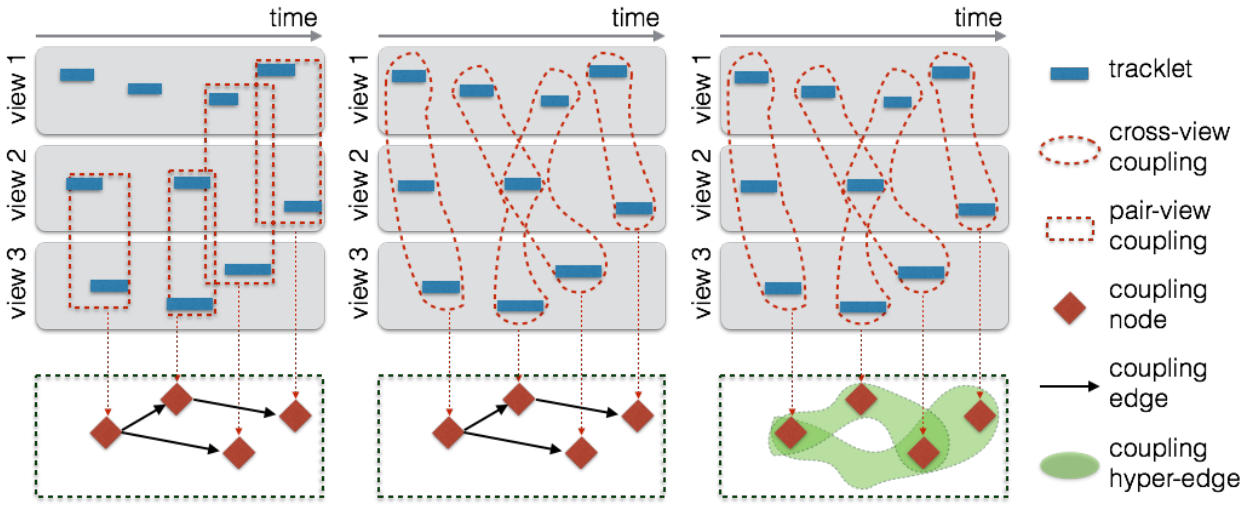
We review the most relevant multi-object tracking methods using both single and multiple camera views.

2.1 Single-Camera Multi-Target Tracking

A traditional approach to multi-target tracking is to predict the states of tracked targets using dynamic Bayesian filtering methods, *e.g.*, Kalman or particle filters [16,33,20,43,29,50]. These methods can track targets state effectively in short durations and run in real-time, but are not effective in handling occlusion and appearance changes that often occur in complex tracking scenarios.

Many recent effective single-view multi-target tracking methods are based on the tracking-by-detection approach, and formulate tracking as a data association problem. The *joint probabilistic data association filter* (JPDAF) [14] and *multiple hypotheses tracking* (MHT) [40] have been proposed to handle the data association problem efficiently. The JPDAF algorithm focuses on estimating the best assignments between the tracked targets and the detections in a probabilistic framework. Different from frame-by-frame association in JPDAF, MHT computes the likelihoods of all candidate assignments over several time steps. However, the number of candidate assignments grows exponentially with the number of frames, which makes MHT not efficient when handling the long-term association. Yu *et al.* [51] presents a data driven Markov Chain Monte Carlo method to accomplish the data association task in multiple frames. The sampling-based inference algorithm may have long "burn-in" time and difficult to evaluate due to the lack of practical check for convergence.

To handle the long-term association problem, several algorithms have been proposed, which differs in the specific optimization methods used, including network flow [52,17,39,7], K-shortest Path (KSP) [4], maximum weight independent set [6], linear programming [19], multi-frame matching [42], hierarchical Hungarian algorithm [15,48,49], tensor power iteration [41], hypergraph based optimization [45]. In particular, the method of [45] is the most related work, because it also uses a weighted hypergraph to represent higher order affinities between 2D tracklets, and directly searches the dense subgraphs on the hypergraph to solve the tracking problem. However, this method is not suitable for the multi-camera multi-target tracking problem. In particular, STV hypergraph models relations among 3D tracklets reconstructed from 2D tracklets of multiple views, and nodes in STV hypergraph have weights reflecting unreliabilities of 3D reconstruction. Such node weights are crucial in our method, which are not addressed in [45].



(a) work of [26] (b) work of [13] (c) this work legends

Fig. 1 Models of two previous methods and this work on multi-camera multi-target tracking. To improve the clarity, we illustrate only a partial set of edges and hyperedges. This figure is better viewed in color.

Furthermore, the number of nodes in STV hypergraph is several orders of magnitude larger than that used in single view tracking [45], which renders the simple optimization algorithm of [45] impractical due to high computation and memory requirements. Most of these works merely focus on using the pairwise similarities of 2D detections/tracklets to complete the tracking task.

2.2 Multi-Camera Multi-Target Tracking

Using multiple camera views can potentially improve the performance of multi-target tracking, but it also brings up some challenging issues. In particular, the tracking algorithm must link 2D tracklets and at the same time reconstruct their 3D trajectories. Early multi-camera multi-target tracking methods [12,46,4,47,32] usually solve the tracking and reconstruction problems in separate stages, which do not take advantage of the mutually bootstrapping relation between these two tasks.

Recently, two existing works attempt to solve the joint problem within a single optimization framework. In [26] (model illustrated in Figure 1(a)), multiple graphs are constructed for tracklets in each view to capture their affinities, and the associations of these tracklets are encoded with another type of graph that are constructed for each pair of camera views. The overall tracking problem is solved using the Dantzig-Wolfe decomposition and branching algorithm. This method is further improved in [13] (model illustrated in Figure 1(b)), where the couplings between 2D tracklets across two or more camera views are formed, and longer tracks are obtained from a directed graph capturing pairwise dependencies of reconstructed 3D tracklets in consecutive frames. However, in both works, only the relations between candidate associations in *consecutive frames* are considered. If a target fails to appear in any camera view due to occlusion or miss detection, using pairwise association in consecutive frames will lead to fragmentations and identity switches, which can significantly deteriorate the overall performance and robustness of the tracking method.

3 Methodology

Our multi-camera multi-target tracking method is based on the space-time-view hypergraph (STV hypergraph) representation of the cross-view and temporal associations of detected 2D tracklets in individual camera views. The process starts with the generation of couplings from tracklets in each single camera view (§ 3.1) and computation of affinity measures (§ 3.2), on which the STV hypergraph is constructed (§ 3.3). Long target trajectories in 3D are obtained from a dense subhypergraph search on STV hypergraph (§ 3.4). The notations used in this paper are listed in Table 1.

3.1 Generating Couplings

We postulate that there are V static camera views, where videos from each view are synchronized with the same frame rate. Furthermore, from each video, tentative short sequences of detected targets (tracklets) are assumed to have been obtained from frame detections (*e.g.*, using [10]) or using single-view tracklet linking methods (*e.g.*, [41,45]). Throughout this paper, we use

Table 1 Notations

V	Number of used camera views.
$q_j^{v,i} = (x_j^{v,i}, y_j^{v,i}, w_j^{v,i}, h_j^{v,i})$	The bounding box of the j-th detection in the i-th tracklet of camera view v, $(x_j^{v,i}, y_j^{v,i})$ is the center pixel location and $(w_j^{v,i}, h_j^{v,i})$ is the dimension.
$D_j^{v,i} = (t_j^{v,i}, q_j^{v,i})$	The j-th detection in the i-th tracklet of camera view v, and $t_j^{v,i}$ is the frame number of the detection.
$T_i^v = \{D_1^{v,i}, \cdots, D_{m_{c,i}}^{v,i}\}$	The i-th tracklet of camera view v, $m_{c,i}$ is the number of detections in the tracklet.
$\mathbf{T}_v = \{T_1^v, \cdots, T_{n_v}^v\}$	The collection of detected 2D tracklets from the v-th camera view.
$\mathcal{T}$	A coupling in the 3D world.
k	The degree of the hypergraph.
ν_i	The i-th vertex (tracklet) in the Space-Time-View hypergraph.
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	The Space-Time-View hypergraph, where $\mathcal{V}$ is the node set and $\mathcal{E}$ is the hyperedge set, *i.e.*, $\mathcal{E} \subset \overbrace{\mathcal{V} \times \cdots \times \mathcal{V}}^{k}$.
e	The k-tuple nodes involved in a hyperedge, *i.e.*, $e = (\nu_1, \cdots, \nu_k)$.
π_i	The i-th connection samples involving $k-1$ nodes.
β^*	The minimal size of the searched dense subgraph.
$\mathcal{G}^* = (\mathcal{V}^*, \mathcal{E}^*)$	The approximate Space-View-Time graph, where $\mathcal{V}^*$ is the same weighted node set as the corresponding STV hypergraph $\mathcal{G}$, and $\mathcal{E}^* = \mathcal{V}^* \times \mathcal{V}^*$ is edge set describing the supports between the nodes.

v to index camera views, i to index the tracklets, and j to index the detections of a tracklet.

We denote the collection of detected 2D tracklets from the v-th camera view as $\mathbf{T}_v = \{T_1^v, \cdots, T_{n_v}^v\}$. A single 2D tracklet, $T_i^v = \{D_1^{v,i}, \cdots, D_{m_{c,i}}^{v,i}\}$, corresponds to a series of frame detections, $m_{c,i}$ is the number of detections in the tracklet, and $D_j^{v,i} = (t_j^{v,i}, q_j^{v,i})$, where $t_j^{v,i}$ is the frame number of the detection, and $q_j^{v,i} = (x_j^{v,i}, y_j^{v,i}, w_j^{v,i}, h_j^{v,i})$ specifies the bounding box of the detection with center pixel location $(x_j^{v,i}, y_j^{v,i})$ and dimension $(w_j^{v,i}, h_j^{v,i})$. We also use $\mathbf{t}_i^v = \{t_1^{v,i}, \cdots, t_{m_{c,i}}^{v,i}\}$ to denote the set of all frame indices of the corresponding 2D tracklet T_i^v. Our definition of 2D tracklets generalizes cases of single detection (*i.e.*, $|\mathbf{t}_i^v| = 1$), or continuous sequence of detections (*i.e.*, $\mathbf{t}_i^v = \{a, a+1, \cdots, b-1, b\}$ where $a < b$ are two integers). We also consider calibrated cameras with known parameters, where targets are moving on a common ground-plane, such that any 2D pixel location (x, y) in the video frame can be back projected to the 3D world coordinates using a mapping function, $\phi^v(x, y) = (X, Y, Z)$.

A 3D *coupling* collects 2D tracklets from different camera views that potentially correspond to the same trajectory of a target in the 3D world. Formally, we define a coupling $\mathcal{T}$ as a non-empty subset of $\bigcup_{v=1}^{V} \mathbf{T}_v$, *i.e.*, $\emptyset \neq \mathcal{T} \subset \bigcup_{v=1}^{V} \mathbf{T}_v$ such that *no more than one* tracklet from any camera view will be included, *i.e.*,

$$\left(T_i^v \in \mathcal{T}\right) \wedge \left(T_{i'}^{v'} \in \mathcal{T}\right) \Rightarrow v \neq v'. \tag{1}$$

As such, the maximum total number of unique couplings is given by $\prod_{v=1}^{V}(n_v + 1) - 1$. This number is obtained as the following: from each camera view, at most one tracklet can be included into the coupling, leaving the total choices for one view as $n_v + 1$. The total is given by $\prod_{v=1}^{V}(n_v + 1) - 1$, where the minus one corresponds to the case that no tracklets are chosen from any views.

We add another constraint for a coupling constructed by *two or more* 2D tracklets, *i.e.*, for each 2D tracklet included in the coupling, it must have overlapping frame indices with *at least one* other 2D tracklets that are also in the coupling, as:

$$T_i^v \in \mathcal{T} \Rightarrow \exists T_{i'}^{v'} \in \mathcal{T} \wedge \mathbf{t}_i^v \cap \mathbf{t}_{i'}^{v'} \neq \emptyset. \tag{2}$$

Note that (2) is weaker than requiring *all* 2D tracklets to have overlapping time, which is justified by the nature of multi-camera tracking scenario that the target may not be observed in all camera views due to occlusion.

We define the frame indices of a coupling as the union of frame indices of its composing 2D tracklets, as $\mathbf{t}_{\mathcal{T}} = \bigcup_{T_i^v \in \mathcal{T}} \mathbf{t}_i^v$. For each frame of the coupling $t \in \mathbf{t}_{\mathcal{T}}$, we also maintain a data structure that backtracks its composing 2D frame detections at t, as

$$\mathcal{D}_t^{\mathcal{T}} = \{(v, i, j) : T_i^v \in \mathcal{T} \wedge t \in \mathbf{t}_{\mathcal{T}} \wedge t = t_j^{v,i}\}, \tag{3}$$

where (1), (2), and (3) together ensures three conditions: (i) 2D tracklet T_i^v must be included in coupling $\mathcal{T}$, (ii) frame index t is in the frame indices of $\mathcal{T}$, and (iii) T_i^v has a detection at frame t. Using $\mathcal{D}_t^{\mathcal{T}}$, we compute the predicted 3D position of the coupling at t as the average 3D positions obtained from the corresponding 2D tracklets,

$$\mathcal{P}_t^{\mathcal{T}} = \frac{1}{|\mathcal{D}_t^{\mathcal{T}}|} \sum_{(v,i,j) \in \mathcal{D}_t^{\mathcal{T}}} \phi^v\left(x_j^{v,i}, y_j^{v,i}\right), \tag{4}$$

and the scattering (uncertainty) of the predicted 3D position,

$$\epsilon_t^{\mathcal{T}} = \frac{1}{|\mathcal{D}_t^{\mathcal{T}}|} \sum_{(v,i,j)\in\mathcal{D}_t^{\mathcal{T}}} \left\| \mathcal{P}_t^{\mathcal{T}} - \phi^v\left(x_j^{v,i}, y_j^{v,i}\right) \right\|^2, \quad (5)$$

where (5) is used as a measure of coherence of the set of 2D tracklets in forming the 3D trajectory.

3.2 Affinity Among Couplings

The obtained 3D couplings may correspond to segments of a longer 3D trajectory belonging to a moving target being tracked in multiple camera views. To evaluate the likelihood of a set of couplings in forming a longer trajectory, similar to [45], we introduce three affinity measures for appearances, motion continuity, and trajectory smoothness. As couplings overlapping in time cannot be associated with one target, we set all affinity measures to zero in that case. Subsequently, we consider only couplings with no overlapping frame indices in the appearance Eq.(6), motion continuity Eq.(7), and trajectory smoothness (8) affinities calculation.

Appearance affinity. The appearance affinity between a pair of non-overlapping couplings $\mathcal{T}$ and $\mathcal{T}'$, with $\mathcal{T}$ preceding $\mathcal{T}'$, is computed from three image features of detections from the last frames in $\mathcal{T}$ and the first frames in $\mathcal{T}'$ [1]. The features we used are histograms of color, shape gradient and local binary patterns as in [37].

Color affinity $\psi_c(\mathcal{T},\mathcal{T}') = 0$, if there is no common camera view between frame detections of the last frame of $\mathcal{T}$ and those of the first frame of $\mathcal{T}'$. Otherwise, for each common camera view of the two sets of frames, we extract color histograms of the corresponding two detections and evaluate their Bhattacharyya distance. $\psi_c(\mathcal{T},\mathcal{T}')$ is computed as the average of such Bhattacharyya distances over all common views. The similarity based on histograms of shape gradient $\psi_s(\mathcal{T},\mathcal{T}')$ and local binary patterns $\psi_b(\mathcal{T},\mathcal{T}')$ are computed similarly. We denote the correspondence between a hypergraph node and a coupling as $\nu \sim \mathcal{T}$. The appearance affinity of node set $\boldsymbol{\nu} = (\nu_1, \cdots, \nu_k)$ with $\nu_i \sim \mathcal{T}_i$ in ascending order of time is computed as

$$\Psi_{\text{app}}(\boldsymbol{\nu}) = \sum_{i,j} e^{\lambda_1\psi_c(\mathcal{T}_i,\mathcal{T}_j)+\lambda_2\psi_s(\mathcal{T}_i,\mathcal{T}_j)+\lambda_3\psi_b(\mathcal{T}_i,\mathcal{T}_j)}, \quad (6)$$

where λ_1, λ_2 and λ_3 are parameters controlling the sensitivity of the affinity score with regards to each type of appearance features.

[1] The last frame index of $\mathcal{T}$ and the first frame index of $\mathcal{T}'$ may correspond to multiple detections from different camera views.

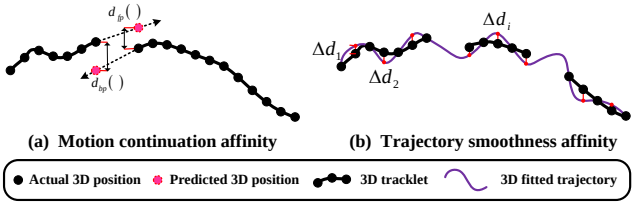


Fig. 2 (a) Motion continuation affinity calculation of a pair of couplings. (b) Trajectory smoothness affinity calculation of a set of couplings. See text for more details.

Motion continuation affinity. The motion continuation affinity between a pair of non-overlapping couplings $\mathcal{T}$ and $\mathcal{T}'$, with $\mathcal{T}$ preceding $\mathcal{T}'$, is based on the forward-backward predictions between the last frame detections of $\mathcal{T}$ and the first frame detections of $\mathcal{T}'$.

We first estimate the "ending" velocity of $\mathcal{T}$ by dividing the difference of 3D positions (computed with (4)) of its last two frame detections with their corresponding time lapse. The predicted position for the start of $\mathcal{T}'$ is obtained by projecting the 3D position of the last frame detection of $\mathcal{T}$ with the estimated ending velocity, multiplied by the time lapse between the last frame detection of $\mathcal{T}$ and that of the first frame detection of $\mathcal{T}'$, Figure 2(a). We then compute the ℓ_2 distance between the actual 3D position of the first frame of $\mathcal{T}'$ and its *forward* prediction from $\mathcal{T}$, as $d_{fp}(\mathcal{T},\mathcal{T}')$.

Similarly, the *backward* prediction of the 3D position of the last frame detection of $\mathcal{T}$ is obtained with the 3D position of the first frame detection of $\mathcal{T}'$ and the estimated beginning velocity from its first two frame detections, Figure 2 (a). We compute the ℓ_2 distance between the 3D position of the last frame of $\mathcal{T}$ and its backward prediction from $\mathcal{T}'$, as $d_{bp}(\mathcal{T},\mathcal{T}')$. Then the motion continuation affinity of node set $\boldsymbol{\nu} = (\nu_1, \cdots, \nu_k)$ with $\nu_i \sim \mathcal{T}_i$ in ascending order of time is computed as

$$\Psi_{\text{mot}}(\boldsymbol{\nu}) = \sum_{i=1}^{k-1} e^{-\lambda_4\left(d_{fp}(\mathcal{T}_i,\mathcal{T}_{i+1})+d_{bp}(\mathcal{T}_i,\mathcal{T}_{i+1})\right)}, \quad (7)$$

where λ_4 is the parameter controlling the sensitivity of the affinity score to the prediction errors.

Trajectory smoothness affinity. A common assumption for visual tracking task is that tracked targets should move continuously and smoothly for most of the time. The trajectory smoothness affinity evaluates the spatial-temporal coherence of a long trajectory formed from a set of non-overlapping couplings $\mathcal{T}_1, \cdots, \mathcal{T}_k$. Specifically, we first compute the 3D positions of these couplings with (4). We then fit a piecewise second order smooth parametric trajectory with cubic spline interpolation to a subset of these 3D positions, Figure 2(b). The ℓ_2 distance, $d_{\text{int}}(\mathcal{T}_1, \cdots, \mathcal{T}_k)$, of the remaining 3D positions with their predictions based on the interpolated smooth curve is computed, which evaluates the

smoothness of the fitted trajectory (where small values indicate coherent fit). The trajectory smoothness affinity score of node set $\boldsymbol{\nu} = (\nu_1, \cdots, \nu_k)$ with $\nu_i \sim \mathcal{T}_i$ in ascending order of time is computed from $d_{\text{int}}(\mathcal{T}_1, \cdots, \mathcal{T}_k)$ as

$$\Psi_{\text{smo}}(\boldsymbol{\nu}) = e^{-\lambda_5 d_{\text{int}}(\mathcal{T}_1, \cdots, \mathcal{T}_k)}, \tag{8}$$

where parameter λ_5 controls the sensitivity of the affinity score to the deviation of smooth trajectories.

3.3 The Space-Time-View Hypergraph (STV hypergraph)

The STV hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ encodes both the reconstruction and linking of detected 2D tracklets in different camera views, and is the central data structure of our multi-camera multi-target tracking method. See Figure 1(c) for an illustrated example.

A **node** ν in STV hypergraph corresponds to a coupling $\mathcal{T}$ (as described in § 3.1) and is associated with a weight reflecting its reliability:

$$\mathcal{A}(\nu) = e^{-\lambda_6 \max_{t \in \mathfrak{t}_\mathcal{T}} \{\epsilon_t^\mathcal{T} - \lambda_7 |\mathcal{D}_t^\mathcal{T}|\}}, \tag{9}$$

where $\epsilon_t^\mathcal{T}$ (from (5)) is the scattering of coupling $\mathcal{T}$ at frame t, and $|\mathcal{D}_t^\mathcal{T}|$ (see (3)) is the number of 2D tracklets associated with the coupling at frame t. λ_6 controls the sensitivity of the weight to the reliability score, and λ_7 represents the trade-off between lower scattering and larger number of associated 2D views. Higher node weights suggest increasing likelihood of the composing 2D tracklets corresponding to a single 3D tracklet. To avoid the case where couplings corresponding to a single 2D view dominate the weight (where the scattering is always zero), we also penalize couplings with smaller number of associated 2D views through parameter λ_7.

A **hyperedge** in the STV hypergraph connects multiple nodes with a weight, whose corresponding couplings potentially form a longer trajectory as shown in Figure 1(c). We only consider hyperedge of degree k, *i.e.*, each hyperedge in STV hypergraph is associated with k nodes [2].

We enforce two constraints that are important to reduce the number of hyperedges. First, for all nodes connected by one hyperedge, their couplings should not overlap in time. Second, we evaluate the distance between the last and first detections of either pair of couplings in a hyperedge. If the distance is significantly larger than the maximum possible velocity (*e.g.*, < 5 m/s for a pedestrian), then the two nodes should not be grouped together by a hyperedge. The weight of a hyperedge in STV hypergraph, $e = (\nu_1, \cdots, \nu_k)$ is computed using the appearance, motion continuity and trajectory smoothness affinities defined in § 3.2,

$$W(e) = \lambda_8 \Psi_{\text{app}}(e) + \lambda_9 \Psi_{\text{mot}}(e) + \lambda_{10} \Psi_{\text{smo}}(e), \tag{10}$$

where parameters λ_8, λ_9, and λ_{10} balance the three types of affinity scores.

3.4 Dense Subhypergraph Search

We formulate the problem of recovering longer trajectories of the targets as searching for "dense" subhypergraphs on the STV hypergraph. Here a dense subhypergraph corresponds to a group of reliable nodes (couplings) that are inter-connected with a set of hyperedges with high weights. We extend a local search algorithm for dense subgraphs in graphs [31] to search the dense subhypergraphs on the STV hypergraph. The basic idea is to find a dense neighborhood for each node in STV hypergraph and then remove the conflicts between such neighborhoods to obtain the longer target trajectories. To accommodate the large number of nodes in STV hypergraph, we further employ a sampling based algorithm to accelerate such search.

3.4.1 Problem Formulation

For a node ν, we denote its *neighborhood* as $\mathcal{N}(\nu)$, which is the set of nodes containing direct neighbors of ν (*i.e.*, connected with one hyperedge to ν) on the STV hypergraph. We then aim to find a subset of $\mathcal{N}(\nu)$ with m nodes such that they jointly form an β-subhypergraph that has the maximum weights combining both the hyperedges and nodes. To this end, we introduce an indicator variable $z_{\nu'}$ for each $\nu' \in \mathcal{N}(\nu)$, which is $1/\beta$ if ν' is in a dense subhypergraph and 0 otherwise. The search of dense subhypergraph can then be formulated as the following discrete optimization problem

$$\begin{aligned} &\underset{z_{\nu'}:\nu' \in \mathcal{N}(\nu)}{\text{argmax}} \sum_{e \in \mathcal{U}_\nu} W(e) \prod_{\nu' \in e} z_{\nu'} + \sum_{\nu' \in \mathcal{N}(\nu)} \mathcal{A}(\nu') z_{\nu'} \\ &\text{s.t.} \quad \sum_{\nu' \in \mathcal{N}(\nu)} z_{\nu'} = 1, \\ &\qquad \forall \nu' \in \mathcal{N}(\nu), \quad z_{\nu'} \in \{0, 1/\beta\}, \end{aligned} \tag{11}$$

in which $\mathcal{U}_\nu$ is the hyperedge set corresponding to the node set $\nu \cup \mathcal{N}(\nu)$, $\mathcal{A}(\nu)$ and $W(e)$ are the weights of nodes in (9) and hyperedges in (10), respectively. The first term in the objective function encourages the inclusion of hyperedges in ν's subhypergraph with larger weights, while the second term penalizes the inclusion

[2] Note that this is different from the degree of the nodes, which specifies how many hyperedges can associate with one node.

of nodes correspond to less reliable couplings indicated by a lower $\mathcal{A}(\nu)$. The first constraint in (11) requires that the subhypergraph should include β nodes, and the second constraint enforces that the label can only take two values.

The optimization problem in (11) is different from the one formulated for single-view multi-target tracking as in [45]. The hypergraph in [45] does not have node weights reflecting uncertainty in forming 3D tracklets. Furthermore, the optimization algorithm in [45] will run inefficiently if it is directly applied to solve (11), due to the large size of our optimization problem. For instance, for 3 camera views with each containing 10 tracklets, a 3-degree STV hypergraph has $10^3 = 1000$ nodes and $\binom{10^3}{3} \approx 1.67 \times 10^8$ hyperedges.

3.4.2 Constructing STV Graph

Motivated by [30], we propose an approximate approach to search dense subhypergraph for efficient optimization of (11), the basic idea is to perform a more efficient search on a graph approximation to the hypergraph. This method thus strikes a balance between the expressiveness of the hypergraph representation and computational efficiency of the graph approximation.

To be specific, we construct a *space-time-view graph* (STV graph) to the STV hypergraph. Then, we search for dense *subgraphs* on STV graph, from which dense subhypergraphs of STV hypergraph can be recovered. Unlike in previous works [26,13], we construct STV graph from STV hypergraph to explicitly capture high-order temporal correlations while maintaining efficacy. We essentially combine the advantages of using a hypergraph to capture high order correlations and the computational efficiency of a graph. We analyze the effectiveness of our approach in § 4.

We construct STV graph by keeping all the nodes in STV hypergraph and sampling from the hyperedges a set of *connection samples* (CSs), $\{\boldsymbol{\pi}_1, \cdots, \boldsymbol{\pi}_i, \cdots\}$. Each CS $\boldsymbol{\pi}_i$ is a set of $k-1$ nodes from STV hypergraph, constructed by traversing each node ν for ξ times, and for each time randomly select other $k-2$ nodes from $\mathcal{N}(\nu) - \{\nu\}$. Obviously, the number of sampled CS pairs is much smaller than the total number of hyperedges. Thus, it is more efficient to search the dense subgraph using sampling strategy than traverse all the hyperedges in [45]. In total, we can obtain $n \cdot \xi$ sampled CSs, where n is the number of nodes in STV hypergraph. For each node in the STV hypergraph, we form a new hypothetical hyperedge with each CS by pooling all nodes together. The scores of all nodes in a CS $\boldsymbol{\pi}_i$ are collected, $\boldsymbol{\mathcal{S}}(\boldsymbol{\pi}_i) = \{\mathcal{S}_1(\boldsymbol{\pi}_i), \cdots, \mathcal{S}_n(\boldsymbol{\pi}_i)\}$. If the j-th node belongs to $\boldsymbol{\pi}_i$, we set $\mathcal{S}_j(\boldsymbol{\pi}_i)$ to a predefined confident score threshold μ. Otherwise, we use

$$\begin{aligned}\mathcal{S}_j(\boldsymbol{\pi}_i) = \lambda_8 \cdot \Psi_{\text{app}}(\boldsymbol{\pi}_i \cup \{\nu_j\}) + \lambda_9 \cdot \Psi_{\text{mot}}(\boldsymbol{\pi}_i \cup \{\nu_j\}) \\ + \lambda_{10} \cdot \Psi_{\text{smo}}(\boldsymbol{\pi}_i \cup \{\nu_j\}).\end{aligned} \tag{12}$$

For each hyperedge e, we define its approximate weight after sampling $\boldsymbol{\pi}_i$ as:

$$W^{[i]}(e) = \max\{W^{[i-1]}(e), \min_{j=\{1,\cdots,k\}} \mathcal{S}_{\nu_j}(\boldsymbol{\pi}_i)\}. \tag{13}$$

The approximate weight of each hyperedge satisfies $0 \leq W^{[1]}(e) \leq \cdots W^{[i]}(e) \leq W(e)$, where i is the number of sampled CSs. As i increases, $W^{[i]}(e)$ approximates $W(e)$ gradually. We do not need to store $W^{[i]}(e)$ of each hyperedge in the sampling procedure. Instead, the scores $\{\boldsymbol{\mathcal{S}}(\boldsymbol{\pi}_1), \cdots, \boldsymbol{\mathcal{S}}(\boldsymbol{\pi}_i), \cdots\}$ are stored. They contain hyperedge weight of the included node and CS pairs, which represent crucial information of our method.

We construct the STV graph $\mathcal{G}^* = (\mathcal{V}^*, \mathcal{E}^*)$ using the scores from the nodes and hyperedges of STV hypergraph. Specifically, $\mathcal{V}^*$ is the same weighted node set as the corresponding STV hypergraph $\mathcal{G}$, and $\mathcal{E}^* = \mathcal{V}^* \times \mathcal{V}^*$ is edge set describing the supports between the nodes. Intuitively, if two nodes belong to the same dense subhypergraph on $\mathcal{G}$, they are expected to simultaneously appear in several hyperedges with large weights. Specifically, for node ν and ν', we set the weight of the edge connecting them in STV graph to reflect the number of hyperedges including both ν and ν' with large weights in the STV hypergraph, based on the scores $\{\boldsymbol{\mathcal{S}}(\boldsymbol{\pi}_1), \cdots, \boldsymbol{\mathcal{S}}(\boldsymbol{\pi}_i), \cdots\}$. We define $\Omega_i = \{\nu_j | \mathcal{S}_j(\boldsymbol{\pi}_i) \geq \mu, j = 1, \cdots, n\}$ to be the reliable node set with the score larger than μ, then the weight of the edge connecting them in STV graph is

$$W^*(\nu, \nu') = \sum_{i=1}^{n \cdot \xi} \sum_{\nu, \nu', \nu_j \in \Omega_i} \rho_i \cdot \mathcal{S}_j(\boldsymbol{\pi}_i), \tag{14}$$

where $\rho_i = \binom{|\Omega_i|-3}{k-3}$ is the total number of hyperedges containing vertices ν and ν' in the original STV hypergraph [3].

3.4.3 Dense Subgraph Search on STV graph

After constructing the STV graph $\mathcal{G}^*$, we search the dense subgraphs on it to complete the tracking task.

[3] The calculation of the number of hyperedges, including nodes ν, ν' and ν_j is a combinational problem, that is to choose $k-3$ nodes from the reliable node set $\Omega_i - \{\nu, \nu', \nu_j\}$. Specifically, we set $\rho_i = 0$ for $|\Omega_i| < 3$, since there does not exist enough nodes to construct a hyperedge in that case.

Similar to (11), the problem is formulated as

$$\begin{aligned}\max_{z_{\nu'}:\nu'\in\widetilde{\mathcal{N}}(\nu)} & \sum_{(\nu_j,\nu')\in\mathcal{U}_\nu^*} W^*(\nu_j,\nu')z_{\nu_j}z_{\nu'} + \sum_{\nu'\in\widetilde{\mathcal{N}}(\nu)} \mathcal{A}(\nu')z_{\nu'} \\ \text{s.t.} \quad & \sum_{\nu'\in\widetilde{\mathcal{N}}(\nu)} z_{\nu'} = 1, \\ & \forall \nu' \in \widetilde{\mathcal{N}}(\nu), \quad z_{\nu'} \in \{0, 1/\beta\},\end{aligned} \tag{15}$$

where $\widetilde{\mathcal{N}}(\nu)$ is the neighborhood of node ν, $\mathcal{U}_\nu^*$ is the edge set corresponding to the node set $\nu \cup \widetilde{\mathcal{N}}(\nu)$, $\hat{\gamma}_\nu$ is the searched dense subgraph corresponding to the node ν, and $\hat{\varpi}_\nu$ is the confident score [4] of the dense subgraph. Optimization problem in (15) is an NP-hard discrete optimization problem [31]. To reduce its complexity, we relax the discrete constraint $z_{\nu'} \in \{0, 1/\beta\}$ to its continuous counterpart $z_{\nu'} \in [0, 1/\beta]$, and thus convert (11) into a continuous optimization problem. Meanwhile, to avoid degeneracy, we require the minimal size of the subgraph to be a fixed number $\beta^* \leq \min_{\nu\in\mathcal{V}} |\widetilde{\mathcal{N}}(\nu)| \leq \beta$, where $\widetilde{\mathcal{N}}(\nu)$ is a set containing the direct neighbors of ν on the graph $\mathcal{G}^*$. Thus, the constraint is further converted as $z_{\nu'} \in [0, 1/\beta^*]$. An efficient method based on pairwise coordinate update given in [31] is used to solve (15) for each node in $\mathcal{G}^*$ to obtain the dense subgraphs $\hat{\Gamma} = \{\hat{\gamma}_i\}_{i=1}^n$ and the corresponding confident score, which will be described as follows.

3.4.4 Optimization Using Pairwise Updates

We adopt the pairwise update algorithm to optimize (15) as in [31]. The formulation in (15) is a constrained optimization problem, we introduce Lagrangian multipliers a, $b_1, \cdots, b_u$, and $c_1, \cdots, c_u$ for each variable z_i, $i \neq \nu$ and $i \in \widetilde{\mathcal{N}}(\nu)$, *i.e.*, where $a \geq 0$ and $b_i \geq 0$ and $c_i \geq 0$ for all $i = 1, \cdots, u$, u is the number of nodes in the neighborhood $\widetilde{\mathcal{N}}(\nu)$. The Lagrangian of the original problem (15) is

$$\begin{aligned}\mathcal{M}(\mathbf{z}, a, b, c) = \frac{1}{2}\cdot f(\mathbf{z}) - a\cdot(\sum_{i=1}^{n} z_i - 1) + \sum_{i,i\neq\nu} b_i \cdot z_i \\ + \sum_{i,i\neq\nu} c_i \cdot (1/\beta - z_i),\end{aligned} \tag{16}$$

where $\mathbf{z} = (z_1, \cdots, z_u)$ is the indicator vector ($z_i = 1$ means the node i is involved in the dense subgraph and $z_i = 0$ means the node i is excluded from the dense subgraph), and β is the number of nodes included in the searched dense subgraph. Similar to [31], we introduce a *reward* score $r_i(\mathbf{z}) = \sum_l W_{il}^* \cdot z_l + \frac{1}{2}\mathcal{A}_i$ at node i reflecting the total weights of node i to other nodes described by the indicator $\mathbf{z}$. Then, we have $\frac{\partial f}{\partial z_i}(\mathbf{z}^*) = 2 \cdot r_i(\mathbf{z}^*)$, *i.e.*, the *reward* score is proportional to the gradient of $f(\mathbf{z})$ at $\mathbf{z}$.

Any local maxima $\mathbf{z}^*$ must satisfy the Karush-Kuhn-Tucker (KKT) conditions [24],

$$\begin{cases} 2\cdot r_i(\mathbf{z}^*) - a + b_i - c_i = 0, i \neq \nu; \\ \sum_{i,i\neq\nu} z_i^* \cdot b_i = 0; \\ \sum_{i,i\neq\nu} c_i \cdot (1/\beta - z_i^*) = 0. \end{cases} \tag{17}$$

Since z_i^*, b_i, and c_i are all nonnegative, and $\sum_{i,i\neq\nu} z_i^* \cdot b_i = 0$, we have two rewritten constraints: (1) $\forall i \neq \nu$, if $z_i^* > 0$, then $b_i = 0$; (2) $\forall i \neq \nu$, if $z_i^* < 1/\beta$, then $c_i = 0$. Thus, for nodes $i \neq \nu$, the KKT conditions can be further reformulated as:

$$r_i(\mathbf{z}^*) = \begin{cases} \leq a/2, & z_i^* = 0; \\ = a/2, & 0 < z_i^* < 1/\beta; \\ \geq a/2, & z_i^* = 1/\beta. \end{cases} \tag{18}$$

Similar to [31], the node set in $\mathcal{G}^*$ can be divided into three disjoint subsets, $\Xi_1(\mathbf{z}) = \{i | z_i = 0\}$, $\Xi_2(\mathbf{z}) = \{i | z_i \in (0, 1/\beta)\}$ and $\Xi_3 = \{i | z_i = 1/\beta\}$.

Using Theorem 1 in [31], we increase a component z_i and decrease z_j to increase $f(\mathbf{z})$, as

$$\hat{\mathbf{z}} = \begin{cases} z_l, & l \neq i, l \neq j; \\ z_l + \alpha, & l = i; \\ z_l - \alpha, & l = j, \end{cases} \tag{19}$$

and define $r_{ij} = W_{ii}^* + W_{jj}^* - 2W_{ij}^*$. Then, we have

$$\begin{aligned}\Delta f(\mathbf{z}) = f(\hat{\mathbf{z}}) - f(\mathbf{z}) &= (W_{ii}^* + W_{jj}^* - 2W_{ij}^*)\alpha^2 \\ &+ 2\Big(\sum_l W_{il}^* \cdot z_l - \sum_l W_{jl}^* \cdot z_l + \frac{1}{2}(\mathcal{A}_i - \mathcal{A}_j)\Big)\alpha \\ &= r_{ij}\cdot\alpha^2 + 2\Big(r_i(\mathbf{z}) - r_j(\mathbf{z})\Big)\alpha.\end{aligned} \tag{20}$$

Note that we can assume $r_i(\mathbf{z}) \geq r_j(\mathbf{z})$, when $r_i(\mathbf{z}) < r_j(\mathbf{z})$, we can exchange i and j To maximize $\Delta f(\mathbf{z})$, α can be calculated based on (20) and the constraints in (15), that is $\alpha = \min(z_j, 1/\beta - z_i)$, if $r_i(\mathbf{z}) > r_j(\mathbf{z})$ and $r_{ij} \geq 0$; $\alpha = \min(z_j, 1/\beta - z_i, \frac{r_i(\mathbf{z}) - r_j(\mathbf{z})}{r_{ij}})$, if $r_i(\mathbf{z}) > r_j(\mathbf{z})$ and $r_{ij} < 0$; and $\alpha = \min(z_j, 1/\beta - z_i)$, if $r_i(\mathbf{z}) = r_j(\mathbf{z})$ and $r_{ij} > 0$. After that, we can compute the local maximizer $\mathbf{z}^*$ of (15) by iteratively using the update strategy (19) and calculating α based on the discussions above from any starting points. We adopt the kNN(o) strategy given in [31] to complete the initialization. A complete analysis of this algorithm can be found in [31].

[4] The confident score $\hat{\varpi}_\nu$ is the function value of the objective in (15) corresponding to the optimal solution.

3.4.5 Conflicts Remove

After identifying the dense subgraphs $\hat{\Gamma}$ and the corresponding confident scores $\hat{\boldsymbol{\varpi}}$ from all nodes, we use some post-processing strategies to filter out the conflicts involved in $\hat{\Gamma}$, *e.g.*, one node may appear in multiple clusters. We use a similar post-processing strategy as in [45]. We first produce an ordered cluster set $\Gamma = \{\gamma_i\}_{i=1}^{n}$ according to the corresponding confident scores $\hat{\varpi}_i$ in descending order from the searched dense subgraphs Γ. Let Γ^* be the dense subgraphs without conflicts. We initialize $\Gamma^* = \emptyset$. For the i-th searched dense subgraph in Γ, *i.e.*, $\gamma_i \in \Gamma$, if $\gamma_i \cap \gamma_j^* = \emptyset$, $\forall j$, $\gamma_j^* \in \Gamma^*$, we add γ_i directly to Γ^*, *i.e.*, $\Gamma^* \leftarrow \Gamma^* \bigcup \{\gamma_i\}$. Otherwise, a greedy approach is designed by directly adding γ_i to the existing clusters γ_j^*, *i.e.*, $\gamma_j^* \leftarrow \gamma_j^* \bigcup \gamma_i$.

3.4.6 Handle Long Videos

For long videos (*e.g.*, with more than 500 frames), constructing the STV hypergraph on all couplings and performing search require large memory and computation. To improve running efficiency, we divide long videos into non-overlapping segments of fixed length. For each segment, the STV hypergraph is constructed and the dense subhypergraph search is performed. We then treat the recovered 3D trajectories as a new coupling, and construct a new STV hypergraph as § 3.3, from which another round of dense subhypergraph search and trajectory linking are performed. This procedure continues until the whole video sequence is merged into a single hypergraph, where the subhypergraph search yields final trajectories. In addition, to avoid exclusion of correct trajectories generated in previous layers, we append the nodes that are not included in the searched dense subgraphs while satisfying the length requirement after the conflict removal step in each layer. That is, we first consider the set of nodes that are excluded from the detected dense subgraph set $\Theta = \{\nu_1, \cdots, \nu_n\} - \bigcup_{i=1}^{\tau} \gamma_i^*$, where $\Gamma^* = \{\gamma_i^*\}_{i=1}^{\tau}$ is the set of detected dense subgraphs. For $\theta_i \in \Theta$, if $\mathcal{L}(\theta_i) \geq \ell$, we add θ_i to Γ^*, *i.e.*, $\Gamma^* \leftarrow \Gamma^* \bigcup \{\theta_i\}$, where ℓ is the preset minimal length of the target trajectory. In this way, the multi-view multi-target tracking task can be completed efficiently.

4 Experiments

4.1 Dataset

We evaluate the performance of our approach and compare with several state-of-the-art methods on the PETS 2009 benchmark dataset [11], which includes three video sequences obtained from multiple synchronized cameras:

- **S2.L1**: low target density, 19 moving pedestrians in 795 frames;
- **S2.L2**: medium target density, 43 pedestrians spreading in 436 frames;
- **S2.L3**: high target density, 44 pedestrians moving together in 240 frames.

These videos represent practical challenges in multi-target tracking, including frequent target occlusions, close targets with similar appearance, and low frame rate (∼7 frame-per-second). In our experiments, we compare tracking results using single and multiple camera views for each of the three PETS 2009 sequences. We use frame detections obtained with the DPM algorithm [10] as the initial tracklets with length one. We use ground truth annotation provided in [34].

4.2 Parameters

We use the following default values for the parameters in our algorithm, which are chosen empirically: in (6): $\lambda_1 = 0.8$, $\lambda_2 = 0.1$, and $\lambda_3 = 0.1$; in (7): $\lambda_4 = 0.01$; in (8): $\lambda_5 = 0.05$; in (9): $\lambda_6 = 1.0$ and $\lambda_7 = 0.01$; in (10): $\lambda_8 = 0.8$, $\lambda_9 = 0.1$, and $\lambda_{10} = 0.1$; the minimal size of subgraph $\beta^* = 2$; the degree of the hyperedge is set as $k = 3$; the score threshold $\mu = 0.2$ and the $\xi = 5$ in CS generation. The minimal length of the target trajectory $\ell = 5$, *i.e.*, each tracked trajectory must contain 5 detections.

4.3 Evaluation Metrics

To quantitatively evaluate the performance, we adopt two CLEAR MOT metrics for multi-target tracking [44]: (i) Multi-Object Tracking Accuracy **(MOTA)**, a consolidated score of false/miss detection rates of ground truth and identity switches of tracked trajectories; and (ii) Multi-Object Tracking Precision **(MOTP)**, the average distance between the tracking results and the ground truth normalized to the hit/miss threshold. The MOTA score is perhaps the widely used figure to evaluate the performance of the tracker, since it combines three errors (*i.e.*, False Negatives (FN), False Positives (FP), and Identity Switches (IDS)) into a single number [27]. To measure the performance of the tracker, we match the locations of tracked targets and the ground truth within a hit/miss distance threshold. To describe the performance of the tracker completely, we plot the MOTA score at the hit/miss distance thresholds varied from 0 to 2 meter. We use the MOTA score at the

Table 2 Single-camera multi-target tracking results in the PETS2009 dataset with single camera view. All methods use the video sequence capture by camera #1 to complete the tracking task. 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 detection results and ground truth 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[%] ↓	FM ↓	IDS ↓
PETS S2.L1	Breitenstein *et al.* [5]*	75.00	60.00	19	-	-	-	-
	Kuo *et al.* [25]*	-	-	19	78.90	**0.00**	23	1
	Yang *et al.* [48]*	-	-	19	89.50	**0.00**	**9**	**0**
	Shi *et al.* [41]*	**96.10**	**81.80**	19	94.70	**0.00**	**6**	4
	Dehghan *et al.* [8]*	90.40	63.12	19	**95.00**	**0.00**	-	**3**
	Berclaz *et al.* [4]	75.05	77.00	19	63.16	**0.00**	63	38
	Andriyenko *et al.* [1]	73.44	78.20	19	52.63	15.79	15	34
	Andriyenko *et al.* [2]	89.05	78.10	19	84.21	**0.00**	21	26
	Pirsiavash *et al.* [39]	81.59	71.80	19	68.42	**0.00**	71	63
	Wen *et al.* [45]	94.43	74.50	19	94.74	**0.00**	16	13
	Hofmann *et al.* [13][5](1)	91.57	80.30	19	94.74	**0.00**	38	52
	Ours(1)	**95.44**	**80.80**	19	**100.00**	**0.00**	10	10
PETS S2.L2	Berclaz *et al.* [4]	41.60	63.00	43	2.33	13.95	416	244
	Andriyenko *et al.* [1]	35.21	69.50	43	9.30	25.58	**91**	**118**
	Andriyenko *et al.* [2]	49.99	64.30	43	9.30	2.33	261	292
	Pirsiavash *et al.* [39]	34.50	**69.90**	43	9.30	4.65	793	2509
	Wen *et al.* [45]	**55.32**	58.40	43	**11.63**	**2.33**	**205**	**141**
	Hofmann *et al.* [13][5](1)	55.11	**70.30**	43	9.30	6.98	303	350
	Ours(1)	**59.15**	65.70	43	**34.88**	**0.00**	259	239
PETS S2.L3	Berclaz *et al.* [4]	39.85	60.60	44	15.91	**18.18**	159	196
	Andriyenko *et al.* [1]	**51.65**	57.20	44	**29.55**	**18.18**	99	153
	Andriyenko *et al.* [2]	46.12	58.90	44	20.45	20.45	126	168
	Pirsiavash *et al.* [39]	49.79	**65.40**	44	**27.27**	25.00	149	172
	Wen *et al.* [45]	50.30	55.10	44	22.73	22.73	**47**	**38**
	Hofmann *et al.* [13][5](1)	46.60	**65.20**	44	20.45	34.09	**64**	**88**
	Ours(1)	**53.36**	59.20	44	25.00	**18.18**	115	100
Average	Berclaz *et al.* [4]	52.17	66.87	-	27.13	10.71	212.67	159.33
	Andriyenko *et al.* [1]	53.43	68.30	-	30.49	19.85	**68.33**	**101.67**
	Andriyenko *et al.* [2]	61.72	67.10	-	37.99	7.59	136.00	162.00
	Pirsiavash *et al.* [39]	55.29	**69.03**	-	35.00	9.88	337.67	914.67
	Wen *et al.* [45]	**66.68**	62.67	-	**43.03**	**8.35**	**89.33**	**64.00**
	Hofmann *et al.* [13][5](1)	64.43	**71.93**	-	41.50	13.69	135.00	163.33
	Ours(1)	**69.32**	68.57	-	**53.29**	**6.06**	128.00	116.33

hit/miss distance threshold 1 meter as the representative score to rank each tracking algorithm.

In addition, we also report the the number of the ground truth trajectories **(GT)**, the ratio of ground truth trajectories that are tracked for more than 80% of total length **(MT)**, the ratio of ground truth trajectories that are tracked for less than 20% of total length **(ML)**, the number of times that a ground truth trajectory is detected with several separate trajectories **(FM)**, and the number of times that a tracked trajectory changes its matched identity **(IDS)** at the hit/miss distance threshold 1 meter. In previous works, tracking performance on the PETS 2009 dataset has been evaluated in either the whole camera view [26,25,48] or a predefined area in the intersection of all views [36,45]. In this work, we use the whole camera view for both single-view and multi-camera evaluations, as it is more relevant to practical tracking scenarios.

4.4 Performance Evaluation and Comparison

We evaluate our method on both single-camera and multi-camera multi-target tracking tasks and discuss the results in the following sections. We use the same input frame detections obtained by the DPM algorithm [10] and the ground truth annotation provided in [34] for all the evaluated methods. Our main concern here is to discount the difference in the detection methods, so as to perform a comprehensive evaluation of our method on the data association part, and provide fair comparison with other methods. On the other hand, because of this setting, performances of many evaluated methods may differ from their published results.

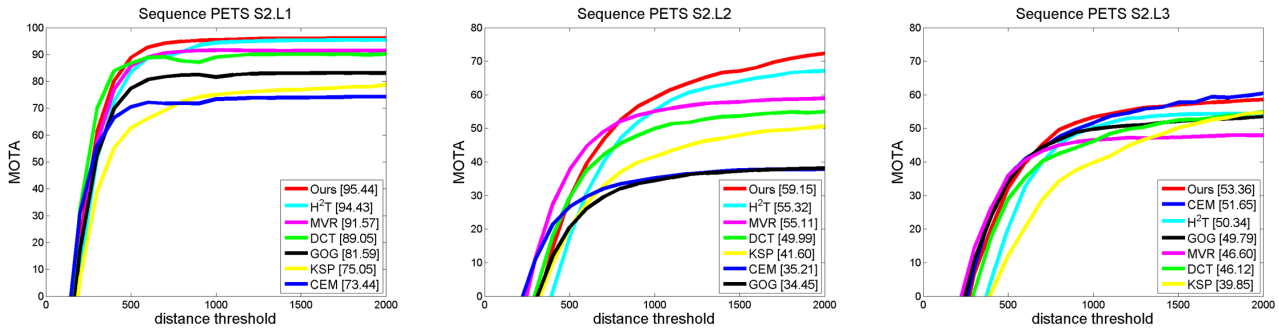


Fig. 3 Plots of the MOTA score of our method and several state-of-the-art methods, *i.e.*, H^{2}T [45], DCT [2], CEM [1], GOG [39], KSP [4], and MVR [13], for single-view multi-target tracking in PETS 2009 S2.L1, S2.L2 and S2.L3 sequences. The performance score for each tracker is presented in the legend.

Table 3 The tracking performance of our method on the 3D MOT Challenge. Refer to the benchmark website for the detailed results of other state-of-the-art methods. 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.

Method	MOTA[%] ↑	MOTP[%] ↑	FAF ↓	MT[%] ↑	ML[%] ↓	FP ↓	FN ↓	IDS ↓	FM ↓
DBN [22]	51.1	61.0	2.3	28.7	17.9	2077	5746	380	418
LPSFM [28]	35.9	54.0	2.3	13.8	21.6	2031	8206	520	601
LP3D [35]	35.9	53.3	4.0	20.9	16.4	3588	6593	580	659
KalmanSFM [38]	25.0	53.6	3.6	6.7	14.6	3161	7599	1838	1686
Ours(1)	-	-	-	-	-	-	-	-	-

4.4.1 Single-Camera Tracking Evaluation

PETS 2009 Dataset. We first evaluate our method in handling the single-view multi-target tracking task on the first view of each sequence in PETS 2009 dataset, *i.e.*, the first view of PETS 2009 S2.L1, S2.L2, and S2.L3 sequences. To demonstrate the favorable performance of our method, we compare it with several state-of-the-art single view multi-target tracking algorithms [5, 25, 48, 41, 8, 4, 13, 1, 2, 39, 45], which is presented in Table 2 and Fig. 3. As previously mentioned, for a fair comparison with the multiple views based methods, we use the same frame detections obtained with the DPM algorithm [10] as the input to all methods.

As shown in Table 2 and Fig. 3, our method performs the best in two sequences, *i.e.*S2.L2 and S2.L3, with more than 3.83% and 1.71% MOTA score gain, and improves more than 2.64% and 10.26% MOTA and MT scores on average. Comparing with the previous methods [5, 25, 48, 41, 8, 4, 13, 1, 2, 39] merely using the similarities between pairwise tracklets, our method exploits the high-order similarities among multiple tracklets in a hypergraph such that full motion information can be used to improve the performance, especially in the crowded scenes, *e.g.*, S2.L2 and S2.L3 sequences.

In addition, compared to [45] the other hypergraph based single-view tracking method, our method also achieves better performance for single-view tracking with 2.64% and 10.26% gain of MOTA and MT scores on average. This is due to the use of calibrated cameras in 3D space, as depth information from the ground-plane assumption improves the motion and trajectory smoothness affinity estimations, which improves overall tracking performance.

MOT 3D Benchmark

4.4.2 Multi-Camera Tracking Evaluation

In Table 4 and Fig. 4, we compare quantitative performance of our method with several state-of-the-art multi-camera multi-target tracking methods on the PETS 2009 sequences. Additional qualitative tracking results can be found in supplementary materials. We include performances from two existing multi-camera multi-target tracking methods [26, 13] as a baseline for comparison [5]. Similar to the single camera case, for a fair comparison with the multiple views based methods, we use the same frame detections obtained with the DPM algorithm [10] as the input to all methods.

We highlight several points regarding the quantitative results in Table 4. Tracking performance is improved by using multi-cameras for all three datasets,

[5] We cannot obtain the source code, binary executable or detection results that can reproduce the performances reported in [13]. As such, results in Table 4 are based on our own implementation of this work, with our best effort to follow the steps given in the original paper. We will make our method and our implementation of [13] along with the tracking results available after the paper decision.

Table 4 Multi-camera multi-target tracking results in the PETS2009 dataset. 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 detection results and ground truth 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	Camera IDs	MOTA[%] ↑	MOTP[%] ↑	GT	MT[%] ↑	ML[%] ↓	FM ↓	IDS ↓
PETS S2.L1	Berclaz *et al.* [18]*	1+3+5+6+8	82.00	56.00	19	-	-	-	-
	Laura *et al.* [26]	1+5	-	-	-	-	-	-	-
	Laura *et al.* [26]	1+7	-	-	-	-	-	-	-
	Laura *et al.* [26]	5+7	-	-	-	-	-	-	-
	Laura *et al.* [26]	1+5+7	-	-	-	-	-	-	-
	Hofmann *et al.* [13][5](2)	1+5	91.89	79.50	19	94.74	**0.00**	29	41
	Hofmann *et al.* [13][5](2)	1+7	-	-	-	-	-	-	-
	Hofmann *et al.* [13][5](2)	5+7	-	-	-	-	-	-	-
	Hofmann *et al.* [13][5](3)	1+5+7	91.66	79.40	19	94.74	**0.00**	31	45
	Ours(2)	1+5	**95.51**	**80.60**	19	**100.00**	**0.00**	**12**	**14**
	Ours(2)	1+7	-	-	-	-	-	-	-
	Ours(2)	1+7	-	-	-	-	-	-	-
	Ours(3)	1+5+7	**95.08**	**79.80**	19	**100.00**	**0.00**	**13**	**13**
PETS S2.L2	Hofmann *et al.* [13][5](2)	1+2	58.97	**65.80**	43	25.56	2.33	288	385
	Hofmann *et al.* [13][5](3)	1+2+3	58.85	**66.00**	43	30.23	2.33	293	388
	Laura *et al.* [26]	1+2	-	-	-	-	-	-	-
	Laura *et al.* [26]	1+2+3	-	-	-	-	-	-	-
	Ours(2)	1+2	**67.00**	61.50	43	**51.16**	**0.00**	**239**	**239**
	Ours(3)	1+2+3	**65.24**	61.80	43	**44.19**	**0.00**	**246**	**249**
PETS S2.L3	Hofmann *et al.* [13][5](2)	1+2	**54.39**	**60.20**	44	25.00	25.00	**67**	**106**
	Hofmann *et al.* [13][5](3)	1+2+4	49.79	**63.00**	44	**29.55**	25.00	**80**	123
	Laura *et al.* [26]	1+2	-	-	-	-	-	-	-
	Laura *et al.* [26]	1+2+4	-	-	-	-	-	-	-
	Ours(2)	1+2	**57.06**	59.30	44	**38.64**	**15.91**	120	129
	Ours(3)	1+2+4	**54.39**	54.90	44	**29.55**	**20.45**	99	**92**

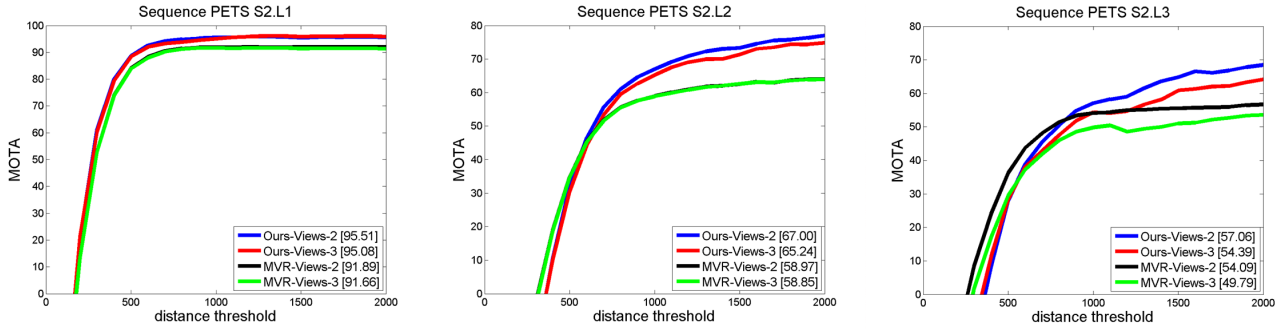


Fig. 4 Plots of the MOTA score of our method and the method MVR [13] for multi-camera multi-target tracking in PETS 2009 S2.L1, S2.L2 and S2.L3 sequences. The performance score for each tracker is presented in the legend.

where the performance gains for videos with higher target densities (S2.L2 and S2.L3) are particularly significant. This is due to the complementary information provided from multiple camera views, which helps to resolve appearance ambiguity and occlusions. However, performance gain with multi-cameras decreases in the cases of low target density (S2.L1), where single-camera tracking already saturates the performance metric. Yet, as pointed out in [26] and [13], further increasing the number of camera views does not usually lead to a monotonic increase in performances. This is due to errors in camera calibration that lead to inaccuracies in the mapping function ϕ^v. These errors result in incorrect associations that accumulate with increasing frame detections in multiple views, and lead to incorrect couplings.

In comparison with two state-of-the-art multi-camera multi-target tracking algorithms [26,13] based on associating temporally adjacent pairs of 3D tracklets, our method achieves better performance as reflected by higher MOTA and MT scores and relative lower IDS and FM scores. This shows the effectiveness of using higher-order temporal correlations between couplings encoded by STV hypergraph, which greatly reduces the association ambiguities, indicated by the lower IDS and FM scores.

Table 5 Effect of different components in the proposed 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	Camera IDs	MOTA[%] ↑	MOTP[%] ↑	GT	MT[%] ↑	ML[%] ↓	FM ↓	IDS ↓
PETS S2.L1	Hofmann *et al.* [13][5](2)	1+5	91.89	79.50	19	94.74	0.00	29	41
	Ours-P(2)	1+5	96.40	80.80	19	100.00	0.00	10	6
	Ours(2)	1+5	95.51	80.60	19	100.00	0.00	12	14
PETS S2.L2	Hofmann *et al.* [13][5](2)	1+2	58.97	65.80	43	25.56	2.32	288	385
	Ours-P(2)	1+2	63.50	61.70	43	51.16	0.00	252	249
	Ours(2)	1+2	67.00	61.50	43	51.16	0.00	239	239
PETS S2.L3	Hofmann *et al.* [13][5](2)	1+2	54.39	60.20	44	25.00	25.00	67	106
	Ours-P(2)	1+2	57.60	59.40	44	31.82	13.63	122	120
	Ours(2)	1+2	57.06	59.30	44	38.64	15.91	120	129
Average	Hofmann *et al.* [13][5](2)	1+5	68.42	**68.50**	-	48.43	9.11	**128.00**	177.33
	Ours-P(2)	1+5	**72.50**	**67.30**	-	**60.99**	**4.54**	**128.00**	**125.00**
	Ours(2)	1+5	**73.19**	67.13	-	**63.27**	**5.30**	**123.67**	**127.33**

4.5 Discussion

Effectiveness of dense subgraphs and hypergraph representation. To have a detailed understanding of the contribution of each component of our method, we construct a baseline tracker that uses pairwise correlation of 3D couplings (tracklets) as in [13], and apply the dense subgraph search [31] as the tracking solution. We compare this baseline algorithm (marked as *Ours-P(2)*) in Table 5 with the full method using hypergraph (Ours(2)) and that of [13] for tracking with two camera views. This baseline algorithm improves 4.17% MOTA and 12.55% MT scores, and reduces 4.57% ML and 29.5% IDS scores on average performance compared to the network flow optimization based method proposed by Hofmann *et al.* [13], showing that our formulation of tracking as searching subgraphs or sub-hypergraphs is important in improving the overall performance. In addition, our full method further improves 0.7% MOTA and 2.28% MT scores, and reduces 3.38% FM score on average performance in comparison with the baseline algorithm, which demonstrates the advantage of using hypergraph as a representation for multi-target tracking.

Effect of the hyperedge degree. We further perform experiments to validate the influence of the hyperedge degree k on tracking performance. We construct STV hypergraph with $k = 2, \cdots, 8$ while keeping all other parameters fixed, and report the changes in the MOTA score in Figure 5. As these results show, tracking performance decreases as k increases when $k \geq 6$, because STV hypergraph with hyperedge degree that is too high fails to describe the motion pattern well enough, for the case of targets moving in drastically different speed and directions. Also, these results confirm our choice of $k = 3$ as it leads to the overall optimal empirical performance.

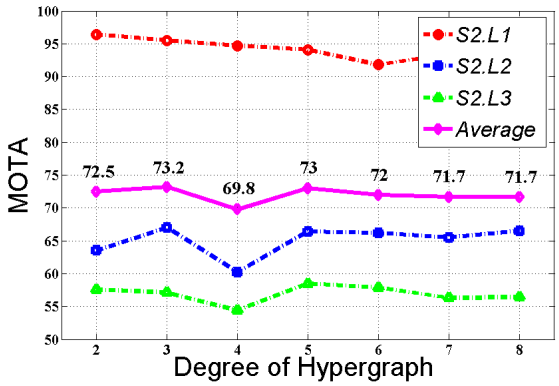


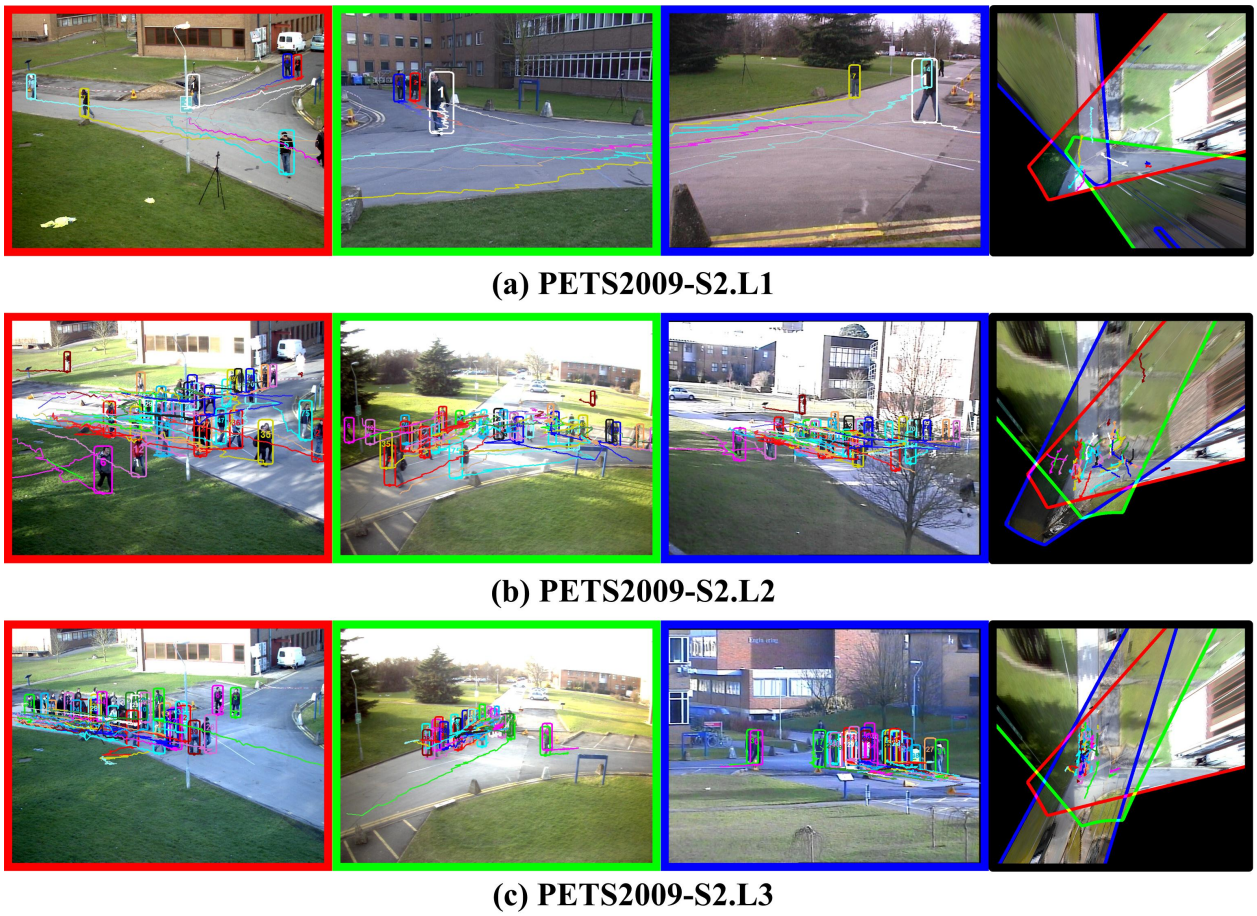
Fig. 5 Effect of the degree of hypergraph k (number of nodes associated with each hyperedge) on the tracking performance. Note that the hypergraph degenerates to a graph when $k = 2$.

4.6 Running Time

In addition to its performance, our method also affords efficient running time. Table 6 reports the running time measured in frame-per-second (fps) on the PETS2009 dataset over 1-3 camera views after given the detection results. These running time is based on an implementation with unoptimized C++ code, single thread execution on a workstation with Intel 2.67GHz CPU and 128 GB memory. Note that, as shown in Table 4, our method runs faster in S2.L3 than S2.L2 over all camera views. Although S2.L3 has higher target density than S2.L2, the highly complex pedestrian interactions in S2.L2 result in more hyperedges are included in STV hypergraph. Thus, the dense subhypergraph search is slower in S2.L2 than that in S2.L3.

5 Conclusion

Incorporating multiple cameras is an effective solution to improve the performance and robustness of multi-



(a) PETS2009-S2.L1

(b) PETS2009-S2.L2

(c) PETS2009-S2.L3

Fig. 6 STV hypergraph hyper-graph tracking results on PETS2009 videos. We show results using three camera views and two different frames, as well as the top down view of the overall tracking results. The colors of the borders of frame captures correspond to the camera views of the same color in the bottom row. Full video results are provided in the supplementary materials. This figure is better viewed in color.

Table 6 The running speed of our method in different sequences with different camera views. Frame-per-second (fps) is used to measure the speed of the tracker. In comparison, we also show the frame rate of the original PETS2009 videos.

Sequence	1-View	2-Views	3-Views	PETS Frame Rate
S2.L1	**30.6**	**16.8**	**10.9**	7.0
S2.L2	**7.1**	1.8	0.9	7.0
S2.L3	**9.0**	3.2	2.5	7.0

target tracking to occlusion and appearance ambiguities. In this paper, we propose a new multi-camera multi-target tracking method based on a space-time-view hypergraph that encodes higher-order constraints (*i.e.*, beyond pairwise relations) on 3D geometry, appearance, motion continuity, and trajectory smoothness among 2D tracklets within and across different camera views. We solve tracking in each single view and reconstruction of tracked trajectories in 3D environment simultaneously by formulating the problem as an efficient search of dense subhypergraphs on the space-time-view hypergraph using a sampling based approach. Experimental results on the PETS 2009 benchmark dataset demonstrate that our method outperforms state-of-the-art methods in both single-view and multi-camera multi-target tracking, while achieving close to real-time running efficiency. We also provide experimental analysis of the influence of various aspects of our method to the final tracking performance.

There are several directions we would like to further improve the current work. First, the current method relies on the knowledge of camera parameters, it is useful to be able to recover camera parameters along with multi-target tracking and 3D reconstruction. This is possible with recent advances that recover camera parameters from multiple image sequences [21,23]. Second, the current method also assumes a static camera, and a more challenging scenario that we will explore is when some views are from cameras with ego motion (*e.g.*, PTZ cameras). Also, there exist alternative formulations of the subhypergraph search algorithm such as those based on hypergraph Laplacians [53]. Subsequently we would like to investigate and compare differ-

ent optimization strategies to solve the dense subhypergraph search problem. Last, we would like to test push the limit of multi-camera tracking methods, and extend similar methods to scenarios where camera views have less overlapping.

References

1. Andriyenko, A., Schindler, K.: Multi-target tracking by continuous energy minimization. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1265–1272 (2011)
2. Andriyenko, A., Schindler, K., Roth, S.: Discrete-continuous optimization for multi-target tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1926–1933 (2012)
3. Attanasi, A., Cavagna, A., Castello, L.D., Giardina, I., Jelic, A., Melillo, S., Parisi, L., Pellacini, F., Shen, E., Silvestri, E., Viale, M.: GReTA – a novel global and recursive tracking algorithm in three dimensions. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(1), 1 (2015)
4. Berclaz, J., Fleuret, F., Türetken, E., Fua, P.: Multiple object tracking using k-shortest paths optimization. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(9), 1806–1819 (2011)
5. Breitenstein, M.D., Reichlin, F., Leibe, B., Koller-Meier, E., Gool, L.J.V.: Online multi-person tracking-by-detection from a single, uncalibrated camera. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(9), 1820–1833 (2011)
6. Brendel, W., Amer, M.R., Todorovic, S.: Multiobject tracking as maximum weight independent set. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1273–1280 (2011)
7. Chari, V., Lacoste-Julien, S., Laptev, I., Sivic, J.: On pairwise costs for network flow multi-object tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 5537–5545 (2015)
8. Dehghan, A., Tian, Y., Torr, P.H.S., Shah, M.: Target identity-aware network flow for online multiple target tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1146–1154 (2015)
9. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(4), 743–761 (2012)
10. Felzenszwalb, P.F., McAllester, D.A., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (2008)
11. Ferryman, J.M., Shahrokni, A.: PETS2009: Dataset and challenge. In: Winter-PETS, pp. 1–6 (2009)
12. Fleuret, F., Berclaz, J., Lengagne, R., Fua, P.: Multicamera people tracking with a probabilistic occupancy map. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(2), 267–282 (2008)
13. Hofmann, M., Wolf, D., Rigoll, G.: Hypergraphs for joint multi-view reconstruction and multi-object tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (2013)
14. Hong, L., Cui, N.: An interacting multipattern joint probabilistic data association (imp-jpda) algorithm for multitarget tracking. Signal Processing **80**(8), 1561–1575 (2000)
15. Huang, C., Li, Y., Nevatia, R.: Multiple target tracking by learning-based hierarchical association of detection responses. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(4), 898–910 (2013)
16. Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. International Journal of Computer Vision **29**(1), 5–28 (1998)
17. Izadinia, H., Saleemi, I., Li, W., Shah, M.: $(MP)^2T$: Multiple people multiple parts tracker. In: Proceedings of European Conference on Computer Vision, pp. 100–114 (2012)
18. Jerome, B., Francois, F., Pascal, F.: Multiple Object Tracking using Flow Linear Programming. In: Winter-PETS (2009)
19. Jiang, H., Fels, S., Little, J.J.: A linear programming approach for multiple object tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2007)
20. Khan, Z., Balch, T.R., Dellaert, F.: MCMC-based particle filtering for tracking a variable number of interacting targets. IEEE Transactions on Pattern Analysis and Machine Intelligence **27**(11), 1805–1918 (2005)
21. Kim, J., Dai, Y., Li, H., Du, X., Kim, J.: Multi-view 3D reconstruction from uncalibrated radially-symmetric cameras. In: Proceedings of IEEE International Conference on Computer Vision, pp. 1896–1903 (2013)
22. Klinger, T., Rottensteiner, F., Heipke, C.: Probabilistic multi-person tracking using dynamic bayes networks **II-3/W5**, 435–442 (2015)
23. Kostrikov, I., Horbert, E., Leibe, B.: Probabilistic labeling cost for high-accuracy multi-view reconstruction. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1534–1541 (2014)
24. Kuhn, W., Tucker, A.: Nonlinear programming. In: Proceedings of 2nd Berkeley Symposium, pp. 481–492 (1951)
25. Kuo, C.H., Nevatia, R.: How does person identity recognition help multi-person tracking? In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1217–1224 (2011)
26. Leal-Taix, L., Pons-Moll, G., Rosenhahn, B.: Branch-and-price global optimization for multi-view multi-object tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1987–1994 (2012)
27. Leal-Taixé, L., Milan, A., Reid, I.D., Roth, S., Schindler, K.: Motchallenge 2015: Towards a benchmark for multi-target tracking. CoRR **abs/1504.01942** (2015)
28. Leal-Taixé, L., Pons-Moll, G., Rosenhahn, B.: Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In: Workshops in Conjunction with IEEE International Conference on Computer Vision, pp. 120–127 (2011)
29. Leven, W.F., Lanterman, A.D.: Unscented kalman filters for multiple target tracking with symmetric measurement equations. IEEE Transaction on Automatic Control **54**(2), 370–375 (2009)
30. Liu, H., Yan, S.: Efficient structure detection via random consensus graph. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 574–581 (2012)
31. Liu, H., Yang, X., Latecki, L.J., Yan, S.: Dense neighborhoods on affinity graph. IJCV **98**(1), 65–82 (2012)

32. Liu, Y., Li, H., Chen, Y.Q.: Automatic tracking of a large number of moving targets in 3d. In: Proceedings of European Conference on Computer Vision, pp. 730–742 (2012)
33. Marchesotti, L., Marcenaro, L., Ferrari, G., Regazzoni, C.S.: Multiple object tracking under heavy occlusions by using kalman filters based on shape matching. In: Proceedings of IEEE International Conference on Image Processing, pp. 341–344 (2002)
34. Milan, A.: Continuous energy minimization tracker. http://www.milanton.de/contracking/index.html
35. Milan, A., Leal-Taixé, L., Schindler, K., Roth, S., Reid, I.D.: Multiple object tracking benchmark: 3d mot 2015. https://motchallenge.net/results/3D_MOT_2015/
36. Milan, A., Roth, S., Schindler, K.: Continuous energy minimization for multitarget tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **36**(1), 58–72 (2014)
37. Ojala, T., Pietikäinen, M., Mäenpää, T.: Gray scale and rotation invariant texture classification with local binary patterns. In: Proceedings of European Conference on Computer Vision, pp. 404–420 (2000)
38. Pellegrini, S., Ess, A., Schindler, K., Gool, L.J.V.: You'll never walk alone: Modeling social behavior for multi-target tracking. In: Proceedings of IEEE International Conference on Computer Vision, pp. 261–268 (2009)
39. Pirsiavash, H., Ramanan, D., Fowlkes, C.C.: Globally-optimal greedy algorithms for tracking a variable number of objects. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1201–1208 (2011)
40. Reid, D.B.: An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control **24**, 843–854 (1979)
41. Shi, X., Ling, H., Hu, W., Yuan, C., Xing, J.: Multi-target tracking with motion context in tensor power iteration. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 3518–3525 (2014)
42. Shu, G., Dehghan, A., Oreifej, O., Hand, E., Shah, M.: Part-based multiple-person tracking with partial occlusion handling. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1815–1821 (2012)
43. Smith, K., Gatica-Perez, D., Odobez, J.M.: Using particles to track varying numbers of interacting people. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 962–969 (2005)
44. Stiefelhagen, R., Bernardin, K., Bowers, R., Garofolo, J.S., Mostefa, D., Soundararajan, P.: The CLEAR 2006 evaluation. In: CLEAR, pp. 1–44 (2006)
45. Wen, L., Li, W., Yan, J., Lei, Z., Yi, D., Li, S.Z.: Multiple target tracking based on undirected hierarchical relation hypergraph. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 3457–3464 (2014)
46. Wu, Z., Hristov, N.I., Kunz, T.H., Betke, M.: Tracking-reconstruction or reconstruction-tracking? comparison of two multiple hypothesis tracking approaches to interpret 3D object motion from several camera views. In: Proceedings of the IEEE Workshop on Motion and Video Computing, pp. 1–8 (2009)
47. Wu, Z., Kunz, T.H., Betke, M.: 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, pp. 1185–1192 (2011)
48. Yang, B., Nevatia, R.: 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, pp. 1918–1925 (2012)
49. Yang, B., Nevatia, R.: An online learned CRF model for multi-target tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 2034–2041 (2012)
50. Yang, M., Liu, Y., Wen, L., You, Z., Li, S.Z.: A probabilistic framework for multitarget tracking with mutual occlusions. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2014)
51. Yu, Q., Medioni, G.G.: Multiple-target tracking by spatiotemporal monte carlo markov chain data association. IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(12), 2196–2210 (2009)
52. Zhang, L., Li, Y., Nevatia, R.: Global data association for multi-object tracking using network flows. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
53. Zhou, D., Huang, J., Schölkopf, B.: Learning with hypergraphs: Clustering, classification, and embedding. In: Advances in Neural Information Processing Systems, pp. 1601–1608 (2006)