GMMCP Tracker: Globally Optimal Generalized Maximum Multi Clique Problem for Multiple Object Tracking

Afshin Dehghan Shayan Modiri Assari Mubarak Shah
Center for Research in Computer Vision, University of Central Florida
{adehghan, smodiri, shah}@cs.ucf.edu

Abstract

Data association is the backbone to many multiple object tracking (MOT) methods. In this paper we formulate data association as a Generalized Maximum Multi Clique problem (GMMCP). We show that this is the ideal case of modeling tracking in real world scenario where all the pairwise relationships between targets in a batch of frames are taken into account. Previous works assume simplified version of our tracker either in problem formulation or problem optimization. However, we propose a solution using GMMCP where no simplification is assumed in either steps. We show that the NP hard problem of GMMCP can be formulated through Binary-Integer Program where for small and medium size MOT problems the solution can be found efficiently. We further propose a speed-up method, employing Aggregated Dummy Nodes *for modeling occlusion and miss-detection, which reduces the size of the input graph without using any heuristics. We show that, using the speed-up method, our tracker lends itself to real-time implementation which is plausible in many applications. We evaluated our tracker on six challenging sequences of Town Center, TUD-Crossing, TUD-Stadtmitte, Parking-lot 1, Parking-lot 2 and Parking-lot pizza and show favorable improvement against state of art.*

1. Introduction

Tracking is a fundamental problem in computer vision and for decades researchers have tried to solve this problem. Every year many trackers are proposed and results in many dataset have already reached their ceilings. The big question is: *"Do we need yet another tracker?"*. Looking back at recent tracking papers, despite great performance reported on specific scenarios, still many challenges remain unsolved including occlusion, abrupt motion, appearance changes and, etc. This leaves the room open for new trackers which can better deal with tracking challenges in various scenarios. The aforementioned aspects mostly affect the performance of the pre-trained object detectors which are used to generate the input to data-association based tracking algorithms. Reviewing the literature, most data association methods have considered a simplified version of the problem and focused on approximate inference methods which can be solved efficiently. On the other side, those algorithms which incorporate more accurate formulation of tracking scenario in real world suffer from greedy optimizations and local minima.

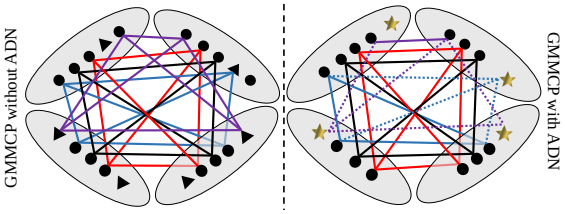


Figure 1. The two versions of the input graph used in our tracking. The graph on the left uses regular dummy nodes (shown with triangles) for occlusion handling and the speed-up version of our tracker uses only one dummy node per cluster called aggregated-dummy-node (shown with stars). Our method finds all the cliques in a graph that maximize the score function simultaneously. In this example 4 cliques are found, each shown in a different color. Each circle represents one tracklet.

In this paper we take a different direction and propose a method which does not involve any simplification in problem formulation nor in optimization. Our method has several advantages: First, it mimics the real world tracking scenario precisely by incorporating all temporal pairwise relationship in a batch of frames, i.e the graph is k-partite complete. Second, we formulate the proposed graph theoretic problem using Binary-integer Program without simplifying the original problem. Third, it allows including high-order relationship between targets in our cost function. Fourth, it can robustly handle short/long-term occlusion. And, finally it lends itself to real-time implementation on a desktop computer. Next we shall critique some important previous data association methods in more detail and highlight how the proposed method differs from them.

Data association is the core of multiple object tracking

algorithms. One group of data-association techniques are temporally local. Bipartite matching is probably the most popular method in this category and exact solution to that can be found using Hungarian algorithm [6]. On the other hand, the popularity of the global data association based tracking methods have recently increased due to their ability to better deal with the challenges caused by noisy detection inputs. In global data association the temporal-locality during optimization is increased which means that the optimization is done over a batch of frames instead of just two/few consecutive frames [20, 21, 8, 27]. This allows incorporating more global properties of targets during optimization. Reviewing the literature we can divide such algorithms into two main groups. One group assumes a simplified version of the problem where only relationship between observations in consecutive frames are taken into account. These methods are mostly followed by an optimization for which there exists an exact solution. Such methods are computationally efficient and plausible for many applications.

Network flow algorithm [20, 21, 8] is an example of such approaches. In network flow, a directed acyclic graph (DAG) is formed given the detection hypotheses in each frame, and the solution is found through minimum-cost maximum-flow algorithm. Zhang et al. [20] showed that the exact solution to network flow problem can be found in polynomial time through push relabel algorithm. Pirsiavash et al. in [21] showed that a high quality sub-optimal solution can be found using dynamic programing.

Different variations of network flow are also used in MOT recently [12, 10, 22]. Authors in [10] incorporate constant velocity motion model in network flow graph and proposed a Lagrangian relaxation solution to min-cost max-flow problem. The work of [22] presents a multi-commodity network flow to better incorporate the appearance consistency between group of people during tracking. Authors in [12] use a variation of multi-commodity flow graph in an inner loop of a structured learning tracker and solve detection and data association simultaneously for multiple objects. However, this simplification of the original problem comes with a price. Limiting the cost calculated for a track to observations in consecutive frames make tracker prone to ID-switches.

The second group of data association methods assume no simplification in problem formulation and consider a model which is closer to the tracking scenario in real world. However, due to the complexity of their models, the proposed solutions are approximate [3, 4, 27]. Andriyenko et al. in [3] formulate tracking in a non-convex optimization framework where the goal is to fit a set of trajectories to the data which best satisfies some constraints mimicking tracking in real world scenarios. Even though the model is closer to the real world tracking scenario compared to network flow, the solution found to the non-convex function is prone to local minima.

GMCP tracker proposed in [27] considered a more complete representation of the tracking problem, where all the pairwise relationships between detections in the temporal span of a video is considered. This makes the input to the data association to be a complete k-partite graph. In a k-partite complete graph a track of a person will form a clique (a subgraph in which all the nodes are connected to each other), thus the MOT can be formulated as constraint maximum weight clique problem. In [27], a cost is assigned to each clique and the one that maximizes the score function is selected as the best clique(track). However, as mentioned earlier the solver used in these methods is sub-optimal. First GMCP tracker does not follow a joint optimization for all the tracks simultaneously and finds the tracks one by one. In addition, the proposed solver follows a greedy local neighborhood search which is prone to local minima. Moreover the heuristic line fitting approach for removing outliers make the tracks prone to ID-switch especially when people are walking close to each other. Some failure cases of GMCP tracker are shown in Figure 2.

In this paper we propose a multiple object tracking approach which in contrast to previous works does not involve any simplification neither in *problem formulation* nor in the *optimization*. We formulate tracking as a *Generalized Maximum Multi Clique Problem* which have not been explored before. Our graph incorporates the pairwise relationship between all the observations in a batch of frames and our cost function allows incorporating higher order costs between all the candidates in a track. Although we prove that GMMCP is an NP hard problem, we do not follow an approximate solution for it. We formulate the NP hard problem through Binary Integer Program (BIP), and show that the solution can be found accurately for small and medium size MOT problems. Our tracking algorithm addresses the aforementioned problems of [27] by not only finding the tracks jointly for all the objects but also providing an accurate solution without relaxing the original problem.

In addition, we propose an approach for speeding up the proposed BIP solution with a more efficient occlusion handling strategy. Our speed-up solution formulates GMMCP as a Mixed-Binary Integer Program that reduce the computational complexity significantly without any loss in the performance. We show that for small and medium size multiple object tracking problem the solution to the NP hard problem can still be found efficiently using our pipeline and favorable results against state-of-art can be achieved. Our experiments involve challenging sequences and the results support the effectiveness of the proposed pipeline. We show qualitatively that our method can handle very long term occlusions, sometimes up to 150 frames, and can preserve the identity of targets better than previous work.

The rest of the paper is organized as follow: In Section 2

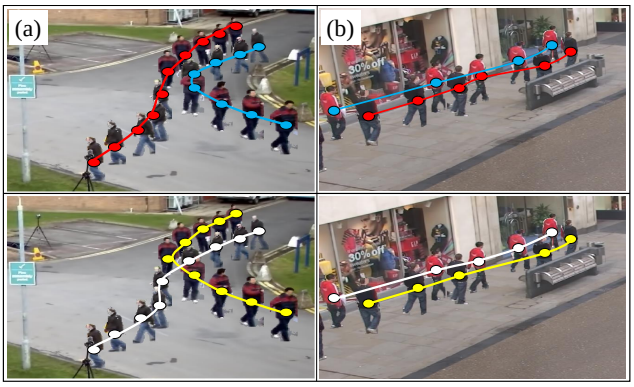


Figure 2. Example of failure cases of the GMCP tracker in [27] (top row).(a) indicates the effect of joint optimization where two targets with similar appearance are walking in the scene. In GMCP first the red track that has a very high score is selected and later the blue track is picked, which has a very low similarity score. Both tracks in this case are wrong. (b) Shows two targets walking in parallel for 6 frames with very different appearance but spatially close (each target has one miss detection). The red track is selected first which includes an ID-Switch. The reason is that, the outliers selection is based on spatial location only and it is sensitive to RANSAC parameters. Bottom row shows the groundtruth tracks

we review our tracking pipeline and introduce the Generalized Maximum Multi Clique Problem. Next we discuss how tracking is formulated using GMMCP (Section 3) and propose our Binary-Integer Program solution. In section 4 we discuss our proposed algorithm to speed up the optimization introduced in Section 3.1. The appearance and motion affinities used in our tracking are discussed in section 5. The experimental results are presented in 6 and finally section 7 concludes the paper.

2. Proposed Framework

We adopt a two layer tracking framework in which tracks with shorter length are combined in each layer to form longer tracks. We start with a set of *low-level tracklets* for different temporal segments of a video. To further increase the reliability of these low-level tracklets we limit their length to a maximum of 10 frames and the ones shorter than 5 frames are discarded. Later the low-level tracklets in different segments $(4-6)$ are used to create the input to our *GMMCP* tracker. This will form the first layer of our tracking where the output is a set of *mid-level tracklets*. The mid-level tracklets are later used as input to the next layer to form the final trajectories. Below we shall introduce GMMCP and we show that it is NP hard.

Generalized Maximum Multi Clique Problem. Given a k-partite complete graph, where there is an edge between every pair of nodes that do not belong to the same cluster (in our case cluster represents tracklets in a segment of a video); the objective is to find a sub-graph which forms K cliques, in which the sum of edges of the cliques are maximized and exactly K nodes are selected from every cluster. To give a more formal definition, the input to the GMMCP is a graph $G(V, E, W)$, where V, E and W are the nodes, edges and their corresponding weights. V is divided into a set of disjoint clusters where v_i^j defines the i^{th} node in the j^{th} cluster. The goal is now to pick a set of K *cliques* by selecting exactly K nodes from each cluster that maximize the total score. The closest problem to ours is Generalized Maximum Clique Problem (GMCP) where the aim is to find only *one clique* with maximum score. It is shown that GMCP is NP hard [16, 19] and a set of approximate solutions, such as multi-greedy heuristics and local search, are proposed to solve GMCP [27, 2]. In order to prove that Generalized Maximum K clique problem is NP hard we use reduction and show that GMCP can be reduced to GMMCP. We will show below that GMMCP is also hard to solve:

Proposition. *Generalized Maximum K Clique Problem* is NP hard.

Proof. Consider a graph with h clusters where we aim to find the clique with maximum score. We add $K-1$ nodes to each cluster. The added nodes in different clusters are connected to each other with $\inf$ edge weights and the edges connecting the added nodes to the nodes of the original graph have the weight zero. Now the solution to Generalized Maximum K clique problem in the new graph is equal to the solution to GMCP in original graph, after excluding the $K-1$ $\inf$ cliques. So GMCP is reduced to GMMCP and we have shown that GMMCP is at least as hard as GMCP, thus it is NP-hard.

3. Tracking using Generalized Maximum Multi Clique Problem

Creating Input Graph. The input to our tracker is a k-partite complete graph where there is an edge between every pair of nodes that are not in the same cluster. For the first layer we divide the video into segment of 10 frames and each segment defines one cluster. The nodes of our graph in the first layer are low-level tracklets (mid-level tracklets are used to create the input to the second layer) found in each segment with maximum length of 10 frames and minimum length of 5 frames. These low-level tracklets are found using a simple overlap criteria where bounding boxes that overlap more than 60% in consecutive frames are connected. It is worth to mention that tracklets have been previously used as reliable inputs in many tracking algorithm [26, 11, 13, 22]. In our method, tracklets not only help reducing the computational complexity but also allow incorporating motion similarity into edge cost of our graph. Because the edges in our graph will connect more than two detection hypotheses which are essential for encoding a motion cost. Thus each edge in our graph, G, is assigned a weight which incorporates both appearance and motion of

the target. We later show in Section 5 how these weights are calculated.

3.1. Solving GMMCP using Binary Integer Program

We formulate GMMCP as a Binary Integer Program. To the best our knowledge this problem is not solved before. Any binary integer program can be formulated as follows:

$$\begin{cases} \text{maximize} & \mathbf{C}^T\mathbf{x}, \\ \text{subject to} & \mathbf{Ax} = \mathbf{b} \text{ and } \mathbf{Mx} \leq \mathbf{n}. \end{cases} \tag{1}$$

The objective function to maximize is $\mathbf{C}^T\mathbf{x}$, where $\mathbf{C}$ is the weight matrix and $\mathbf{x}$ is boolean column vector. $\mathbf{Ax} = \mathbf{b}$ and $\mathbf{Mx} \leq \mathbf{n}$ define the equality and inequality constraints respectively. For every node and edge in the graph there is binary variable in vector $\mathbf{x}$. Let us define the v_i^j as the binary variable for each node (the i^{th} node in the j^{th} cluster) and e_{ij}^{mn} to be the binary variable for the edge between the nodes v_i^j and v_m^n. In order to guarantee a feasible solution to GMMCP, $\mathbf{x}$ needs to satisfy three constraints.

The first **constraint** enforces the sum of nodes in each cluster to be equal to K, which is the number of cliques we need to find (We explain later in the experiment section how we set the number K).

$$\{\forall j | 1 \leq j \leq h\} : \sum_{i=1}^{l} v_i^j = K, \tag{2}$$

where h is the number of clusters and l is the number of nodes within that cluster.

The second **constraint** ensures that, if a node is selected then $(h-1)$ of its edges should be included in the solution. This is because of the fact that in each clique, one node from each cluster is included.

$$\sum_{j=1}^{h}\sum_{i=1}^{l} e_{mn}^{ij} = v_m^n.(h-1).\{\forall m, n | 1 \leq n \leq h, 1 \leq m \leq l\} \tag{3}$$

Finally we need the third **constraint** to ensure that the solution found by $\mathbf{x}$ will form a clique.

$$e_{ij}^{i'j'} + e_{i'j'}^{i''j''} \leq 1 + e_{i''j''}^{ij}. \quad \{\forall e_{ij}^{mn} \in E\} \tag{4}$$

Constraints in Equations 2 and 3 are used together to form the equality constraint defined by matrix $\mathbf{A}$ and vector $\mathbf{b}$. Matrix $\mathbf{M}$ and vector $\mathbf{n}$ are also constructed so that the constraint in Equation 4 is satisfied. The combination of these three constraints will ensure that $\mathbf{x}$ will provide a valid solution to the GMMCP. Once $\mathbf{x}$ is found, each clique will represent a track of a person.

3.2. Occlusion Handling using Dummy Nodes

The solution to GMMCP is a set of cliques where one node from each cluster is selected in each clique. In our formulation each node will represent a tracklet of person which may not necessarily be present in all the frames (cluster) or may be occluded or miss-detected. In order to avoid selecting irrelevant nodes in a track of a person, we introduce an additional set of nodes in each cluster called *dummy nodes*. Dummy nodes are treated the same as the rest of the nodes in the graph with only one difference. The weights of the edges connected to each dummy node are fixed to a pre-defined value of c_d. Our dummy nodes will ensure that the tracks for each person will be free of outliers. In other words, when there is no confident tracklet for a clique in a particular cluster, a dummy node from that cluster is selected. In figure 1 (left), 4 cliques are selected, each shown in different color and dummy nodes are shown with triangles. The figure shows that dummy nodes are used to fill the miss-detection spots whenever needed.

Considering the dummy nodes in the graph we can expand the cost function in 1 into four terms as shown bellow:

$$\begin{aligned} \text{maximize} \sum_{j_1} \overbrace{c_{j_1}x_{j_1}}^{RealEdges} + \sum_{j_2} \overbrace{c_{j_2}x_{j_2}}^{DummyEdges} \\ + \sum_{j_3} \overbrace{c_{j_3}x_{j_3}}^{RealNodes} + \sum_{j_4} \overbrace{c_{j_4}x_{j_4}}^{DummyNodes}, \end{aligned} \tag{5}$$

where $x_{j_1}, x_{j_2}, x_{j_3}$ and x_{j_4} are the four types of variables in column vector $\mathbf{x}$. x_{j_1} defines the variables specified to real edges in the graph, x_{j_2} are used to define the variables for dummy edges, e.g edges which are connected to dummy nodes, x_{j_3} is the variable for real nodes representing the tracklets in each cluster and finally x_{j_4} represent the dummy nodes in the graph. The cost associated to each type of variable is defined using $c_{j_1}, c_{j_2}, c_{j_3}$ and c_{j_4}. In our formulation $c_{j_2} = c_d$ and c_{j_3} and c_{j_4} are set to zero. However one can also define a cost for the nodes in the graph, e.g average detection confidences of one tracklet can define the score of a node. c_{j_1} is defined based on motion and appearance similarity of the two tracklets. In Section 5 we explain in detail how to compute c_{j_1}.

Given the number of clusters and the number of nodes in each cluster, one can define the upper bound for the number of dummy nodes which needs to be added to each cluster:

$$N_d^i = \sum_{j \neq i} N^j, \tag{6}$$

where N_d^i is the number of dummy nodes added to cluster i and N^j is the number of true-nodes in cluster j. One should note that this is the upper bound for the number of dummy nodes, where the assumption is that for each track

there is only one true node among all the clusters. However, in practice we found that using only a small number of dummy nodes is sufficient, e.g when considering 5 clusters in GMMCP, $\frac{\sum_{j\neq i} N^j}{3}$ dummy nodes are enough. We show in our experiments that our dummy nodes are able to robustly replace miss detections as well as detection hypothesis with low global appearance and motion similarity with the rest of the tracks.

4. Speed-Up

In previous section we showed that one can easily obtain the upper bound for the number of dummy nodes added to each cluster. However, in practice only a small number of these dummy nodes are sufficient to handle miss detections in cliques. Adding more dummy nodes will increase the computational complexity as the number of variables during optimization will increase. In order to void such cases, we introduce *Aggregated Dummy Nodes (ADN)*. Our ADN will no longer be boolean variable and can take any integer value. This allows us to add only one ADN to each cluster which will account for all the outliers in that cluster.

Our new graph with aggregated dummy nodes is similar to the original graph with the difference that there is no edge connecting dummy nodes to other nodes. Integer-valued dummy nodes, will no longer allow solving GMMCP through BIP introduced in Section 3.1. Moreover, removal of edges correspond to dummy nodes require us with a new set of constraints in order to ensure the cliques found during optimization are valid solutions to the GMMCP.

We propose to solve GMMCP with ADN through Mixed-Binary-Integer Programming in which we aim to minimize the objective function $\mathbf{C}^T\mathbf{x}$. Where $\mathbf{C}$ is the weight matrix and $\mathbf{x}$ is a vector containing both boolean and integer variables. To ensure the solution is a valid solution to GMMCP we enforce the following three constraints:
Constraint 1 is similar to the one in Equation 4 which ensures that the solution will form a clique.

$$e_{ij}^{i'j'} + e_{i'j'}^{i''j''} \leq 1 + e_{i''j''}^{ij}. \quad \{\forall e_{ij}^{mn} \in E\} \tag{7}$$

Constraint 2 enforces the sum of outgoing edges of one node entering another cluster to be less than or equal to one. Thus one clique does not include more than one node from each cluster.

$$\sum_{m=1}^{l} e_{mn}^{ij} \leq 1. \quad \{\forall e_{ij}^{mn} \in E\} \tag{8}$$

Constraint 3 guarantees that K nodes from each cluster is selected.

$$\{\forall j | 1 \leq j \leq h\} : \sum_{i=1}^{l} e_{mn}^{ij} + vd^j = (h-1) \times K, \tag{9}$$

where vd^j defines the ADN in cluster j. The cost function remains the same as the one defined in Equation 5 with the difference that we no longer have the term for dummy edges, instead c_{j_4} which corresponds to setting dummy node cost to $\frac{c_d}{2}$, and c_{j_1} and c_{j_3} remain the same as before. The solution can be found optimally through Mixed-Binary Integer Program. For optimization we used ILOG CPLEX package provided at [1]. Our experiments show that using ADNs, one can significantly reduce the computational complexity and achieve performance close to real time on a desktop computer.

5. Affinity Measures

As mentioned earlier once our graph, G, is constructed given the tracklets in each segment, the edge connecting each pair of tracklets in different clusters are assigned a weight (c_{j_1} in Equation 5). The weight assigned to each edge is calculated considering both motion and appearance similarity between the two tracklets. Below we shall describe how these two affinities our computed in our tracker:

Appearance Affinity. For appearance representation of a node, we use color histogram [14]. The histogram is computed for all the detections of one node (tracklet) and the median appearance of the detections is selected as the appearance representation of the node. Given the appearance descriptor of two nodes in the graph ($\phi(T_i)$ and $\phi(T_j)$), the appearance affinity is calculated by computing the histogram intersection between the two tracklets.

$$c_{Appearance}(T_i, T_j) = k(\phi(T_i), \phi(T_j)). \tag{10}$$

Motion Affinity. Motion model is one of the major components in any tracking method. When it comes to tracking people in fixed surveillance cameras, the most practical motion model is constant velocity. Similar to [27], we employed a global constant velocity model. In our graph each node represents a tracklet. This allows encoding the motion affinity in the edge weight of the graph because each edge in the graph will connect more than two detection hypothesis. Given two tracklets and their corresponding state vectors, $T_1 = \left[\mathbf{y}_{s_1}, \mathbf{y}_{s_1+1}, \cdots, \mathbf{y}_{e_1}\right]$ and $T_2 = \left[\mathbf{y}_{s_2}, \mathbf{y}_{s_2+1}, \cdots, \mathbf{y}_{e_2}\right]$, the deviation error is computed using the following equation:

$$d = \sum_{j=0}^{\lfloor m/2 \rfloor} \sum_{i=0}^{\lfloor l/2 \rfloor} \overbrace{\left[\mathbf{y}_{s_2+j} - (\dot{\mathbf{y}}_{e_1-i})(s_2 - e_1 + j - i)\mathbf{y}_{e_1-i}\right]}^{forward\ deviation\ error}$$
$$+ \sum_{j=0}^{\lfloor m/2 \rfloor} \sum_{i=0}^{\lfloor l/2 \rfloor} \overbrace{\left[\mathbf{y}_{e_1-i} - (\dot{\mathbf{y}}_{s_2+j})(e_1 - s_2 + i - j)\mathbf{y}_{s_2+j}\right]}^{backward\ deviation\ error}.$$

where l and m define the length of first and second tracklets

Metric	Description
MOTA	Takes into account false positives, false negatives and ID-Switches
MOTP	It measures the tightness of the tracking results and groungtruth.
MT	Percentage of tracks that are successfully tracked for more than 80%
ML	Percentage of tracks that are successfully tracked for less than 20%
IDS	Total number of times that an output track changes its identity

Table 1. Description of metrics used in our evaluations.

respectively. y_i defines the location of the target in frame i, s_1 and s_2 are the start frames and e_1 and e_2 are the end frames. $(\dot{\mathbf{y}}_{e_1-i})$ defines the velocity vector ending at node in frame $e_1 - i$ and $(\dot{\mathbf{y}}_{s_2+j})$ defines the velocity vector starting from node $s_2 + j$. The first part of the above equation calculates the forward motion error and the second part calculates the backward error. Once the deviation error d is found the similarity is computed using Equation 11.

$$c_{Motion}(T_i, T_j) = \frac{1}{2} exp(\frac{-d}{\sigma}). \tag{11}$$

Given the appearance and motion affinity for each pair of nodes in the graph, the final edge cost is defined using linear combination of these two as:

$$c_{j_1} = (\nu) c_{Appearance} + (1 - \nu) c_{Motion}, \tag{12}$$

where ν and $(1 - \nu)$ are the corresponding weights for the two affinities. ν is set to 0.7 in first layer of our pipeline. However, in the second layer and where the final trajectories are found, global constant velocity motion model does not hold all the time. For the second layer of our tracking framework, where the long tracks are formed, we consider a damping factor which reduce the contribution of motion similarities when the two tracklets are far away in time. For the second layer the edge weight for two tracklets T_i^l and T_j^m, where l and m are the indexes of the GMMCP clusters, is computed using the equation below:

$$\begin{aligned} c_{j_1} &= (1 - \tau) c_{Appearance} + (\tau) c_{Motion}, \\ \text{where } \tau &= \exp(\frac{|l - m|}{\gamma})(1 - \nu). \end{aligned} \tag{13}$$

6. Experiments

We performed exhaustive experiments and evaluated our tracker on six sequences. Five of the sequences are publicly available sequences including *Town Center* [7], *Parking-Lot 1* [23], *Parking-Lot 2* [24], *TUD-Crossing* [3] and *TUD-Stadmitte* [17] and the last sequence is a new sequence called *Parking-Lot Pizza*. For publicly available sequences which the tracking results have been already reported (*Town Center*, *Parking-Lot 1*, *TUD-Crossing TUD-Stadmitte*), we compared our method with the state-of-art trackers, borrowing the numbers from the authors' papers. On the other two sequences which no tracking results are reported, we compare our method with competitive approaches which for we have access to their code including KSP [8], DCT [4], CMOT [5], IHTLS [13] and GMCP [27]. The results reported for these trackers could be considered as lower bound for which the parameters are set to the default as suggested by the authors. [1]

Dataset	Method	MOTA	MOTP	MT	ML	IDS
Town Center	MPT	72.9	71.3	-	-	-
	GMCP	75.59	71.93	-	-	-
	Ours	**77.37**	**66.38**	**86.09**	**4.35**	**68**
TUD Crossing	MWIS	85.9	73	-	-	2
	GMCP	91.63	75.6	-	-	0
	Ours	**91.9**	**70**	**75**	**0**	**2**
TUD Stadmitte	DLP	79.3	73.9	-	-	4
	GMCP	77.7	63.4	-	-	0
	Ours	**82.4**	**73.9**	**80**	**0**	**0**
Parking Lot 1	H2T	88.4	81.9	78.57	0	21
	GMCP	90.43	74.1	-	-	-
	Ours	**92.9**	**73.6**	**92.86**	**0**	**4**

Table 2. Quantitative Analysis of our method on Town Center, TUD-Crossing, TUD-Stadmitte and Parking-lot 1 with state-of-art methods of DLP[18], H2T [25], WMWIS [9] and PartTrack [23].

Implementation Details. We used deformable part based model [15] to get the detection hypothesis in each frame. The weights for the dummy nodes is found empirically and is set to 0.3. K which is the number of tracks found by GMMCP is set to a high number (50 in our experiments). A clique must have at least one real tracklet in its solution to be counted as a valid tracklet otherwise it is discarded. In Equation 11 the parameter σ is set to 20 and in Equation 12 the parameter ν and γ are set to 0.7 and 5 respectively.

6.1. Quantitative Evaluation and Discussion

For quantitative evaluation we used CLEAR MOT metrics (MOTA and MOTP) as well as Trajectory-Based measures,(Mostly Track, Mostly Lost and ID-Switch). CLEAR MOT metrics examine the video as a whole while TBM aim to evaluate each groundtruth track individually considering their completeness. The description of the metrics used in our evaluation is shown in Table 1.

Town Center. This is challenging sequence which contains 4500 annotated frames. The number of cluster in the first layer is set to 5, yielding to tracklet of maximum 50 frames. For final merging we considered 6 clusters which create track of maximum length 300 frames. During generating the final track we followed similar approach to [26, 27] by considering overlapping segments, thus connecting tracks in each step that overlap. The quantitative

[1] For more information and qualitative results please visit http://crcv.ucf.edu/projects/GMMCP-Tracker/.

Dataset	Method	MOTA	MOTP	MT	ML	IDS
Parking Lot 2	KSP	45.4	57.8	46.15	0	531
	DCT	60.1	56.1	76.92	0	234
	CMOT	80.7	58	84.62	0	61
	GMCP	75.6	58.1	61.54	0	76
	IHTLS	78.8	57.9	84.62	0	50
	Ours	**87.6**	**58.1**	**92.31**	**0**	**7**
Parking Lot Pizza	KSP	51.8	65.7	39.13	0	249
	DCT	53.5	65.8	69.57	0	185
	IHTLS	57.6	66.8	43.48	4.35	105
	CMOT	56.9	63.3	30.43	4.35	87
	GMCP	57.6	68.6	26.9	4.35	52
	Ours	**59.5**	**64.1**	**30.43**	**0**	**55**

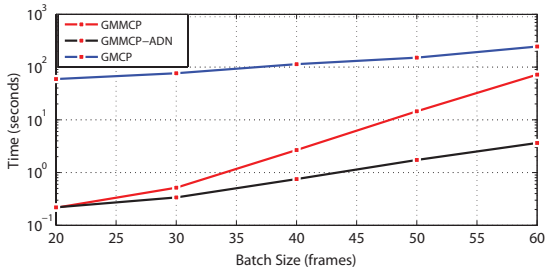
Table 3. Quantitative Analysis of our method on Parking-lot Pizza and Parking-lot 2 with competitive approaches of KSP [8], DCT [4], CMOT [5], IHTLS [13] and GMCP [27].

results are shown in Table 2.

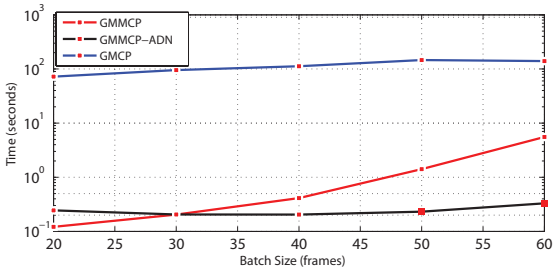
Parking Lot 1 and Parking Lot 2. Parking-lot 1 (PL1) contains 748 annotated frames and up to 14 pedestrians in the scene. The main challenges in this sequence are occlusion and confusion cause by targets walking close by with similar appearance . Parking Lot 2 is relatively more difficult as it includes a fighting scene with targets walking with abrupt motion. Parking-Lot 2 contains 900 annotated frames and up to 13 pedestrians. On PL1 we show $\sim 4\%$ improvement in MOTA compared to state-of-art. For PL2 we compare with 5 other methods and show favorable improvement especially in number of ID-Switches. IHTLS [13] which estimates the underlying dynamic of the target without assuming a prior motion model performs better compared to other four methods. However, we show that using our holistic motion model followed by a damping factor we can still handle abrupt motion to some extent and outperform competitive approaches. (To be fair we want to mention that, our tracker also takes advantage of appearance information which is not the case in IHTLS)

TUD Dataset. TUD-Crossing and Stadtmitte are two sequences in this data set with low camera angle and frequent occlusions. Crossing includes 201 and Stadtmitte contains 179 frames. The results are provided in Table 2.

Parking-lot Pizza. Although we show significant improvement on the five publicly available sequences, the performance of state-of-art trackers on those surveillance type sequences have almost reached its ceiling. Our new sequence *Parking-lot Pizza* contains a semi-crowded scenario with a lot of occlusions, pose variations and abrupt motions. A high-resolution surveillance camera (4000×3000) is used to record the sequence. We compare our method with 5 trackers and the results are reported in Table 3.



(a) Parking-lot 1



(b) TUD-Crossing

Figure 3. run-time comparison of our GMMCP tracker with ADN and without ADN as well as the GMCP tracker proposed in [27] on *Parking-lot 1* and *TUD-Crossing* sequences. The plots are in semilogarithmic scale

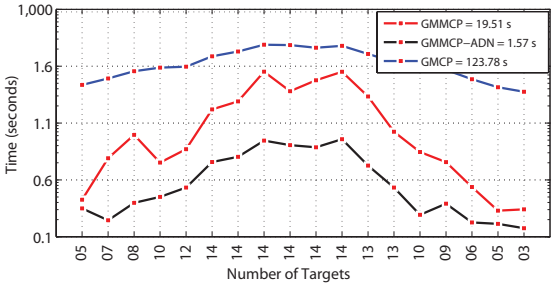


Figure 4. Run time comparison of our method for different number of targets in a batch of 50 frames on Parking-lot 1. The numbers in the legend show the average run-time across all the segments.

6.2. Run Time Comparison

As discussed in section 4, the more dummy nodes we add to each cluster, the slower the optimization becomes. Although we show that without ADNs we still can efficiently find the tracks, the implementation of proposed GMMCP tracker with aggregated dummy nodes will help reducing the computational complexity significantly. We conducted two set of experiments to show the speed-up achieved using ADNs. First, in figure 3, we compare our implementation with and without ADNs when the average run-time per segment of two sequences are reported. For this experiment we changed the number of frames in the batch from 20 to 60 and recorded the time to find the tracks within that batch. In the second experiment in figure 4 we show how the run-

Figure 5. Qualitative results of our tracker on experimented sequences.

time increases as the number of people increase in the scene for a batch size of 50 frames. For both experiments we also compared our results with GMCP tracker [27] where the publicly available implementation is used to record the time complexity. In both experiments we achieved significant speed-up, up to two order of magnitude, when using the MBIP implementation of our approach. All the numbers reported in Figure 4 and 3 are obtained using a 3.2GHz PC while utilizing only a single core.

It is worth to mention that the computational complexity of our method is highly dependent on the problem size. Although we show performance close to real-time using a non-optimized code on tested sequences, the complexity increases as the size of the graph increases. This issue highlights itself even more, when we are dealing with an NP hard problem. Our observations show that for crowded scenes (more than 30 pedestrians) we need to decrease the batch size in-order to be able to track targets efficiently.

7. Conclusion

In this paper we formulate multiple target tracking as a Generalized Maximum Multi Clique problem where it is solved through Binary Integer Programing. We later show that by using aggregated dummy nodes and reformulating GMMCP through a Mixed-Binary-Integer Program we can achieve a speed-up up to two order of magnitude. In our experiment we show that we can improve the state-of-art on six challenging sequences.

Acknowledgment. This material is based upon work supported by, the U.S. Army Research Lab, the U.S. Army Research Office under grant number W911NF-14-1-0294.

References

[1] Ibm ilog cplex optimizer, www-01.ibm.com/software/integration/optimization/cplex-optimizer. 5

[2] E. Althaus, O. Kohlbacher, H. Lenhof, and P. Muller. Recomb. In *A combinatorial approach to protein docking with flexible side chains*, 2000. 3

[3] A. Andriyenko and K. Schindler. Multi-target Tracking by Continuous Energy Minimization. In *CVPR*, 2011. 2, 6

[4] A. Andriyenko, K. Schindler, and S. Roth. Discrete-Continuous Optimization for Multi-Target Tracking. In *CVPR*, 2012. 2, 6, 7

[5] S.-H. Bae and K.-J. Yoon. Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In *CVPR*, 2014. 6, 7

[6] Y. Bar-Shalom, T. Fortmann, and M. Scheffe. Joint probabilistic data association for multiple targets in clutter. In *Information Sciences and Systems*, 1980. 2

[7] B. Benfold and I. Reid. Stable multi-target tracking in real-time surveillance video. In *CVPR*, pages 3457–3464, June 2011. 6

[8] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua. Multiple Object Tracking Using K-Shortest Paths Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819, 2011. 2, 6, 7

[9] W. Brendel, M. Amer, and S. Todorovic. Multiobject tracking as maximumweight independent set. In *CVPR*, 2011. 6

[10] A. A. Butt and R. T. Collins. Multi-target tracking by lagrangian relaxation to min-cost network flow. In *CVPR*, 2013. 2

[11] S. Chen, A. Fern, and S. Todorovic. Online multi-person tracking-by-detection from a single, uncalibrated camera. In *CVPR*, 2014. 3

[12] A. Dehghan, Y. Tian, P. H. S. Torr, and M. Shah. Target identity-aware network flow for online multiple target tracking. In *CVPR*, 2015. 2

[13] C. Dicle, M. Sznaier, and O. Camps. The way they move: Tracking targets with similar appearance. In *ICCV*, 2013. 3, 6, 7

[14] J. Domke and Y. Aloimonos. Deformation and viewpoint invariant color histograms. In *BMVC*, 2006. 5

[15] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. 6

[16] C. Feremans, M. Labbe, , and G. Laportee. European journal of operational research. In *Information Sciences and Systems*, 2003. 3

[17] J. Ferryman and A. Shahrokni. Dataset and challenge. In: Performance Evaluation of Tracking and Surveillance. 2009. 6

[18] A. K. K.C. and C. D. Vleeschouwer. Discriminative label propagation for multi-object tracking with sporadic appearance features. In *ICCV*, 2013. 6

[19] A. Koster, S. V. Hoesel, and A. Kolen. Operations research letters 23. In *The partial constraint satisfaction problem: Facets and lifting theorems*, 1998. 3

[20] Z. Li, L. Yuan, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008. 2

[21] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects. In *CVPR*, 2011. 2

[22] H. B. Shitrit, J. Berclaz, F. Fleuret, and P. Fua. Multi-commodity network flow for tracking multiple people. In *PAMI*, 2013. 2, 3

[23] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah. Part-based Multiple-Person Tracking with Partial Occlusion Handling. In *CVPR*, 2012. 6

[24] G. Shu, A. Dehghan, and M. Shah. Improving an Object Detector and Extracting Regions using Superpixels. In *CVPR*, 2013. 6

[25] L. Wen, W. Li, J. Yan, Z. Lei, D. Yi, and S. Z. Li. Multiple target tracking based on undirected hierarchical relation hypergraph. In *CVPR*, 2014. 6

[26] B. Yang and R. Nevatia. An online learned crf model for multi-target tracking. In *CVPR*, 2012. 3, 6

[27] A. R. Zamir, A. Dehghan, and M. Shah. GMCP-Tracker: Global Multi-object Tracking Using Generalized Minimum Clique Graphs. In *ECCV*, 2012. 2, 3, 5, 6, 7, 8