GLOBAL DATA ASSOCIATION FOR MULTIPLE PEDESTRIAN TRACKING

by

AFSHIN DEHGHAN
M.S. University of Central Florida, 2014

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Electrical Engineering and Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2016

Major Professor: Mubarak Shah

ABSTRACT

Multi-object tracking is one of the fundamental problems in computer vision. Almost all multi-object tracking systems consist of two main components; *detection* and *data association*. In the detection step, object hypotheses are generated in each frame of a sequence. Later, detections that belong to the same target are linked together to form final trajectories. The latter step is called data association. There are several challenges that render this problem difficult, such as occlusion, background clutter and pose changes. This dissertation aims to address these challenges by tackling the data association component of tracking and contributes three novel methods for solving data association.

Firstly, this dissertation will present a new framework for multi-target tracking that uses a novel data association technique using the Generalized Maximum Clique Problem (GMCP) formulation. The majority of current methods, such as bipartite matching, incorporate a limited temporal locality of the sequence into the data association problem. This makes these methods inherently prone to ID-switches and difficulties caused by long-term occlusions, a cluttered background and crowded scenes. On the other hand, our approach incorporates both motion and appearance in a global manner. Unlike limited temporal locality methods which incorporate a few frames into the data association problem, this method incorporates the whole temporal span and solves the data association problem for one object at a time. Generalized Minimum Clique Graph (GMCP) is used to solve the optimization problem of our data association method. The proposed method is supported by superior results on several benchmark sequences.

GMCP leads us to a more accurate approach to multi-object tracking by considering all the pairwise relationships in a batch of frames; however, it has some limitations. Firstly, it finds target trajectories one-by-one, missing joint optimization. Secondly, for optimization we use a greedy solver, based on local neighborhood search, making our optimization prone to local minimas. Finally GMCP tracker is slow, which is a burden when dealing with time-sensitive applications. In

order to address these problems, we propose a new graph theoretic problem, called *Generalized Maximum Multi Clique Problem* (GMMCP). GMMCP tracker has all the advantages of the GMCP tracker while addressing its limitations. A solution is presented to GMMCP where no simplification is assumed in *problem formulation* or *problem optimization*. GMMCP is NP hard but it can be formulated through a Binary-Integer Program where the solution to small- and medium-sized tracking problems can be found efficiently. To improve speed, *Aggregated Dummy Nodes* are used for modeling occlusions and miss detections. This also reduces the size of the input graph without using any heuristics. We show that using the speed-up method, our tracker lends itself to a real-time implementation, increasing its potential usefulness in many applications. In test against several tracking datasets, we show that the proposed method outperforms competitive methods.

Thus far we have assumed that the number of people do not exceed a few dozens. However, this is not always the case. In many scenarios such as, marathon, political rallies or religious rites, the number of people in a frame may reach few hundreds or even few thousands. Tracking in high-density crowd sequences is a challenging problem due to several reasons. Human detection methods often fail to localize objects correctly in extremely crowded scenes. This limits the use of data association based tracking methods. Additionally, it is hard to extend existing multi-target tracking to track targets in highly-crowded scenes, because the large number of targets increases the computational complexity. Furthermore, the small apparent target size makes it challenging to extract features to discriminate targets from their surroundings.

Finally, we present a tracker that addresses the above-mentioned problems. We formulate online crowd tracking as a Binary Quadratic Programing, where both detection and data association problems are solved together. Our formulation employs target's individual information in the form of appearance and motion as well as contextual cues in the form of neighborhood motion, spatial proximity and grouping constraints. Due to large number of targets, state-of-the-art commercial quadratic programing solvers fail to efficiently find the solution to the proposed optimization. In order to overcome the computational complexity of available solvers, we propose to use the most

recent version of Modified Frank-Wolfe algorithms with SWAP steps. The proposed tracker can track hundreds of targets efficiently and improves state-of-the-art results by significant margin on high density crowd sequences.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Mubarak Shah, and committee, Dr. Ulas Bagci, Dr. Shaojie Zhang, Dr. GuoJun Qi and Dr. Qipeng Zheng, for their guidance throughout the research process. I feel very fortunate for having been part of Dr. Shah's research group and for being given the opportunity to be involved in and learn from a wide range of projects. I would like to thank all of the past and present members of the Center for Research in Computer Vision (CRCV), particularly Dr. Amir Roshan Zamir, Dr. Omar Oreifej, Dr. Guang Shu, Dr. Enrique Ortiz, Dr. Haroon Idrees, Shayan Modiri, Yicong Tian, Mahdi Kalayeh, Shervin Ardeshir, Gonzalo Vaca, Ruben Villegas, Cherry Place and Tonya LaPrarie for their support, the good times, and the great memories. Also, I was honored to work with Dr. Philip H. S. Torr, Dr. Arnold Smeulders, Dr. Massimo Piccardi, Dr. Rita Cucchiara. Lastly, I want to thank my father, my mother and my sister for their endless support. This journey has been long but I am so thankful for those that assisted me through it.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

We humans use our eyes and brains to understand and interpret what we see. Computer vision is the science that aims to give machines the same capability and it has begun to play a part in our everyday lives. We use and interact with different machines, equipped with computer vision algorithms already, and many of us may not have even noticed. For instance, face detection is integrated into almost every camera these days. Face recognition [16, 17] and clustering techniques are two more examples of computer vision algorithms that are used by applications in order to improve photo album organization. However, some computer vision problems have proven to be more intractable. Among those, multi-target tracking is undoubtedly a fundamental problem that has attracted a lot of interest among many researchers for decades.

Multi-object tracking, sometimes referred to as multi-target tracking, is the problem of automatically detecting the objects of interest and inferring the trajectories of those objects in a video sequence. Figure 1.1 illustrates a multi-person tracking problem where given a video sequence, the task is to construct the trajectory of each person. There are many real-world applications of multi-object tracking, which help to explain the wide-ranging interest among researchers and companies in the problem. Self-driving cars have begun to find their way into headlines in the mainstream media, and will soon be a part of our lives. Soon cars will be equipped with sensors, and collision avoidance systems will benefit from tracking information of surrounding objects. Multi-target tracking is also helpful in analyzing sports games and can provide the coache and audience with game and player statistics [18, 19]. In virtual reality, multi-target tracking helps to connect humans with machines by tracking gestures or hand movements [20–22]. Biologists have also found multi-target tracking helpful in their studies [23–25], for example in tracking shoals of fish or flocks of birds to gain insight into their behavior. Medical professionals have started to use systems equipped with tracking capabilities to track cells, bacteria or protein motion that will eventually help them in studying body organs or for drug discovery [26–29].

Figure 1.1: Multi-target tracking problem. Given a sequence of images, the task is to detect and construct the trajectory of each individual in the scene.

However, among all these applications, visual surveillance is without doubt the one that has attracted the most attention. There are millions of hours of video recorded every day from surveillance cameras. In the United States it is estimated that 30 million surveillance cameras shoot over four billion hours of footage every week. When we walk through a shopping mall, parking lot, school or a small private store, we are being watched. Examples of images taken by surveillance cameras are shown in Figure 1.2). Law enforcement increasingly relies on videos provided by surveillance cameras, and several commercial applications offer features such as detecting abandoned luggage, monitoring abnormal activities or focusing on aggressive or unusual behavior.

While there is no doubt about the importance of surveillance cameras, manual supervision of all this footage is almost impossible. Being able to analyze these videos automatically and reduce the amount of manual work becomes vital. An important step in analyzing surveillance videos is to successfully reconstruct the trajectories of targets in a scene. These trajectories are later used as a pre-requisite to most of the above-mentioned visual tasks, such as abnormal activity detection or crowd analysis.

Figure 1.2: Samples of image sequences taken by surveillance cameras at airports and parking lots [1].

Most recent approaches in multi-target tracking follow a so called tracking-by-detection paradigm. In tracking-by-detection methods, at the input level, the detection hypotheses in every time stamp are found first. Later the detections that belong to the same target are linked together to reconstruct the tracks of the targets in the scene. The latter step is called data association. The popularity of these methods is due to the great advances made in the field of object detection in the past decade [30–33]. In most visual surveillance scenarios, the object detection methods are able to provide reasonable results. This is the main reason that the focus of most multi-target tracking methods has been to design a better data association technique. The challenge that renders the data association problem difficult is noisy observation data. Noisy observations include not only the mis-localization of the objects but also false alarms that are caused by the clutter, occlusion, pose change or changes in camera view point. Several offline trackers were proposed in recent years that have improved the state-of-the-art significantly. However, the results in many scenarios are still far from perfect, leaving data-association as one of the most active areas of research in computer vision.

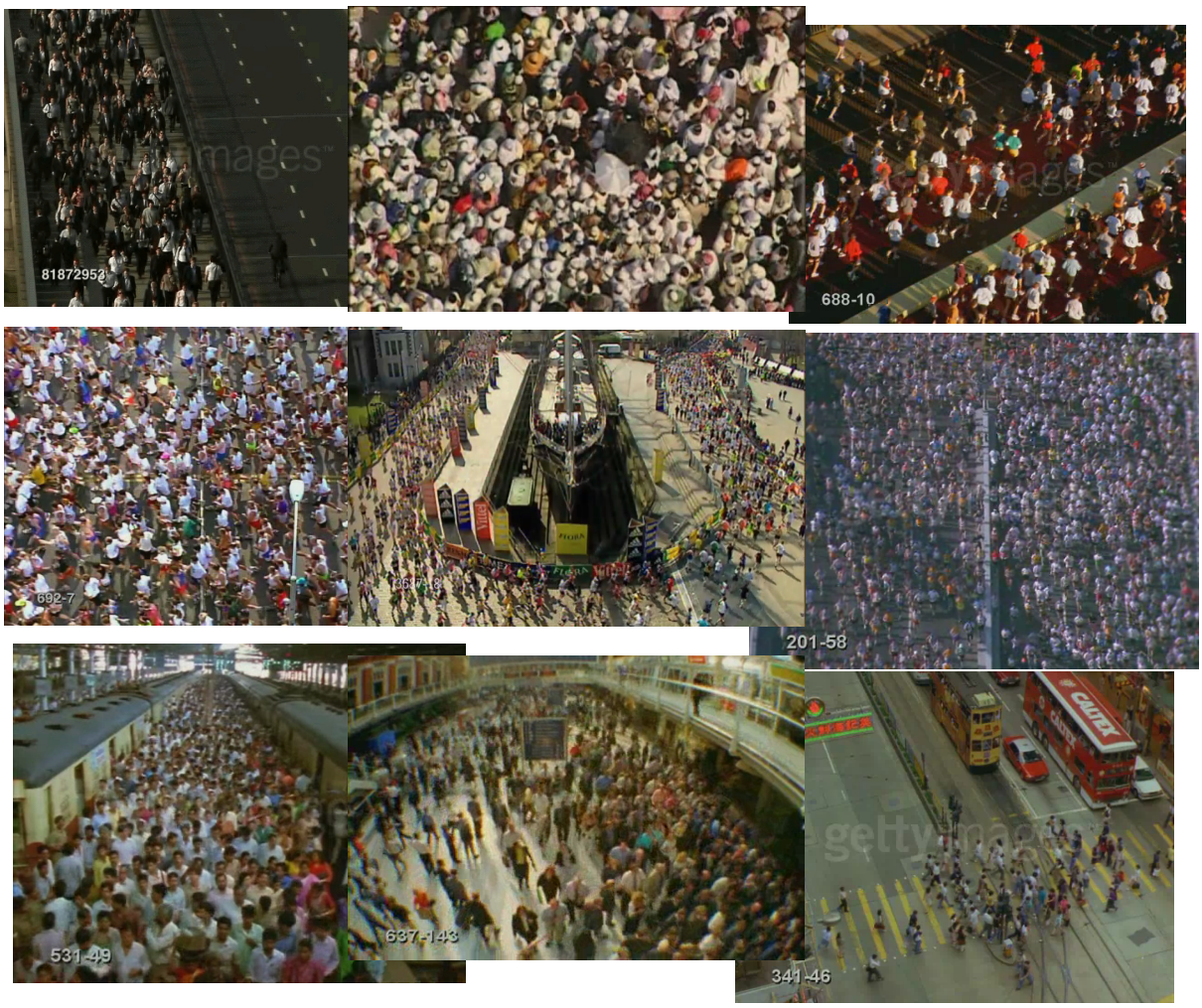


Figure 1.3: Example of high density crowd sequences used in [2]. The examples include scenarios such as marathons, train stations and religious gatherings. One can see that targets are very small and sometimes hard to distinguish from nearby targets.

On the other hand, there is a limit to what data association-based tracking methods can achieve when dealing with noisy observation data. If an object detector fires too many false alarms, or misses too many true detections, the data association-based tracking methods will often fail to construct the trajectories of targets correctly. In particular, when dealing with crowded scenes, the performance of object detectors degrades significantly. Figure 1.3 illustrates several examples of

such sequences. Tracking targets in highly-crowded scenes is a relatively new area of research and only a few papers have focused on this problem [2, 5, 6, 34].

In this dissertation we aim to address multi-target tracking in various dense crowds by proposing several new algorithms. We not only propose new data association methods for small and medium density crowd scenarios, but also an online multi-target tracking algorithm that is designed to track hundreds of people in extremely crowded scenes. More specifically, two new global data association techniques are presented that are based on the Generalized Maximum Clique Problem and the Generalized Maximum Multi Clique Problem formulations that use information from many frames to infer the trajectories of pedestrians. This is different from most previous methods that incorporate the information from only two or a few frames during data association. The proposed data association methods aim to address the confinements of noisy observation data provided at input level by providing a formulation that is closer to real-world scenarios. Additionally, to efficiently track targets in extremely crowded scenes and replace noisy pre-trained object detectors, the dissertation proposes a Binary-Quadratic Programing formulation for crowd tracking problems. In summary, this dissertation makes the following important contributions.

- Tracking with global data association, using the Generalized Maximum Clique Problem (GMCP), where all the pairwise relationships between targets in a temporal span are taken into account. In the proposed framework, the whole temporal span of the sequence is incorporated into the data association problem, but we focus on one object at a time, rather than addressing all of them simultaneously. The proposed method yields a better approach to data association which is supported by our superior results.

- Multi-target tracking is re-formulated as a new graph theoretic problem, using a Generalized Maximum Multi Clique Problem (GMMCP). The GMMCP tracker aims to address the limitations of the GMCP tracker by adding joint optimization as well as a faster and more accurate solver for the optimization. We show that the GMMCP tracker improves the per-

formance of the GMCP tracker in most scenarios. This is in addition to the fact that the GMMCP tracker is almost two orders of magnitude faster than GMCP.

- We are the first to formulate tracking in crowded scenes as multi-object tracking where all targets are tracked simultaneously. This differs from previous methods which track targets one-by-one, making them prone to ID-switches. The proposed method is better for multi-target tracking in crowded scenes and this is supported by the experimental results on challenging sequences.

1.1 GMCP Tracker: Generalized Maximum Clique Problem for Multi-target Tracking

Data association is the problem of finding detections corresponding to one particular object in different frames of a video. It is the backbone of most multi-target tracking algorithms. The input to the data association problem in a sequence can ideally be represented by a graph in which all the detections in each frame are connected to all the other detections in the other frames (a k-partite complete graph), regardless of their closeness in time. Similarly, the output can be ideally represented by several subgraphs of the input in which the detections belonging to common entities are connected. Finding the ideal subgraphs which represent the exact solution to data association requires solving an optimization problem which remains unsolved to date due to its extreme complexity. Therefore, approximation methods are employed in order to simplify the conditions and find an acceptable solution. The most natural, and probably the simplest, approximation is considering a limited temporal locality, e.g. two or a few frames of the input graph, and solving the optimization problem for the smaller, less complex subgraph. This natural approximation forms one main category of data association methods and has been investigated thoroughly in the literature. The best known examples of such methods are bi-partite matching (considering only two frames) and its extensions [1, 35], which use an extended, yet limited, number of frames.

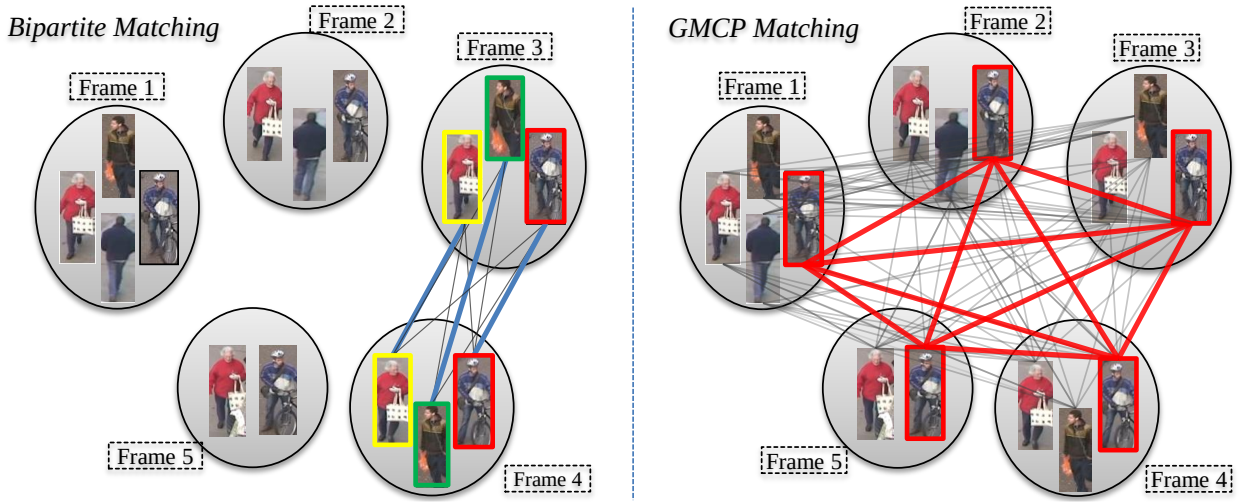


Figure 1.4: Bi-partite vs. GMCP matching.Gray and colored edges represent the input graph and optimized subgraph, respectively. Bi-partite matches all objects in two frames. On the other hand, the GMCP matches one object at a time across full temporal span (five frames here), while incorporating the rest of the objects implicitly.

Our proposed method for data association incorporates both appearance and motion in a global way. In the proposed framework, the whole temporal span of the sequence is incorporated into the data association problem, but we focus on one object at time rather than addressing all of them simultaneously. This is to avoid dealing with an extremely complex optimization problem. Although we focus on solving the data association problem for one object at time, we also incorporate all the other objects implicitly. Therefore, our approximation in the object-domain is significantly less restrictive than those used by other approximation methods, such as limited temporal locality. This is because limited temporal locality methods are blind to information outside of the temporal neighborhood they are focused on, while the proposed method incorporates the whole of the approximation domain, i.e. all objects, implicitly. This fundamental difference, along with the new framework, leads to a better construction of the data association problem which is supported by our superior results

The proposed method and bi-partite matching are both shown schematically in Fig. 1.4

for a sequence of 5 frames. The gray edges show the input graph for the data association for one iteration of the algorithms. The colored thick edges show the optimized subgraph representing the results of that iteration. As shown on the right, the resultant subgraph of this method determines the detections of one person over the whole temporal span, while keeping a notion of other pedestrians in the optimization process. In contrast, the limited-locality-methods do not carry the information out of their current locality (i.e. no edges to or from the frames other than 3 and 4). The proposed framework comprises detecting humans in frames [33], calculating tracklets in video segments and merging them into trajectories. We use Generalized Minimum Clique Graphs to solve our data association problem.

Reviewing the literature, global data association methods, including the GMCP tracker proposed in previous section, can be divided into two main groups. One group assumes a simplified version of the problem where only relationships between observations in consecutive frames are taken into account. These methods are mostly followed by an optimization for which there exists an exact solution. Network flow algorithm [12, 36, 37] 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. Such methods are computationally efficient and plausible for many applications. However, this simplification of the original problem comes at a price. Limiting the cost calculated for a track to observations in consecutive frames makes the tracker prone to ID-switches.

The second group of data association methods assumes no simplification in problem formulation and considers a model which is closer to the tracking scenarios in the real world. However, due to the complexity of their models, the proposed solutions are approximate [13,38]. The GMCP tracker proposed in the previous section lies within this group as well. Even though incorporating all pairwise relationships is important, GMCP has its own limitations. Firstly, the GMCP tracker does not follow a joint optimization for all the tracks simultaneously rather it finds the tracks one-by-one. In addition, the proposed solver follows a greedy optimization approach which is prone to

local minimas. Moreover, the heuristic line-fitting approach for removing outliers makes the tracks prone to ID-switch, especially when people are walking close to each other. Finally, GMCP tracker is slow and takes several seconds to solve the association for a batch of frames. Some failure cases of GMCP tracker are shown in Figure 1.5.

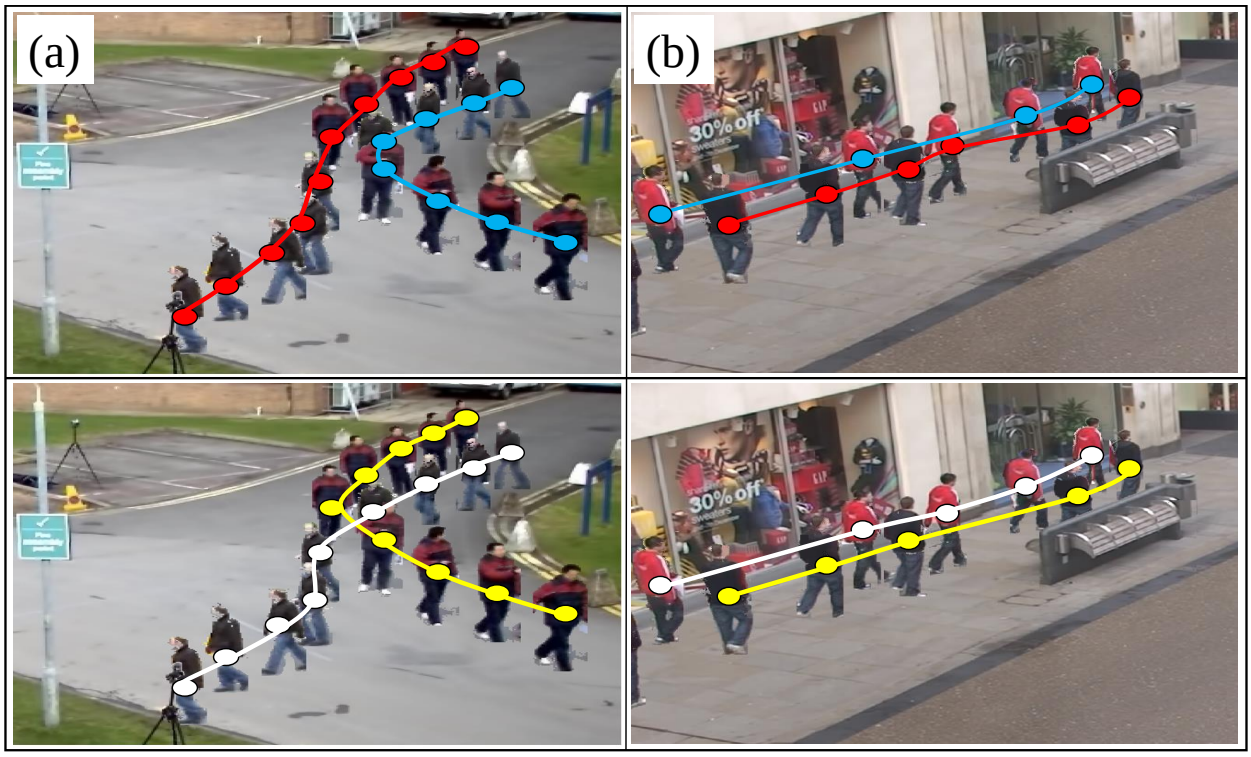


Figure 1.5: GMCP failure cases. Example of failure cases of the GMCP tracker (top row) (a) indicates the effect of joint optimization where two targets with similar appearance are walking in the scene. In GMCP first the red track that has a very high score is selected and later the blue track is picked, which has a very low similarity score. Both tracks in this case are wrong. (b) Shows two targets walking in parallel for six frames with very different appearance but spatially close (each target has one missed detection). The red track is selected first which includes an ID-Switch. The reason is because of the greedy outlier selection technique used in GMCP tracker. Bottom row shows the groundtruth tracks .

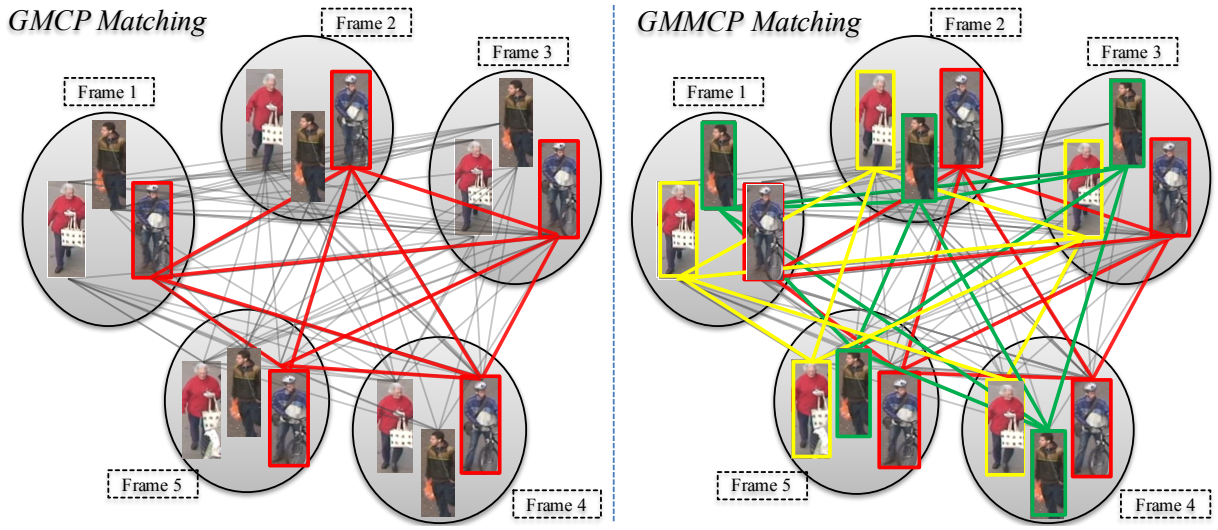


Figure 1.6: GMCP matching vs. GMMCP matching. Gray and colored edges represent the input graph and optimized subgraph, respectively. GMCP matches one object at a time across full temporal span, while incorporating the rest of the objects implicitly. On the other hand, GMMCP follows a joint optimization strategy and matches all the objects simultaneously in the full temporal span

1.2 GMMCP Tracker: Generalized Maximum Multi-clique Problem for Multi-target Tracking

In order to address the limitations of the GMCP tracker we propose a new graph theoretic problem called, the Generalized Maximum Multi Clique Problem (GMMCP). Similar to a GMCP tracker, our graph incorporates the pairwise relationship between all observations in a batch of frames, and our cost function allows the incorporation of higher order costs between all candidates in a track. However, during optimization, GMMCP is capable of selecting multiple cliques that maximize the score of the objective function, allowing joint optimization of target tracks. In Figure 1.6 we show the major difference between GMCP and GMMCP trackers. GMCP selects one subgraph that has the least cost in the k-partite complete graph. On the other hand, GMMCP is able to select multiple subgraphs with the best scores. This is essential for having a more accurate multi-object tracking formulation.

Although we show that GMMCP is an NP hard problem, we do not follow an approximate solution for it. Instead we formulate GMMCP through a Binary Integer Program (BIP), from which the solution can be found accurately for small- and medium-sized MOT problems. The tracking algorithm addresses the aforementioned problems of GMCP tracker by not only finding the tracks jointly for all the objects but also providing an accurate solution without relaxing the original problem. In order to address the complexity issue of GMCP tracker, a speed-up technique with a more efficient occlusion handling strategy is proposed. Our speed-up solution formulates GMMCP as a Mixed-Binary Integer Program that reduces the computational complexity significantly without any loss in the performance. For small and medium size multiple object tracking problems the solution to this problem can still be found efficiently using this pipeline and favorable results against other state-of-the-art systems can be achieved.

1.3 Crowd Tracker: Binary Quadratic Programing for Online Tracking of Hundreds of People in Extremely Crowded Scenes

When dealing with high-density crowd scenes, where there are hundreds of pedestrians, the problem becomes more difficult. It is even challenging, if not impossible, for humans to track individuals in such sequences (An example of such a sequence is shown in Figure. 1.7). The first challenge that one would be faced with, while designing a tracker, is that pre-trained object detectors fail to detect individuals in high-density crowd scenes. The main reason is that most crowded sequences are captured using cameras facing down, where the full body is not visible. Moreover face/head detection methods have shown poor performances in such scenarios [39]. This makes the use of data-association based tracking methods [12, 36, 37] impossible where they rely entirely on the outcome of a pre-trained object detector.

Figure 1.7: This figure shows an example of a crowd sequence. The yellow boxes show the targets that are being tracked in this sequence. On the borders we show the close-up version of some of the targets. As can be seen targets are very small and the discriminative appearance cues are very low. More over targets look very similar which confuses most trackers.

The other challenge in dealing with these sequences is the small apparent target size. The number of pixels covering each target is small, which makes it hard to discriminate the target from others or sometimes from the background. Additionally, the large number of targets to be tracked increases the computational complexity and makes the design of an efficient tracker even more challenging. The latter issue is one of the main reasons that all previous trackers [2, 5, 6, 34], focused on high-density crowds, track one target at a time instead of jointly optimizing the objective function for all the targets simultaneously. Although focusing on tracking one target at a time helps reduce the computational complexity, joint optimization is essential for optimal multi-target tracking. For example, multiple targets cannot occupy the same location at the same time. Thus for an optimal assignments of new locations, objects in the scene should compete for each candidate location simultaneously, which is not the case in [2, 5, 6, 34]. Additionally

modeling interactions between targets can greatly benefit multi-target tracking algorithms. This is only feasible when target tracks are optimized jointly.

To this end, we propose a Binary Quadratic Programing solution to crowd-tracking that aims to accomplish the above-mentioned goals. To be more specific, we are the first to formulate tracking in high-density crowds as multi-target tracking. This means that our joint optimization allows updating tracks of all targets simultaneously. Further, our method considers multiple candidate locations for each target within the optimization and determines the correct location without assuming availability of target locations. Additionally, we use five essential components for tracking individuals in crowds that capture the target's individual information as well as contextual cues. We show that each of these components can be encoded in our objective function as a linear or a quadratic term. The first component captures the information necessary to discriminate each target from its background by using its appearance information. Our appearance term is based on an online discriminative learning approach, where we train a regression model for each target. This is different from previous works where a generative model such as template based tracking is used as a baseline [2, 5, 6, 34]. The second and third components in our objective function capture the motion of the crowd. One encodes the target motion which is obtained based on the past trajectory of each target. The other one is related to the neighborhood motion, which captures the effect of neighbors. The fourth term in our objective function is the spatial proximity term that aims to discourage the co-selection of targets that are too close to each other. Finally, the last term encodes the group formation. It encourages the co-selection of targets that help maintain the group formations from frame to frame.

The first two terms are the building blocks of most current trackers. But the third and fourth terms are especially important for tracking targets in crowded scenes. When tracking targets in high-density crowds, relying on only the observations from an individual's target tracks is not sufficient. In such scenarios, modeling contextual information and interactions between targets becomes vital. In crowded scenes, where a large number of people are bound to move in a small

area, the motion of each individual is affected not only by its own behavior, but also by the motion of its neighbors. This neighborhood motion helps us to improve tracks of the targets when an individual's appearance and motion cues are not that strong. This happens frequently due to the low apparent target size and similar looking moving targets. This neighborhood motion is captured through the third component of our objective function which encourages each individual to walk with similar motion as his/her coherently moving neighbors. The coherent motion groups are found using the information from past trajectories of targets. Our model is simpler compared to the previous works, which require prior information about the scene such as floor fields, [5] or the heuristics such as instantaneous flows to deal with anomalies or unstructured scenes [2].

Another important component in our formulation which has not been used before in crowd-tracking is the spatial proximity constraint. The most commonly used constraint in almost all multi-target tracking methods is that each location should be assigned to only one target. While this constraint is essential, we found this not to be sufficient in crowded scenes. In data association based trackers, sparse detections are provided at input level, thus having the constraint that two tracks should not share a detection suffices. However, since we do not assume the availability of target detections, at every frame we have candidates sampled densely over the entire frame. Having targets with similar appearance and densely sampled candidates, it often happens that tracks of different targets overlap considerably while not sharing the same candidate location (i.e they select two locations that are too close to each other). Additionally, when dealing with crowd sequences in aerial videos, having detections that overlap is restricted. This is because most sequences are captured using cameras facing down and targets do not occlude each other. Figure. 1.8 illustrates an example where two targets with similar appearance are running next to each other and the tracker gets confused after a few frames. In order to handle such failure cases, encouraging the co-selection of targets that are not too close to each other becomes essential. We later show in the section 5.4 that the spatial proximity constraint helps significantly improve the performance on most sequences.

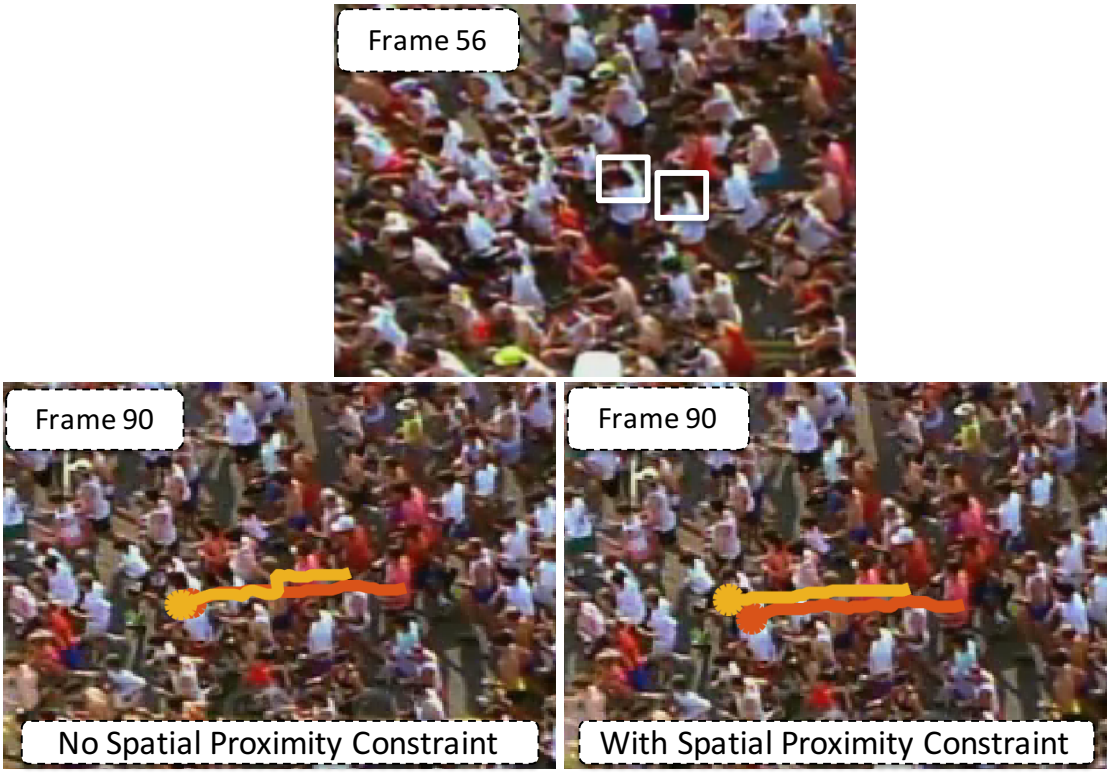


Figure 1.8: This figure aim to motivate the spatial proximity constraint used in our formulation. The top figure shows the two targets with very similar appearance. The bottom figures demonstrate the tracking results of our method, with and without proximity constraint. As can be seen when no spatial proximity constraint is used, the tracker gets confused and track of one target jumps to the near by one with similar appearance and motion. However, we are able to correctly track the two targets when we use the spatial proximity constraint as shown in bottom right figure.

In addition to our proximity constraint, we introduce another term which accounts for group formation. Group information has been recently integrated into several multi-target tracking algorithms and has shown its effectiveness [40–42]. Our BQP formulation allows adding any constraint in the form of a linear or a quadratic term. We show that the group constraint can be formulated as an additional quadratic term and later show in our experiments that the group constraint helps to further improve the results. Please refer to Figure. 1.9 for a summary of the constraints used in our formulation.

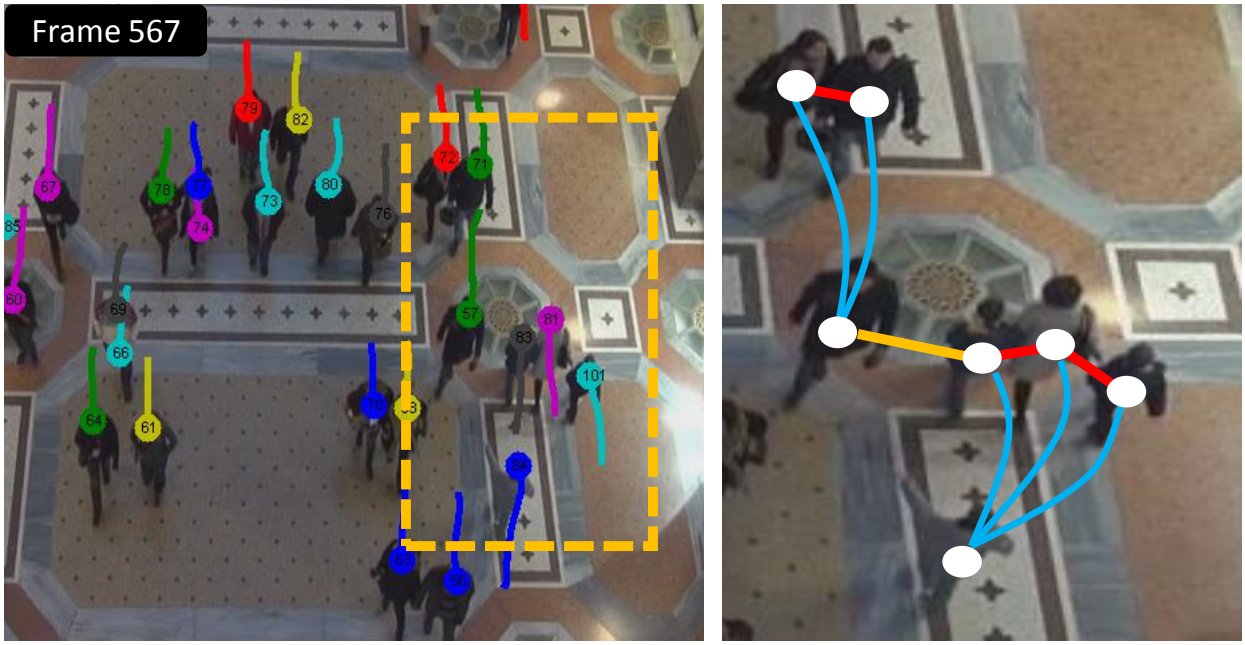


Figure 1.9: This figure is a graph illustration of the contextual constraints used in our formulation. The figure on the left shows the tracks and the figure on the right shows the constraints between the targets in yellow box. Each target is a node in the graph and all targets are connected with an edge. In practice there is a connection between every pair of targets but here for simplicity we are showing only some of the connections. The figure illustrates two groups walking in opposite directions as well as two individuals. Each edge is assigned a different cost depending on the grouping information and distance of targets from each other. The red edges that connect people in the same group encode the grouping and proximity constraints. The blue lines contain the neighborhood motion information and exist only between targets with coherent motion. The yellow edge only contains the spatial proximity information between two nearby targets.

One of the main concerns for optimizing the proposed BQP is the number of variables which correspond to potential locations that targets can occupy. Publicly available QP solvers such as CPLEX [43] or MOSEK [44], can handle up to a few dozen of targets. We show in our experiments that as the number of targets grows, the optimization becomes extremely inefficient. In some previous works the quadratic function is converted into a linear one by adding additional equality/inequality constraints to the objective function [45], one constraint for every pair of variables in the quadratic term. This may help reduce the complexity, but still it is not scalable to our problem

size and it will not have the advantage of having a soft quadratic constraint in the objective function. Additionally, adding millions of constraints for large number of targets is not desirable. We instead use the modified Frank Wolfe algorithm with SWAP steps to directly solve the quadratic objective function. In our experiments we show that the Frank-Wolfe with SWAP steps can handle sequences with up to a few hundred people. Lastly, we propose a straight-forward technique to reduce the number of candidates for further speed-up. We show that our speed-up technique reduces the computational complexity significantly without loss in the tracking accuracy.

1.4 Dissertation Organization

The rest of the dissertation is structured as follows: In Chapter 2, we review existing literature on single object tracking as well as multi-object tracking in various crowd densities. In Chapters 3 and 4, we present new approaches to data association and describe the GMCP-based and GMMCP-based multi-object tracking methods. Chapter 5 focuses on multi-target tracking in high-density crowd scenes and present our binary quadratic optimization framewrok.

CHAPTER 2: LITERATURE REVIEW

Tracking is one of the fundamental problems in computer vision. Tracking has been studied for decades and great advances have been made in this field. Hundreds of tracking methods have been proposed over the past decades with a focus on different applications. Large series of papers in this area focus on surveillance [1,37,38,46–50]. There is a series of papers with a focus on sport video analysis [18, 19, 51], where targets undergo abrupt motion changes and articulate widely. Multi-target tracking has also found its way into medical image analysis to help doctors [26–29] in drug discovery and studying body organs. Hand tracking [20–22] is another application of multi-target tracking method which has been used in augmented reality.

Tracking methods can be divided in to two main categories. One focuses on tracking one object only [52–55], while the goal of the trackers in the other category is to track multiple targets simultaneously. Although the focus of this dissertation is multi-object tracking, for the sake of completeness, we first review the prominent approaches in the area of single target tracking. Next, we review multi-target tracking methods with a focus on surveillance applications and highlight their differences with our methods.

2.1 Single-object tracking

In single object tracking, given the bounding box in first frame, the task is to establish the target location over a sequence of frames. In most single object tracking methods the goal is to build an appearance model to discriminate the target from the background. This model should be powerful enough to deal with several challenges, such as changes in illumination and scale change, occlusion, abrupt motion and background clutter. What makes this task more challenging is that only one training sample is provided to the tracking systems.

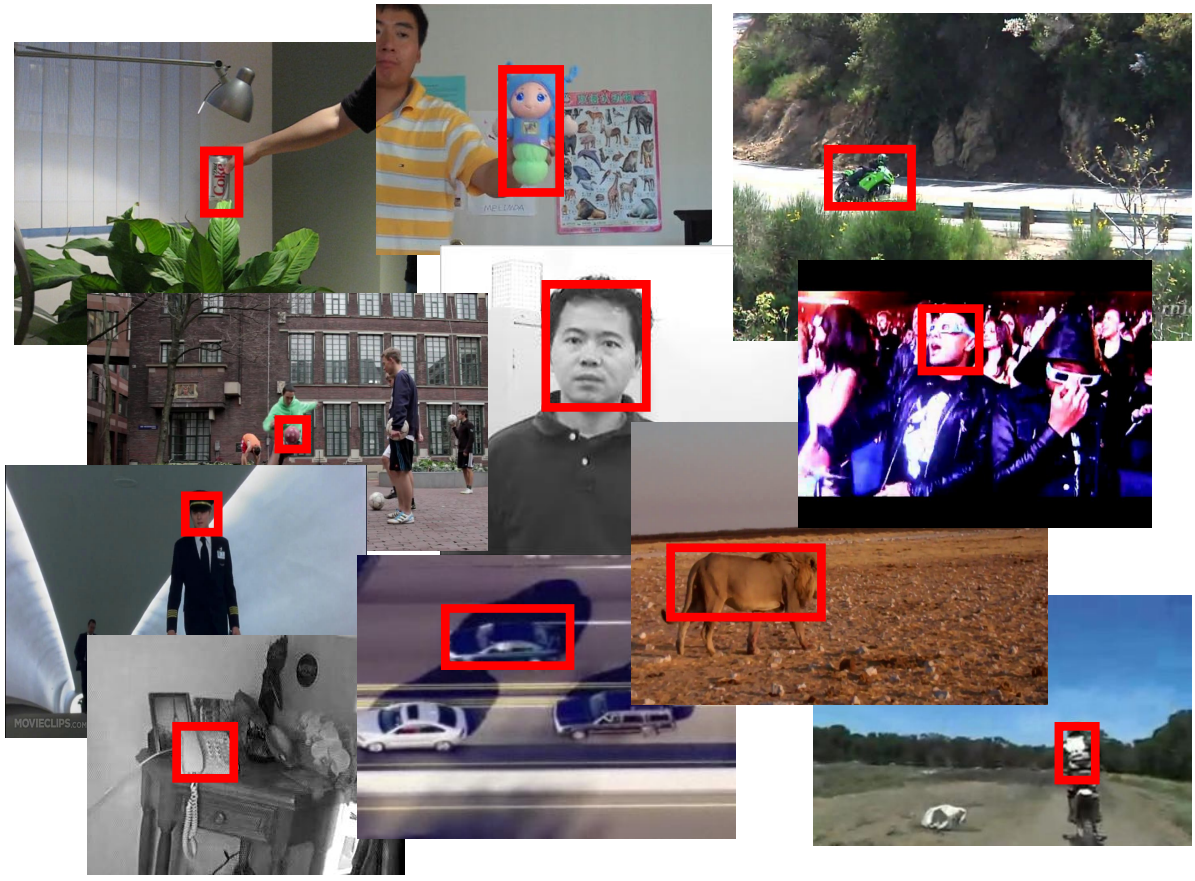


Figure 2.1: Examples of the sequences in tracking benchmark of ALOV++ [3]. The red box indicates the target that needs to be tracked.

Early works in single target tracking use a matching strategy between the target representation in the previous frame(s) and the current frame to track the target [52–54, 56–58]. Normalized Cross-Correlation [58] is one of the most basic matching techniques that is still being used as an underlying tracker in many systems. It is shown in [3] that the simple NCC-based tracker of [58] works better than some recent, more sophisticated trackers in the ALOV++ dataset in [3]. A mean-shift tracker [54] is another popular method. The simple tracker of [54] uses matching between RGB-color histograms of the target and candidate windows. Later mean-shift is used to find the mode of the function, the one that maximizes the Bhattacharyya distance. The tracker proposed

in [53] is a more recent version of such tracking algorithms that use matching. The method in [53] uses the parametrized Earth Mover's Distance between the average HSV values of super pixels extracted from the target region and each candidate window. Even though these methods are computationally efficient, they are too simple to handle real world scenarios.

All the above methods follow a generative tracking strategy. On the other hand, discriminative trackers have recently gained popularity due to their superior performance [55, 59–68]. Struck [55] is one of the most successful tracking methods that have been highlighted several times in recent studies [3, 69]. Hare et al. [55] introduced a structured SVM for single object tracking and reported significant improvements. Correlation filter-based tracking methods [61–63] are another example of successful tracking methods based on online discriminative learning. This is mostly due to low computational complexity of these methods which makes them suitable for many real-time applications. The trackers proposed in [65, 67] are specifically useful when it comes to long-term tracking. Kalal et. al in [65] proposed a tracker that employs two experts to identify the new location of the target, one based on optical flow and the other one based on the result of a detector. The tracker of Hong et. al [67] is another attempt to employ dual component tracker for long term tracking. Their method uses integrated correlation filter for short-term tracking as well as key point-matching-tracking and RANSAC estimation for the long-term tracking component.

Given the wide variety of aspects in tracking circumstances, having a large dataset covering these aspect is essential. However, all papers until very recently used less than ten video sequences in their experiments. Several recent studies such as [3, 69, 70] aim to advance this field by providing standardized quantitative evaluations and larger dataset. Smeulders et. al in [3] provided the largest tracking dataset, called ALOV++, which contains 315 videos covering 14 important aspects in tracking. Some of the aspects in ALOV++ include, illumination changes, occlusion, clutter, camera motion, low contrast and specularities. The authors in [3], tested 19 trackers on ALOV++ and compared their performances under several conditions. The survival curves of 19 trackers with respect to F-score are shown in Figure 2.2. We can see that the best performing tracker has a

F-score of only 66% which is far from being perfect in several scenarios.

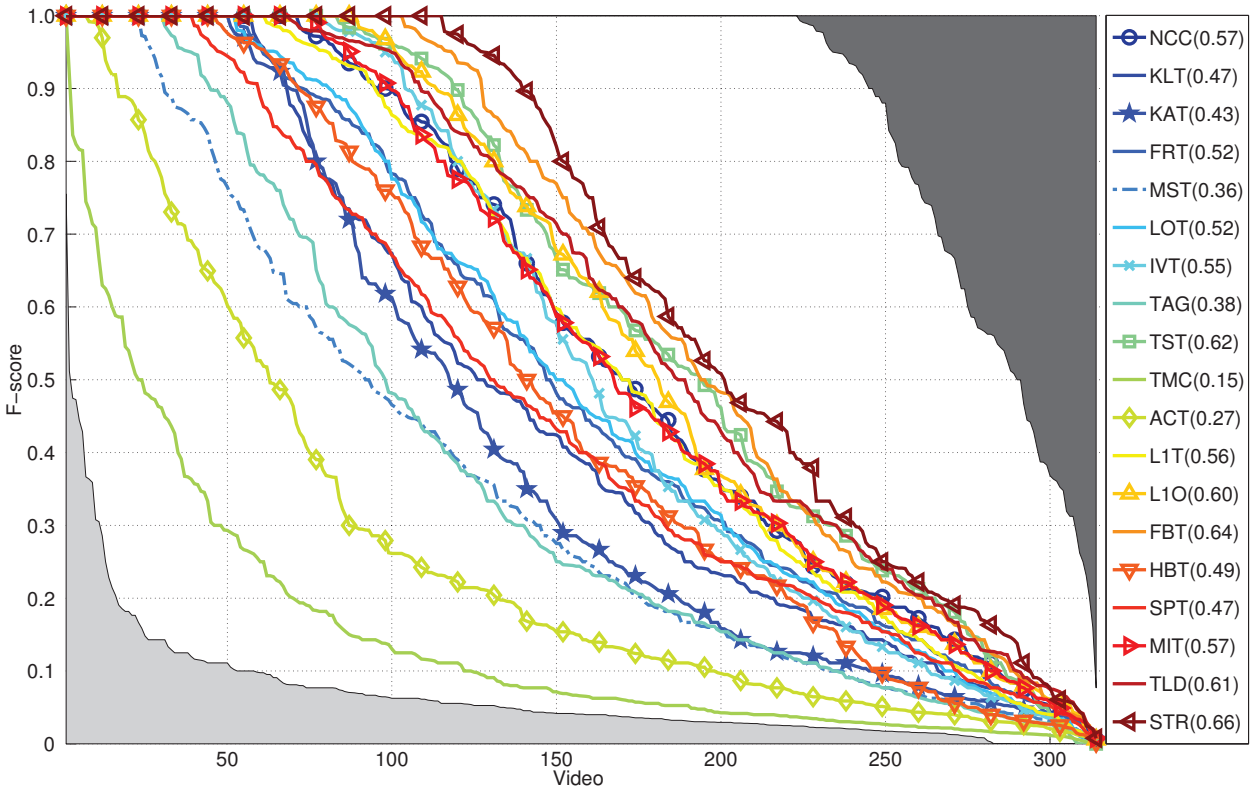


Figure 2.2: The survival curve of the 19 trackers in [3] with respect to F-score.

Due to these new datasets and uniform evaluations, significant progress has been made in this area over the past few years. For example the average F-score reported on ALOV++ has been already improved by 7% in [67]. Despite great advances made in single object tracking, one area that remains relatively unexplored is the extension of single target trackers to track multiple objects. These methods are especially important for scenarios where pre-trained object detectors do not perform well or the object of interest is not known a priori.

The work of Zhang and Maaten [71] is probably the first attempt to apply online discriminative learning in tracking multiple objects. In [71], the spatial constraint among the targets is modeled during tracking. It is shown that the tracker performs well for fixed number of targets and

only when the structure among the objects remains the same (or changes very slowly). However, this is only applicable to very limited scenarios and it will perform poorly in others, especially when the targets are moving independently. The work of Chen et. al [51] is another attempt towards this direction. The targets in [51] are annotated in the first frame and the association is being performed between supervoxels through a constraint sequential labeling process that assigns labels to the supervoxels. Dehghan et. al in [72] extended the structured SVM-based single target tracker [55] to track multiple targets. They used a variation of network flow graph to solve the inference of structured SVM. The results presented in [72] show the effectiveness of the method. However, the complexity increases significantly as the number of targets increase.

In chapter 5, we show how we successfully extend a single object tracker to track multiple targets in extremely crowded scenes. The proposed method, enables us to track hundreds of objects for which there do not exist good pre-trained detectors. Moreover it helps in better dealing with common challenges in dense crowds such as small apparent target size, similarly looking targets and high computational complexity. Our binary quadratic programing based tracker, achieves state-of-the-art accuracy on several challenging crowded sequences.

2.2 Multi-object tracking

Multi-target tracking is a significantly harder problem than single object tracking. In multi-object tracking, we not only need to discriminate the target from the background, but we also need to distinguish targets from each other by assigning separate IDs to them. Algorithms developed for Multiple Object Tracking (MOT) can be traced back to 1970s [73–77], when there was a primary focus on radar and sonar. The goal was to perform data association in real time between the given tracks and current observations. Until this time, the appearance of objects did not play a crucial role in the tracking of multiple objects with strong emphasis on target motion.

One of the oldest and most popular tracking filters introduced in the sixties is the Kalman

filter [78]. The Kalman filter provides an analytic way to estimate the state of a linear dynamic system from a set of noisy measurements. The exact solution to the probabilistic model is found if the observation noise is assumed to be a zero mean Gaussian distribution. Kalman filter assumes a linear model for both appearance and motion of the target. However, a more complex version of the Kalman filter called the extended-Kalman filter [79] was introduced eliminating the previous linear assumption. The existence of non-Guassian noise as well as non-linearity of many systems led to the design of another popular tracking technique, called particle filter. Particle filter tracking methods can better deal with non-Guassian noise. However, the computational complexity of these methods is higher than the Kalman filter and extended-Kalman filter. One common drawback of the above-mentioned methods is that, although they are used in many multi-tracking systems, they are designed to solve a dynamic system for only one target at a time.

Over the past decade, the popularity of tracking-by-detection methods have increases greatly. In tracking-by-detection, first detection hypotheses in every frame are provided using an object detector. The detections that belong to the same person are later linked together, during the data association step, to form the final trajectories. The popularity of these methods is mostly due to the great progress of object detection methods [30–33] in detecting different categories such as person and car. These categories have been the focus of most multi-object tracking methods. Some recent papers such as the work of Shu et. al [1] focused on improving the performance of a pre-trained object detector to better deal with occlusion. Using the improved detections, they have shown they can achieve better tracking results. Authors in [80] proposed a model based on bag-of-word of superpixels that incorporates visual information of a sequence such as color to improve the performance of a generic object detector. While few methods have focused on improving the accuracy of existing object detectors, most multi-object tracking methods, on the other hand, have focused on designing a better data association technique. While there exist many data association methods, one can divide them into two main classes.

2.2.1 *Local Association*

These methods are temporally local, which means they consider only a few frames while solving the association problem. The best example of such approaches is bi-partite matching and its extensions [1, 35, 76, 81]. In [81], the association probabilities are computed jointly across all targets to deal with ambiguities in association. Shu et. al in [1] use a greedy approach to combine the responses of part detectors to form a joint likelihood model of multiple cues to associate detections and object hypotheses. Whilst this class of methods are computationally inexpensive, their assumption to use only a few frames during data association makes them prone to ID-switches. A more sophisticated version of these data-associaton methods is called multiple hypothesis tracker (MHT) [76]. The idea here is to keep all possible association from the past observations in memory and pick the best assignment given current detections. Even though MHT uses more information from the past compare to bipartite-matching, it still uses only the very next frame. Moreover, due to exponential growth in hypotheses it is nearly impossible to use these methods in practice without efficient pruning steps.

2.2.2 *Global Association*

To better deal with the above problems, another set of data association techniques have recently received a lot of attention. In global association methods, the number of frames is increased and sometimes the entire video is processed at once to infer the tracks [10, 12, 36, 38, 48, 49, 82–85]. These methods are suitable for offline tracking where the entire data is available. In practice, sometimes processing the entire video at once is not possible. Thus the video is divided into segments/clips and each clip is processed at once. Although, global data association methods are not suitable for online processing, they can theoretically obtain the global optimal solution.

Increasing the temporal locality in global data association methods allows one to incorporate more global properties of targets during optimization. One group of global data association

methods assume a simplified version of the problem, where only the relationship between observations in consecutive frames is taken into account. These methods are normally followed by an optimization for which there exists an exact solution. Such methods are computationally efficient and plausible for many applications. Network flow algorithm [12, 36, 37] 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. [36] showed that the exact solution to network flow problem can be found in a polynomial time through push relabel algorithm. Pirsiavash et al. in [37] showed that a high quality sub-optimal solution can be found using dynamic programing. The authors in [12] also model multi-object tracking using network flow and propose a linear programing solution to that. Different variations of network flow have also been used in MOT recently [86, 87]. Authors in [86] incorporate constant velocity motion model in network flow graph and proposed a Lagrangian relaxation solution to min-cost max-flow problem. The work of [87] presents a multi-commodity network flow to better incorporate the appearance consistency between a group of people during tracking and reduces ID-switch.

Another group of data association methods assume no simplification in problem formulation and consider a model which is closer to the tracking scenarios in the real world. However, due to the complexity of their models, the proposed solutions are approximate [13, 38]. Milan et al. in [38] 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.

Besides data association and online learning of detectors, high-order social constraints have recently caught more attention in the context of MOT. As social agents, people make decisions about their future steps based on not only individual desires, but also based on the interactions with other people around them. Pellegrini et. al. in [88] proposed a dynamic social behavior model

by taking into account people's environment for collision avoidance. But the important social grouping factor is ignored. Authors in [89] designed an explicit energy function based on social behavior model incorporating individual and grouping factors. Similarly Leal-Taixé et. al [90] incorporates social and grouping within a global optimization framework which is solved trough linear programing. Authors in [91] developed a group CRF model to estimate group memberships and individual target trajectories jointly. The experiments show the positive influence of social grouping on the tracking results. Qin et. al. [92] treated the social grouping behavior as a high-order factor and incorporated it into the general data-association based tracking. The quality of social grouping is measured by the distance between a trajectory with the group mean trajectory. In all these methods, the group memberships are considered static during the entire tracking procedure and group merge and split are not handled. Qin et. al. [93] are able to handle group merge and split, but their method needs tracklets generation as a first step and group structure is not modeled. In recent work of [94], authors remove hand-specified social force models and propose to learn the features that capture the relationship between motion of the targets.

The data association methods proposed in this dissertation are different from previous approaches in several aspects. The proposed data association technique in both GMCP (Chapter 3) and GMMCP (Chapter 4) trackers forms a k-partite complete graph from the detection candidates. This is different from previous works that tried to reduce the pairwise relationships between the targets to simplify the problem. The algorithms presented herein are more accurate way of formulating multi-target tracking and allows the modeling of appearance and motion in a global manner. Both GMCP and GMMCP are NP hard problems. In the GMCP tracker, a greedy solver based on local neighborhood search, which even though gives significant improvements over the state-of-the-art, is still prone to local minima. Moreover GMCP is missing joint optimization of target location which is essential in some scenarios to avoid ID-switch. The aforementioned problems are solved in the GMMCP tracker. GMMCP tracker uses a new graph theoretic problem that allows joint selection of all the targets simultaneously. We propose a Mixed Binary Integer Programing

solution that is able to find the tracks in real-time for small and medium density crowded scenes (up to few dozen people).

2.3 Crowd Tracking

Target tracking in a highly crowded scene is a relatively new area of research, and only a few papers have focused on this problem [2, 5, 6, 34]. The methods proposed in [2, 5, 6, 34] track each target separately by training an online tracker for each individual separately. Ali and Shah [5] proposed using high-level knowledge about the scene in the form of floor fields to improve tracking in crowded scenes. The floor fields along with appearance information are used for tracking. The method in [5] fails when the crowd is dynamic, when there are anomalies or when the camera moves. The series of papers by Kratz and Nishino [95–97] follow a similar approach. They learn the motion patterns and then use them as prior information to improve the track of each individual. Rodriguez et al. [6] proposed a Correlated Topic Model to model crowd behavior at each location. Each scene is modeled using a set of behavior proportions, where behaviors represent distribution of low-level motion features. Their model does not have the assumption of [5] and can handle the multi-modality in crowd behavior. However, it still requires learning the dynamics of the scene and is prone to the above-mentioned problems. In their approach, words correspond to low level quantized motion features and topics correspond to crowd behaviors. Bera et. al in [34] propose a real-time tracker that uses a mixture of motion models, to deal with complex movements in crowded scenarios. The recent work of [2] is the closest to ours and addressed some of the problems with previous works. In that approach no prior assumption about the scene is considered. Instead contextual information is incorporated in a greedy manner to improve tracking. Although contextual information was used before in tracking [98], its extension to high-density crowds was explored first in [2].

One major draw-back of previous crowd trackers, such as [2, 5, 6, 34], is that they track

each target separately, lacking a joint optimization of target tracks. One of the main reasons for that is the complexity of joint optimization techniques. When dealing with hundreds of people, modeling the interaction of targets becomes cumbersome and finding an efficient optimization is quite challenging. When tracking one target at a time, we are limited to using information from that target only. This makes it impossible to model the interaction between the targets. In order to overcome this limitation, previous works have either used the prior information from the scene (which is not available and applicable all the time) or have tried to model interactions in a greedy manner by using information from other tracks [2]. Given the above issues, it is crucial that multiple object tracking in crowded scenes should be formulated such that all target tracks are optimized simultaneously. This allows one to model the interactions between targets and include assumptions that are fundamental in modeling behavior of targets in physical world. In order to address the aforementioned problems, we present an online multi-object tracking framework where all the targets are tracked simultaneously. Our method provides a flexible formulation where one can include different individual and contextual terms.

Our binary quadratic formulation consists of three linear terms and two quadratic terms. Two linear terms capture the properties of the individual tracks. The third linear term as well as the quadratic terms are responsible for modeling interactions between the targets. We show that the proposed quadratic objective function could be solved efficiently using an accelerated version of the modified Frank-Wolfe algorithm which takes advantage of *SWAP* steps for further speed up [99].

Frank-Wolfe optimization algorithm was introduced in 1956 to solve constraint quadratic programing, which has been revisited recently and used in many machine learning applications [99–101]. Authors in [101] used Frank-Wolfe with away steps and proposed a faster optimization strategy for structured support vector machine. Allende et. al in [99] showed that Frank-Wolfe could be applied to solve the quadratic programing in support vector machine and achieved significant speed up for large scale problems compared to other methods such as projected gradient de-

scent. Moreover, the authors in [102] adopted Frank-Wolfe with away steps to solve the quadratic objective function for image/video co-localization. In our work, we show that commercial software such as CPLEX [43] and MOSEK [44] which use Barrier Optimization techniques are not able to handle a large number of people. The accelerated FW, that we use in our work, not only can solve the problem efficiently for large numbers of targets, but also is faster than commercial software and other versions of Frank Wolfe used in [100–102].

Finally, due to the difficulty of human detection in dense crowds, and in order to keep the primary focus on tracking, all previous works in this area [2, 5, 6, 34] assume that a manual initialization of templates on individuals in the crowd is afforded to the algorithm. The template refers to a bounding box around the individual we intend to track. For our approach, like previous works, we also assume that initial templates (bounding boxes) are provided and our goal is to track them across the scene.

2.4 Summary

In this chapter we first reviewed the prominent approaches in the area of single target tracking. Next we reviewed multi-target tracking methods which focus on surveillance applications. We categorized these methods based on the crowd density of the scenarios they can be applied to. We also reviewed methods that incorporate social constraints in their tracking formulation. In the next chapter we present our data-association-based tracking method using Generalized Maximum Clique Problem.

CHAPTER 3: GMCP TRACKER: GENERALIZED MAXIMUM CLIQUE PROBLEM FOR MULTI-OBJECT TRACKING

In this chapter, we propose our global data association method for multi-target tracking that utilizes Generalized Maximum Clique Problem (GMCP) formulation. Our formulation incorporates both appearance and motion in a global way. In the proposed framework, we incorporate the whole temporal span of the sequence into the data association problem, but we focus on one object at time rather than addressing all of them simultaneously. This is to avoid dealing with an extremely complex optimization problem. Although we focus on solving the data association problem for one object at time, we also incorporate all the other objects implicitly. Therefore, our approximation in the object-domain is significantly less restrictive than those used by other approximate methods, such as limited-temporal-locality. This is because the limited-temporal-locality methods are literally *blind* to the information outside of the temporal neighborhood they are focused on, while the proposed method incorporates the whole approximation domain, i.e. all objects, implicitly.

Given human detections [33] in frames of a video, our method is consist of two main steps, calculating mid-level tracklets in video segments (subsec. 3.1.1) and merging them into trajectories(subsec. 3.1.2). We utilize Generalized Minimum Clique Graphs to solve our data association problem (subsec. 3.1.1.1). We tested the proposed method on the diverse sequences of Town Center [103], TUD-corssing [104], TUD-Stadtmitte [105], PETS2009 [106], and Parking Lot sequence [1], all with promising results.

3.1 GMCP-Tracker

The block diagram of the proposed global data association algorithm is shown in Fig. 3.1. The first step is to detect the humans in each frame. We used Felzenszwalb et al.'s [33] part-based

human detector. However, any other detector could be used. Next, we divide the input video into a number of segments and find the mid-level tracklets of pedestrians within each segment using the proposed global method for tracklet generation utilizing GMCP. In the last step, we merge the mid-level tracklets found in all of the segments to form the trajectory of each person over the course of the whole video.

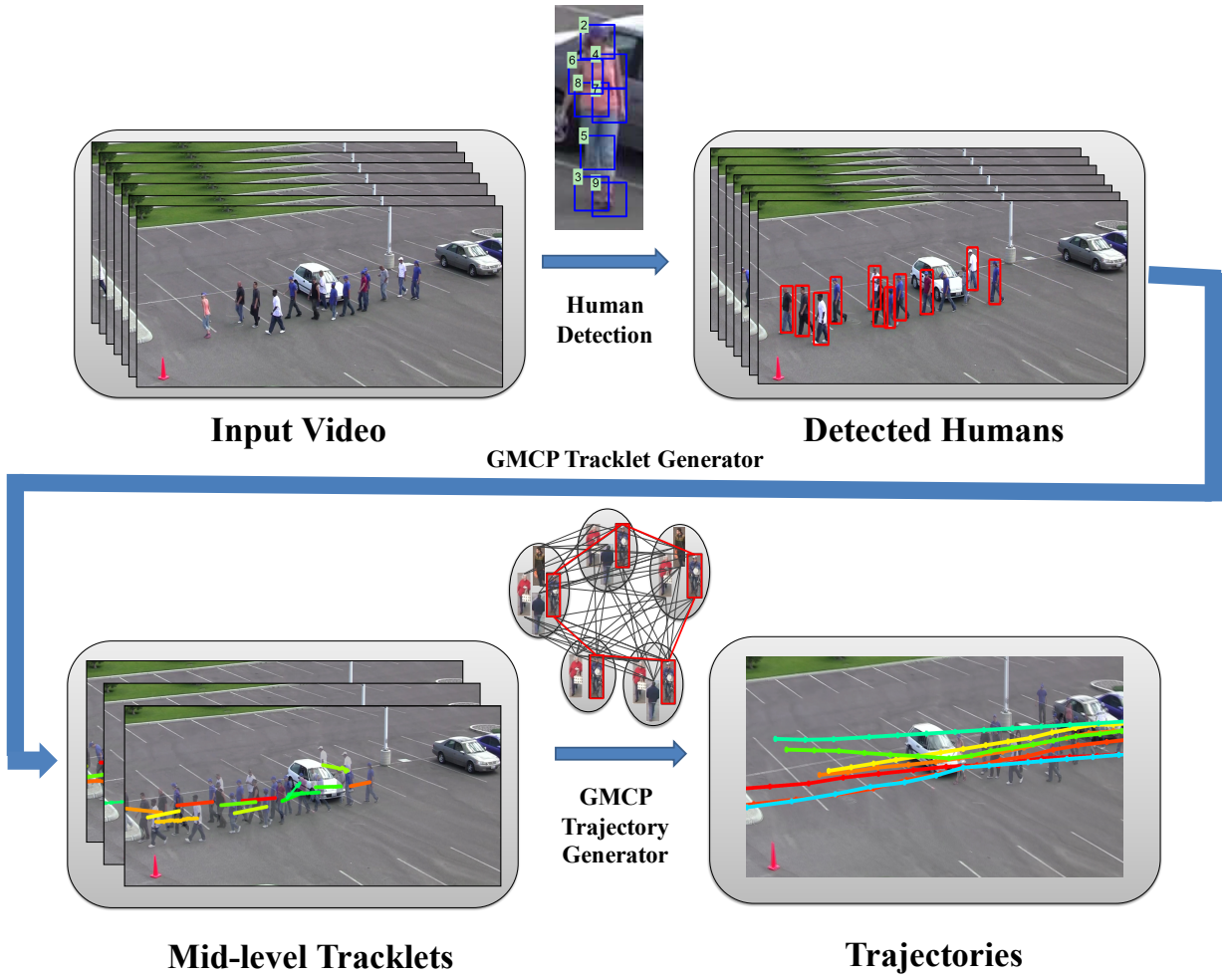


Figure 3.1: The block diagram of the proposed GMCP tracker. We followed a hierarchical tracking framework, where detections are first linked together to form short tracks, called mid-level tracklets. These mid-level tracklets are used in the second level of data association to form the final trajectories. In both steps the data association is done using Generalized Maximum Clique Graphs.

Despite the appearance of the pedestrians remaining rather consistent throughout a video, the pattern of motion tends to differ significantly in short and long term. In principle, it's difficult to model the motion of one person for a long duration without having the knowledge of the destination, structure of the scene, interactions between people, etc. However, the motion can be modeled sufficiently using constant velocity or acceleration models over a short period of time. Therefore, the way motion is incorporated into the global data association process should be different in short and long terms. This motivated us to employ the hierarchical approach, i.e. finding mid-level tracklets first and then merging them into full trajectories.

3.1.1 Finding Mid-level Tracklets using GMCP

We divide a video into s segments of f frames each. We propose a data association method for finding mid-level tracklets which are globally consistent in terms of motion and appearance over the course of a segment. The input to our data association problem for finding mid-level tracklets is a graph $G = (\vec{V}, E, w)$, where $\vec{V}$, E and w denotes the set of nodes, set of edges and weights of edges, respectively. $\vec{V}$ is divided into f disjoint clusters. Each represents one frame, and the nodes therein represent the human detections in that particular frame. Let C_i, where $i \in \mathbb{Z} : 1 \leq i \leq f$, denotes the frame ($\equiv$cluster) i, and v_m^i denotes the mth detection ($\equiv$node) in the ith frame. Therefore $\vec{C_i} = \{v_1^i, v_2^i, v_3^i, ...\}$. The edges of the graph are defined as $E = \{(v_m^i, v_n^j) | i \neq j\}$, which signifies that all the nodes in G are connected as long as they do not belong to the same cluster. A node, v_m^i, is associated with a location feature, $\vec{x}_m^i$, which is the 2-dimensional spatial coordinates of the center of the corresponding detection and appearance features, $\underset{l}{\vec{\phi}_m^i}$, which represents the color histogram of the lth body part of the detection v_m^i. The weight of an edge between two nodes, $w : E \rightarrow \mathbb{R}^+$, represents the similarity between the two corresponding detections:

$$w(v_m^i, v_n^j) = \sum_{l=1}^{8} k(\underset{l}{\vec{\phi}_m^i}, \underset{l}{\vec{\phi}_n^j}), \tag{3.1}$$

where k represents histogram intersection kernel.

The task of finding the mid-level tracklet of one particular person in a segment requires identifying the detections of that person in each frame. Therefore, a feasible solution to this problem can be represented by a subgraph of G in which one node ($\equiv$detection) is selected from each cluster($\equiv$frame). We call this subgraph which represents a feasible solution $G_s = (\vec{V}_s, E_s, w_s)$. Therefore, G_s contains a set of nodes which includes the general form $\vec{V}_s = \{v_a^1, v_b^2, v_c^3, ...\}$ which means the ath node from 1st cluster, bth one from 2nd cluster, and so on are selected to be in $\vec{V}_s$. By definition, $E_s = \{E(p,q)|p \in \vec{V}_s, q \in \vec{V}_s\}$ and $w_s = \{w(p,q)|p \in \vec{V}_s, q \in \vec{V}_s\}$. Bear in mind that the feasible solution G_s represents the mid-level tracklet of *one person* and not all the people visible in the segment. Fig 3.2 (a) shows the human detections in a small segment of 6 frames along with the graph G they form in (b). (d) shows a feasible solution G_s with the mid-level tracklet it forms in (c). Since the set of nodes $\vec{V}_s$ is enough to form G_s, we use $\vec{V}_s$ to denote a feasible solution hereafter.

We define the appearance cost of the feasible solution V_s as:

$$\gamma_{appearance}(\vec{V}_s) = \frac{1}{2}(\sum_{i=1}^{f} \sum_{j=1, j\neq i}^{f} w(\vec{V}_s(i), \vec{V}_s(j))), \tag{3.2}$$

which is the cost of the complete graph induced by the nodes in $\vec{V}_s$. Eq. 3.2 is a global cost function since it is based on comparing all pairs of detection in a feasible solution no matter how close they are temporally. This is based on the assumption that the appearance of people does not change drastically in a segment. Overlapping bounding boxes, occlusion, noisy descriptors, background clutter, etc in part of a trajectory can potentially cause ID-switches in the majority of current methods, in particular the ones using limited-temporal-locality. The formulation defined in eq. 3.2 minimizes the chance of such cases of ID-switches, as all possible pairs of detections are compared regardless of their temporal order.

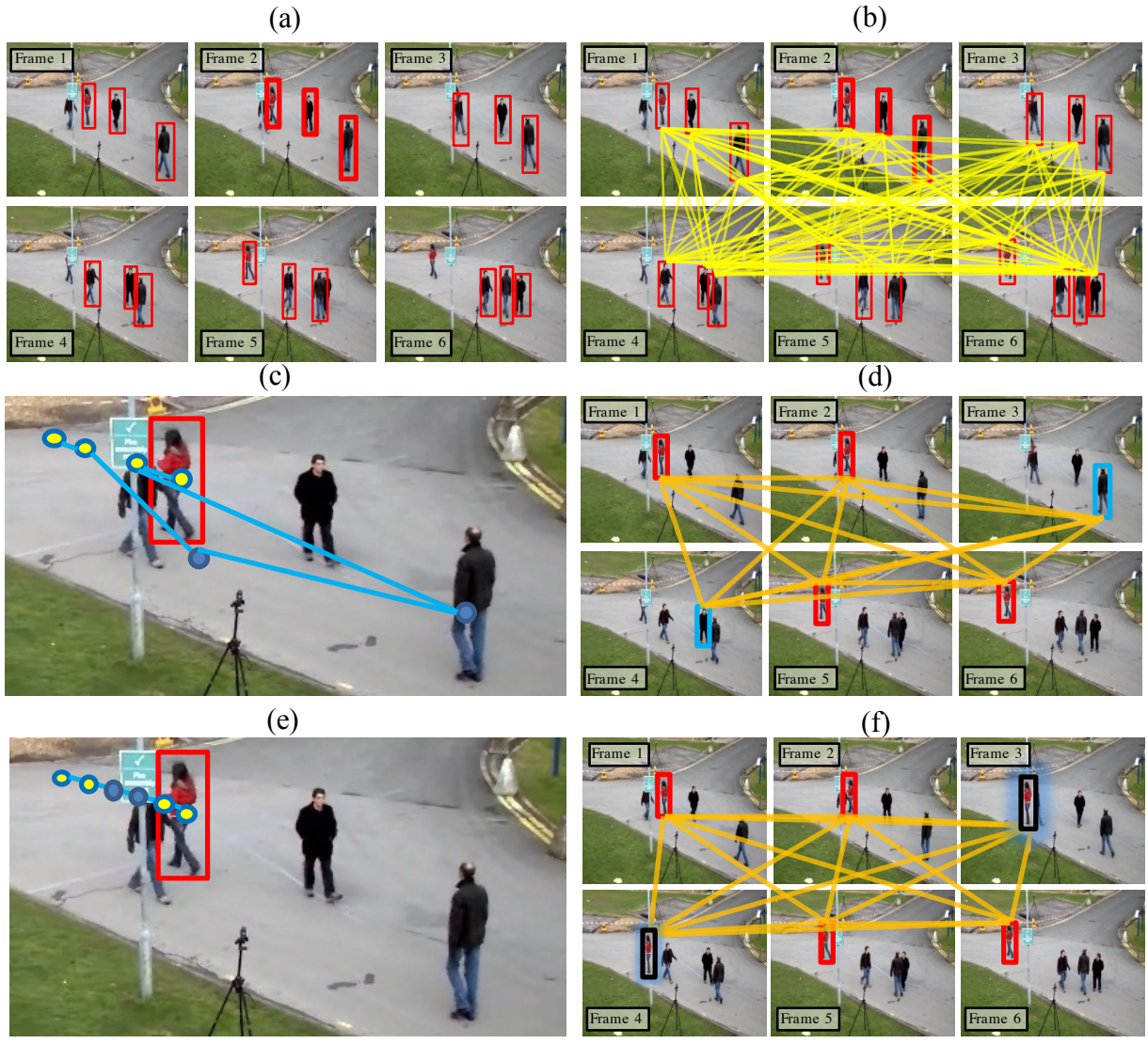


Figure 3.2: Finding a mid-level tracklet for a small segment of 6 frames. The first row (Figures (a) and (b)) shows the detections in each frame along with graph G they induce. The middle row (Figures (c) and (d)) shows the feasible solution with minimal cost along with the tracklet it forms, *without* adding hypothetical nodes. The third row (Figures (e) and (f)) shows the feasible solution with minimal cost *with* hypothetical nodes added for handling occlusion, along with the tracklet it forms.

By finding the feasible solution with the minimum appearance cost, i.e. $\underset{\vec{V}_s}{\operatorname{argmin}}(\gamma_{appearance}(\vec{V}_s))$, the mid-level tracklet of the person with the most stable color histogram features in the segment will be found. In the following subsection, we explain that Generalized Minimum Clique Graph is a perfect fit for our problem, and can be used for solving the aforementioned optimization task for finding mid-level tracklets.

3.1.1.1 Generalized Minimum Clique Problem (GMCP)

Generalized Graph Problems, more formally known as Generalized Network Design Problems [107], are a class of problems which are commonly built on generalizing the standard subgraph problems. In these problems the definition of a node is expanded to a cluster of nodes. An example of such problems is the standard Traveling Salesman Problem (TSP) where the goal is to find the minimal Hamiltonian cycle which visits all the nodes of the input graph exactly once. In the *Generalized* Traveling Salesman Problem, the nodes of the input graph are grouped into disjoint clusters, and the objective is to find the minimal Hamiltonian cycle which visits all the clusters of the input graph exactly once. From each cluster, exactly one node should be visited.

Similarly, in the Generalized Minimum Clique Problem (GMCP) the nodes of the input graph are grouped into disjoint clusters. The objective is to find a subset of the nodes that include exactly one node from each cluster, while requiring the minimum cost for the complete graph they produce [107]. Recently, GMCP has been used in the fields of biology and telecommunications [107] as well as computer vision [108]. However, its potential applications in Computer Vision have not been studied to date.

In order to have a more formal definition for GMCP, assume a graph $G = (\vec{V}, E, w)$ exists, where G is undirected and weighted, $\vec{V}$ is the set of all nodes, E is the set of edges, and $w : E \rightarrow \mathbb{R}^+$ is the weight of a given edge. The set of nodes V is divided into f clusters $\vec{C}_1, \vec{C}_2, \ldots, \vec{C}_f$ such that all of the clusters are completely disjoint: $\vec{C}_1 \cup \vec{C}_2 \cup \ldots \cup \vec{C}_f = V$ and $\vec{C}_1 \cap \vec{C}_j = \emptyset$ $(1 \leq i \neq j \leq k)$. A feasible solution of the GMCP instance is a subgraph

$G_s = (\vec{V}_s, E_s, w_s)$, where $\vec{V}_s$ is a subset of $\vec{V}$ which encompasses only one node from a given cluster. E_s is a subset of E which includes the nodes V_s induces, and w_s is their corresponding weights from w. The goal of the GMCP is to find the feasible solution with the minimal cost, where the cost is defined to be the sum of all the weights along the solution subgraph.

In this formulation, there exists an edge in E for all possible pairs of nodes of $\vec{V}$, as long as they do not belong to the same cluster. Therefore, the subgraph G_s is essentially complete, which makes any feasible solution of GMCP a clique.

As can be seen from the formulation of our data association problem explained in 3.1.1, GMCP essentially solves the same optimization problem, we need to solve for finding a mid-level tracklet if the input graph G is formed as explained in 3.1.1. Therefore, by solving GMCP for the graph G, the optimal solution which corresponds to the feasible solution with the most consistency in appearance features over the course of the segment, i.e. $\underset{\vec{V}_s}{\operatorname{argmin}}(\gamma_{appearance}(\vec{V}_s))$, is found. More details about solving GMCP will be discussed in section 5.4.

In order to incorporate motion, as well as appearance, into the data association problem, we add one more term to the cost function and define our global data association task as the following optimization problem:

$$\hat{\vec{V}}_s = \underset{\vec{V}_s}{\operatorname{argmin}}(\gamma_{appearance}(\vec{V}_s) + \alpha.\gamma_{motion}(\vec{V}_s)), \tag{3.3}$$

where $\hat{\vec{V}}_s$ is the optimal solution to determine the data association for *one* mid-level tracklet, and α is the mixture constant which balances the contribution of appearance and motion.

Finding $\hat{\vec{V}}_s$ by solving eq. 3.3 yields the mid-level tracklet of *one* person in the segment. Therefore, in order to find the mid-level tracklets of all the pedestrians in the segment, the optimization problem of eq. 3.3 has to be solved several times. The first time eq. 3.3 is solved, the algorithm finds the mid-level tracklet which has the lowest total cost, i.e. the most stable appearance features and most consistent motion with the model. Then, the vertices selected in $\hat{\vec{V}}_s$ are

excluded from G and the above optimization process is repeated to find the mid-level tracklet for the next person, and so on. This process is repeated until zero or few nodes remains in G.

Since the algorithm finds the mid-level tracklets in order of how stable and consistent their appearance and motion features are, the mid-level tracklets which are less likely to be confused are calculated and excluded from G first. Therefore, our method does not lower the chance of successful extraction of the mid-level tracklets found at the last iterations. Our global motion-cost model, which defines the term $\gamma_{motion}(\vec{V}_s)$, is explained in the next subsection.

3.1.1.2 Tracklet-global motion cost model

In order to incorporate motion into the optimization process of eq. 3.3, we need to compute a cost for the feasible solution $\vec{V}_s$ based on motion. The spatial velocity vector for the feasible solution $\vec{V}_s$ is defined as: $\dot{\vec{X}}_s(i) = \vec{X}(i+1) - \vec{X}(i)$, where $1 \leq i \leq (f-1)$. One common approach to computing the motion cost is to calculate the deviation from a presumed model, such as constant velocity. This can be done by using each velocity vector to predict the spatial location of the detection immediately after it, and summing up the errors between the predicted locations and corresponding locations in the feasible solution. This piecewise approach is mainly used in bipartite matching and similar approaches [1, 35]. However, in our global framework, one feasible solution is meant to represent one mid-level tracklet over the course of the whole segment; therefore we can calculate the motion cost in a more effective way, which assures both piecewise and global consistency within the model. We assume the constant velocity model for the motion of pedestrians in one segment and calculate the motion cost as:

$$\gamma_{motion}(\vec{V}_s) = \sum_{i=1}^{s} \sum_{j=1}^{s-1} \overbrace{|\vec{X}_s(i) - \underbrace{[\vec{X}_s(j) + \dot{\vec{X}}_s(j).(i-j)]}_{prediction}|}^{deviation}, \tag{3.4}$$

where the term in brackets in eq. 3.4 is the predicted location for the node $\vec{V}_s(i)$ using $\dot{\vec{X}}_s(j)$. In eq. 3.4, we assumed a person moves at a constant velocity manner in one segment, and each element of $\dot{\vec{X}}_s$ vector is used to predict the location of all other nodes in the feasible solution $\vec{V}_s$.

Fig. 3.3 shows this in more detail. Fig. 3.3 (a) shows a feasible solution which is being generated for the person with the red boundary. However, three detections of another person are mistakenly selected in the feasible solution. Therefore, we expect the three wrong selections to add a large value to the motion cost, while the rest of the selected nodes, which are consistent, to add low cost values. The value of eq. 3.4 is shown for two nodes of $i = 6$ and $i = 3$ in parts (b) and (c), respectively. The black circles show the predicted locations for the node i. The red lines depict the distance between the predicted locations and $\vec{X}_s(i)$, which shows the deviation from the model. The value of node i, which adds to the motion cost, is the sum of these distances. As can be seen, the node $i = 6$, which is not consistent with the majority of the tracklet, adds a large value to the cost, while $i = 3$ adds a lower value.

Therefore, in contrast to the piecewise motion models, the motion cost in eq. 3.4 is calculated by measuring the deviation from the constant velocity model in a tracklet in a global manner, because all nodes are contributing to the cost of the other nodes. Although, we use the constant velocity model in eq. 3.4, the extension to the constant acceleration and higher order models for more complicated scenarios is straightforward.

3.1.1.3 Handling occlusion using Hypothetical Nodes

In some cases, a given frame may not include a detection for a particular person due to occlusion, missed detection, etc. In order to cope with this issue, we add a Hypothetical Node to each cluster, thus if one frame does not include an appropriate detection, the hypothetical node is selected. Fig. 3.4 (a) shows an example of 3 occluded frames in a segment of 13 frames.

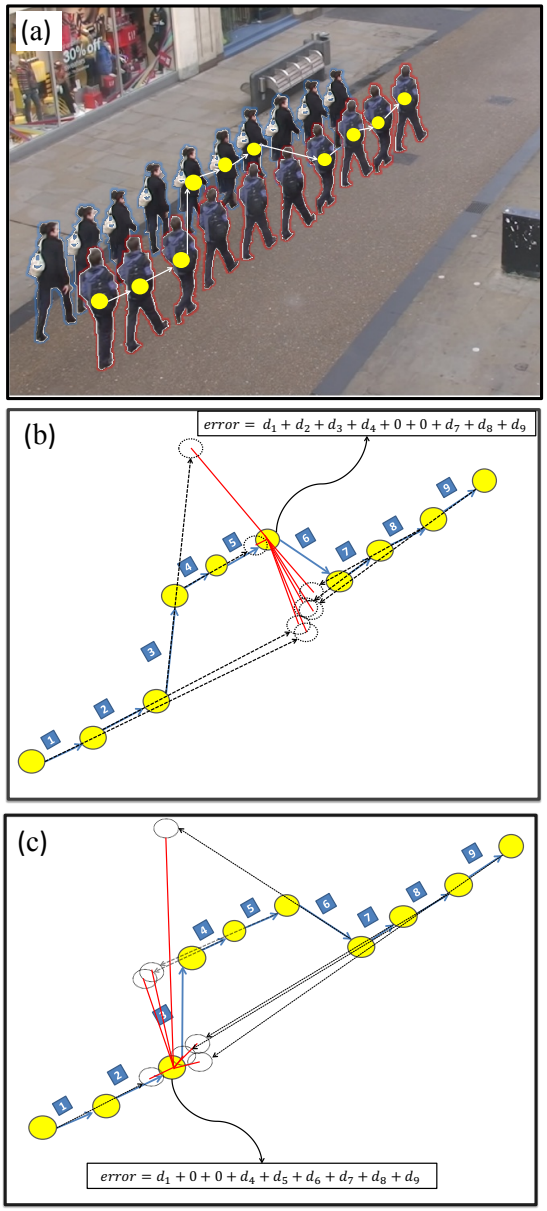


Figure 3.3: Tracklet-Global motion cost. (a) shows the mid-level tracklet of a feasible solution with three outliers. (b) and (c) show the cost for an outlier and inlier, respectively. It is clear, that the error increases as the number of outliers increases.

We need to define appearance and motion features for the hypothetical nodes, as each node in G has these two features. Solving the optimization problem of eq. 3.3 is an iterative process.

In each iteration these two features for the Hypothetical Nodes are re-estimated using the method explained in the rest of this subsection and the hypothetical nodes are updated.

$\vec{V_s}$ is expected to include the detections for one person (inliers), along with other detections (outliers) for the frames, which do not include a detection for that particular person. Due to the short temporal span of one segment, we assume a person moves at a constant velocity in a segment and use this assumption in order to identify inliers and outliers in $\vec{V_s}$. According to 2-dimensional constant velocity motion, the spatial location of the detections can be modeled using $\vec{X_s}(i) = \vec{a_1}i + \vec{a_o}$, where $\vec{a_1}$ and $\vec{a_0}$ are 2-dimensional constant vectors. Therefore, we identify the inliers and outliers to the constant velocity model determined by $\vec{a_1}$ and $\vec{a_0}$ using:

$$\vec{V_s}^{inliers} = \{\vec{V_s}(i) : |\vec{a_1}i + \vec{a_0} - \vec{X_s}(i)| < \delta\}, \tag{3.5}$$

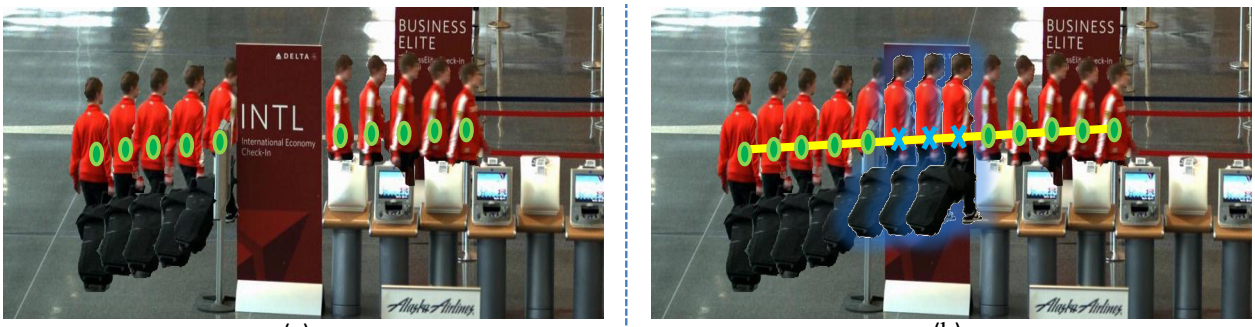
where δ is the fitting tolerance. The best parameters of the tracklet's constant velocity model are the ones which maximize the number of inliers:

$$\hat{\vec{a_1}}, \hat{\vec{a_0}} = \underset{\vec{a_1}, \vec{a_0}}{\textit{argmax}}(\#\{\vec{V_s}^{inliers}\}), \tag{3.6}$$

where $\#$ represents the cardinality of the set. Since V_s is composed of inliers and outliers, we employ $RANSAC$, which is a robust estimation method, to compute $\vec{a_1}$ and $\vec{a_0}$ in eq. 3.6 with the error criterion of eq. 3.5

The inlier nodes in V_s are those which survive the criterion of eq. 3.5 using $\hat{\vec{a_1}}$ and $\hat{\vec{a_0}}$. The spatial coordinates of the hypothetical nodes are computed using the estimated model $\vec{x}^i_H = \hat{\vec{a_1}}i + \hat{\vec{a_0}}$, where $\vec{x}^i_H$ denotes the spatial location of the hypothetical node of cluster i. The appearance feature of the hypothetical nodes is the average appearance of the inlier nodes' appearances (mean of their color histograms). However, a constant penalty is added to the weights of the edges connected to the hypothetical nodes in G, in order to avoid selecting the hypothetical node if the frame includes a proper detection.

As mentioned earlier, the hypothetical nodes are updated at the end of each iteration when solving the optimization problem of eq. 3.3. In the first few iterations, hypothetical nodes are not likely to be selected as the algorithm is still selecting the existing detections. However, as the optimization process progresses, the clusters which include correct detections are exhausted and the hypothetical nodes will start to contribute until the algorithm converges to the final solution $\hat{V}_s$. Fig. 3.2 (f) shows $\hat{\vec{V}}_s$ two hypothetical nodes selected for the frames with occlusion. The trajectory $\hat{\vec{V}}_s$ forms is shown in (e).



(a) (b)

Figure 3.4: Hypothetical Nodes: (a): The detections for one person in a segment of 13 frames. The person is fully occluded in three frames. (b): Hypothetical detections are shown at their spatial locations for the frames with occlusion.

3.1.2 Merging mid-level tracklets into trajectories using GMCP

As explained earlier, we divide the video into s segments and find the mid-level tracklets of all the pedestrians in each segment using the method described in subsection 3.1.1. In order to generate a trajectory of a person over the course of the full video, we need to merge the mid-level tracklets belonging to each person. This is a data association problem for which we can use any available data association method, such as bi-partite matching [1, 35]. However, in order to have a fully global framework, we use the same GMCP-based data association method we used for finding mid-level tracklets to merge them. Therefore, the clusters and nodes in G' now represent segments

and mid-level tracklets, respectively (vs. representing frames and human detections in 3.1.1)[1]. The appearance feature of a node, which represents one mid-level tracklet, is defined as the average appearance of the human detections in the mid-level tracklet (the mean of their color histograms), and its spatial location, $\vec{x'}^{i}_{m}$, is defined as the spatial location of the middle point of the mid-level tracklet. Fig. 3.5 (a) shows six consecutive segments with their mid-level tracklets, along with the complete graph their representative nodes induce in (b). Only four mid-level tracklets out of fifteen are shown to avoid cluttering of the plots.

Note that the data association at the track level is fundamentally different from the one used for finding mid-level tracklets; there we assumed a pedestrian moves at a constant velocity within one segment, but modeling the human motion over long periods of time in a track becomes extremely difficult. Generally, it's difficult to model the motion of pedestrians for a long duration without the knowledge of scene structure, intentions, destination, social interaction, etc. This major difference prevents us from using the global motion cost model explained in subsection 3.1.1.2 and the hypothetical node feature estimation described in subsection 3.1.1.3. However, the full trajectory is expected to be temporally smooth. Therefore, at this level, it is essential to change the method for computing the motion cost and estimating hypothetical nodes to piecewise, rather than global, as follows.

Motion-Cost: In order to compute the motion cost for a feasible solution $\vec{V}'_s$, we use the piecewise extension of the velocity vector immediately preceding each node:

$$\gamma'_{motion}(\vec{V}'_s) = \sum_{i=3}^{s} |\vec{X}'_s(i) - [\vec{X}'_s(i-2) + 2\dot{\vec{X}}'_s(i-2)]|. \tag{3.7}$$

Compared to the global approach of eq. 3.4, the second summation in eq. 3.4 is omitted, so that only the preceding piece of the trajectory contributes to the motion cost, because of this eq. 3.7 represents a piecewise extension.

[1]We show the notations related to tracklet merging level with prime ($\prime$) to preserve their correspondence to the ones in mid-level tracklet generation level, yet not confusing them.

Figure 3.5: Merging mid-level tracklets into trajectories. (a) shows six consecutive segments with four mid-level tracklets in each, along with $\vec{G}'$. (b) shows a feasible solution without adding the hypothetical nodes to handle tracklet-occlusion. (c) shows the converged solution, $\hat{\vec{V}}'_s$, along with the generated full trajectory.

Handling occluded mid-level tracklets by hypothetical nodes: Short term occlusions and missed detections are already handled at the mid-level tracklet level by introducing hypothetical nodes. However, if a pedestrian remains occluded for a period longer than the temporal span of

a segment, e.g. over 50 frames, then he/she will not have a mid-level tracklet in the corresponding segment, which leads to the *tracklet occlusion*. In order to handle such cases, we add hypothetical nodes to the clusters of the tracklet association input graph G' with piecewise prediction for computing their spatial location:

$$\vec{x'}^{i}_{Hf} = |\vec{X}'_s(i-2) + 2\ \dot{\vec{X}}'_s(i-2)|. \tag{3.8}$$

In eq. 3.8, only the preceding piece of the trajectory is used to compute the spatial location of the hypothetical node, rather than using the global method of subsection 3.1.1.3 which incorporated all the nodes.

At the mid-level tracklet association level, we add two hypothetical nodes to each cluster rather than one: The *Forward* hypothetical node, denoted by Hf, in which the preceding piece of trajectory is used for prediction, and the *backward* hypothetical node, denoted by Hb, for which the following piece of the trajectory is leveraged for prediction. The backward hypothetical node is necessary for the cases where the mid-level tracklets at the beginning of the trajectory are occluded. That way, since there does not exist any tracklet before the beginning of the trajectory to be used to predict the location of the hypothetical nodes of the missing tracklets, the backward hypothetical nodes are added. They use the tail of the trajectory to predict the location of the hypothetical nodes for the missing tracklets at the beginning of the trajectory. Therefore, the spatial location of the backward hypothetical node is computed using $\vec{x'}^{i}_{Hb} = |\vec{X'}_s(i+2) + 2\ \dot{\vec{X}}'_s(i+1)|$.

The appearance feature of the forward and backward hypothetical nodes are the same as the nodes, which were used to predict the spatial location, i.e. $\vec{V'}_s(i-2)$ for the forward and $\vec{V'}_s(i+2)$ for the backward ones. The hypothetical nodes added at the level of mid-level tracklet association are useful for solving the problem of entry/ exit as well. For instance, if one pedestrian exits the field of view before the end of the video, or enters the view later than the beginning of the video, there will be some segments in which they won't have a mid-level tracklet. The optimization

process will select hypothetical nodes for those segments. However, the computed spatial location of such hypothetical nodes will be out of the view of the frame as they correspond to the time before the pedestrian enters the view or after exiting.

Table 3.1: Description of metrics used in our evaluations.

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

3.2 Experimental Results and Discussion

We evaluated our method on four publicly available sequences, which provide a wide range of significant challenges: TUD-Crossing [109], TUD-Stadtmitte [110], Town Center [103], and sequence S2L1 from VS-PET2009 benchmark [111]. We also present experimental results on a new data set called Parking Lot. We set δ to 5 in eq. 3.5 for all the test sequences. We normalize appearance and motion cost values in eq. 3.3 in order to make them comparable. A typical value for α in eq. 3.3 is one which assigns equal weights to appearance and motion in the overall cost. However, based on our experiments, the appearance features are more informative than motion in our formulation. In fact, in many cases using appearance features only is sufficient for finding the appropriate mid-level tracklets.

Standard CLEAR MOT [112] are used as evaluation metrics. False positives, false negatives and ID-Switches are measured by MOTA. MOTP is defined as the average distance between the ground truth and estimated targets locations. MOTP shows the ability of the tracker in estimat-

ing the precise location of the object, regardless of its accuracy at recognizing object configurations, keeping consistent trajectories, and so forth. Therefore, MOTA has been widely accepted in the literature as the main gauge of performance of tracking methods.

Town Center [103]: The sequence consists of 4500 frames. The size of each segment is 50 frames in this experiment. The quantitative results of the competitive methods for this sequence are presented in [113] and shown in Table 1. Our precision and recall values for this sequence are 92.65% and 81.64% respectively.

Table 3.2: Tracking results for Town Center data set.

	MOTA	MOTP	MODP	MODA
Benfold et al. [103]	64.9	80.4	80.5	64.8
Zhang et al. [36]	65.7	71.5	71.5	66.1
Pellegrini et al. [47]	63.4	70.7	70.8	64.1
Yamaguchi et al. [114]	63.3	70.9	71.1	64.0
Leal-Taixe et al. [113]	67.3	71.5	71.6	67.6
Ours/GMCP	**75.59**	**71.93**	**72.01**	**75.71**

Table 3.3: Tracking results for Parking Lot data set.

	MOTA	MOTP	Prec	Rec
Shu et al. [1]	74.1	79.3	91.3	81.7
Ours/GMCP	**90.43**	**74.1**	**98.2**	**85.3**

PET2009-S2L1-View one [111]: The sequence consists of 800 frames. In sec. 3.1.1.3, we assumed the motion of pedestrians in one segment is near constant velocity in order to identify the outliers and estimate the location of hypothetical nodes. Therefore, segment size should be set in a way that this assumption is not severely violated. Otherwise, the location of hypothetical nodes will be inaccurate which results in a misplaced mid-level tracklet. Typically, the segment size is determined with regard to the frame rate of the video. Therefore, regarding the low frame rate of

PET2009-S2L1, we chose a smaller segment size of 15 frames. The quantitative comparison is provided in Table 2. All reported results are calculated using original tracking outputs provided by the authors [12, 48, 82, 105] with the same overlap threshold for CLEAR MOT metrics.

TUD Data set [38]: TUD-Crossing and TUD-Stadtmitte are two sequences in this data set with low camera angle and frequent occlusions. Crossing and Stadtmitte include 201 and 179 frames respectively. Due to the short length of these sequences, we divided each one into three segments. Quantitative results are provided in Table 2 [11, 115].

Parking Lot: [1] This sequence consists of 1,000 frames of a relatively crowded scene with up to 14 pedestrians walking in parallel. It includes frequent occlusions, missed detections, and parallel motion with similar appearances. Quantitative results are shown in Table 3. No ID-switch was observed in our results. Six sample segments and the merging results are shown in fig. 3.6.

As can be see in Tables 1, 2 and 3, the proposed method constantly outperforms the state of the art on all the standard sequences.

Table 3.4: Tracking results for TUD and PETS 09 sequences.

Dataset	MOTA	MOTP	Prec	Rec	IDSw
TUD-Crossing. [115]	84.3	71.0	85.1	98.6	2
TUD-Crossing. [11]	85.9	73.0	89.2	98.8	2
TUD-Crossing-Ours	**91.63**	**75.6**	**98.6**	**92.83**	**0**
TUD-Stadtmitte. [105]	60.5	65.8	-	-	7
TUD-Stadtmitte-Ours	**77.7**	**63.4**	**95.6**	**81.4**	**0**
PET2009-View1. [12]	80.00	58.00	81.00	60.00	28
PET2009-View1. [82]	81.46	58.38	90.66	90.81	19
PET2009-View1. [105]	81.84	73.93	96.28	85.13	15
PET2009-View1. [48]	84.77	68.742	92.40	94.03	10
PET2009-View1-Ours	**90.3**	**69.02**	**93.64**	**96.45**	**6**

3.2.1 Implementation Details

Several application-oriented methods for solving GMCP, such as branch-and-cut algorithm and multi-greedy heuristics [107], have been proposed to date. Inspired by the solutions proposed for the other generalized graph problems, such as GMST [107], we employ Tabu-search to solve our optimization problem of eq. 3.3. The size of the search neighborhood is changed at each iteration in order to make the optimization process faster and avoid becoming stuck in suboptimal regions. The main contributing factors to the complexity of GMCP are the number of clusters and number of nodes within each cluster. Regarding the small size of GMCP instances we need to solve in our tracking problem, one instance of the optimization problem of eq. 3.3 is typically solved in the negligible time of a fraction of a second. For a segment of 50 frames with approximately 15 pedestrians, processing a frame using the full proposed framework, excluding human detection, takes an average time of 4.4 seconds on a 4core 2.4 GHz machine running non-optimized Matlab code. Using an optimized parallel implementation in C, the algorithm is likely to work in real time.

3.3 Summary

In this chapter we presented a global framework for data association. We utilized Generalized Minimum Clique Problem to solve the formulated optimization problem. Our method is based on shifting the approximation from the temporal domain to the object domain while keeping a notion of the full object domain implicitly. We argued that this framework yields a better developed approach to data association. A two level method, i.e. finding mid-level tracklets first and forming full trajectories later, is employed to cope with different characteristics of human motion in the short and long term. Also, we utilized a global approach to compute the motion cost at the mid-level tracklet level. Experiments showed superior results over the state of the art.

In next chapter we argue the limitation of the GMCP tracker. We will demonstrate a new graph theoretic problem, called Generalized Maximum Multi Clique Problem (GMMCP), and

show how it is used to formulate multi-target tracking. We later show that the proposed GMMCP tracker is capable of addressing GMCP tracker's limitations.

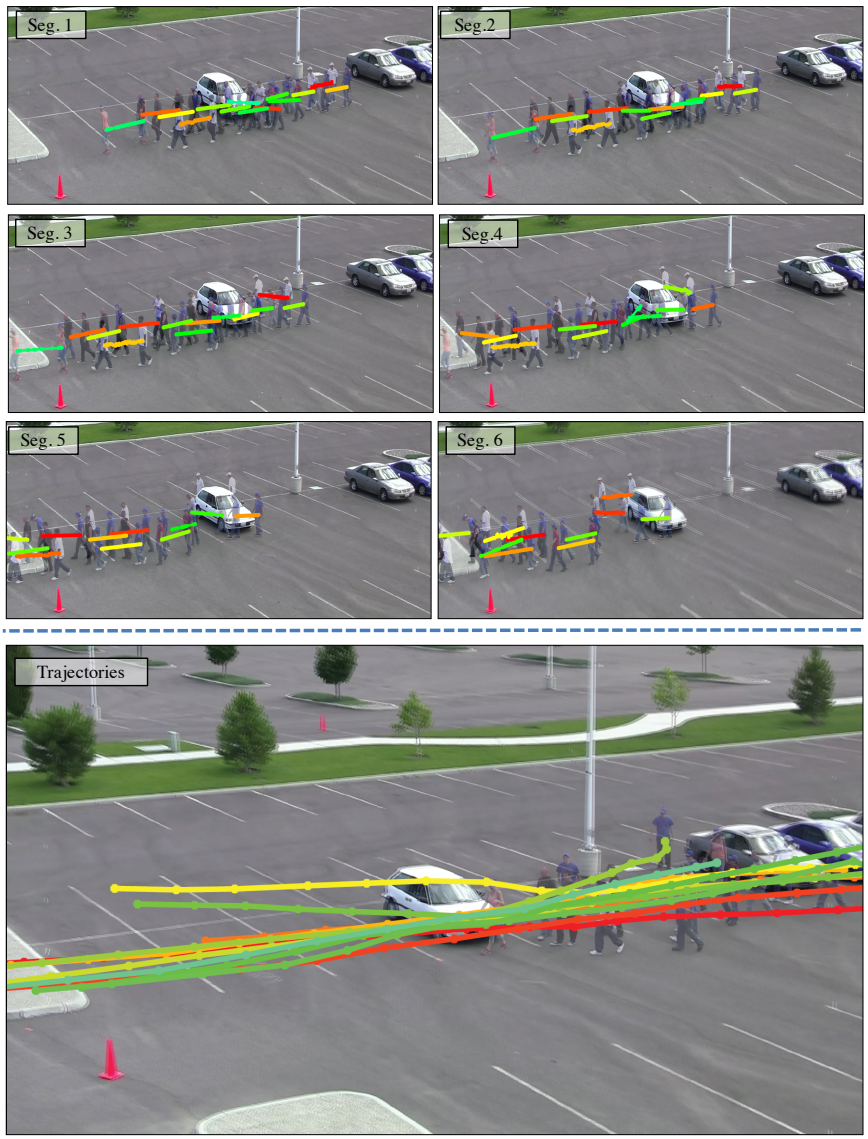


Figure 3.6: Top: The mid-level tracklets of six sample consecutive segments from Parking Lot sequence. Bottom: The trajectories resulting from associating mid-level tracklets of all the segments.

CHAPTER 4: GMMCP TRACKER: GENERALIZED MAXIMUM MULTI-CLIQUE PROBLEM FOR MULTI-OBJECT TRACKING

Previous data association methods, including GMCP tracker proposed in Chapter 3, assume a simplification either in *problem formulation* or in the *optimization*. In this chapter, we describe a new data association technique that provides a more accurate formulation for multi-object tracking where no simplification is assumed in either steps. We follow the same idea as GMCP tracker but aim to address its three main limitations. GMCP finds one target at a time, missing joint optimization; which is essential in some scenarios to avoid ID-switch. GMCP uses a greedy optimization which makes it prone to local minimas. Finally GMCP tracker is slow, which is a concern when dealing with time-sensitive applications.

We describe in this chapter, how to formulate data association as a new graph theoretic problem called, *Generalized Maximum Multi Clique Problem* . Similar to GMCP tracker, the input to our data-association is a k-partite complete graph, where all the pairwise relationships between all the observations in a batch of frames are considered. Our cost function allows incorporating higher order costs between all the candidates in a track. Similar to GMCP, GMMCP is an NP hard problem. However, 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 efficitenly for small and medium size MOT problems using available commercial softwares [43, 44]. Our tracking algorithm addresses the aforementioned problems of GMCP tracker by not only finding the tracks jointly for all the objects but also providing an accurate solution without relaxing the original problem.

Finally, we present 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.

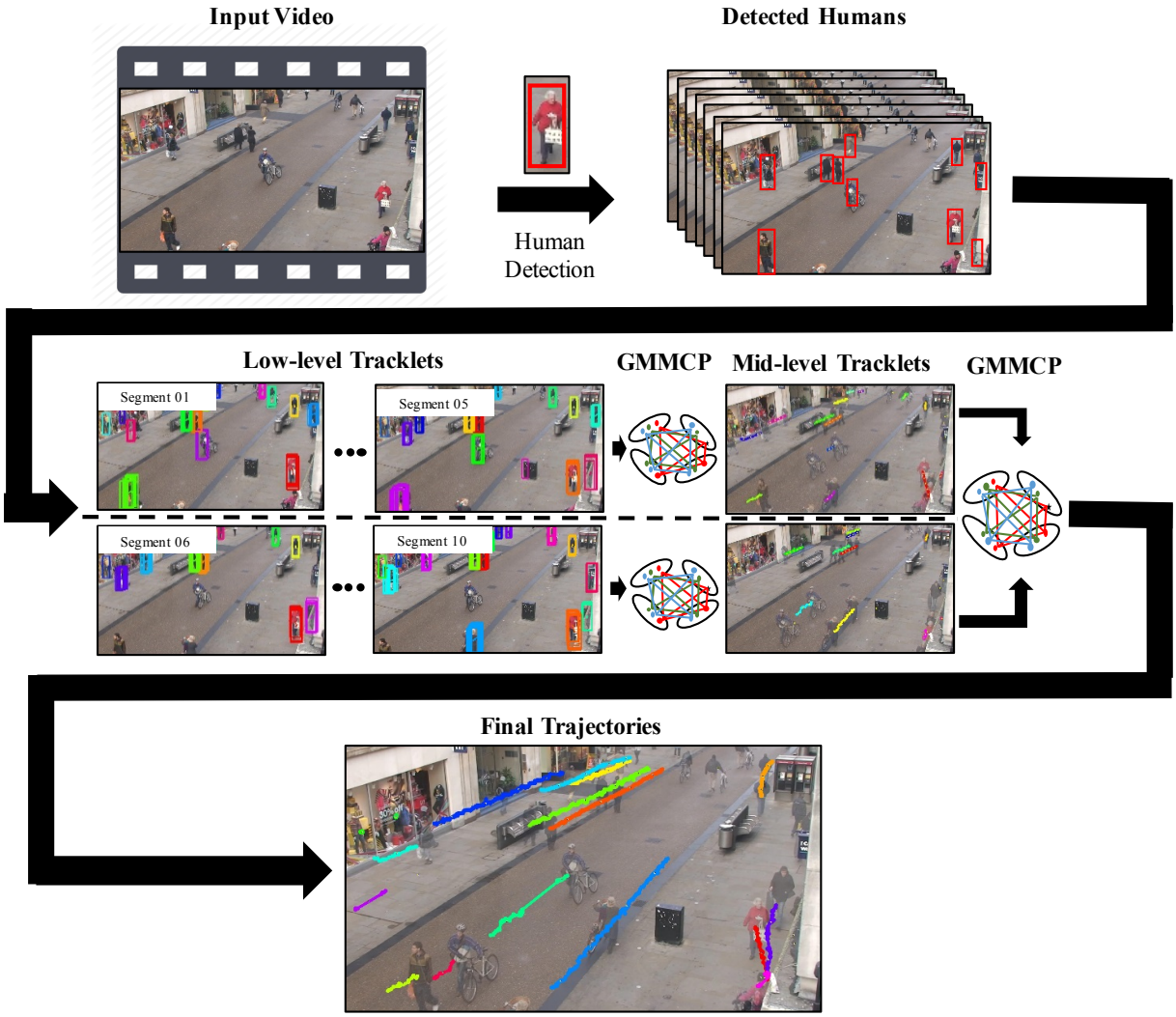


Figure 4.1: This figure illustrates the GMMCP three-layer tracking pipeline. First low-level tracklets are found through simple overlap constraint. Next low-level tracklets are connected through GMMCP to form the mid-level tracks. Final trajectories are formed by applying the same data association technique to the mid-level tracklets.

4.1 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. Figure 4.2 illustrates our framework. 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 [4, 116] and a set of approximate solutions, such as multi-greedy heuristics and local search, are proposed to solve GMCP [4, 117]. 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 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 $K-1$ cliques with $\inf$ edges. So GMCP is reduced to GMMCP and we have shown that GMMCP is at least as hard as GMCP, thus it is NP-hard.

4.2 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 [15, 49, 87, 118]. 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 4.4 how these weights are calculated.

4.2.1 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:

$$\begin{cases} \text{maximize} & \mathbf{C}^T\mathbf{x}, \\ \text{subject to} & \mathbf{Ax} = \mathbf{b} \text{ and } \mathbf{Mx} \leq \mathbf{n}. \end{cases} \tag{4.1}$$

The objective function to maximize is $\mathbf{C}^T\mathbf{x}$, where $\mathbf{C}$ is the weight vector 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).

$$\{\forall j | 1 \leq j \leq h\} : \sum_{i=1}^{l} v_i^j = K, \tag{4.2}$$

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.

$$\sum_{j=1}^{h}\sum_{i=1}^{l} e_{mn}^{ij} = v_m^n.(h-1).\{\forall m, n | 1 \leq n \leq h, 1 \leq m \leq l\} \tag{4.3}$$

Finally we need the third **constraint** to ensure that the solution found by **x** will form a clique.

$$e_{ij}^{i'j'} + e_{i'j'}^{i''j''} \leq 1 + e_{i''j''}^{ij}. \ \{\forall e_{ij}^{mn} \in E\} \tag{4.4}$$

Constraints in Equations 5.2 and 5.3 are used together to form the equality constraint defined by matrix **A** and vector **b**. Matrix **M** and vector **n** are also constructed so that the constraint in Equation 4.4 is satisfied. The combination of these three constraints will ensure that **x** will provide a valid solution to the GMMCP. Once **x** is found, each clique will represent a track of a person.

4.2.2 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 4.2 (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 4.1 into four

terms as shown bellow:

$$\text{maximize} \sum_{j_1} \overbrace{c_{j_1} x_{j_1}}^{RealEdges} + \sum_{j_2} \overbrace{c_{j_2} x_{j_2}}^{DummyEdges} + \sum_{j_3} \overbrace{c_{j_3} x_{j_3}}^{RealNodes} + \sum_{j_4} \overbrace{c_{j_4} x_{j_4}}^{DummyNodes} , \tag{4.5}$$

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} represents 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 4.4 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:

$$N_d^i = \sum_{j \neq i} N^j, \tag{4.6}$$

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.

4.3 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 4.2.1. 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 vector 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 4.4 which ensures that the solution will form a clique.

$$e_{ij}^{i'j'} + e_{i'j'}^{i''j''} \leq 1 + e_{i''j''}^{ij}. \ \{\forall e_{ij}^{mn} \in E\} \tag{4.7}$$

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.

$$\sum_{m=1}^{l} e_{mn}^{ij} \leq 1. \ \ \{\forall e_{ij}^{mn} \in E\} \tag{4.8}$$

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

$$\{\forall j | 1 \leq j \leq h\} : \sum_{i=1}^{l} e_{mn}^{ij} + vd^{j} = (h-1) \times K, \tag{4.9}$$

where vd^{j} defines the ADN in cluster j. The cost function remains the same as the one defined in Equation 4.5 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 [43]. Our experiments show that using ADNs, one can significantly reduce the computational complexity and achieve performance close to real time on a desktop computer.

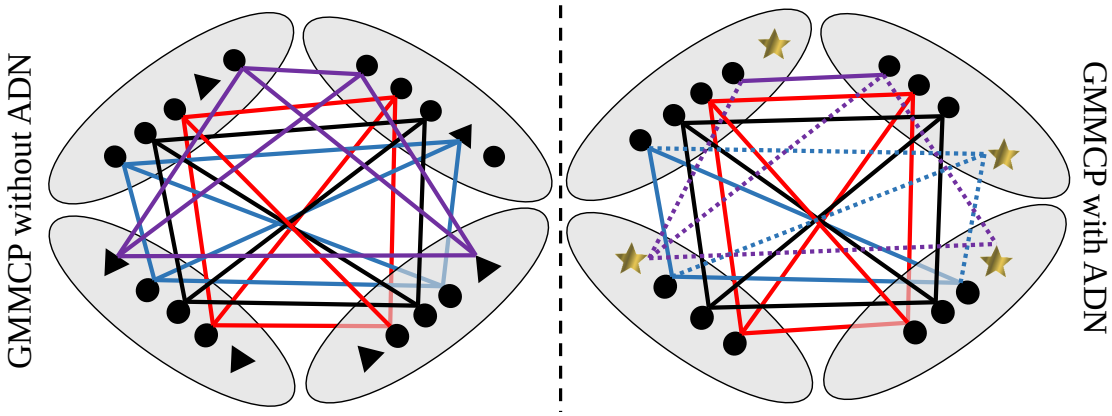


Figure 4.2: The two versions of the input graph used in our tracking. The graph on the left uses regular dummy nodes (shown with triangles) for occlusion handling and the speed-up version of our tracker uses only one dummy node per cluster called aggregated-dummy-node (shown with stars on the right). 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.

4.4 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 5.1). 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 [119]. 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.

$$c_{Appearance}(T_i, T_j) = k(\phi(T_i), \phi(T_j)). \tag{4.10}$$

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 [4], 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 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 4.11.

$$c_{Motion}(T_i, T_j) = \frac{1}{2} exp(\frac{-d}{\sigma}), \tag{4.11}$$

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:

$$c_{j_1} = (\nu) c_{Appearance} + (1 - \nu) c_{Motion}, \tag{4.12}$$

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:

$$c_{j_1} = (1-\tau)c_{Appearance} + (\tau)c_{Motion},$$
$$\text{where } \tau = \exp(\frac{|l-m|}{\gamma})(1-\nu). \tag{4.13}$$

4.5 Experiments

We performed exhaustive experiments and evaluated our tracker on six sequences. Five of the sequences are publicly available sequences including *Town Center* [103], *Parking-Lot 1* [1], *Parking-Lot 2* [80], *TUD-Crossing* [104] and *TUD-Stadtmitte* [105] 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 for which we have access to their code including KSP [12], DCT [13], CMOT [14], IHTLS [15] and GMCP [4]. 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.

Implementation Details. We used deformable part based model [32] 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 4.11 the parameter σ is set to 20 and in Equation 4.12 the parameter ν and γ are set to 0.7 and 5 respectively.

Table 4.1: Quantitative Analysis of our method for Town Center, TUD-Crossing, TUD-Stadmitte and Parking-lot 1 with state-of-art methods of DLP [9], H2T [10], MWIS [11] and PartTrack [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**

4.5.1 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 3.1.

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 [49] by considering overlapping segments, thus connecting tracks in each step that overlap. The quantitative results are shown in Table 4.1.

Table 4.2: Quantitative Analysis of our method on Parking-lot Pizza and Parking-lot 2 with competitive approaches of KSP [12], DCT [13], CMOT [14], IHTLS [15] and GMCP [4].

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**

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 caused 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 [15] 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 4.1.

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 4.2.

4.5.2 Run Time Comparison

As discussed in section 4.3, 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 4.3, 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 4.4 we show how the run-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. 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 4.4 and 4.3 are obtained using a 3.2GHz PC while utilizing only a single core.

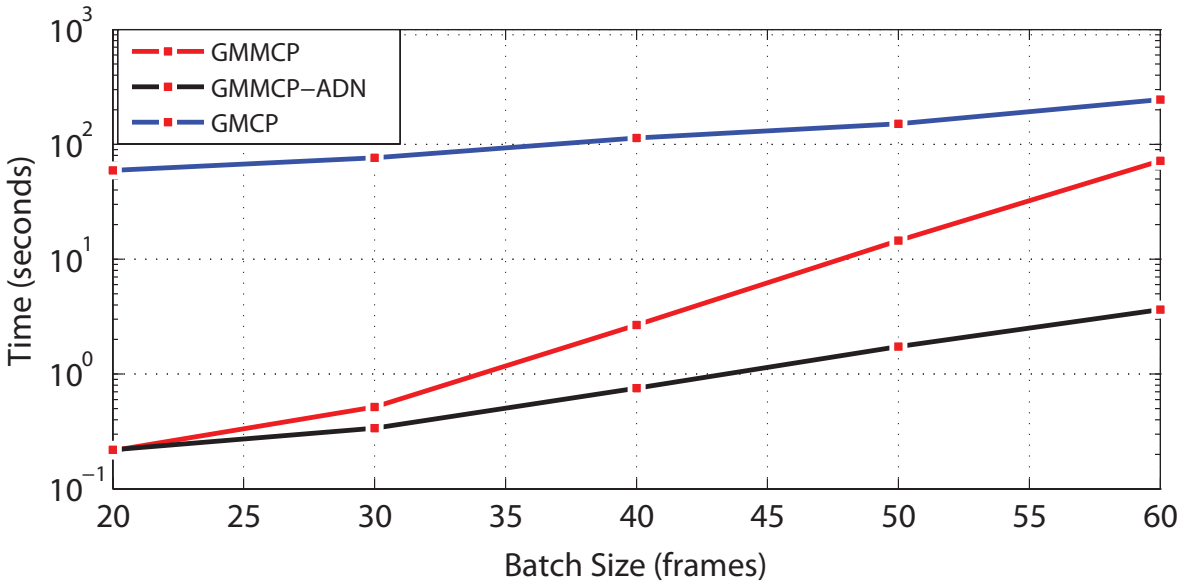
It is worth to mention that the computational complexity of our method is highly dependent on the problem size. Although we show performance close to real-time using a non-optimized code on

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.

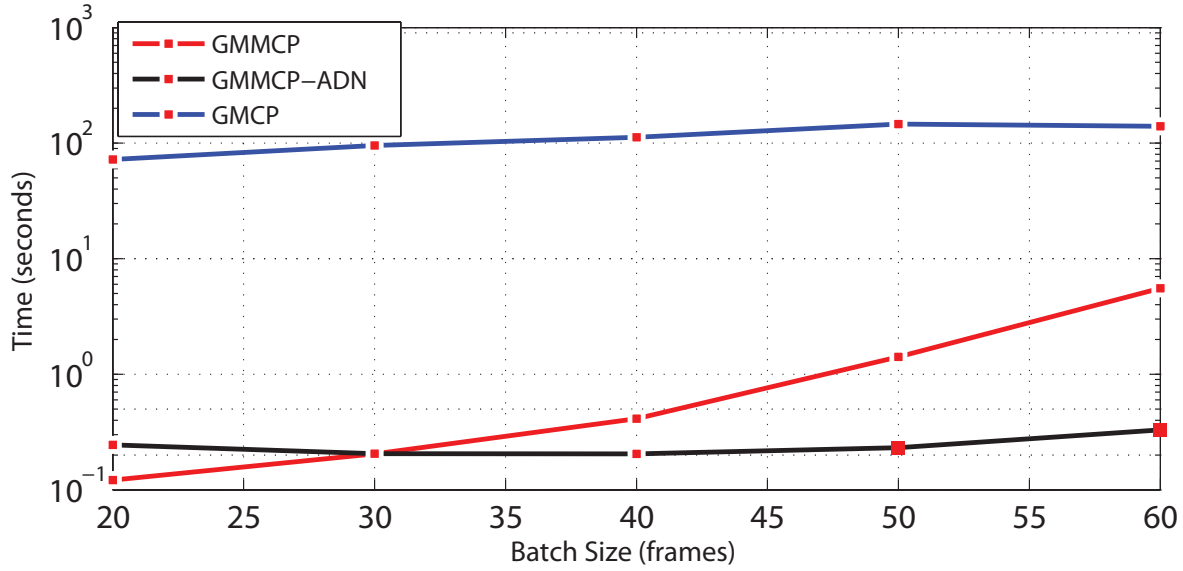
4.6 Summary

In this chapter we formulated multiple target tracking as a Generalized Maximum Multi Clique problem, which it is solved through Binary Integer Programing. We later showed 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 showed that we can improve the state-of-art on six challenging sequences.

In the next chapter, we focus on a relatively unexplored problem in the context of multi-target tracking, which is tracking targets in extremely crowded scenes. We formulate crowd tracking as a binary quadratic programing and incorporate several constraints in our formulation and show that we can efficiently track hundreds of people in high-density crowds.



(a) Parking-lot 1



(b) TUD-Crossing

Figure 4.3: run-time comparison of our GMMCP tracker with ADN and without ADN as well as the GMCP tracker proposed in [4] on *Parking-lot 1*(a) and *TUD-Crossing*(b) sequences. The plots are in semilogarithmic scale

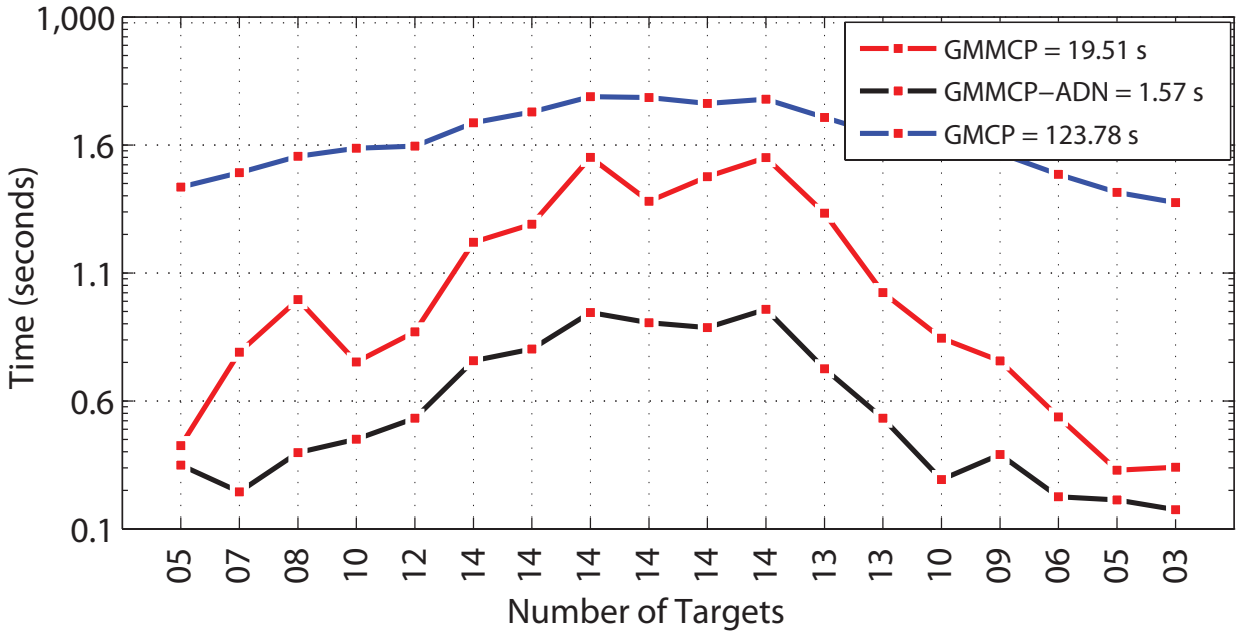


Figure 4.4: Run time comparison of our method for different number of targets in a batch of 50 frames on Parking-lot 1. The numbers in the legend show the average run-time across all the segments.

Figure 4.5: Qualitative results of our tracker on *Parking-lot 2*, *Parking-lot 1*, *TUD-Crossing*, *TUD-Stadtmitte*, *Parking-lot Pizza* and *Town Center* sequences.

CHAPTER 5: BINARY QUADRATIC PROGRAMING FOR ONLINE TRACKING OF HUNDREDS OF PEOPLE IN EXTREMELY CROWDED SCENES

Thus far we have assumed that the number of people do not exceed a few dozens. However, this is not always the case. In many scenarios such as, marathon, political rallies or religious rites, the number of people in a frame may reach few hundreds or even few thousands. Tracking in high-density crowd sequences is a challenging problem due to several reasons. Human detection methods often fail to localize objects correctly in extremely crowded scenes. This limits the use of data association based tracking methods. Additionally, it is hard to extend existing multi-target tracking to track targets in highly-crowded scenes, because the large number of targets increases the computational complexity. Furthermore, the small apparent target size makes it challenging to extract features to discriminate targets from their surroundings. In this chapter we aim to solve the detection and data association in an online manner for high density crowd sequences. Given the initial target locations in the first frame, our method starts by training a discriminative model for each target using linear regression. During inference, the potential candidates for each target are sampled densely around the pervious locations of the target. Each candidate is assigned a cost according to its past trajectory as well as its surrounding neighbors. The goal is then to find new location of each target by minimizing the proposed quadratic objective function. The new target location is later used to update the target models if necessary.

5.1 Objective Function

At every frame the best locations of the targets are found by minimizing the following objective function:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} f(x) = \overbrace{\mathbf{c}_a^T\mathbf{x}}^{appearance} + \overbrace{\zeta\,\mathbf{c}_m^T\mathbf{x}}^{targetmotion} + \overbrace{\eta\,\mathbf{c}_{nm}^T\mathbf{x}}^{neighbormotion} \\ + \overbrace{\mathbf{x}^T\mathbf{C}_{sp}\mathbf{x}}^{spatialproximity} + \overbrace{\mathbf{x}^T\mathbf{C}_g\mathbf{x}}^{grouping}, \end{aligned} \tag{5.1}$$

where $\mathbf{x} \in \mathbb{R}^l$ is a vector which contains all the binary variables, each encoding potential location of the target in next frame. $l = n \times k$ defines the number of candidate locations, where n is the number of targets and k is the number of candidate locations for each target. $\mathbf{c}_a$, $\mathbf{c}_m$, $\mathbf{c}_{nm}$ are affinity vectors which respectively encode the appearance, motion and neighborhood motion cost. The affinity matrix $\mathbf{C}_{sp}$ includes the pairwise proximity cost that discourages the co-selection of locations that are too close to each others. This term is especially important in high density crowded scenes that are mostly captured by cameras facing down. Due to the camera view in these sequences targets will not occlude each other, thus two candidates cannot get closer than a certain distance to each other. The affinity matrix $\mathbf{C}_g$ contains the group information and encourages the people in the same group to keep their formation.

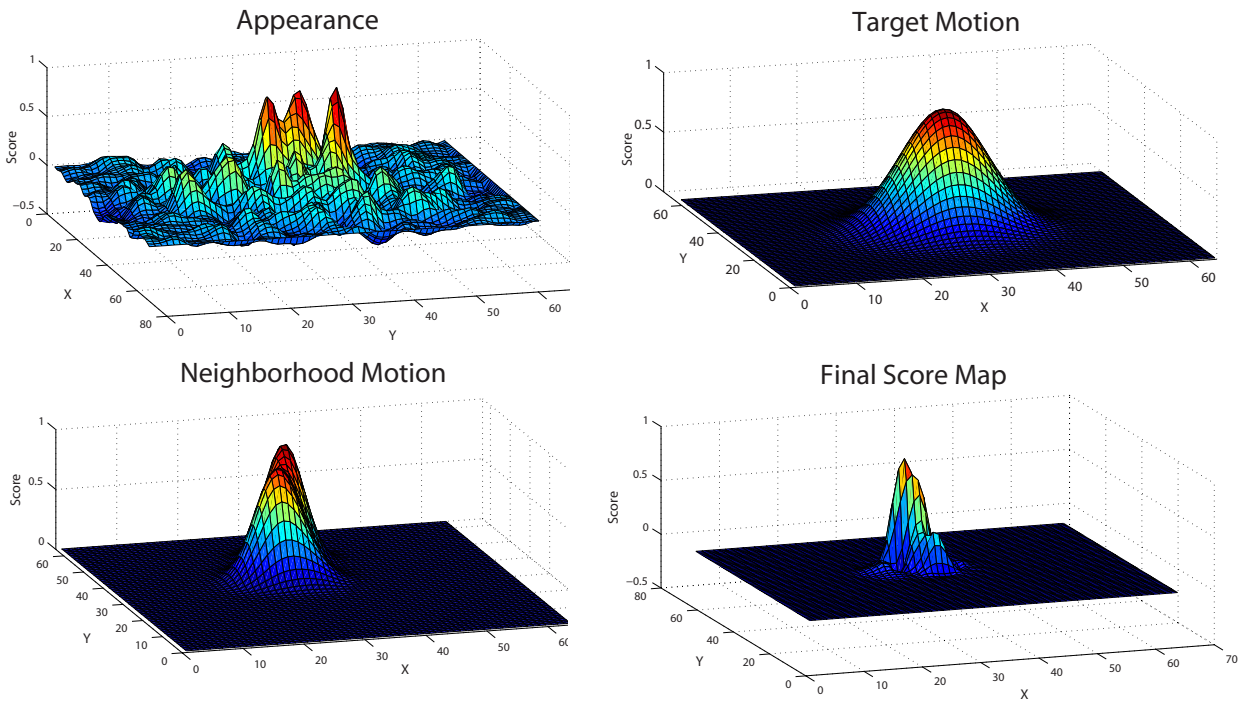


Figure 5.1: An example of the probability maps obtained for a target in seq-4. These maps include appearance, motion and neighborhood motion. It is clear from the figure that the appearance by itself is not always sufficient and the combination of all the three components is more discriminative for tracking.

In order to ensure the solution found by solving Equation. 5.1 is a feasible tracking solution, we need to enforce the following constrains:

$$\sum_{i \in N_j} x_i^j = 1, \quad \{\forall i, j | 1 \leq i \leq k, 1 \leq j \leq n\} \tag{5.2}$$

$$x_i^j \in \{0, 1\}, \quad \{\forall i, j | 1 \leq i \leq k, 1 \leq j \leq n\}, \tag{5.3}$$

where x_i^j, a component of vector $\mathbf{x}$, is a binary variable representing the i^{th} candidate location in the neighborhood of the j^{th} target. k is the total number of sampled candidates for each target and n is the total number of targets to track. The constraint in Eq. 5.2 guarantees that exactly

one location is selected for each target. The constraint in Eq. 5.3 ensures that each location is assigned to at most one target. These constraints along with the cost function in Equation 5.1 form our Binary Quadratic Programing formulation that needs to be solved at every frame. Each term in Equation. 5.1 requires the computation of its own affinity matrix/vector. Below we describe in details how to compute these affinity matrices/vectors.

Appearance information in high density crowd sequence is not as discriminative as in low or medium dense crowd sequences such as the ones used in [1,106]. There are only a small number of pixels covering each target, and the targets look very similar. However, our experiments show that the appearance still plays an important role in our approach. In [2, 5, 6], a template-based method based on Normalized Cross Correlation (NCC) was used to capture such information. In a recent study in [3], it is shown that discriminative tracking methods work better than the generative ones. However, the discriminative models have not been used in previous crowd tracking methods. One reason is the complexity of discriminative models. Training individual models, such as the one used in [55], for a large number of targets is computationally expensive. In this work, we show that one could still use discriminative models while not increasing the complexity.

In our tracker, we train a regressor for each target by minimizing the following objective function:

$$\underset{\mathbf{w}_i}{\text{minimize}} \sum_{j \in T_i} \left(\mathbf{w}_i \phi(x_i^j) - y_i^j\right)^2 + \lambda \left\|\mathbf{w}_i\right\|^2, \tag{5.4}$$

where $\mathbf{w_i}$ is the model parameters for the i^{th} target, $\phi(x_i^j)$ is the feature vector extracted from the candidate location x_i^j, the labels are defined by y_i^j and T_i represents the training samples for i^{th} target. Regression models allow one to avoid binary labeling of training samples. This is shown in [55] to provide a better model. The above optimization has a closed form solution of $\mathbf{w} = (Z^T Z + \lambda I)^{-1} Z^T \mathbf{y}$, where Z is a matrix that has a sample x_i^j per row. Once the models are

trained the appearance cost for each candidate location is found using the following equation:

$$c_{i,a}^{j} = \mathbf{w}_i \phi(x_i^j). \tag{5.5}$$

In [62] it is shown that the solution to Eq. 5.4 and 5.5 can be found efficiently in Fourier domain. We follow the same approach to compute the models for each target, but instead of using pixel intensity values as feature, we employ the multi channel formulation of [62] and use color features.

Motion plays an important role in the context of tracking pedestrians. The motion of pedestrian in crowded scene is effected by their environment. Thus using motion models that predict target location only based on its own behavior is not enough. Several methods have been proposed over the past few years to model the behavior of pedestrians in crowds considering their environment. But none of those models have been used in a crowd tracking framework with efficient joint optimization of target tracks.

In this work, we use two different types of motion information. One that predicts target location based on its past observations, this is captured using $\mathbf{c}_m$. The other incorporates information from the neighboring targets [2] to predict the location of individuals at each time step.

The first term predicts the location of target based on its past observation using a linear velocity model shown below:

$$\mathbf{c}_{i,m} = \mathcal{N}(A\mathbf{s}_i^{t-1}, \mathbf{\Sigma}), \tag{5.6}$$

where $\mathbf{s}_i^t = [p_i, \dot{p}_i]$ is a vector that contains the location and velocity of target i at time t, A is the state transition matrix and $\mathcal{N}(\mu, \Sigma)$ is a $2D$ Gaussian distribution.

The second term, on the other hand, captures the local force that influence the motion of each individual. In order to capture the neighborhood motion, one needs to first find the groups of people with coherent motion. This is obtained by computing the correlation between the tracks based on their past observations (frames $t-10$ to $t-1$) and clustering them into different groups.

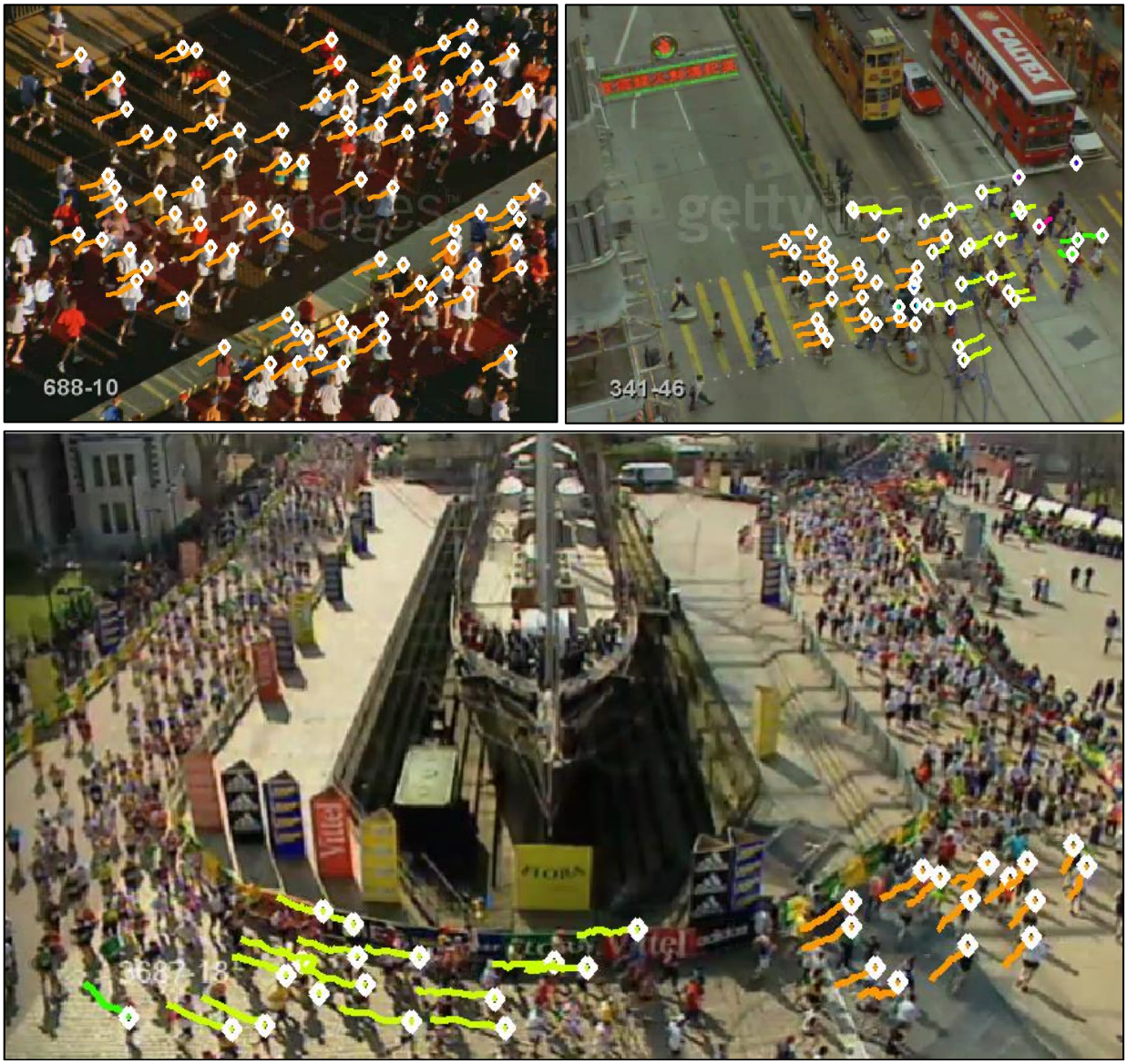


Figure 5.2: An example of groups that move with coherent motion in Seq-5 (each color corresponds to one group). These groups are used to incorporate the neighborhood motion effect in our optimization.

An example is shown in Figure 5.2. During this unsupervised clustering we also take into account the distance between the members. This approach is simple and effective and can be computed online without requiring much computations. Considering p_i to be the neighbor of target i which

moves coherently with it. N_i is a set that includes all the neighbors of target i (In our experiments the number of neighbors in N_i is limited to 5). The influence of neighborhood motion is captured using the following equations.

$$c_{i,nm} = \sum_{j \in N_i} w_j . \mathcal{N}(A\mathbf{s}_{ij}^{t-1}, \mathbf{\Sigma}), \tag{5.7}$$

where $\mathbf{s}_{ij}^t = [p_i, \dot{p}_j]$ encodes the position of target i and velocity of the j^{th} neighbor. The influence of each neighbor is captured using the weight coefficient w_j which is obtained using the distance of the neighbor to the target:

$$w_j = \frac{exp(\|p_j - p_i\|_2)}{\sum_{k \in P_i} exp(-\|p_k - p_i\|_2)}. \tag{5.8}$$

An example of the three linear terms for a target is shown in Figure 5.1. It is clear from the figure that the appearance cue by itself is not always sufficient. One can clearly see that the combination of appearance, motion and neighborhood motion is able to discriminate target from its background better than using each feature individually.

Spatial Proximity Constraint is another important factor in our formulation. In top view high density crowd sequences, targets tend to look similar. This similarity confuses most trackers, as the small apparent target size makes it difficult to extract useful appearance features. Our spatial proximity constraint discourages the tracker to select locations that are very close to each other. Additionally this is a soft constraint, meaning if the targets get too close (When the camera is not facing down and we have perspective effect) it still allows the targets to get close to each other as long as the appearance cues exist.

In our formulation, $\mathbf{C}_{sp}$ contains the proximity cost for each pair, where $\mathbf{C}_{sp} = I - D^{-1/2} S D^{1/2}$ is the normalized Laplacian matrix [120]. D is the diagonal matrix composed of row sums of similarity matrix S. $S \in \mathbb{R}^{l \times l}$ is the similarity matrix that encodes the spatial proximity cost and discourages the co-selection of locations that are too close (this is defined based on

the target size). The entries of matrix S are defined as follows:

$$S_{ij} = exp(\frac{-\|p_i - p_j\|_2^2}{2\sigma^2}), \tag{5.9}$$

where σ is set to half the target size. It is important to note that we could not set $\mathbf{C}_{sp} = \mathbf{S}$. The reason is that matrix $\mathbf{S}$ is not positive-semi-definite and this makes our objective function non-convex. The Laplacian trick helps us to keep $\mathbf{C}_{sp}$ positive-semi-definite while still imposing the same effect to our optimization.

Figure 5.3: An example of our minimum spanning tree group structure model. The figure on the left shows the groups found in frame 638 of Galleria1 sequence (nearby tracks with similar color are one group). The figure on the right illustrates the model created for the group inside the yellow box. p_i identifies the location of the target and e_ij determines the relative distance between the two targets i and j.

Grouping Constraint is another important factor that affects pedestrians behavior in crowded scenes. People walking in a group tend to keep their formations for a short time. This provides valuable information to any tracking algorithms. However, the biggest challenge is: *How to incorporate group information in a tracking framework?* To utilize group information one needs to incorporate pairwise information in the tracking formulation. In our case, this is done by simply adding another quadratic term in our objective function that captures the group formations.

Let $G_i = \{p_1, p_2, ..., p_m\}$ be the i^{th} group which contains m targets, where p_j defines the location of the j^{th} target. We adopt a minimum spanning tree pictorial structure model to represent each group. The main advantage of having a tree model instead of considering all the pairwise relationships is bifold. First, considering fewer pairwise relationships, helps reducing the computational complexity. Second, having fewer constraints is less restrictive and better allows small changes in target formations, which is likely to happen in practice. The parameters of our pictorial structure model are also learned online during tracking, and do not involve large training data like most previous works. The grouping information in our formulation is encoded by the matrix $\mathbf{C}_g$, where $\mathbf{C}_g = I - D^{-1/2}\Gamma D^{1/2}$ is the normalized Laplacian matrix [120]. Here D is the diagonal matrix composed of row sums of similarity matrix Γ. $\Gamma \in \mathbb{R}^{(l)\times l}$ is the similarity matrix that encodes the information and encourages the selection of candidate locations that keep the formation of targets within each group. Each entries of Γ is obtained using the equation below:

$$\Gamma_{ij} = exp(\frac{-(\|p_i - p_j\|_2 - e_{ij})}{2\sigma^2}), \tag{5.10}$$

where e_{ij} is the distance between target i and target j in our tree model for that group. An example of our tree model for one group is shown in Figure. 5.3. We update the group information every τ frames ($\tau = 10$ in our experiments), thus the values of e_{ij} are updated every τ frames. We found that it is important to update the groups frequently, because targets within the same group are likely to change their relative distance.

5.2 Optimization

Since $\mathbf{C}_{sp}$ and $\mathbf{C}_g$ are positive semi definite, one can use publicly available QP software such as ILOG CPLEX to find the solution to Eq. 5.1. However, we observed that the solver becomes extremely slow as the number of targets exceeds five, thus it is not feasible to solve the BQP directly. One option is to relax the discrete non convex binary constraint in Eq. 5.3 to its convex hull. The barrier optimization techniques used in commercial softwares such as CPLEX has complexity of $O(N^3)$, which makes it inefficient when solving for hundreds of targets. On the other hand, the structure of our problem allows one to solve the linearized version of Eq. 5.1 very efficiently using projected gradient descend methods. This property opens the room to the powerful Frank Wolfe optimization technique that has been revisited recently. We adopt the most recent version of Frank Wolfe algorithm that takes advantage of the SWAP steps to speed up the optimization. We show that using Frank-Wolfe with SWAP steps one can find the solution efficiently for hundreds of targets. We also compare the run-time of our optimization with CPLEX and other variants of Frank-Wolfe algorithm and show that we can find the solution faster.

5.2.1 Frank Wolfe Optimization

Algorithm 1: Frank Wolfe

Data: $x_0 \in D$.
Result: z
Result: Initialization: $k = 0$, $z = x_0$, max_it
for $k = 0, 1...K$ **do**
  Compute $s^* \leftarrow \underset{s \in D}{\text{argmin}} < s, \bigtriangledown f(x_k) >$,
  $d_{FW} = s^* - x_k$
  Line Search : $\lambda_{FW} = \underset{\lambda \in [0,1]}{\text{argmin}} f(x_k + \lambda(d_{FW}))$
  Update : $x_{k+1} = (1 - \lambda_{FW})x_k + (\lambda_{FW})s^*$
end
Perform the rounding $z \leftarrow rounding(x_K)$
return z

Given our convex quadratic function $f(\mathbf{x})$ in Eq. 5.1 and a set of convex constraints D, Frank Wolfe algorithm finds a solution to this problem by solving the iterative optimization given in Algorithm. 1. At every iteration it solves the linearized version of the objective function, $g(\mathbf{x})$. The minimizer is then used to find the descent direction after performing a line search. In our problem, the linearized version of our objective function is given by [1]:

$$g(\mathbf{x}) =< \mathbf{s}, \bigtriangledown f(\mathbf{x}) >= \mathbf{s}(\mathbf{C}_{sp}\mathbf{x} + \mathbf{C}_g\mathbf{x} + \mathbf{c}_a + \mathbf{c}_m + \mathbf{c}_{nm}), \tag{5.11}$$

subject to the set of convex constraints defined by D. This could be solved efficiently using any projected gradient descend method. One can define the step size in Algorithm. 1 by $\lambda_{FW} = 2/(2+k)$, where k is the current iterate. However, the exact step size is found by solving the line-search problem given below:

$$\lambda_{FW} = \underset{\lambda \in [0,1]}{\operatorname{argmin}} f(\mathbf{x} + \lambda(\mathbf{s} - \mathbf{x})). \tag{5.12}$$

The optimization problem in Equation. 5.12 has a closed form solution that can be obtained by setting the derivative of Equation 5.12 with respect to λ equal to zero, and replacing the function f with its definition in Equation 5.1. This will lead to the following closed form solution:

$$\begin{aligned} \frac{\partial}{\partial \lambda} f(\mathbf{x} + \lambda(\mathbf{s} - \mathbf{x})) = \bigtriangledown f(\mathbf{x} + \lambda(\mathbf{s} - \mathbf{x}))^T(\mathbf{s} - \mathbf{x}) = 0, \\ \lambda_{FW} = \frac{\bigtriangledown f(\mathbf{x})^T(\mathbf{s} - \mathbf{x})}{(\mathbf{s} - \mathbf{x})(\mathbf{C}_{sp} + \mathbf{C}_g)(\mathbf{s} - \mathbf{x})}. \end{aligned} \tag{5.13}$$

One should note that the solution in Equation 5.13 will not add much overhead to the computation. Because most of the terms have been already computed in other steps. The only part in Algorithm. 1 that is left unexplained is the rounding. The solution found at the end of FW

[1] please note that for simplicity, we removed the index k in Equations 5.11, 5.12 and 5.13 , which shows the k^{th} iteration

optimization does not satisfy the discrete constraint of Eq. 5.3, thus requires rounding. In order to find the best binary solution one needs to solve the following optimization.

$$\underset{\mathbf{z}\in D}{\operatorname{argmin}} \left\|\mathbf{z}-\mathbf{x}_K\right\|_2, \tag{5.14}$$

where $\mathbf{x}_K$ is the solution found by FW at the end of the optimization at iteration K and $\mathbf{z}$ is the final rounded solution. Due to the structure of our problem, the optimization in 5.14 reduces to solving the above optimization for each target separately, which is equivalent to taking the argmin of the $\mathbf{x}$ for those candidates of each target.

5.2.2 Frank Wolfe with SWAP Steps

The convergence rate of original Frank-Wolfe algorithm is shown to slack near the optimal solution. This makes the original FW method intractable for large scale problems. Instead of using the original FW, we use an accelerated version of Modified-Frank-Wolfe algorithm that takes advantage of a trick called SWAP steps to speed up the optimization. The full optimization procedure is given in Algorithm 2. The idea is that at each iteration we find the descend vertex, $\mathbf{x}_k$ as well as the ascend vertex, $\mathbf{y}_k$, over the face spanned by current solutions. Beside the FW step of $d_{fw} = \mathbf{x}_k - \mathbf{z}$, we consider the SWAP step defined as $\mathbf{x}_k - \mathbf{y}_k$. The SWAP could be considered as a step that moves the current solution in the away direction and at the same time in the direction of the toward step. This is shown in the equation below.

$$\overbrace{\mathbf{x}_k + \lambda(\mathbf{s}_k - \mathbf{y}_k)}^{SWAP\ step} = \overbrace{\frac{1}{2}(\mathbf{x}_k + \lambda(\mathbf{s}_k - \mathbf{x}_k))}^{toward\ step} + \overbrace{\frac{1}{2}(\mathbf{x}_k + \lambda(\mathbf{x}_k - \mathbf{y}_k))}^{away\ step}. \tag{5.15}$$

Once we define the toward and SWAP steps, we find the improvement update using each step. The one that gives the best improvement is selected to perform the move in the current iteration. In order to pick the best improvement one needs to perform two line-searches compared to

one line-search step that was used in previous accelerated versions of Frank-Wolfe [102]. However, since the estimation of the objective function at each iteration is more accurate, it requires less iterations to converge. Additionally, the optimal value of line-segment problem is found analytically and does not require much computations. The computation of δ_{fw} and δ_{swap} involves terms already computed in the line-search and therefore does not add any additional overload. We present several experiments in Section 5.4 to validate the discussions above. Clipping the line search is to ensure the solution remains in the convex set *D*. SWAP-add/drop step is used to update the active set S [121].

Algorithm 2: Frank Wolfe with SWAP Steps

Data: $\mathbf{x}_0 \in D, \varepsilon > 0$.
Result: $\mathbf{z}$
Result: Initialization: $k = 0$, $\mathbf{z} = \mathbf{x}_0$, $S_0 = \{\mathbf{x}_0\}$, max_it
while $duality\_gap(\mathbf{z}) > \varepsilon$ *and* $k < max\_it$ **do**
  $k \leftarrow k + 1$
  (descent direction) $\mathbf{s}_k \leftarrow \underset{\mathbf{s} \in D}{\text{argmin}} < \mathbf{s}, \bigtriangledown f(\mathbf{x}_k) >$
  (ascent direction)$\mathbf{y}_k \leftarrow \underset{\mathbf{y} \in S_{k-1}}{\text{argmax}} < \mathbf{y}, \bigtriangledown f(\mathbf{x}_k) >$
  Line Search : $\lambda_{fw} = \underset{\lambda \in [0,1]}{\text{argmin}} f(\mathbf{x}_k + \lambda(\mathbf{s}_k - \mathbf{x_k}))$
  Line Search : $\lambda_{swap} = \underset{\lambda \in [0,1]}{\text{argmin}} f(\mathbf{x}_k + \lambda(\mathbf{s}_k - \mathbf{y}_k))$
  Compute $\delta_{fw} = f(\mathbf{x}_k + \lambda_{fw}(\mathbf{s}_k - \mathbf{x}_k)) - f(\mathbf{x}_k)$ (Improvement of fw step)
  Compute $\delta_{swap} = f(\mathbf{x}_k + \lambda_{swap}(\mathbf{s}_k - \mathbf{y}_k)) - f(\mathbf{x}_k)$ (Improvement of SWAP step)
  Compute $\delta_k = max(\delta_{swap}, \delta_{fw})$
  if $\delta_k = \delta_{swap}$ **then**
    Clip the line serach parameter, $\lambda_{swap*} = max(\lambda_{swap}, \alpha_k(\mathbf{y}_k))$
    if $\lambda_{swap*} = \alpha_k(\mathbf{y}_k)$ mark it as SWAP-drop step
    if $\lambda_{swap*} = \lambda_{swap}$ mark it as SWAP-add step
    Perform the SWAP step $\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_{swap*}(\mathbf{s}_k - \mathbf{y}_k)$
  end
  else
    Perform the FW step
    $\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_{fw*}(\mathbf{s}_k - \mathbf{x}_k)$
  end
end
Perform the rounding $\mathbf{z} \leftarrow rounding(\mathbf{x}_K)$
return $\mathbf{z}$

5.3 Speed-UP

Although Frank-Wolfe algorithm speeds up the optimization significantly, we noticed that there is a room for even further speed up. This is important specially when the number of targets in the scene reaches to a few hundreds. The main motivation behind this is that, a lot of candidate locations can be removed which leads to reduction in the number of variables in the optimization (An example is shown in Figure 5.4(b,d)). One naive way of removing the undesired candidate locations is thresholding the confidence values of our detector ($\mathbf{c}_a$) or thresholding the confidence value of all three linear terms ($\mathbf{c}_a + \mathbf{c}_m + \mathbf{c}_{nm}$). This is similar to what a pre-trained object detector does. However, this may results in removing useful candidates that represent the targets and are assigned low scores due to pose changes or occlusion. Moreover, keeping only the samples with high confidence values in the linear terms (Figure 5.4(c)) in Equation 5.1, limits the effect of our quadratic terms that capture the spatial proximity as well as grouping.

Instead, we incorporate a better way of sampling candidate locations, which does not drop the accuracy and at the same time is capable of showing the effect of each term in our optimization. Our sampling starts with first selecting the top m extrema points in the probability map obtained from the linear terms in Equation 5.1 (m is set to three). An example is shown in Figure 5.4(c). In order not to limit ourselves to the high confidence locations found by $\mathbf{c}_a + \mathbf{c}_m + \mathbf{c}_{nm}$, we further sample an extra 10 candidates in a small neighborhood, (6×6) of each extrema point (Figure 5.4(d)). The latter step will allow the quadratic terms to make the necessary changes to the target locations in order to improve the optimization cost. We observed in our experiments that, if only the extrema points are selected, the average performance for the 9 sequences is 1.5% lower than the ones reported in Table 5.1. However, when we use our sampling technique, the performance is only 0.2% lower compare to when all the candidate locations are used.

An example of our sampling technique is shown in Figure 5.4. This procedure reduces the number of candidate more than an order of magnitude which results in significant speed up.

Furthermore, in Figure 5.5 we show the effect of m in the speed-up. As can be seen when the number of candidates are reduced by an order of magnitude, we get almost six times speed-up in the optimization. As also mentioned earlier, the performance almost remains the same when we set $m = 3$.

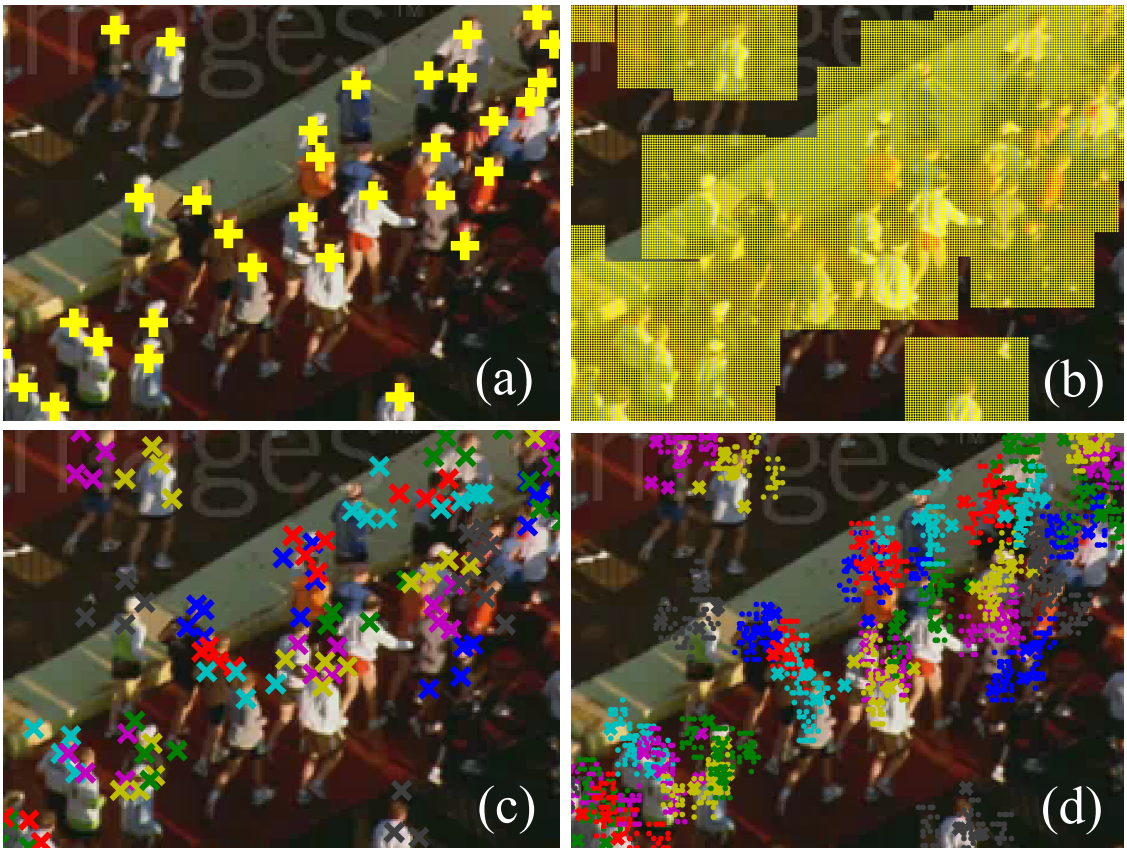


Figure 5.4: A comparison of our speed-up sampling technique vs dense sampling of candidate locations. (a) shows the targets to be tracked. (b) illustrates the candidates sampled densely in the neighborhood of each target. (c) shows the selected extrema locations (each candidate is shown with a cross). (d) visualizes the final candidates that survived using the proposed speed-up technique (each candidate is shown with a small dot).

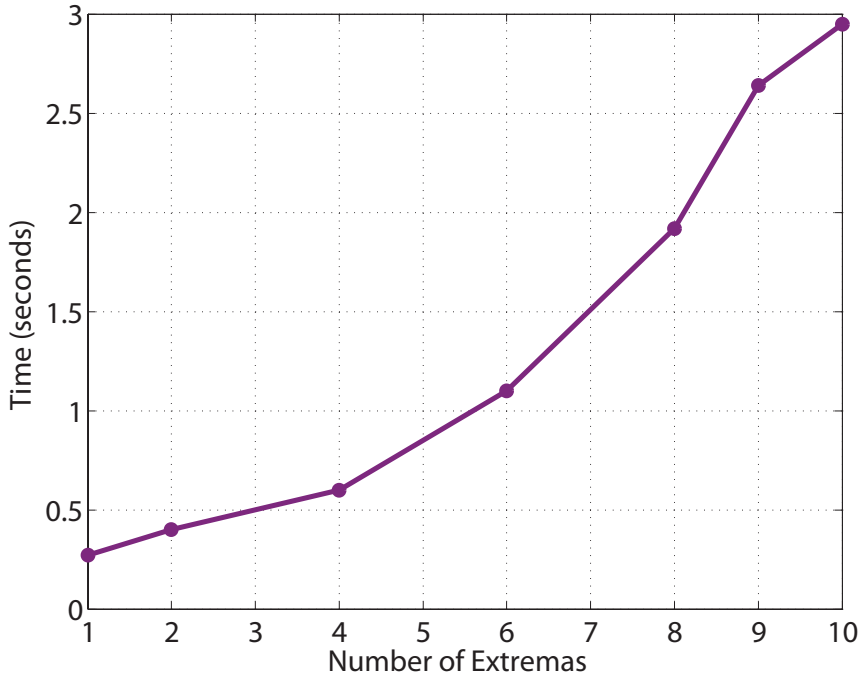


Figure 5.5: This figure illustrates the run-time vs the number of extremas (m).

5.4 Experiments

We perform exhaustive experiments on nine high density crowd sequences of [2] and two new sequences with medium crowd density. These sequences include different scenarios and challenges. All sequences are taken using cameras facing down.

High-density Crowd Sequences: First frames of the high density crowd sequences of [2] are shown in Figure. 5.11. Seq-3, seq-4, seq-5 and seq-6 show marathon events where targets with similar appearance run close to each other. Seq-1 shows daily commute of crowds. Targets look very similar to their background and they often get confused with the background. Seq-2 is taken from Hajj event, seq-7 shows the crowd in a train station. Seq-8 is taken from airport lobby and seq-9 contains people crossing a street. This dataset contains structured and unstructured crowd sequences with lots of anomalies even in structured crowd sequences. The crowd density in each

video is different. Some statistics of the dataset is shown in Table. 5.1. The number of individuals annotated in each sequences ranges from 57 to 747.

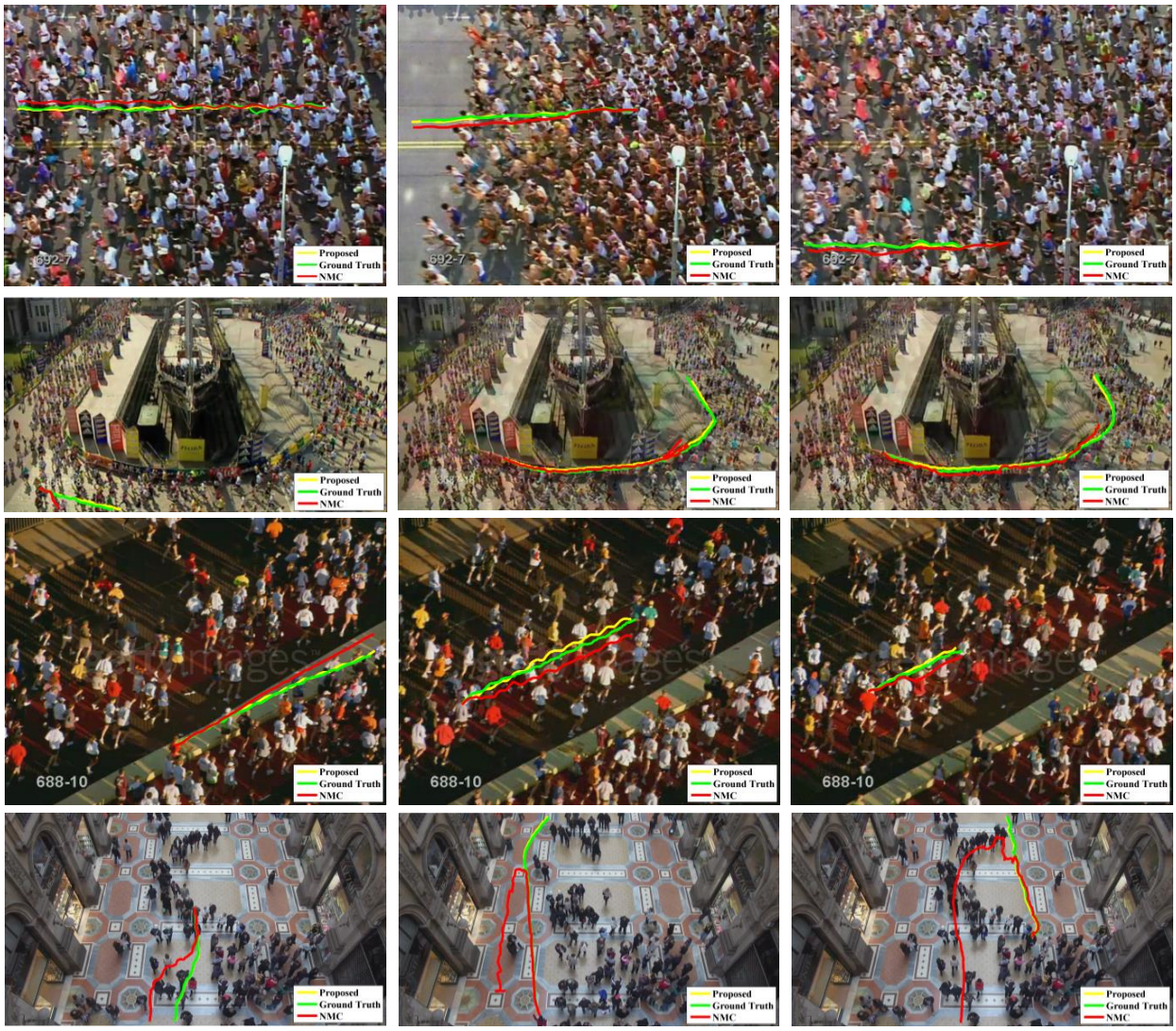


Figure 5.6: This figure shows quantitative results of our method for a few challenging targets. We also compare our results with the one in [2]. For some sequences the proposed method gives tracks very close to the ground truth tracks. Since green is superimposed on yellow, due to that tracks in yellow may not be that visible.

Medium-density Crowd Sequences: We annotated two new sequences in addition to the high density crowd sequences to show effectiveness of our method for medium density crowd

sequences as well. We have named them, Galleria and Galleria2. These two sequences are recorded from Galleria mall in Milan and have been used in the vision community for other tasks such as group detection [122]. We annotated the first 2000 frames of each sequence and included them in our evaluation. With the permission from the group which published the sequences [123], we plan to release the videos along with their annotations upon the acceptance of the paper. The sequences are recorded using a cameras facing down. Galleria1 contains 200 targets and Galleria2 contains 215 targets. Some example frames of these two sequences along with qualitative results are shown in Figure 5.6.

5.4.1 Overall Performance

We compare our method with previous methods designed to track individuals in high-density crowds. Below we provide a summary of each approach:

- Floor Fields method of Ali and Shah (FF) [5] : This method is based on the assumption that all targets follow global crowd behavior at every location in the scene. The prior information they learn, called floor fields, restrict the motion of each individual in a scene severely.

- Correlated Topic Model (CTM) [6]: CTM tracker utilized a Correlated Topic Model to model crowd behavior at each location. Their model does not have the assumption of [5] in which targets are restricted to take only one direction at each location. But still it needs to learn dynamic model of the scene given some training data. In their construction, words correspond to low level quantized motion features and topics correspond to crowd behaviors

- Mean-shift Belief Propagation (MSBP) [7]: MSBP tracker models the contextual relationship between the target in an MRF framework. The mean-shift belief propagation technique was used for the optimization.

- Prominence Neighborhood Motion Concurrence (NMC) [2]: The PNMC tracker utilized a

template based tracker at its core. The targets are tracked individually in an ordered fashion employing information from the neighborhood and confidence from the template based tracker.

We used the manual annotation of initial target locations provided with the dataset. The template size is the same as the one used in [2].

Table 5.1: Quantitative results of our method in terms of **Tracking Accuracy** when pixel threshold is set to 15. We compared our method with six competitors on nine sequences of [2].

	Seq1	Seq2	Seq3	Seq 4	Seq 5	Seq 6	Seq 7	Seq 8	Seq 9
#Frames	840	134	144	492	464	133	494	126	249
#People	152	235	175	747	171	600	73	58	57
NCC	49%	85%	58%	52%	33%	52%	50%	86	33%
MS	19%	67%	16%	8%	7%	36%	28%	43%	10%
MSBP	57%	97%	71%	69%	51%	81%	68%	94%	40%
FF	74%	99%	83%	88%	66%	90%	68%	93%	47%
CTM	76%	100%	88%	92%	72%	94%	65%	94%	x%
NMC	80%	100%	92%	94%	77%	94%	67%	92%	63%
Proposed	**86%**	**99%**	**96%**	**97%**	**78%**	**96%**	67%	90%	78%

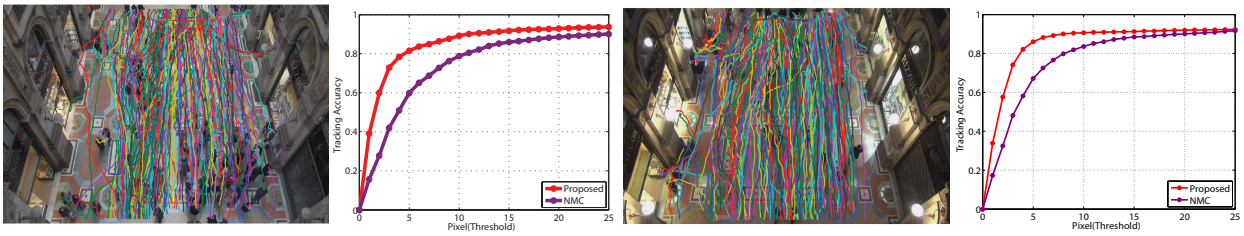


Figure 5.7: This figure shows tracks (show on the left) and quantitative results of our method. We compare our method with NMC [2] which achieves the best results on the 9 high-density crowd sequences.

5.4.2 *Quantitative Results*

We followed the same metrics as the one in [2] to quantitatively compare our method with others. The tracking accuracy for different pixel-error threshold is shown in Figure. 5.11 for the 9 high-density crowd sequences of [2]. The results on Galleria1 and Galleria2 sequences are also shown in Figure 5.7. Similar to [2] we also provide the accuracy for when the pixel threshold is set to 15 in Tables 5.1 and 5.2. Our method outperforms the existing approaches in most sequences. The performance increase in sequence 9 is worth a special mention here. In this sequences people are walking in the opposite directions, which is the scenario that [5] is not designed to handle. The heuristic such as instantaneous flow proposed in [2] are not suitable for tracking in these scenarios. In Figure 5.12 we show some of the failure cases of our method.

Table 5.2: Quantitative Comparison of our method with the tracker of [2] when pixel threshold is set to 15 on two new sequences of Galleria1 and Galleria2. On average we improve NMC tracker of [2] by 4.5% on these two sequences.

	Galleria1	Galleria2
#Frames	2000	2000
#People	200	215
PNMC	86%	88%
Proposed	**92%**	**91%**

5.4.3 *Contribution of the Proposed Terms*

In order to show the effectiveness of the proposed terms in the objective function we conducted an experiment with different setups. The detailed results for each sequence is shown in Table. 5.3. We tried a combination of different terms. Our baseline (B(NCC)) is just a template-based tracker without the other terms in Eq. 5.1. Comparing the baseline of [2] that also uses a NCC based tracker with the equivalent of ours (B+Mo(NCC)), we clearly see that formulating the problem as multi-target tracking with joint optimization can improve the performance a lot. Our

baseline is 30% higher that the one of [2].

Table 5.3: Quantitative comparison of **Tracking Accuracy** using different terms in cost function of Equation. 5.1.B is the baseline. Mo is the motion term (c_m), SP is the spatial proximity term(c_{sp}), NMo is neighborhood motion cost (c_{nm}). NCC represent the template based tracker based on Normalized Cross Correlation and KCF represents the discriminative model based on kernalized correlation filters.

	Seq1	Seq2	Seq3	Seq 4	Seq 5	Seq 6	Seq 7	Seq 8	Seq 9
#Frames	840	134	144	492	464	133	494	126	249
#People	152	235	175	747	171	600	73	58	57
B(NCC)	78.1%	98.5%	85.5%	91.2%	65.3%	93.5%	66.8%	88.8	63.7%
B+Mo(NCC)	76.9%	98.8%	87.6%	92.7%	72.2%	92.9%	73.8%	91.7%	64.1%
B+Mo+SP(NCC)	77.3%	99.0%	93.7%	92.3%	76.1%	94.3%	**74.4%**	**93.0%**	67.7%
B+Mo+Gr(NCC)	79.7%	99.0%	93.1%	93.7%	77.1%	94.3%	72.9%	93.0%	64.1%
B+Mo+SP+Gr+NMo(NCC)	80.3%	99.0%	93.4%	95.2%	76.7%	94.7%	74.0%	92.1%	69.8%
B+Mo+SP+Gr+NMo(KCF)	**86.1%**	**98.6%**	**96.4%**	**96.8%**	**77.7%**	**95.8%**	67.2%	90.4%	**77.9%**

We next add different components of the objective function in 5.1 to the baseline tracker one by one, in order to evaluate their potency. $B + Mo$ is our baseline tracker where we add the linear motion constraint to it. $B + Mo + SP$ and $B + Mo + Gr$ are the same as $B + Mo$ with an additional spatial proximity constraint and group constraint respectively. We later add the neighborhood motion term ($B+Mo+Sp+NMo$) to evaluate its effectiveness and finally we replace the generative template based tracker with our discriminative model ($B+Mo+Ov+NMo(disc)$)and show that it further improves the overall performance. From the results in Table 5.3, we observe that the improvement of different terms depends on different factors in the scene including the density of crowd. For example in sequence 7 and 8 the performance slightly decreases when adding the neighborhood motion term. This is mostly due to heterogeneity of target movement in those sequences. Spatial proximity and grouping terms in all the sequences improve the performance which show their potency. One should note that people rarely form groups in extremely crowded scenes. However, our observation shows that enforcing the consistency in formation of the coherently moving targets that are close to each will help improving the results.

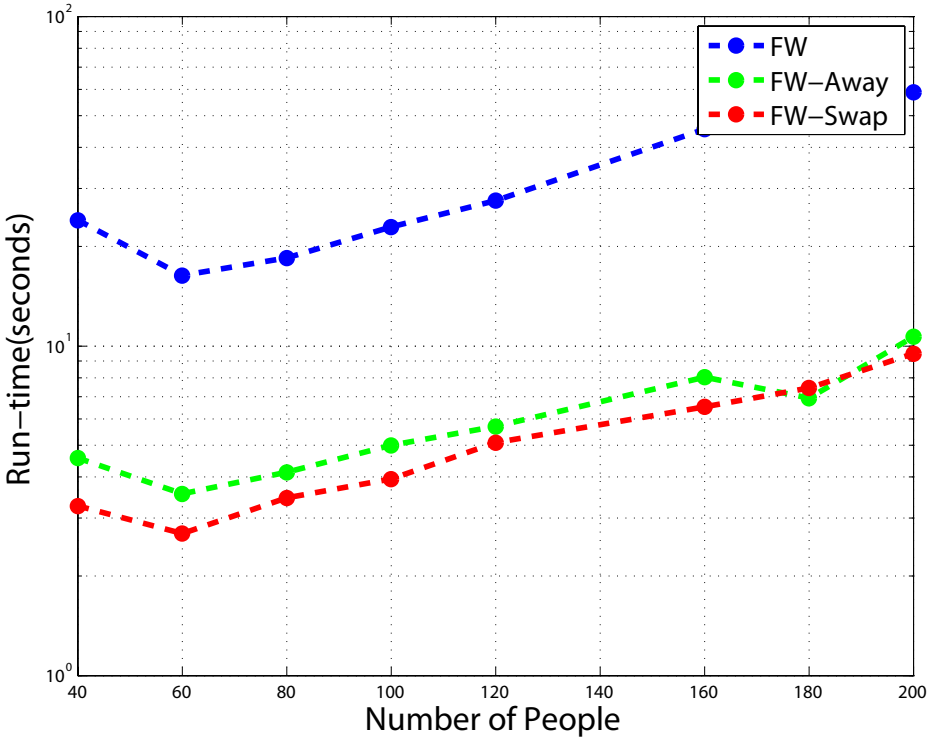


Figure 5.8: The figure shows the run-time comparison of Frank-Wolfe(FW), Frank-Wolfe with Away steps (FW-Away) and Frank-Wolfe with Swap steps (FW-Swap) on one of our sequences. It is clear that the Away step technique improves the run-time of FW significantly. However, using the Swap step we can further speed-up the optimization.

5.4.4 Run-time Comparison

We conduct several experiments to compare run-time of our method with competitive optimization methods. Firstly, we provide a runtime comparison of the FW-Swap in crowd tracking with previous versions of FW, including the original FW [124] and widely used version of FW with Away steps [102]. The results are shown in Figure 5.8. As can be seen the run-time increases as the number of people increases. The duality-gap which determines the stopping criteria and quality of the final solution is set to $\varepsilon = 0.00001$. In practice, we select a higher number $\varepsilon = 0.01$, however the reason we set it to a lower number is that, in this experiment, we are interested in the convergence of these methods.

In the second experiment, we compare our optimization with publicly available QP solvers. We selected ILOG CPLEC [43], which is one of most popular solvers and is used extensively in research community. We compare the run-time of CPLEX with FW-Swap for different duality gap values (in our experiments we set $\varepsilon = 0.01$). The results are shown in Fig. 5.9. It is clear that the complexity of CPLEX, even after relaxing the binary constraint, is still very high as the number of targets increases. Finally, we show the number of iterations that our algorithm takes to converge for different setups in Figure 5.10. All the experiments are performed on a quad-core 3.0 GHz machine.

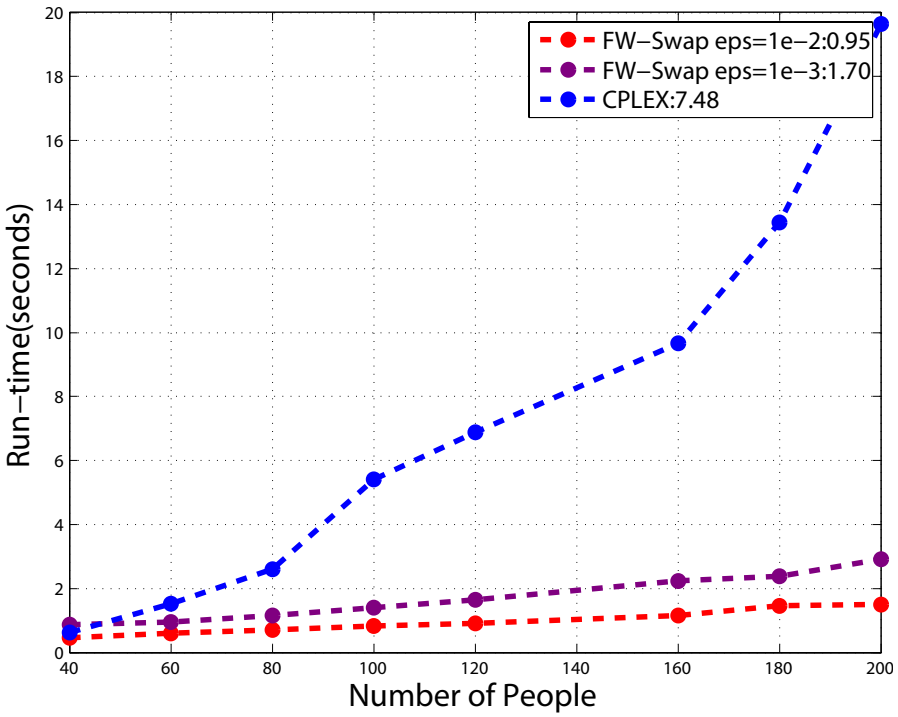


Figure 5.9: This figure illustrates the run-time comparison of the proposed method for different values of duality gaps ε. We also compare the run-time of FW-Swap with ILOG CPLEX. It is evident that the FW method scales to much larger problem size and requires far less computations.

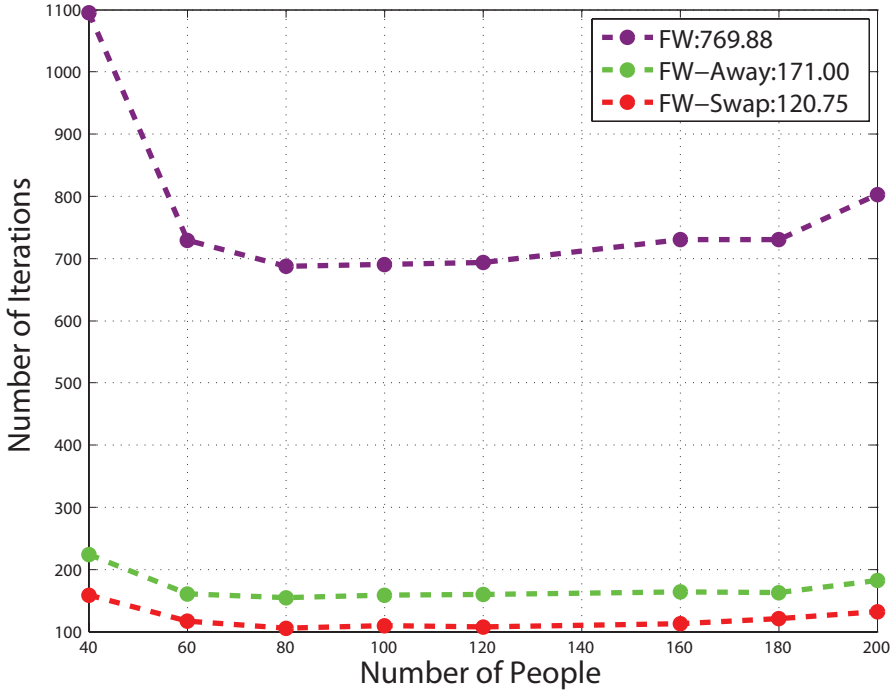


Figure 5.10: This figures illustrates the number of iterations that it takes for the optimization to converge for original Frank-Wolfe (FW), Frank-Wolfe with Away steps (FW-Away) and Frank-Wolfe with Swap steps (FW-Swap). The value of ε is set to 0.0001.

5.5 Summary

In this chapter, we showed that the multi-target tracking in high density crowded scenes can be formulated through Binary Quadratic Programming. Our formulation includes several components that are important in designing a good tracker that works for crowded scenes. Those components include, appearance, motion, neighborhood motion, pairwise spatial relationship and pairwise group information. We show that the proposed formulation can be efficiently solved using Frank-Wolfe optimization with SWAP steps. Additionally, the proposed speed-up technique can reduce the computational complexity, which is necessary when dealing with large number of people. We tested our algorithm on publicly available sequences as well as new sequences and showed state of the art performance.

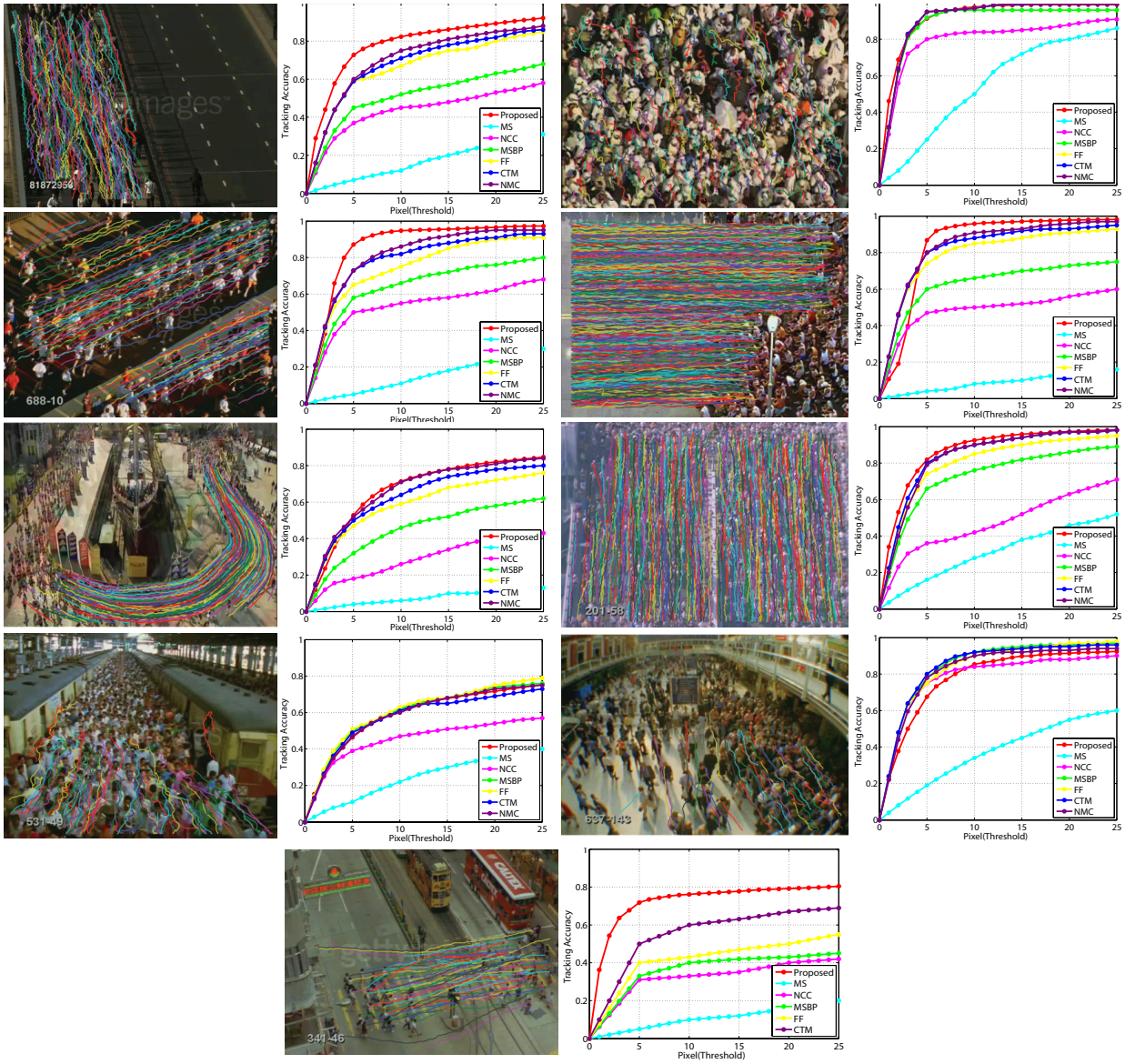


Figure 5.11: This figure shows tracks and quantitative results of our method with competitive approaches of FF [5], CTM [6], MSBP [7], PNMC [2] and MS [8]. For each sequence we show the qualitative results of all tracks and on the right we show the quantitative comparison. Each plot shows the tracking accuracy vs pixel error.

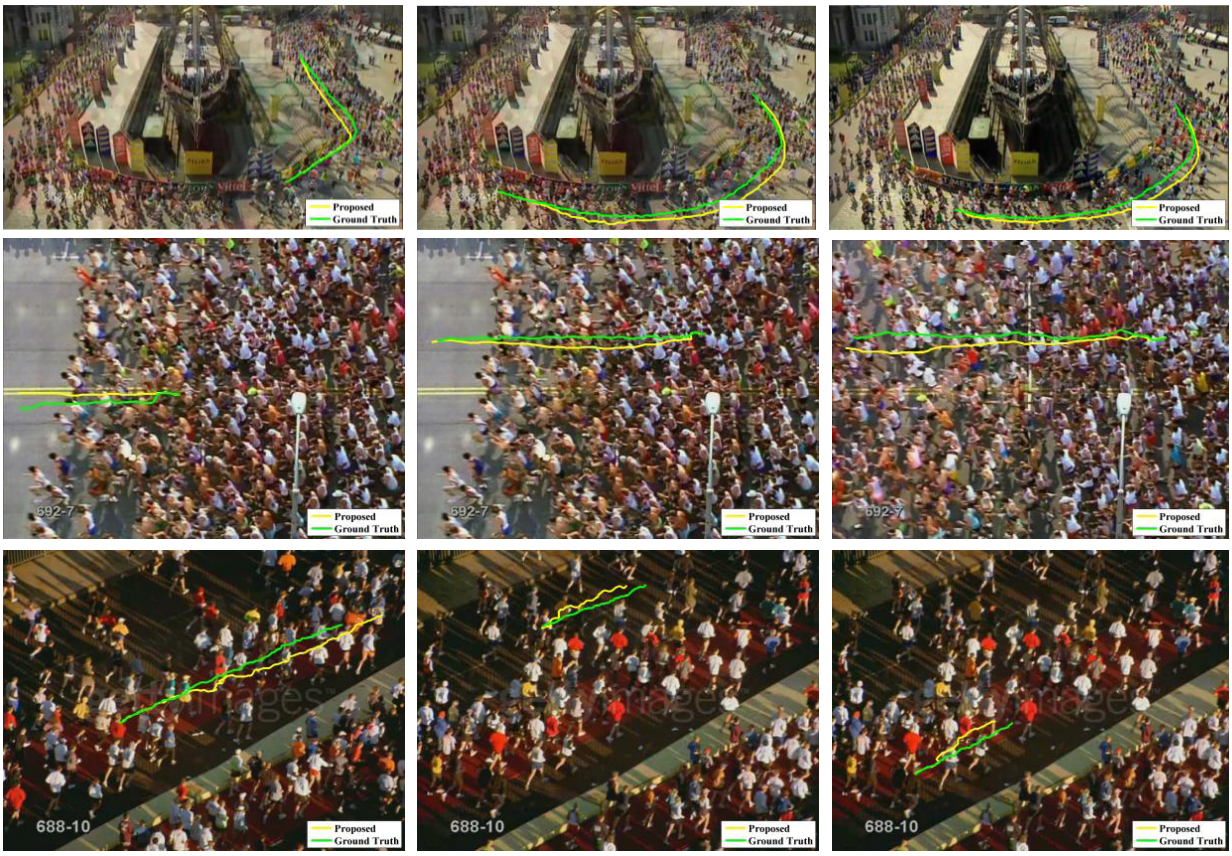


Figure 5.12: This figure shows the failure cases of our method in three different sequences. The failure cases mostly belong to the targets with inaccurate initialization which leads to poor appearance information. The drift usually happens in the first few frames.

CHAPTER 6: CONCLUSION AND FUTURE WORK

In this dissertation explores the fundamental problem of multi-target tracking in surveillance videos. Chapter 3 and Chapter 4 focus on data association which is the backbone to many multi-target tracking algorithms. In these two chapters we show how maximum clique problem can be used to solve data association and achieve state of the art performance on several datasets. We show that the proposed trackers can track dozens of targets efficiently in surveillance videos. We move on by further proposing a new tracker in Chapter 5 that is designed to track hundreds of targets in extremely crowded sequences where data association based tracking mathods fail.

The first data association technique proposed in this dissertation utilizes Generalized Minimum Clique Problem. Our method is based on shifting the approximation from the temporal domain to the object domain while keeping a notion of the full object domain implicitly. We argued that this framework yields a better developed approach to data association. A two stage tracking method, i.e. finding mid-level tracklets first and forming full trajectories later, is employed to cope with different characteristics of human motion in the short and long term. Also, we utilized a global approach to compute the motion cost at the tracklet level.

In order to address the limitations of GMCP tracker we proposed a new graph theoretic problem in Chapter 4. We formulate multiple target tracking as a Generalized Maximum Multi Clique problem where it is solved through Binary Integer Programing. GMMCP finds the tracks for all the targets simultaneously while GMCP is able to tracks one-by-one. Additionally, GMMCP yields itself to a real-time implementation for small- and medium-sized sequences, which was not the case for GMCP tracker. The speed up is achieved through a better occlusion handling technique, which uses aggregated dummy nodes.

When dealing with high density crowd sequences, taken by overhead cameras, data association based tracking methods fail. In order to address the relatively unexplored problem of tracking individuals in crowded scenes, we present a new tracking method in Chapter 5. We showed

that the multi-target tracking in high density crowded scenes can be formulated through Binary Quadratic Programming. Our formulation includes several components that are important in designing a good tracker that works for crowded scenes. These components include, appearance, motion, neighborhood motion, pairwise spatial relationship and pairwise group information. We show that the proposed formulation can be efficiently solved using Frank-Wolfe optimization with SWAP steps. Additionally, the proposed speed-up technique can further reduce the computational complexity, which is necessary when dealing with large number of people.

In summary, this dissertation shows that formulating data association in a global manner, where all the pairwise relationships between targets in a batch of frames are taken into account, using a k-partite complete graph, provides a more accurate formulation to multi-target tracking. Either we find the approximate solution to this formulation, which is the case in GMCP tracker, or we find the exact solution, as was done in GMMCP tracker, the performance is improved significantly. Additionally, to address the cases that pre-trained object detectors fail, such as high-density crowd sequences, we provide an efficient framework to solve detection and data association simultaneously in an online manner. Finally, several new datasets were used to evaluate the performance of our approaches in this dissertation. These sequences will be useful for other researchers who are working in the area of multi-target tracking.

6.1 Future Work

Here, we highlight some extensions and areas of future research that are likely to improve performance of proposed approaches.

GMCP and **GMMCP**. The results are likely to improve using a non-linear motion model. This is especially important when tracking sport players. Non-linear motion models allow one to increase the temporal span during data association without worrying about tracking targets with abrupt motion. Another direction is to design an efficient greedy optimization for the Generalized

Maximum Multi Clique problem that provides a high quality solution. In Chapter 4, we find the exact solution using Binary Integer Programing. However, the complexity becomes a burden as the number of people in the scene exceeds a few dozen. Thus having an efficient optimization will help using this tracker for crowded sequences.

Furthermore, there is room for improving the affinity measures used in GMCP or GMMCP tracker. Recently deeply learned features have shown to be effective in tracking single objects. Utilizing such technique to design global features that can be used in our trackers, can help improve the performance.

Crowd tracker. When solving detection and data association simultaneously, the biggest challenge is that, tracks of some targets may drift after few frames. Drifting happens when targets get occluded or when the appearance/motion cues are not strong. Right now we rely on different components in our formulation such as neighborhood motion or group information to help avoid drifting. However, one future direction could be to increase the temporal locality. This means one can consider more than two frames during data association and incorporate the temporal smoothness. This consequently requires a faster optimization. As the number of targets or the number of frames during data association increases the complexity increases as well and Frank Wolfe optimizations becomes slow for large number of targets. This will open the room for investigating faster QP solvers.

LIST OF REFERENCES

[1] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling," in *Conf. on Computer Vision and Pattern Recognition*, 2012.

[2] H. Idrees, N. Warner, and M. Shah, "Tracking in dense crowds using prominence and neighborhood motion concurrence," vol. 32, pp. 14 – 26, 2014.

[3] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual Tracking: an Experimental Survey," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2013.

[4] A. R. Zamir, A. Dehghan, and M. Shah, "GMCP-Tracker: Global Multi-object Tracking Using Generalized Minimum Clique Graphs," in *European Conf. on Computer Vision*, 2012.

[5] S. Ali and M. Shah, "Floor fields for tracking in high density crowd scenes," 2008.

[6] M. Rodriguez, S. Ali, and T. Kanade, "Tracking in unstructured crowded scenes," 2009.

[7] M. Park, Y. Liu, and R. Collins, "Efficient mean shift belief propagation for vision tracking,," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[8] R. V. Dorin Comaniciu and P. Meer., "Real-time tracking of non-rigid objects using mean shift," in *CVPR*, 2000.

[9] A. K. K.C. and C. D. Vleeschouwer, "Discriminative label propagation for multi-object tracking with sporadic appearance features," in *Int'l. Conf. on Computer Vision*, 2013.

[10] L. Wen, W. Li, J. Yan, Z. Lei, D. Yi, and S. Z. Li, "Multiple target tracking based on undirected hierarchical relation hypergraph," in *Conf. on Computer Vision and Pattern Recognition*, 2014.

[11] W. Brendel, M. Amer, and S. Todorovic, "Multiobject tracking as maximum weight independent set," in *Conf. on Computer Vision and Pattern Recognition*, 2011.

[12] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, "Multiple object tracking using k-shortest paths optimization," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2011.

[13] A. Andriyenko, K. Schindler, and S. Roth, "Discrete-Continuous Optimization for Multi-Target Tracking," in *Conf. on Computer Vision and Pattern Recognition*, 2012.

[14] S.-H. Bae and K.-J. Yoon, "Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning," in *Conf. on Computer Vision and Pattern Recognition*, 2014.

[15] C. Dicle, M. Sznaier, and O. Camps, "The way they move: Tracking targets with similar appearance," in *Int'l. Conf. on Computer Vision*, 2013.

[16] Y. Taigman, M. Yang, M. A. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification.," in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[17] A. Dehghan, E. Ortiz, R. Villegas, and M. Shah, "Who do i look like? determining parent-offspring resemblance via gated autoencoders.," in *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[18] P. J. Figueroa, N. J. Leite, and R. M. Barros., "Tracking soccer players aiming their kinematical motion analysis.," in *ELSEVIER Journal on Computer Vision and Image Understanding*, 2006.

[19] R. T. Collins and P. Carr., "Hybrid stochastic/deterministic optimization for tracking sports players and pedestrians.," in *European Conf. on Computer Vision*, 2014.

[20] H. Hamer, K. Schindler, E. Koller-Meier, and L. Van Gool, "Tracking a hand manipulating an object.," in *Int'l. Conf. on Computer Vision*, 2009.

[21] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt, "Fast and robust hand tracking using detection-guided optimization.," in *Conf. on Computer Vision and Pattern Recognition*, 2015.

[22] S. Melax, L. Keselman, and S. Orsten., "Dynamics based 3d skeletal hand tracking.," in *Canadian Information Processing Society*, 2013.

[23] A. D. Straw, K. Branson, T. R. Neumann, and M. H. Dickinson., "Multi-camera real-time three-dimensional tracking of multiple flying animals.," in *Journal of The Royal Society Interface*, 2010.

[24] Y. Liu, H. Li, and Y. Q. Chen, "Automatic tracking of a large number of moving targets in 3d.," in *European Conference on Computer Vision*, 2012.

[25] Z. Khan, T. Balch, and F. Dellaert, "Mcmc data associa- tion and sparse factorization updating for real time multitarget tracking with merged and multiple measurements.," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1972.

[26] e. a. Matov, Alexandre, "Analysis of microtubule dynamic instability using a plus-end growth marker.," in *Nature methods*, 2010.

[27] A. Matov, M. M. Edvall, G. Yang, and G. Danuser, "Optimal-flow minimum-cost correspondence assignment in particle flow tracking," in *ELSEVIER Journal on Computer Vision and Image Understanding*, 2011.

[28] K. T. A. et al., "plustiptracker: quantitative image analysis software for the measurement of microtubule dynamics.," in *Journal of structural biology*, 2011.

[29] e. a. Godinez, William J., "Deterministic and probabilistic approaches for tracking virus particles in time-lapse fluorescence microscopy image sequences.," in *Medical image analysis*, 2009.

[30] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Conf. on Computer Vision and Pattern Recognition*, 2005.

[31] X. Wang and T. Han, "An hog-lbp human detector with partial occlusion handling," in *Int'l. Conf. on Computer Vision*, 2009.

[32] 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*, vol. 32, no. 9, pp. 1627–1645, 2010.

[33] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2010.

[34] A. Bera, N. Galoppo, D. Sharlet, A. Lake, and D. Manocha, "Adapt: Real-time adaptive pedestrian tracking for crowded scenes," 2014.

[35] K. Shafique and M. Shah, "A noniterative greedy algorithm formultiframe point correspondence," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2005.

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

[37] H. Pirsiavash, D. Ramanan, and C. Fowlkes, "Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects," in *Conf. on Computer Vision and Pattern Recognition*, 2011.

[38] A. Andriyenko and K. Schindler, "Multi-target Tracking by Continuous Energy Minimization," in *Conf. on Computer Vision and Pattern Recognition*, 2011.

[39] e. a. Idrees, Haroon, "Multi-source multi-scale counting in extremely dense crowd images," 2013.

[40] e. a. Chen, Xiaojing, "An online learned elementary grouping model for multi-target tracking,," in *CVPR*, 2014.

[41] Z. Qin and C. R. Shelton., "Improving multi-target tracking via social grouping.," in *CVPR*, 2012.

[42] S. Pellegrini, A. Ess, and L. V. Gool., "Improving data association by joint modeling of pedestrian trajectories and groupings.," in *ECCV*, 2010.

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

[44] E. D. Andersen and K. D. Andersen., "The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm.," in *High performance optimization.*, 2000.

[45] V. Chari, S. Lacoste-Julien, I. Laptev, and J. Sivic, "On pairwise costs for network flow multi-object tracking.," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[46] Y. Li, C. Huang, and R. Nevatia, "Conf. on Computer Vision and Pattern Recognition," in *Learning to associate: Hybridboosted multi-target tracker for crowded scene*, 2009.

[47] S. Pellegrini, A. Ess, and L. van Gool, "Improving data association by joint modeling of pedestrian trajectories and groupings," in *European Conf. on Computer Vision*, 2010.

[48] J. F. Henriques, R. Caseiro, and J. Batista, "Globally optimal solution to multi-object tracking with merged measurements," in *Int'l. Conf. on Computer Vision*, 2011.

[49] B. Yang and R. Nevatia., "An online learned crf model for multi-target tracking," in *Conf. on Computer Vision and Pattern Recognition*, 2012.

[50] A. Dehghan, H. Idrees, A. R. Zamir, and M. Shah, "Automatic detection and tracking of pedestrians in videos with various crowd densities," in *Pedestrian and Evacuation Dynamics*, 2012.

[51] S. Chen, A. Fern, and S. Todorovic, "Multi-object tracking via constrained sequential labeling," in *Conf. on Computer Vision and Pattern Recognition*, 2014.

[52] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram.," in *Conf. on Computer Vision and Pattern Recognition*, 2006.

[53] D. L. S. Oron, A. Bar-Hillel, and S. Avidan, "Locally orderless tracking.," in *Conf. on Computer Vision and Pattern Recognition*, 2012.

[54] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift.," in *Conf. on Computer Vision and Pattern Recognition*, 2000.

[55] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Int'l. Conf. on Computer Vision*, 2011.

[56] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework.," in *IJCV*, 2004.

[57] H. T. Nguyen and A. W. M. Smeulders, "Fast occluded object tracking by a robust appearance filter.," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2004.

[58] K. Briechle and U. D. Hanebeck, "Template matching using fast normalized cross correlation,," in *in Proc SPIE*, 2001.

[59] S. Avidan, "Support vector tracking.," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2004.

[60] S. Avidan, "Ensemble tracking," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2007.

[61] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui., "Visual object tracking using adaptive correlation filters.," in *Conf. on Computer Vision and Pattern Recognition*, 2010.

[62] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High- speed tracking with kernelized correlation filters.," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2015.

[63] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection withkernels.," in *European Conf. on Computer Vision*, 2012.

[64] B. Babenko, M. Yang, and M. Hsuan, "Visual tracking with online multiple instance learning," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2010.

[65] Z. K. andKrystian Mikolajczyk and J. Matas, "Tracking-Learning-Detection," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2010.

[66] S. Wang, H. Lu, F. Yang, and M. Yang, "Superpixel tracking," in *Conf. on Computer Vision and Pattern Recognition*, 2011.

[67] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "MUlti-Store Tracker (MUSTer): a Cognitive Psychology Inspired Approach to Object Tracking.," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[68] M. Hahn, S. Chen, and A. Dehghan, "arxiv," in *Deep Tracking: Visual Tracking Using Deep Convolutional Networks.*, 2015.

[69] Y. Wu, J. Lim, and M. H. Yang, "Online Object Tracking: A Benchmark," in *Conf. on Computer Vision and Pattern Recognition*, 2013.

[70] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojir, G. Fernandez, A. Lukežič, A. Dimitriev, A. Petrosino, A. Saffari, B. Li, B. Han, C. Heng, C. Garcia, D. Pangeršič, G. Häger, F. S. Khan, F. Oven, H. Possegger, H. Bischof, H. Nam, J. Zhu, J. Li, J. Y. Choi, J.-W. Choi, J. ao F. Henriques, J. van de Weijer, J. Batista, K. Lebeda, K. Öfjäll, K. M. Yi, L. Qin, L. Wen, M. E. Maresca, M. Danelljan, M. Felsberg, M.-M. Cheng, P. Torr, Q. Huang, R. Bowden, S. Hare, S. Y. Lim, S. Hong, S. Liao, S. Hadfield, S. Z. Li, S. Duffner, S. Golodetz, T. Mauthner, V. Vineet, W. Lin, Y. Li, Y. Qi, Z. Lei, and Z. Niu, "The visual object tracking vot2014 challenge results," 2014.

[71] L. Zhang and L. van der Maaten, "Structure Preserving Object Tracking," in *Conf. on Computer Vision and Pattern Recognition*, 2013.

[72] A. Dehghan, Y. Tian, P. H. S. Torr, and M. Shah, "Target identity-aware network flow for online multiple target tracking," in *Conf. on Computer Vision and Pattern Recognition*, 2015.

[73] R. W. Sittler, "An optimal data association problem in surveillance theory," *Military Electronics, IEEE Transactions on*, vol. 8, no. 2, pp. 125–139, 1964.

[74] T. E. Fortmann and S. Baron, "Problems in multi-target sonar tracking," in *Decision and Control including the 17th Symposium on Adaptive Processes, IEEE Conference on*, vol. 17, pp. 1182–1188, 1978.

[75] P. Smith and G. Buechler, "A branching algorithm for discriminating and tracking multiple objects," *Automatic Control, IEEE Transactions on*, vol. 20, no. 1, pp. 101–104, 1975.

[76] D. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[77] R. B. Klimek, T. W. Wright, and R. S. Sielken, *Color image processing and object tracking system*. National Aeronautics and Space Administration, Lewis Research Center, 1966.

[78] R. E. Kalman, "A new approach to linear filtering and prediction problems.," *Journal of Fluids Engineering*, pp. 35–45, 1960.

[79] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems.," *International Society for Optics and Photonics*, 1997.

[80] G. Shu, A. Dehghan, and M. Shah, "Improving an Object Detector and Extracting Regions using Superpixels," in *Conf. on Computer Vision and Pattern Recognition*, 2013.

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

[82] H. B. Shitrit, J. Berclaz, F. Fleuret, and P. Fua, "Tracking multiple people under global appearance constraints," in *Int'l. Conf. on Computer Vision*, 2011.

[83] B. Wu and R. Nevatia, "Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors," in *IJCV*, 2007.

[84] A. K. K.C. and C. D. Vleeschouwer, "Discriminative Label Propagation for Multi-Object Tracking with Sporadic Appearance Features," in *Int'l. Conf. on Computer Vision*, 2013.

[85] A. Milan, L. Leal-Taix, K. Schindler, and I. Reid, "Joint tracking and segmentation of multiple targets," in *Conf. on Computer Vision and Pattern Recognition*, 2015.

[86] A. A. Butt and R. T. Collins, "Multi-target tracking by lagrangian relaxation to min-cost network flow," in *Conf. on Computer Vision and Pattern Recognition*, 2013.

[87] H. B. Shitrit, J. Berclaz, F. Fleuret, and P. Fua, "Multi-commodity network flow for tracking multiple people," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2013.

[88] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *Computer Vision, IEEE 12th International Conference on*, pp. 261–268, 2009.

[89] K. Yamaguchi, A. C. Berg, L. E. Ortiz, and T. L. Berg, "Who are you with and where are you going?," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pp. 1345–1352, 2011.

[90] B. R. L. Leal-Taixé, G. Pons-Moll, "verybody needs somebody: modeling social and grouping behavior on a linear programming multiple people tracker.," in *IEEE International Conference on Computer Vision Workshops (ICCV). 1st Workshop on Modeling, Simulation and Visual Analysis of Large Crowds*, 2011.

[91] S. Pellegrini, A. Ess, and L. J. V. Gool, "Improving data association by joint modeling of pedestrian trajectories and groupings," in *Computer Vision - European Conf. on Computer Vision, 11th European Conference on Computer Vision*, 2010.

[92] Z. Qin and C. R. Shelton, "Improving multi-target tracking via social grouping," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[93] X. Chen, Z. Qin, L. An, and B. Bhanu, "An online learned elementary grouping model for multi-target tracking," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[94] A. K. B. R. S. S. L. Leal-Taixé, M. Fenzi, "Learning an image-based motion context for multiple people tracking.," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[95] L. Kratz and K. Nishino, "Tracking with local spatio-temporal motion patterns in extremely crowded scenes,," 2010.

[96] L. Kratz and K. Nishino., "Going with the flow: pedestrian efficiency in crowded scenes," in *European Conference on Computer Vision*, 2012.

[97] L. Kratz and K. Nishino, "Tracking pedestrians using local spatio-temporal motion patterns in extremely crowded scenes,," 2012.

[98] P. S, E. A, S. K, and V. G. L, "You'll never walk alone: Modeling social behavior for multi-target tracking.," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[99] H. Allende, E. Frandi, R. Nanculef, and C. Sartori, "A Novel Frank-Wolfe Algorithm. Analysis and Applications to Large-Scale SVM Training ," in *Information Science*, 2014.

[100] M. Jaggi, "Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization," in *ICML*, 2013.

[101] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher, "Block-Coordinate Frank-Wolfe Optimization for Structural SVMs ," in *ICML*, 2013.

[102] A. Joulin, K. Tang, and L. Fei-Fei, "Efficient Image and Video Co-localization with Frank-Wolfe Algorithm," in *ECCV*, 2014.

[103] B. Benfold and I. Reid, "Stable multi-target tracking in real time surveillance video," in *Conf. on Computer Vision and Pattern Recognition*, 2011.

[104] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking.," 2008.

[105] M. Andriluka, S. Roth, and B. Schiele, "Monocular 3d pose estimation and tracking by detection.," 2010.

[106] J. Ferryman and A. Shahrokni, "Dataset and challenge. In: Performance Evaluation of Tracking and Surveillance," 2009.

[107] C. Feremans, M. Labbe, and G. Laporte, "Generalized network design problems," in *EJOR*, 2003.

[108] S. M. Assari, A. R. Zamir, and M. Shah, "Video classification using semantic concept co-occurrences," 2014.

[109] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking," in *Conf. on Computer Vision and Pattern Recognition*, 2008.

[110] M. Andriluka, S. Roth, and B. Schiele, "Monocular 3d pose estimation and tracking by detection," in *Conf. on Computer Vision and Pattern Recognition*, 2010.

[111] J. Ferryman and A. Shahrokni, "Pets2009: Dataset and challenge," in *Performance Evaluation of Tracking and Surveillance, International Workshop on*, 2009.

[112] R. Kasturi and et al., "Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol," in *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2009.

[113] L. Leal-Taixe and et al., "Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker," in *ICCV11Workshops*, 2011.

[114] K. Yamaguchi, A. Berg, L. Ortiz, and T. Berg, "who are you with and where are you going?," in *Conf. on Computer Vision and Pattern Recognition*, 2011.

[115] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *Int'l. Conf. on Computer Vision*, 2009.

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

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

[118] S. Chen, A. Fern, and S. Todorovic, "Online multi-person tracking-by-detection from a single, uncalibrated camera," in *Conf. on Computer Vision and Pattern Recognition*, 2014.

[119] J. Domke and Y. Aloimonos, "Deformation and viewpoint invariant color histograms," in *BMVC*, 2006.

[120] J. Shi and J. Malik, "Normalized cuts and image segmentation,," 2000.

[121] S. Lacoste-Julien and M. Jaggi, "On the Global Linear Convergence of Frank-Wolfe Optimization Variants.," in *NIPS*, 2015.

[122] F. Solera, S. Calderara, and R. Cucchiara, "Socially Constrained Structural Learning for Groups Detection in Crowd.," in *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 2015.

[123] "http://www.csai.disco.unimib.it/csai/space/start/,"

[124] M. Frank and P. Wolfe, "An algorithm for quadratic programming," in *Naval Research Logistics Quarterly*, 1956.