Binary Quadratic Programing for Online Tracking of Hundreds of People in Extremely Crowded Scenes

Afshin Dehghan, *Member, IEEE,* and Mubarak Shah, *Fellow, IEEE*

Abstract—Multi-object tracking has been studied for decades. However, when it comes to tracking pedestrians in extremely crowded scenes, we are limited to only few works. This is an important problem which gives rise to several challenges. Pre-trained object detectors fail to localize targets in crowded sequences. This consequently limits the use of data-association based multi-target tracking methods which rely on the outcome of an object detector. Additionally, the small apparent target size makes it challenging to extract features to discriminate targets from their surroundings. Finally, the large number of targets greatly increases computational complexity which in turn makes it hard to extend existing multi-target tracking approaches to high-density crowd scenarios. In this paper, we propose a tracker that addresses the aforementioned problems and is capable of tracking hundreds of people efficiently. We formulate online crowd tracking as Binary Quadratic Programing. 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, and solves detection and data association simultaneously. In order to solve the proposed quadratic optimization efficiently, where state-of art commercial quadratic programing solvers fail to find the solution in a reasonable amount of time, we propose to use the most recent version of the Modified Frank Wolfe algorithm, which takes advantage of SWAP-steps to speed up the optimization. We show that the proposed formulation can track hundreds of targets efficiently and improves state-of-art results by significant margins on eleven challenging high density crowd sequences.

Index Terms—Multiple object tracking, Crowd tracking, High density crowd, quadratic programing, Frank-Wolfe optimization

1 Introduction

WHY do we study crowds and why is tracking individuals in crowds important to us? The answer is safety. Of all the things which attracts the most attention of the world on the work of computer vision researchers, safety is the one which resonates most deeply as we as a society seek to prevent disasters and to protect individuals. When by dint of circumstance a large number of people move in a small area, safety becomes the biggest concern. Tragic incidents such as Boston marathon bombing [1] or the recent Hajj stampede [2] exemplify why there is a need for visual analysis of crowds. Moreover, understanding the dynamics of large groups of people is critical in the design and management of any type of public events. When dealing with high-density crowd scenarios such as religious rites participations, political rallies, concerts or marathons, modeling crowd dynamics can become quite complex. This could be due to several factors, such as the heterogeneity of participants or their interactions with one and another.

While analyzing crowds, tracking individuals plays an important role and provides the prerequisite to many visual tasks such as crowd management, anomaly detection, activity recognition, crowd understanding and crowd modeling for computer graphics. However, when reviewing the literature, we find most multiple object tracking methods have focused on low- or medium-density crowd sequences [3], [4], [5], [6], where the number of targets are limited to tens of people. Almost none of these methods can be used to track people directly in crowds. This is either due to the complexity of methods or the type of input data they require (e.g. availability of detection candidates in each frame).

Fig. 1. This figure shows one of our test sequences. The yellow boxes show the targets that are tracked in this sequence. On the borders we show the close-up versions of some of the targets. As can be seen targets are very small and there are very few discriminative appearance cues. Moreover targets look very similar which confuses most trackers.

When dealing with high-density crowd scenes, containing 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). The first challenge that one is 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 [7]. This makes the use of data-association based tracking methods [8], [9], [10] impossible since they rely entirely on the outcome of a pre-trained object detector.

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 other objects 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 [11], [12], [13], [14], 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 optimal assignments of new locations, objects in the scene should compete for each candidate location simultaneously, which is not the case in [11], [12], [13], [14]. 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, in this paper 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 propose five essential components for tracking individuals in crowds that capture 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, similar to the one proposed in [15], where we train a regression model for each target. This is different from previous work, where a generative model such as template based tracking is used as a baseline [11], [12], [13], [14]. 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. Please refer to Figure 3 for a summary of the constraints used in our formulation.

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 [16] or MOSEK [17], can handle up to a few tens of targets. We show in our experiments that as the number of targets grows, the optimization becomes extremely inefficient. In some previous work, the quadratic function is converted into a linear one by adding additional equality/inequality constraints to the objective function [18]; 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 a 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 hundreds of people. Lastly, we propose a simple 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.

In summary, the paper makes the following important contributions. We are the first ones to formulate tracking in high-density crowds as online multiple object tracking with joint optimization. We propose a flexible binary quadratic programing solution which brings in five essential components for crowd tracking into one single formulation. Our formulation includes both target's individual information as well as contextual cues and models the interaction between targets using neighborhood motion, spatial proximity and grouping constrains. We show that the proposed objective function can be solved efficiently using the most recent version of modified Frank-Wolfe algorithm. Additionally we propose an effective speed up technique that further reduces the computational complexity without loss in the tracking accuracy. Finally, we improve state-of-art on nine challenging sequences of [11] and two new sequences of Galleria1 and Galleria2.

2 Related Work

Multiple target tracking is one of the fundamental problems in computer vision. Most prior work have focused on low and medium density crowd sequences [3], [4], [5], [6], [19], [20], [21], where the goal is to design a better data association technique. Authors in [19] formulate data association as maximum weight independent set. Many successful data association based trackers utilize network flow to formulate tracking [8], [9], [10]. The solution to network flow can be found efficiently using linear programing [10] or a dynamic programing [9]. Authors in [6], [21], [22] formulate data association as maximum clique problem. All of these methods assume that, the detections in each frame are already given. This requires having a good pre-trained object detector [23], [24] that works reasonably well.

When dealing with high-density crowds, the performance of pre-trained object detectors drop significantly. This is due to several factors. In crowded sequences, full bodies

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2687462, IEEE Transactions on Pattern Analysis and Machine Intelligence

of people are not visible and only their heads, faces or upper bodies are visible. For instance, [25] showed that the face or head detectors perform poorly in high density crowds. Instead of using a pre-trained object detector one can use an online object detector, which can be continuously updated. Due to this, there has been a recent interest in online tracking methods. In online tracking methods, pre-trained object detectors are not used and detection and data-association are solved at the same time. Online tracking methods have been used extensively in the context of single object tracking [15], [26], [27]. More recently, researcher have tried to extend these methods to track multiple people. A few example of this line of work are: [28], [29], [30], [31], [32] are a few examples. However, it is not easy to extend these methods to high-density crowd scenarios.

Target tracking in highly crowded scene is relatively a new area of research, and only a handful of papers have focused on this problem [11], [12], [13], [14]. The methods proposed in [11], [12], [13], [14] track each target separately by training an online tracker for each individual separately. Ali and Shah [12] proposed an algorithm which learns the prior information about the scene, called floor fields, that restrict the motion of each individual severely. This would cause failure in tracking when the crowd is dynamic, when there are anomalies or when camera moves. The series of papers by Kratz and Nishino [33], [34], [35] follow similar approach. They learn the motion patterns and then use them as prior information to improve the track of each individual. Rodriguez et al. [13] proposed a Correlated Topic Model to model crowd behavior at each location. Their model does not have the assumption of [12] and allows targets to select among different motion patterns at each location. However, their model needs to learn dynamics of the scene and is prone to the same problems. In their approach, words correspond to low level quantized motion features and topics correspond to crowd behaviors. The recent work of [11] addresses some of the problems with previous work. In their approach, no prior assumption about the scene is used, instead the effect of neighborhood motion is incorporated in a greedy manner to improve tracking. Although the effect of neighbors in tracking was used before in [36], its extension to high-density crowds was explored first in [11].

One major draw-back of previous crowd trackers such as [11], [12], [13], [14] 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 interaction of targets becomes cumbersome and finding an efficient optimization is quite challenging. When tracking one target at a time, we are limited to use information from that target only. This makes it impossible to model interaction between the targets. In order to overcome this limitation, previous work 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 [11]. Given the above issues, it is very 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 a 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.

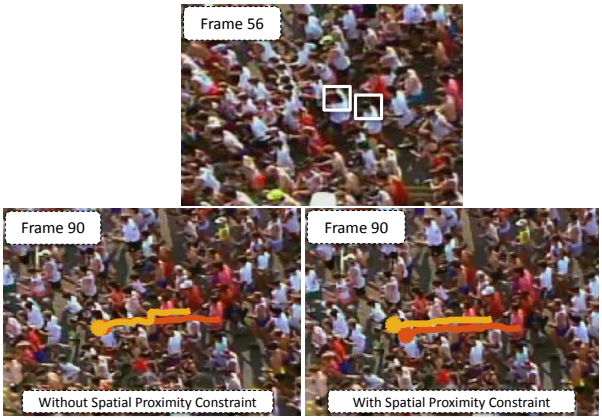


Fig. 2. This figure motivates the spatial proximity constraint used in our formulation. The top figure shows the two targets with very similar appearance. The bottom figures demonstrate the tracking results of our method, with and without proximity constraint. As can be seen when spatial proximity constraint is not used, the tracker gets confused and tracks of one target jump 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.

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 modified Frank-Wolfe algorithm which takes an advantage of *SWAP* steps for further speed up [37]. Quadratic programing has already been used for multi-target tracking [38], [39]. However, this is the first time that it is adapted in an optimization friendly framework for crowd tracking.

The Frank-Wolfe optimization algorithm was introduced in 1956 to solve the constraint quadratic programing, which has been recently revisited and used in many machine learning applications [37], [40], [41]. Authors in [41] used Frank-Wolfe with Away steps and proposed a faster optimization strategy for structure support vector machine. Allende et. al in [37] showed that Frank-Wolfe could be applied to solve the quadratic programing in support vector machine and achieve significant speed up for large scale problems compared to its competitive methods such as projected gradient descend. Moreover, authors in [42] adopted Frank-Wolfe with away steps to solve the quadratic objective for image/video co-localization. In our work, we show that, commercial softwares such as CPLEX [16] and MOSEK [17], 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 number of target, but also is faster than commercial software and other version of Frank Wolfe used in [40], [41], [42].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2687462, IEEE Transactions on Pattern Analysis and Machine Intelligence

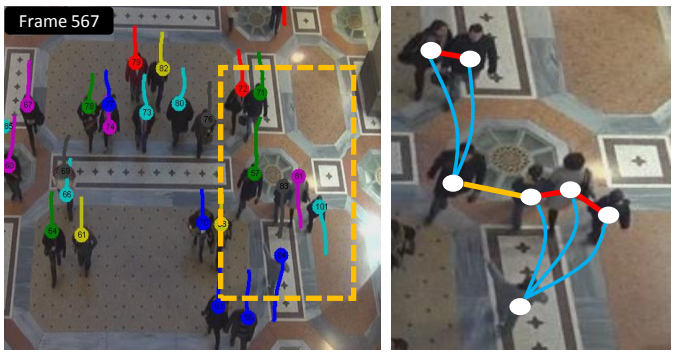


Fig. 3. This figure is a graph illustration of the contextual constraints used in our formulation. The figure on the left shows the tracks and the figure on the right shows the constraints between the targets in yellow box. Each target is a node in the graph and all targets are connected with an edge. 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 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.

The rest of the paper is organized as follow, in Section. 3 we present our framework. In Section 4, we describe our binary quadratic formulation and different terms we consider in our objective function. The Frank Wolfe algorithm used for the optimization is covered in Section. 5. We describe our speed up technique in Section. 6. In Section. 7 we present our quantitative and qualitative results and finally in Section 8 we conclude the paper.

3 Proposed Framework

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.

4 Objective Function

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

$$\underset{\mathbf{x}}{\text{minimize}} f(\mathbf{x}) = \overbrace{\mathbf{c}_a^T\mathbf{x}}^{appearance} + \overbrace{\zeta\,\mathbf{c}_m^T\mathbf{x}}^{targetmotion} + \overbrace{\eta\,\mathbf{c}_{nm}^T\mathbf{x}}^{neighbormotion} + \overbrace{\mathbf{x}^T\mathbf{C}_{sp}\mathbf{x}}^{spatialproximity} + \overbrace{\mathbf{x}^T\mathbf{C}_g\mathbf{x}}^{grouping}, \tag{1}$$

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.
In order to ensure the solution found by solving Equation 1 is a feasible tracking solution, we need to enforce the following constrains:

$$\sum_{i \in N_j} x_i^j = 1, \quad \{\forall i, j | 1 \leq i \leq k, 1 \leq j \leq n\} \tag{2}$$

$$x_i^j \in \{0, 1\}, \quad \{\forall i, j | 1 \leq i \leq k, 1 \leq j \leq n\}, \tag{3}$$

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 candidate locations for each target and n is the total number of targets to track. The constraint in Eq. 2 guarantees that exactly one location is selected for each target. The constraint in Eq. 3 ensures that each location is assigned to at most one target. These constraints along with the cost function in Equation 1 form our Binary Quadratic Programing formulation that needs to be solved at every frame. Each term in Equation. 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 [43], [44]. An example is shown in Figure 1. 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 [11], [12], [13], a template-based method employing Normalized Cross Correlation (NCC) was used to capture such information. In a recent study in [45], it is shown that discriminative based 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 [26], 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, similar to [15], we train a regressor for each target by minimizing the following objective function:

$$\underset{\mathbf{w}_i}{\text{minimize}} \sum_{j \in T_i} \left(\mathbf{w}_i \phi(x_i^j) - y_i^j\right)^2 + \lambda \left\|\mathbf{w}_i\right\|^2, \tag{4}$$

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 [46] that it provides 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 per row $\mathbf{z}_i$. Once the models are trained the appearance cost for each candidate location is found using the following equation:

$$c_{i,a}^j = -\mathbf{w}_i \phi(x_i^j). \tag{5}$$

In [15] it is shown that the solution to Eq. 4 and 5 can be found efficiently in Fourier domain. We follow the same approach to compute the models for each target, but instead of using gray scale images as used in [15], we employ the multi channel formulation of the above equation and use color features.

Motion plays an important role in the context of tracking pedestrians. 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 become 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. Thus using motion models that predict target location only based on its own behavior is not enough. 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 similarly looking moving targets. 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 [11] to predict the location of individuals at each time step.

The first term predicts the location of target using a linear velocity model shown below:

$$\mathbf{c}_{i,m} = \mathcal{N}(A\mathbf{s}_i^{t-1}, \mathbf{\Sigma}), \tag{6}$$

where $\mathbf{s}_i^t = [p_i, \dot{p}_i]$ is a vector that contains the location (p_i) and velocity ($\dot{p}_i$) 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 forces 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. An example is shown in Figure 4. During this unsupervised clustering we also take into account the distance between the members of the group. This approach is simple and effective and can be computed online without requiring much computations. Considering

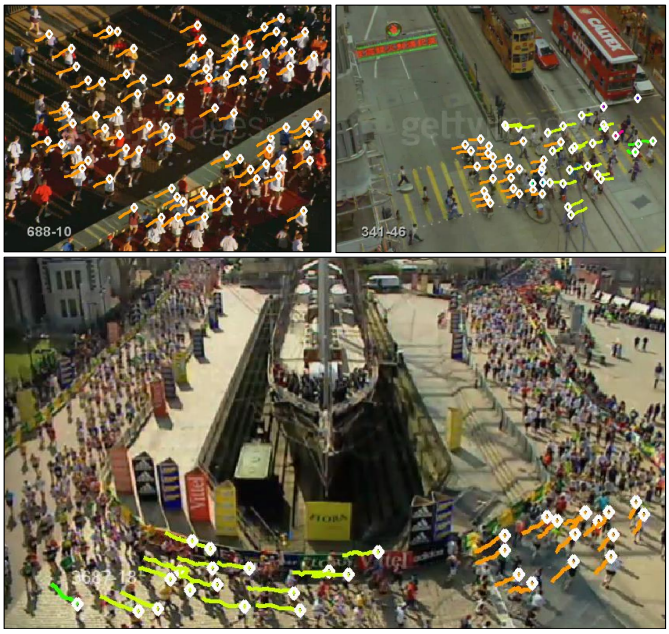


Fig. 4. An example of groups that move with coherent motion in several sequences (each color corresponds to one group). These groups are used to incorporate the neighborhood motion effect in our optimization.

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.

$$c_{i,nm} = \sum_{j \in N_i} w_j . \mathcal{N}(A\mathbf{s}_{ij}^{t-1}, \mathbf{\Sigma}), \tag{7}$$

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:

$$w_j = \frac{exp(-\|p_j - p_i\|_2^2)}{\sum_{k \in P_i} exp(-\|p_k - p_i\|_2^2)}. \tag{8}$$

Spatial Proximity Constraint is another important component in our formulation which has not been used before in crowd-tracking. The most commonly used constraint in almost all multi-target tracking methods is that each location should be assigned to only one target. This is imposed in our formulation using constraint in Eq. 3. 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, in every frame we sample candidates 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 may select two locations that are too close to each other). Additionally, when dealing with crowd sequences of aerial scenes, having detections that overlap is restricted. This is because most sequences

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2687462, IEEE Transactions on Pattern Analysis and Machine Intelligence

are captured using cameras facing down and targets do not occlude each other. Figure 2 illustrates an example where two targets with similar appearance are running next to each other and the tracker gets confused after a few frames. 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 downward and we have a 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 $\mathbf{C}_{sp} = I - D^{-1/2}SD^{-1/2}$ is the normalized Laplacian matrix [47]. 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:

$$S_{ij} = exp(\frac{-\|p_i - p_j\|_2^2}{2\sigma^2}), \tag{9}$$

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 having the same impact on our optimization.

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 [48], [49], [50].. However, the biggest challenge is: *How to incorporate grouping 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 work. 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 [47]. 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 groping information and encourages the selection of candidate locations, that keep the formation of targets within each group. Each entry of Γ is obtained using the equation below:

$$\Gamma_{ij} = exp(\frac{-(\|p_i - p_j\|_2 - e_{ij})}{2\sigma^2}), \tag{10}$$

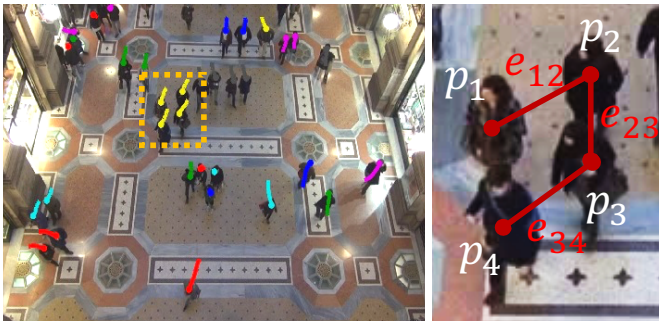


Fig. 5. An example of our minimum spanning tree group structure model. The figure on the left shows the groups found in frame 638 of Galleria1 sequence (nearby tracks with similar color correspond to one group). The figure on the right illustrates the model created for the group inside the yellow box. p_i denotes the location of the target and e_ij denotes the relative distance between the two targets i and j.

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. We update the grouping 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 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. 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. 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. 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.

5.1 Frank Wolfe Optimization

Given our convex quadratic function $f(\mathbf{x})$ in Eq. 1 and a set of convex constraints D (constraint in Equation 2 and relaxed version of the constraint defined in equation 3) ($x_i^j \in [0, 1]$), 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

Algorithm 1: Frank Wolfe

Data: $\mathbf{x}_0 \in D$.
Result: $\mathbf{z}$
Result: Initialization: $k = 0$, $\mathbf{z} = \mathbf{x}_0$
for $k = 0, 1...K$ **do**
  Compute $\mathbf{s}_k \leftarrow \underset{\mathbf{s}\in D}{\operatorname{argmin}} < \mathbf{s}, \bigtriangledown f(\mathbf{x}_k) >$,
  $d_{FW} = \mathbf{s}_k - \mathbf{x}_k$
  Line Search : $\lambda_{FW} = \underset{\lambda\in[0,1]}{\operatorname{argmin}} f(\mathbf{x}_k + \lambda(d_{FW}))$
  Update : $\mathbf{x}_{k+1} = (1 - \lambda_{FW})\mathbf{x}_k + (\lambda_{FW})\mathbf{s}_k$
Perform the rounding $\mathbf{z} \leftarrow rounding(\mathbf{x}_K)$
return $\mathbf{z}$

line search. In our problem, the linearized version of our objective function is given by [1]:

$$\begin{aligned} g(\mathbf{x}) &=< \mathbf{s}, \bigtriangledown f(\mathbf{x}) > \\ &= \mathbf{s}(\mathbf{C}_{sp}\mathbf{x} + \mathbf{C}_g\mathbf{x} + \mathbf{c}_a + \mathbf{c}_m + \mathbf{c}_{nm}), \end{aligned} \tag{11}$$

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:

$$\lambda_{FW} = \underset{\lambda\in[0,1]}{\operatorname{argmax}} f(\mathbf{x} + \lambda(\mathbf{s} - \mathbf{x})). \tag{12}$$

The optimization problem in Equation. 12 has a closed form solution that can be obtained by setting the derivative of Equation 12 with respect to λ equal to zero, and replacing the function f with its definition in Equation 1. This will lead to the following closed form solution:

$$\begin{aligned} \frac{\partial}{\partial\lambda} f(\mathbf{x} + \lambda(\mathbf{s} - \mathbf{x})) = \bigtriangledown f(\mathbf{x} + \lambda(\mathbf{s} - \mathbf{x}))^T(\mathbf{s} - \mathbf{x}) = 0, \\ \lambda_{FW} = \frac{\bigtriangledown f(\mathbf{x})^T(\mathbf{s} - \mathbf{x})}{(\mathbf{s} - \mathbf{x})(\mathbf{C}_{sp} + \mathbf{C}_g)(\mathbf{s} - \mathbf{x})}. \end{aligned} \tag{13}$$

One should note that the solution in Equation 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 optimization does not satisfy the discrete constraint of Eq. 3, thus requires rounding. In order to find the best binary solution one needs to solve the following optimization.

$$\underset{\mathbf{z}\in D}{\operatorname{argmin}} \left\|\mathbf{z} - \mathbf{x}_K\right\|_2, \tag{14}$$

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 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.

1. please note that for simplicity, we removed the index k in Equations 11, 12 and 13 , which shows the k^{th} iteration

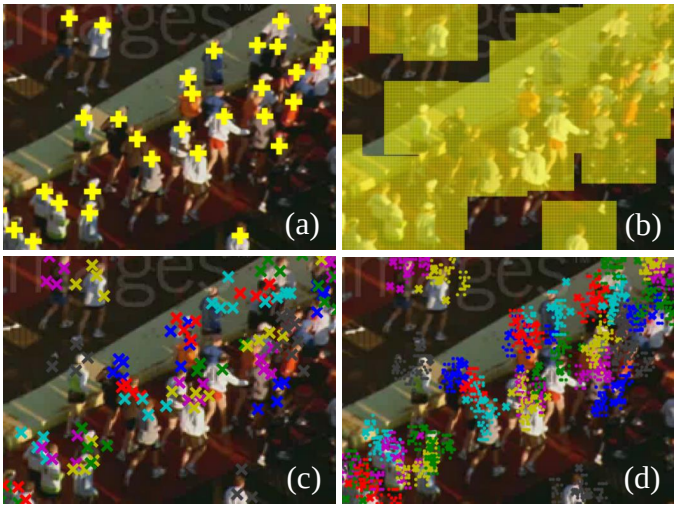


Fig. 6. A comparison of our speed-up sampling technique vs dense sampling of candidate locations. (a) shows the targets to be tracked. (b) illustrates the candidates sampled densely in the neighborhood of each target (each yellow dot represent one candidate location). (c) shows the selected extrema locations (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).

5.2 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.

$$\begin{aligned} \overbrace{\mathbf{x}_k + \lambda(\mathbf{s}_k - \mathbf{y}_k)}^{SWAP\ step} = \overbrace{\frac{1}{2}(\mathbf{x}_k + \lambda(\mathbf{s}_k - \mathbf{x}_k))}^{toward\ step} \\ + \overbrace{\frac{1}{2}(\mathbf{x}_k + \lambda(\mathbf{x}_k - \mathbf{y}_k))}^{away\ step}. \end{aligned} \tag{15}$$

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 [42]. 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.

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
     $\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_{swap*}(\mathbf{s}_k - \mathbf{y}_k)$
  else
    Perform the FW step
    $\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_{fw*}(\mathbf{s}_k - \mathbf{x}_k)$
Perform the rounding $\mathbf{z} \leftarrow rounding(\mathbf{x}_K)$
return $\mathbf{z}$

We present several experiments in Section 7 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 [51].

Original FW algorithm is expected to have a sub-linear convergence rate [52]. However the algorithm described in Alg.2 converges linearly to the optimal value of the objective function. Moreover it achieves a predetermined accuracy ε (primal-dual gap) in at most $\mathcal{O}(\frac{1}{\varepsilon})$ iterations. We refer the reader to [37] for more in depth convergence analysis of the algorithm in Alg. 2.

6 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 6(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 which are assigned low scores due to pose changes or occlusion. Moreover, keeping only the samples with high confidence values in the linear terms (Figure 6(c)) in Equation 1, limits the effect of our quadratic terms that capture the spatial proximity as well as grouping.

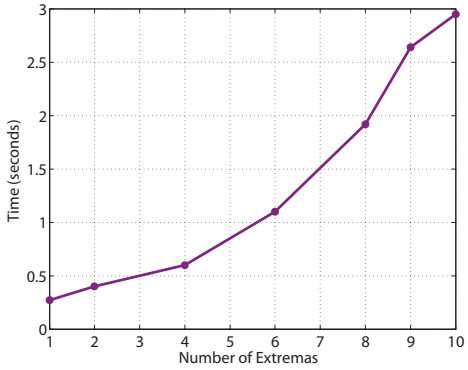


Fig. 7. This figure illustrates the run-time vs the number of extremas (m).

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 1 (m is set to three). An example is shown in Figure 6(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 6(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 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 6. This procedure reduces the number of candidate more than an order of magnitude which results in significant speed up. Furthermore, in Figure 7 we show the effect of m in the speed-up. As can be seen when the number of candidates are dropped 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 $m = 3$.

7 Experiments

We perform exhaustive experiments on nine high density crowd sequences of [11] and two new sequences with medium crowd density. These sequences include different scenarios and challenges. All sequences are taken using cameras facing down. The parameters ζ and η are set to 0.3 and 0.2, which we found through cross validation. The search region for each target is set to twice the size of the target for all sequences.

High-density Crowd Sequences: First frames of the high density crowd sequences of [11] are shown in Figure 10. 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

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2687462, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 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 [11]. On average we improve the best previous tracker by 3.1% in nine sequences.

	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%	66%
PNMC	80%	100%	92%	94%	77%	94%	67%	92%	63%
Proposed	**86%**	**99%**	**96%**	**97%**	**78%**	**96%**	67%	90%	78%

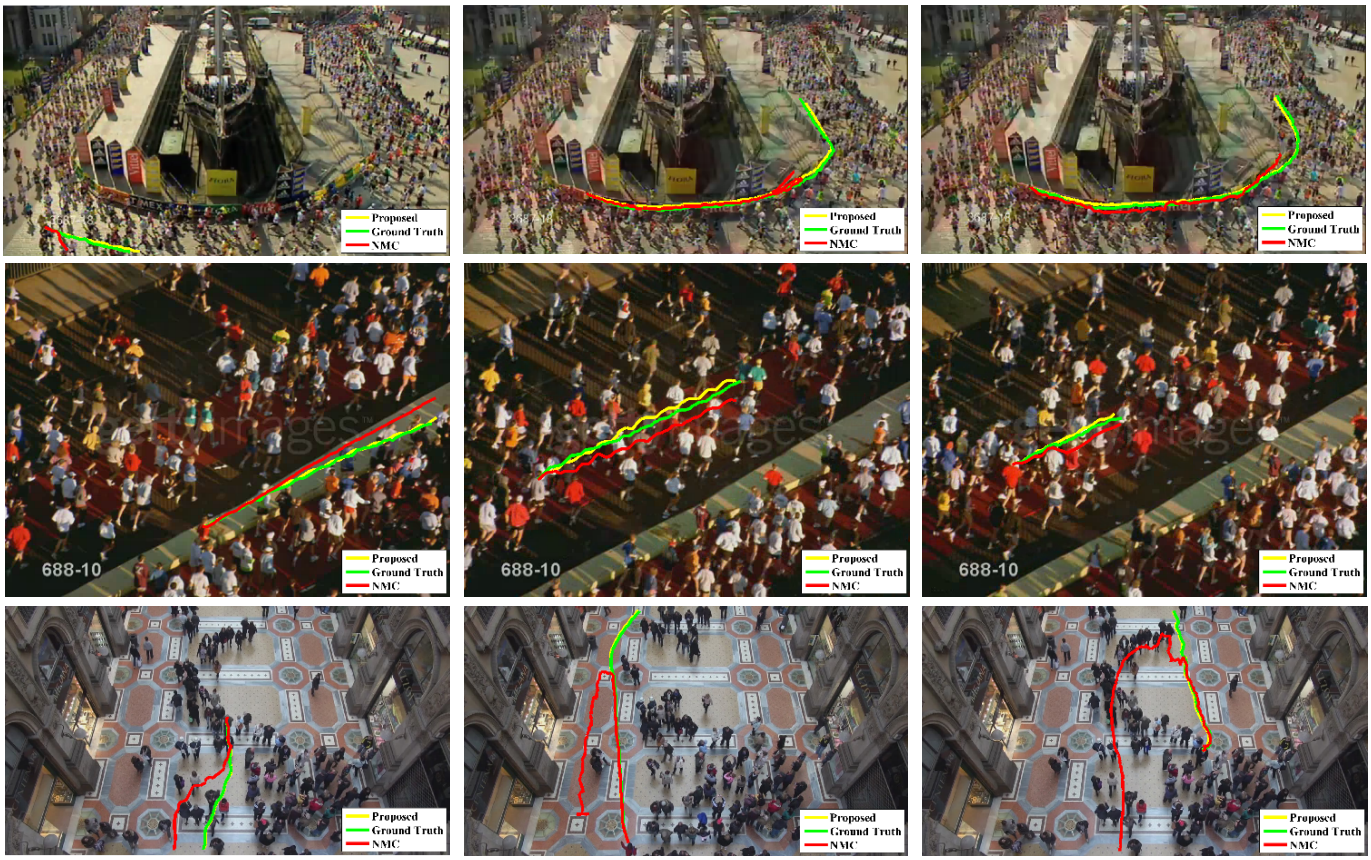


Fig. 8. This figure shows tracks obtained using our method for a few challenging targets. We also compare our results with the one in [11]. For some sequences the proposed method gives tracks very close to the ground truth tracks. Since green is superimposed on yellow, due to that tracks in yellow may not be that visible.

TABLE 2
Quantitative result of our method in terms of tracking accuracy when pixel threshold is set to 5 on all sequences.

	Seq1	Seq2	Seq3	Seq 4	Seq 5	Seq 6	Seq 7	Seq 8	Seq 9	Galleria1	Galleria2
Proposed	72.8%	91.4%	87.1%	86.7%	52.9%	81.9%	46.3%	67.6%	71.8%	81.6%	86.0%

crowd at 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 are shown in Table. 1. The number of individuals annotated in each sequences ranges from 57 to 747. For a fair comparison we used the same groundtruth as [11]. The target tracks are initialized using their locations in the first frame. Most of the targets that appear in these sequences are annotated. However in couple of sequences, some targets are not annotated, thus not tracked.

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, Galleria1 and Galleria2. These two sequences are recorded from Galleria mall in Milan and have been used in the

TABLE 3
Quantitative comparison of **Tracking Accuracy** using different terms in cost function of Equation. 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 represents 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)	78.13%	98.54%	85.54%	91.18%	65.34%	93.49%	66.81%	88.76	63.72%
B(KCF)	81.3%	99.03%	90.63%	92.68%	65.7%	92.95%	67.22%	89.2	64.68%
B+Mo(NCC)	76.9%	98.83%	87.64%	92.68%	72.18%	92.95%	73.80%	91.74%	64.08%
B+Mo(KCF)	82.63%	98.54%	90.63%	92.88%	73.33%	93.65%	67.22%	88.7%	69.80%
B+Mo+SP(NCC)	77.29%	99.03%	93.70%	92.28%	76.08%	94.28%	74.37%	93.00%	67.73%
B+Mo+SP(KCF)	84.24%	98.62%	93.40%	93.12%	76.28%	94.58%	71.24%	90.35%	72.48%
B+Mo+Gr(NCC)	79.74%	99.03%	93.13%	93.73%	77.08%	94.17%	72.92%	93.00%	64.07%
B+Mo+Gr(KCF)	85.22%	98.62%	93.56%	93.2%	76.18%	94.43%	67.23%	90.15%	71.12%
B+Mo+Gr+NMo(NCC)	79.86%	99.03%	93.13%	94.1%	77.08%	94.64%	72.72%	92.76%	66.07%
B+Mo+Gr+NMo(KCF)	85.42%	98.62%	94.56%	93.65%	76.18%	95.13%	67.22%	89.86%	73.42%
B+Mo+SP+Gr+NMo(NCC)	80.32%	99.03%	93.40%	95.18%	76.67%	94.65%	74.0%	92.13%	69.80%
B+Mo+SP+Gr+NMo(KCF)	**86.08%**	**98.62%**	**96.41%**	**96.84%**	**77.73%**	**95.75%**	67.22%	90.35%	**77.88%**

vision community for other tasks such as group detection [53]. 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 [54], we plan to release the videos along with their annotations. Galleria1 contains 200 targets and Galleria2 contains 215 targets. Some example frames of these two sequences along with qualitative results are shown in Figure 8.

TABLE 4
Quantitative Comparison, in terms of tracking accuracy, of our method with the tracker of [11] when pixel threshold is set to 15 on two new sequences of Galleria1 and Galleria2. On average we improve PNMC tracker of [11] by 4.5% on these two sequences.

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

7.1 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) [12] : 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, restricts the motion of each individual in a scene severely.
- Correlated Topic Model (CTM) [13]: CTM tracker utilizes a Correlated Topic Model to model crowd behavior at each location. Their model does not have the assumption of [12], 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) [55]: 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 (PNMC) [11]: 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 use the manual annotation of initial target locations provided with the dataset. The template size is the same as the one used in [11]. Target appearance model in our discriminative online tracker is updated similar to [15]. Our NCC-based tracker updates target appearance model every 10 frame. In order to alleviate drifting, we do not update target appearance when the confidence of the appearance tracker is less than the threshold 0.6.

We chose pixel-threshold = 15 in Tables 1 and 4 in order to be consistent with the results reported in [11]. However, those numbers are much lower if a lower pixel threshold is selected. In Figure 10 we show the tracking accuracy for different pixel thresholds. Additionally we provide exact numbers for when the threshold is set to 5 instead of 15 in Table 2. [2]

7.2 Quantitative Results

We followed the same metrics as the one in [11] to quantitatively compare our method with other methods. The tracking accuracy for different pixel-error threshold is shown in Figure 10 for the 9 high-density crowd sequences of [11]. The results on Galleria1 and Galleria2 sequences are also shown in Figure 9. Similar to [11], we also provide the accuracy for when the pixel threshold is set to 15 in Tables 1 and 4. 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

2. pixel threshold 5 is roughly equal to half the target template size which resembles the common 50% overlap threshold in most detection/tracking papers. We encourage the reader to use this table for quantitative comparison.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2687462, IEEE Transactions on Pattern Analysis and Machine Intelligence

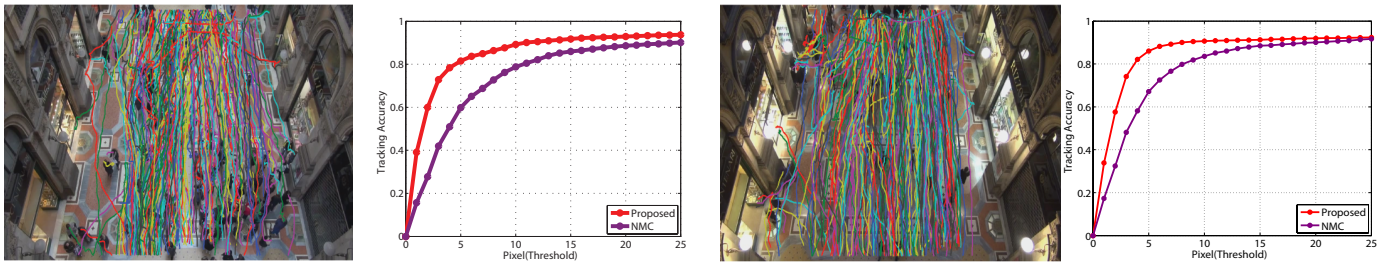


Fig. 9. This figure shows tracks (shown on the left) and quantitative results of our method (shown on the right) . We compare our method with PNMC [11] which achieves the best results on the 9 high-density crowd sequences.

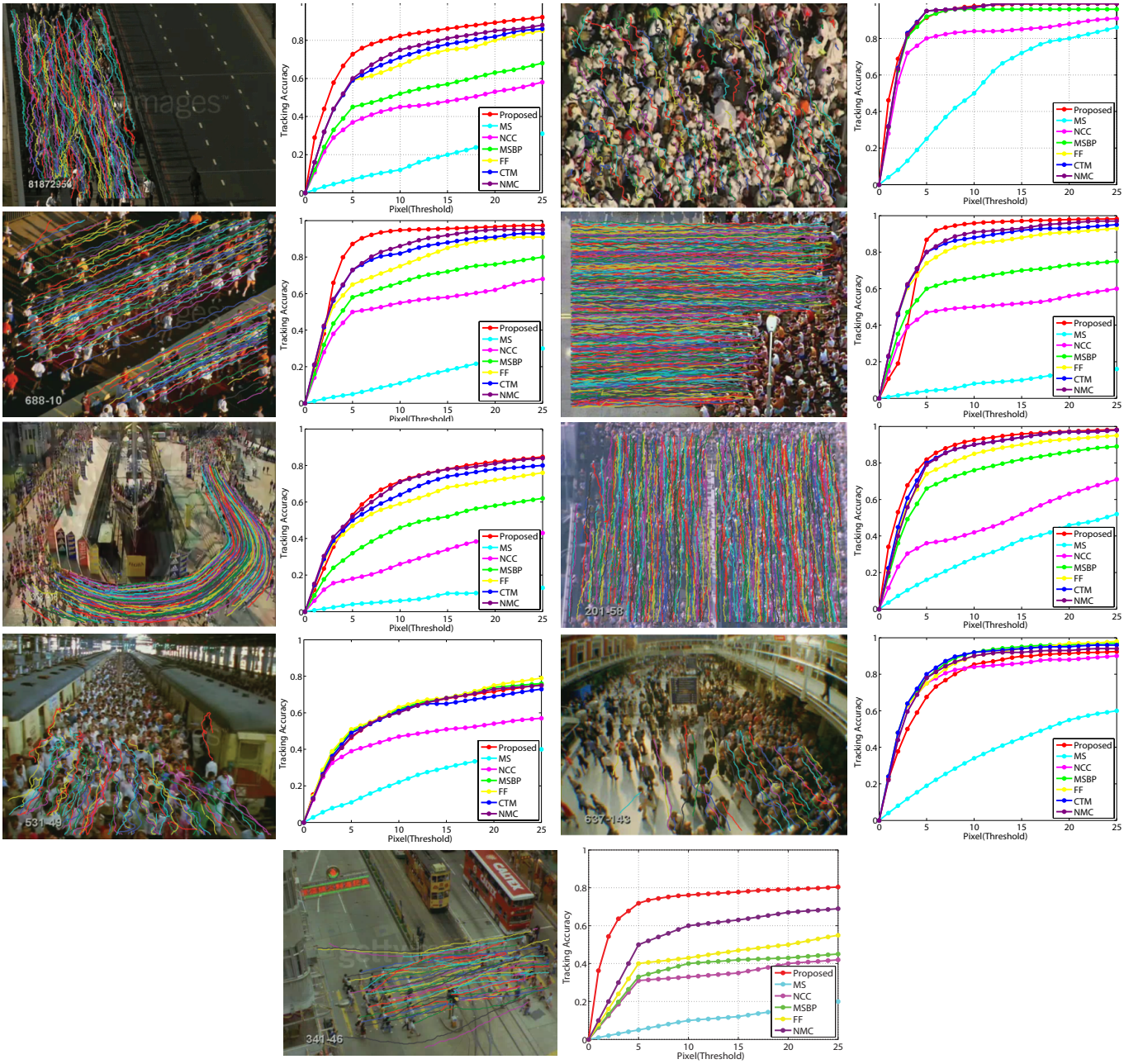


Fig. 10. This figure shows tracks (show on the left) and quantitative results of our method (shown on the right) with competitive approaches of FF [12], CTM [13], MSBP [55], PNMC [11] and MS [56]. For each sequence we show the qualitative results of all tracks and on the right we show the quantitative comparison. Each plot shows the tracking accuracy vs pixel error.

that [12] is not designed to handle. The heuristic such as instantaneous flow proposed in [11] are not suitable for tracking in these scenarios. In Figure 14, we show some of the failure cases of our method. For comparison with [11] on the new sequences, we used the implementation provided by the author.

7.3 Contribution of the each Component

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 are shown in Table. 3. We tried a combination of different terms. Our baseline (B(NCC)) is just a template-based tracker without the other terms in Eq. 1. Comparing the baseline of [11] that also uses a NCC based tracker with the equivalent of ours (B+Mo(NCC)), one can see that formulating the problem as multi-target tracking with joint optimization of target tracks can improve the performance significantly. Our baseline is 30% higher that the one of [11].

Next, we add different components of the objective function in Equation 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 grouping 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 3, we observe that the improvement of different terms depends on different factors in the scene including the density of crowd. For example in sequences 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 effectiveness. 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 other will help improve the results.

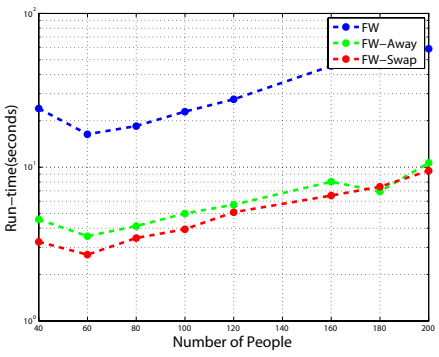


Fig. 11. The figure shows the run-time comparison of Frank-Wolfe(FW), Frank-Wolfe with Away steps (FW-Away) and Frank-Wolfe with Swap steps (FW-Swap) on one of our sequences. It is clear that the Away step technique improves the run-time of FW significantly. However, using the Swap step we can further speed-up the optimization.

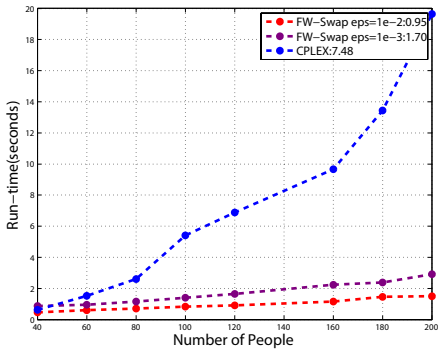


Fig. 12. This figure illustrates the run-time comparison of the proposed method for different values of duality gaps ε. We also compare the run-time of FW-Swap with ILOG CPLEX. It is evident that the FW method scales well to much larger problem size and requires far less computations.

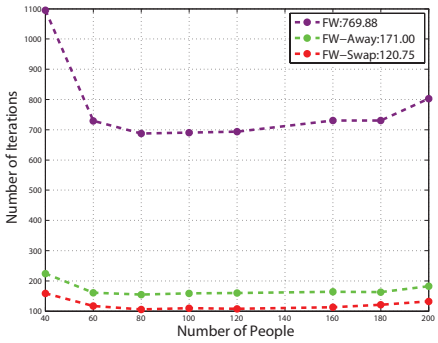


Fig. 13. This figures illustrates the number of iterations taken by the optimization to converge for original Frank-Wolfe (FW), Frank-Wolfe with Away steps (FW-Away) and Frank-Wolfe with Swap steps (FW-Swap). The value of ε is set to 0.0001.

7.4 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 [57] and widely used version of FW with Away steps [42]. The results, tested on one of our sequences, are shown in Figure 11. 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 [16], 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. 12. 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 13. All the experiments are performed on a quad-core 3.0 GHz machine.

Fig. 14. 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.

8 Conclusion

In this paper, we formulated the multi-target tracking in high density crowded scenes 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. We hope that our paper opens up the room for other researchers to further study this important yet challenging problem.

Acknowledgment

This work was made possible by NPRP grant number NPRP 7-1711-1-312 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

[1] T. E. Board, "Bombs at the marathon. the new york times [online]." 2013. [Online]. Available: http://www.nytimes.com/2013/04/16/opinion/bombs-at-the-boston-marathon.html?ref=bostonmarathon

[2] B. Hubbard, "Hajj stampede near mecca leave over 700 dead." 2015. [Online]. Available: http://www.nytimes.com/2013/04/16/opinion/bombs-at-the-boston-marathon.html?ref=bostonmarathon

[3] K. C. A. Kumar and C. D. Vleeschouwer, "Discriminative label propagation for multi-object tracking with sporadic appearance features," in *ICCV*, 2013.

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

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

[6] M. S. Amir Roshan Zamir, Afshin Dehghan, "Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs," in *ECCV*, 2012.

[7] H. I. et al., "Multi-Source Multi-Scale Counting in Extremely Dense Crowd Images,," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.

[8] Z. Li, L. Yuan, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

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

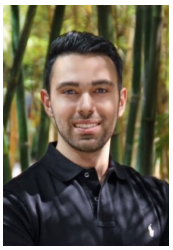
[10] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, "Multiple Object Tracking Using K-Shortest Paths Optimization," vol. 33, no. 9, 2011, pp. 1806–1819.

[11] H. Idrees, N. Warner, and M. Shah, "Tracking in dense crowds using prominence and neighborhood motion concurrence," vol. 32, no. 1, 2014, pp. 14 – 26.

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

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

[14] A. Bera, N. Galoppo, D. Sharlet, A. Lake, and D. Manocha, "Adapt: Real-time adaptive pedestrian tracking for crowded scenes," 2014.
[15] H. J. F., R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters." in *PAMI*, 2015.
[16] "Ibm ilog cplex optimizer, www-01.ibm.com/software/integration/optimization/cplex-optimizer."
[17] 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.
[18] 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.
[19] W. Brendel, M. Amer, and S. Todorovic, "Multiobject tracking as maximumweight independent set," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.
[20] R. Collins, "Multitarget data association with higher-order motion models." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
[21] A. Dehghan, S. M. Assari, and M. Shah, "Gmmcp-tracker:globally optimal generalized maximum multi clique problem for multiple object tracking," 2015.
[22] E. Ristani and C. Tomasi, "Tracking multiple people online and in real time," in *ACCV*. Springer, 2014.
[23] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," vol. 32, no. 9, 2010, pp. 1627–1645.
[24] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *ICCV*, 2005.
[25] e. a. Idrees, Haroon, "Multi-source multi-scale counting in extremely dense crowd images," 2013.
[26] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *ICCV*, 2011.
[27] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-Learning-Detection," in *PAMI*, 2011.
[28] K. Cheng-Hao, C. Huang, and R. Nevatia, "Multi-target tracking by on-line learned discriminative appearance models." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.
[29] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool., "Robust tracking-by-detection using a detector confidence particle filter." in *ICCV.*, 2009.
[30] A. Dehghan, Y. Tian, P. H. Torr, and M. Shah, "Target identity-aware network flow for online multiple target tracking." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.
[31] L. Zhang and L. van der Maaten, "Structure Preserving Object Tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2013.
[32] S.-I. Yu, W. Z. Deyu Meng, , and A. Hauptmann., "The Solution Path Algorithm for Identity-Aware Multi-Object Tracking." in *IEEE Conference on Computer Vision and Pattern Recognition.*, 2016.
[33] L. Kratz and K. Nishino, "Tracking with local spatio-temporal motion patterns in extremely crowded scenes,," 2010.
[34] L. Kratz and K. Nishino., "Going with the flow: pedestrian efficiency in crowded scenes," in *European Conference on Computer Vision*, 2012.
[35] L. Kratz and K. Nishino, "Tracking pedestrians using local spatio-temporal motion patterns in extremely crowded scenes,," 2012.
[36] 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.
[37] 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.
[38] B. Leibe, K. Schindler, and L. V. Gool, "Coupled detection and trajectory estimation for multi-object tracking." in *ICCV.*, 2007.
[39] A. Andriyenko and K. Schindler., "Globally optimal multi-target tracking on a hexagonal lattice." in *ECCV.*, 2010.
[40] M. Jaggi, "Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization," in *ICML*, 2013.
[41] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher, "Block-Coordinate Frank-Wolfe Optimization for Structural SVMs ," in *ICML*, 2013.
[42] A. Joulin, K. Tang, and L. Fei-Fei, "Efficient Image and Video Co-localization with Frank-Wolfe Algorithm," in *ECCV*, 2014.
[43] J. Ferryman and A. Shahrokni, "Pets2009: Dataset and challenge." in *In: Performance Evaluation of Tracking and Surveillance, International Workshop on.*, 2009.
[44] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, "Part-based multiple-person tracking with partial occlusion handling." in *IEEE Conference on Computer Vision and Pattern Recognition.*, 2012.
[45] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual Tracking: an Experimental Survey," in *PAMI*, 2013.
[46] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured Output Tracking with Kernels," in *ICCV*, 2011.
[47] J. Shi and J. Malik, "Normalized cuts and image segmentation,," 2000.
[48] e. a. Chen, Xiaojing, "An online learned elementary grouping model for multi-target tracking,," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.
[49] Z. Qin and C. R. Shelton., "Improving multi-target tracking via social grouping." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.
[50] S. Pellegrini, A. Ess, and L. V. Gool., "Improving data association by joint modeling of pedestrian trajectories and groupings." in *ECCV*, 2010.
[51] S. Lacoste-Julien and M. Jaggi, "On the global linear convergence of Frank-Wolfe optimization variants." in *Advances in Neural Information Processing Systems*, 2015.
[52] J. Guelat and P. Marcotte, "Some comments on Wolfes away step," in *Mathematical Programming*, 1986.
[53] F. Solera, S. Calderara, and R. Cucchiara, "Socially Constrained Structural Learning for Groups Detection in Crowd." in *PAMI.*, 2015.
[54] "http://www.csai.disco.unimib.it/csai/space/start/."
[55] 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.
[56] R. V. Dorin Comaniciu and P. Meer., "Real-time tracking of non-rigid objects using mean shift," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2000.
[57] M. Frank and P. Wolfe, "An algorithm for quadratic programming," in *Naval Research Logistics Quarterly*, 1956.



Afshin Dehghan received his PhD degree in computer science from the University of Central Florida (UCF) in 2016. He has published several papers in conferences and journals such as CVPR, ECCV, ACM Multimedia, and IEEE Transactions on Pattern Analysis and Machine Intelligence. He was a program committee member of the ACM MM. His research interests include multi-target tracking, deep learning, face/facial attribute recognition, complex event recognition, mathematical optimization, and graph theory. He received UCF's computer science doctoral student of the year award in 2014-2015.

Mubarak Shah , the trustee chair professor of computer science, is the founding director of the Center for Research in Computer Vision at the University of Central Florida (UCF). He is an editor of an international book series on video computing, was editor-in-chief of Machine Vision and Applications journal, and an associate editor of ACM Computing Surveys journal. His research interests include video surveillance, visual tracking, human activity recognition, visual analysis of crowded scenes, video registration, UAV video analysis, and so on. He is a fellow of the IEEE, AAAS, IAPR, and SPIE.