arXiv:1705.08314v3 [cs.CV] 8 Sep 2017

A Novel Multi-Detector Fusion Framework for Multi-Object Tracking

R. Henschel[1] L. Leal-Taixé[2] D. Cremers[2] B. Rosenhahn[1]
[1]Leibniz University Hannover [2]Technical University Munich

Abstract

In order to track all persons in a scene, the tracking-by-detection paradigm has proven to be a very effective approach. Yet, relying solely on a single detector is also a major limitation, as useful image information might be ignored. This work demonstrates how to incorporate several detectors into a tracking system, using a novel multi-object tracking formulation. We cast tracking as a weighted graph labeling problem, resulting in a binary quadratic program. As such problems are NP-hard, the solution can only be approximated. Based on the Frank-Wolfe algorithm, we present a new solver that is crucial to handle such difficult problems. As a result, the tracker can take information from many frames and different detectors holistically into account. When applied with head and full-body detections, the fusion helps to recover heavily occluded persons and to reduce false positives. Evaluation on pedestrian tracking is provided for multiple scenarios, showing superior results over single detector tracking and standard QP-solvers. Finally, our tracker performs state-of-the-art on the MOT16 benchmark and is the winner of the MOT17 challenge[1].

1. Introduction

Multiple object tracking, and in particular people tracking, is one of the key problems in computer vision with potential impact for many applications such as video surveillance or crowd analysis [1]. A common approach to generate the trajectories of multiple people is *tracking-by-detection*: first a person detector is applied to each individual frame to find the putative locations of people. Then, these hypotheses are linked across frames to form trajectories. By building on the advances in person detection over the last decade, tracking-by-detection has been very successful [13, 14, 33, 47]. However, the dependence on detection results, typically bounding boxes, is also a major limitation. A lot of potentially useful information is lost during the non-maxima suppression. A tracker typically does not use direct image data, except in the form of appearance models in order to discriminate different people. Recently, a number of approaches have proposed to step away from the standard paradigm [12, 21, 37] and instead tackle the tracking problem more directly, by adding other image features aside from the full-body detections, helping especially to recover partly occluded pedestrians missed by the detector.

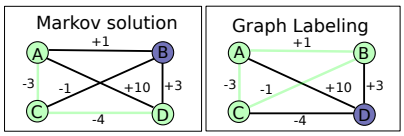

Figure 1. A node represents a detection and any two detections are connected. We draw here only the edge weights, without the costs for each node. The more negative the weight, the more likely the incident nodes belong to one person. **Left**: The result of a tracker that using the 1st order Markov chain assumption, thus finding the best paths (indicated by the colored nodes). It ignores that node A and node D very likley belong to different persons. **Right**: We search for the labeling that minimizes the costs between all equally labeled nodes. Our labeling solution results in long-term consistent trajectories.

In this paper, we present a framework, which performs offline multiple object tracking using multiple detectors. Conceptually, we formulate the tracking task as a graph labeling problem, such that all equally labeled detections belong to a person and build the trajectories. Existing trackers often search for the best paths in order to compute the trajectories [52], especially due to computational reasons, which provides consistency only from a frame to the previous frame. In contrast, our tracking formulation ensures long-term temporal consistency by taking *all* equally labeled detections into account (see Fig. 1). Due to our powerful solver, we are able to optimize globally, that is directly on the input detections, without the need of error-prone tracklets.

We compute the best labeling by solving a Binary Quadratic Problem (BQP). A straightforward approach to solve that BQP would thus be to optimize an equivalent Binary Linear Program (BLP), using branch&bound. However, due to the high dimensionality of the problem, such a BLP is computationally expensive and memory demanding.

[1]See https://motchallenge.net/MOT17_results_2017_07_26.html

We propose to use the Frank-Wolfe algorithm (FW) to solve the relaxation of the BQP and discretize afterwards. By using a standard implementation of FW, the result is often far away from the binary optimal solution. Therefore, we propose several crucial improvements that lead in practice to a much better solution, thus resulting in an improvement of the tracking performance. At the same time, the proposed algorithm is much faster than the standard branch&bound approach. We analyze the impact of our modifications and demonstrate that the proposed approach delivers solutions close the global optimum Finally, an analysis on the effect of the fusion of head detections with full-body detections shows that the best tracking accuracy is obtained by using both input sources. The fusion helps especially to remove false positive full-body detections that are not consistent with the head detections and to recover heavily occluded persons (see Fig. 2).

1.1. Contributions

To summarize, our contribution is three-fold:

- We propose a novel multi-detector multi-object tracking system, which solves a graph labeling problem and is represented by a BQP with very few constraints.
- We propose a new solver that significantly improves over standard BQP solvers, when applied to our discrete optimization problem.
- We present detailed evaluations on the improvements due to our solver as well as the detector fusion. Our framework sets a new state-of-the-art in tracking.

1.2. Related Work

In this section, we point out related methods and how they differ from our approach.

Optimization in tracking. Tracking-by-detection has become the standard paradigm for multi-object tracking. It splits the problem into two steps: object detection and data association. In crowded environments, where occlusions are common, even state-of-the-art detectors [16, 18, 41, 50] are prone to false alarms and missed detections. The goal of the data association step is then to fill in the gaps between detections and filter out false positives. In order to do this robustly, data association is mostly performed for all frames and all trajectories simultaneously. This is usually done in discrete space, using graph based methods [7, 14, 25], or BLPs [3, 26, 40, 52]. Due to the computational challenge, tracking methods that need to solve a BQP have been rare so far, despite many advanced tracking models are naturally expressed as a BQP. The augmentation of the standard network-flow tracking formulation by one additional feature [10] is one example. To solve the BQP in [10], the problem is rewritten as an equivalent BLP. As we show in our experiments, this simple trick is not applicable to our more demanding model, as we revoke the 1st order Markov chain assumption, which results in a larger problem, that delivers long-term consistent trajectories. Due to our proposed solver, our huge BQP can still be solved fast and accurate. Among the methods that employ a BQP, the Frank-Wolfe algorithm is often used. Just recently, it has been applied to online tracking [15]. While this method shows good performance, we propose a hierarchical solving scheme that can be easily integrated into their formulation, thereby further improving their result. Furthermore, during the Frank-Wolfe algorithm, the step size for an iterate update has to be computed. We derive an optimal, algebraic computation, that is cheap to compute and improves over existing methods [2, 15, 31]. Note that our improvements may be applied to methods of other fields in computer vision as well, such as person re-identification [2], co-localization [27] or object segmentation [44].

Data association models. Recently, multiple people tracking has seen improvement by revoking the 1st order Markov chain model assumption [52]. Instead, consistency within all links of a trajectory (within a batch) is ensured [14, 37, 46, 47, 51]. In [51] trajectories are created iteratively, computing one best clique corresponding to exactly one person and then removing the corresponding detections from the loop. This concept has been extended in [14] to obtain the trajectories in a global manner, for all persons at the same time. To keep the approach computationally feasible, the authors follow a hierarchical approach, connecting heuristically computed initial tracklets via the clique search. Other methods following the same consistency goals use a huge set of cycle constraints [14, 46, 47] in a BLP, that has exponential growth. Conceptually close to our tracking formulation is the multi-label CRF tracker [37], that, due to the computational issues, works with initial tracklets. Contrary to all these methods, our quadratic cost function automatically ensures the consistency within equally labeled (feature) nodes. Our model needs only one constraint per detection and optimizes directly on the input, without the need of potentially erroneous initial tracklets.

Incorporating different features. Limiting the input of the tracker to a single detector has clearly several drawbacks, since much of the information of the image is not taken into account, potentially ignoring semi-occluded objects. In recent literature, several works have started incorporating different image features for the task of multi-target tracking. Few works use supervoxels as input for tracking, obtaining as a byproduct a silhouette of the pedestrian. In [12], supervoxels are labeled according to target identities with greedy propagation, starting from manually initialized segmentation masks. In [37], supervoxel labeling is formulated as CRF inference, where the targets are modeled as volumetric

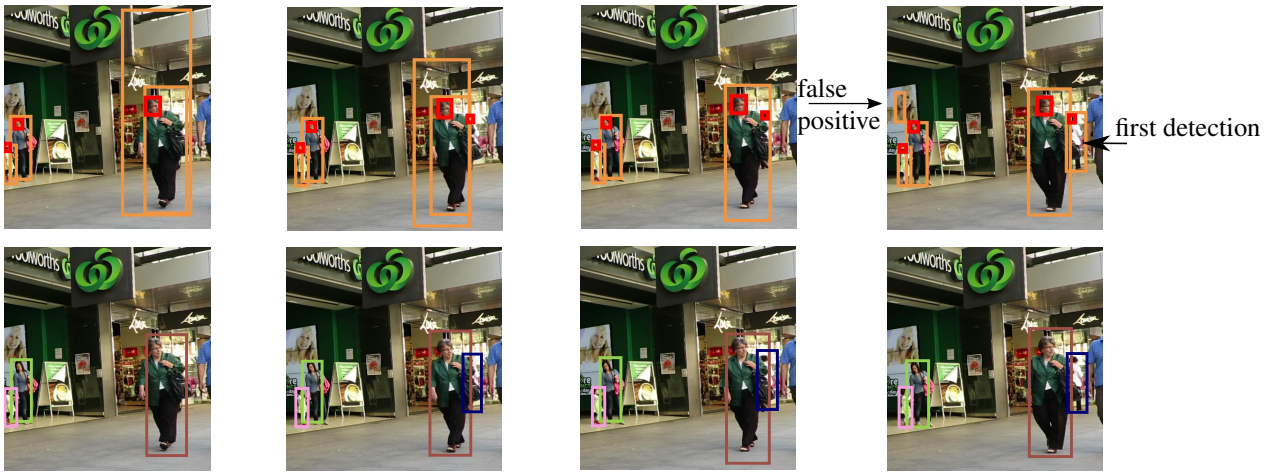

Figure 2. MOT16-09 sequence (from left to right: frame 8,10,12,15). **Top row**: Body detections (orange) and head detections (red). The body detector misses the marked person until frame 15. **Bottom row**: The result by our tracker. It removes the false positive that has no head detection and recovers the heavily occluded person due to the presence of head detections.

'tubes' through the sequence, with the help of initial detection and supervoxel tracklets. In [20], multi-target tracking is tackled by clustering dense feature tracks [8] and further combined with detection-based tracklets in a two-step approach in [21].

In [10], a BQP fuses head and body detections to track pedestrians, modeling non-maxima suppression as well as overlap consistency between features. In contrast to our model, only co-occurrences of active features are considered, while we directly model the grouping of features to different persons, allowing to ensure consistency within each cluster over long time periods. Also in the extension [44] to motion segmentation using superpixels, the per-person consistency is not considered.

2. Multi-Detector Multi-Target Tracking

In this section, we describe the data association that couples multiple detectors and ensures long-term temporal consistency. Head detections are very reliable as they are barely prone to pose variations or occlusions. Hence, we analyze in this paper the coupling of full-body detections with head detections. This is especially useful in crowded scenarios: Fig. 2 shows a heavily occluded pedestrian. While the full-body detector misses that person, its head is still visible so that our tracker localizes that pedestrian. Evaluations in Sect. 4 show that by using our formulation the tracker benefits most from both detectors.

We cast the fusion of the input sources and the trajectory computation as finding the best labeling of the detections to the different persons, using a compact BQP. In order to solve that BQP, we compute the relaxed solution using the Frank-Wolfe [22] solver, which is especially well suited for quadratic problems with linear constraints, as each iteration step involves solving a computationally efficient linear optimization problem. Then, we obtain the binary solution by an efficient rounding step.

When applied to a non-convex problem, like our model, the Frank-Wolfe algorithm delivers only a local optimum [29]. Hence, simply applying the standard algorithm will result in a solution that is far away from the global optimum. We thus focus on enhancing the solution of Frank-Wolfe by: (i) regularizing the cost function, (ii) computing the optimal step size within the solver's algorithm algebraically and (iii) introducing a hierarchical solving scheme that enhances the solution produced by the Frank-Wolfe algorithm.

Our regularizer prevents the Frank-Wolfe algorithm from falling to quickly into a local optimum. The hierarchical solving scheme gains the improvement by revoking or connecting clusters of the discretized solution, while having the guarantee of operating optimally. The presented approach is not specific to the Frank-Wolfe solver. It can be applied after any approximating algorithm. It further allows to correct errors introduced by the initial solver.

Experiments in Sect. 4 show that our proposed solver provides good solutions close to the estimated bound, while being considerably faster than the commercial solver Gurobi [23], which uses the branch-and-bound/cut algorithm [32, 38] to find the globally optimal solution.

2.1. Joint Data Association

We formulate the tracking problem as finding a labeling function $L : V \rightarrow \{0, 1, \ldots, K\}$, on the input detections V. Ideally, the labeling L assigns all detections belonging to a person to a unique ID $k > 0$, whereas false positive detections are assigned to the 0-label and removed. Thus, all equally labeled nodes with label > 0 form a cluster in our graph and are used to create the trajectory of the person. In particular, our formulation ensures that all equally labeled

nodes are consistent, within frames and between frames (see Fig. 1). We consider d sets of detections $D_1, \ldots, D_d$, where D_i contains all detections generated by the i-th detector. The set V is the union of all detections, with some ordering $V = \{1, \ldots, n\}$, for n detections. We build an undirected complete graph $G = (V, E)$ on the vertex set V. Each node represents an input detection and an edge encodes a possible linking of two detections to the same person. In our implementation, we consider $d = 2$ sources, namely head and full-body detections.

The nodes of G have weights $c_v \in \mathbb{R}$ reflecting the confidence of a detection $v \in V$, and the edges have weights $q_{u,v}, (\{u, v\} \in E)$, reflecting how likely two detections belong to a person. For a node v and label $k \in [K] \coloneqq \{1, \ldots, K\}$, the decision variable $x_v^k \in \{0, 1\}$ signals whether detection v belongs to ID k. All decision variables are stacked in a vector $\mathbf{x} \in [0, 1]^{nK}$ in the order given by $\mathbf{x}_i \coloneqq x_v^k$, for $i = v + n(k - 1)$, $v \in [n]$ and $k \in [K]$.

Given the unary and pairwise potentials

$$\mathrm{un}_G(\mathbf{x}) \coloneqq \sum_{v \in V, k \in [K]} c_v x_v^k \tag{1}$$

and

$$\mathrm{pa}_G(\mathbf{x}) \coloneqq \sum_{\{u,v\} \in E, k \in [K]} q_{u,v} x_u^k x_v^k, \tag{2}$$

we define the cost function

$$f_G(\mathbf{x}) = \mathrm{un}_G(\mathbf{x}) + \mathrm{pa}_G(\mathbf{x}). \tag{3}$$

Then our tracking model $\mathrm{BQP}(G, K)$ is described by the labeling problem:

$$\mathrm{BQP}(G, K) \coloneqq \underset{\mathbf{x} \in \mathcal{C}_b(G,K)}{\arg\min}\ f_G(\mathbf{x}), \tag{4}$$

where $\mathcal{C}_b(G, K) \coloneqq \{0, 1\}^{nK} \cap \mathcal{C}(G, K)$ and

$$\mathcal{C}(G, K) \coloneqq \{\mathbf{x} \in [0, 1]^{nK} | \sum_{k \in [K]} x_v^k \leq 1,\ \forall v \in V\}. \tag{5}$$

The constraints (5) ensure that each detection is assigned to at most one label $k \in [K]$, *i.e.* to at most one person, or rejected. Note that $\mathrm{BQP}(G, K)$ has only n linear constraints.

We model the binomial distribution for the selection of nodes v and edges e using logistic regression. Then, finding the most likely selection is equivalent to solving $\mathrm{BQP}(G, K)$, if the costs are defined via the logit function, see [46]. Therefore, we set the unary costs as

$$c_v \coloneqq \log\left((1 - p_v)p_v^{-1}\right) \tag{6}$$

with p_v denoting the probability of detection v (as given by the detector). For the pairwise costs, we learn model parameters θ to obtain probabilities

$$p_{u,v} \coloneqq p(x_u^k = 1 \wedge x_v^k = 1 | \mathbf{f}, \theta), \tag{7}$$

given θ and a feature vector $\mathbf{f}$. Since we model $p_{u,v}$ using logistic regression, the pairwise costs are

$$q_{u,v} \coloneqq \log((1 - p_{u,v})p_{u,v}^{-1}). \tag{8}$$

In Sect. 3, we describe the model features $\mathbf{f}$ used for the classifier.

Detections, which are temporally too far apart can neither be compared reliably nor meaningful. For such edges $\{u, v\}$, we set their weight to $q_{u,v} \coloneqq 0$, so that the edge becomes irrelevant to the BQP minimization (4). This strategy effectively sparsifies the graph G and keeps the proposed approach memory and computationally efficient.

2.2. Frank-Wolfe Optimization

Solving $\mathrm{BQP}(G, K)$ is a challenging task due to the fact that it belongs to the NP-hard problems [42] and that our domain space is very high-dimensional. We thus follow a common practice and consider the relaxed problem:

$$\mathrm{QP}(G, K) \coloneqq \underset{\mathbf{x} \in \mathcal{C}(G,K)}{\arg\min}\ f_G(\mathbf{x}). \tag{9}$$

However, even the relaxation is still NP-hard to solve [39], as f_G is non-convex, in general. Thus even for commercial quadratic solvers like Gurobi [23], solving $\mathrm{BQP}(G, K)$ or $\mathrm{QP}(G, K)$ is computationally very expensive.

This paper proposes to use the Frank-Wolfe algorithm to approximate $\mathrm{QP}(G, K)$, and points out ways to further improve the solution. We present a pseudo-code of the standard Frank-Wolfe algorithm, together with a discretization step, in Alg. 1 and its evaluation, as a baseline, in Sect. 4.

Algorithm 1: Frank-Wolfe Algorithm

Data: Costs f_G, feasible point $\mathbf{x}(0)$, IMAX,ϵ
Result: Solution vector $\mathbf{x}_{\mathrm{FW}}$

1 $c_{min} = \infty$;
2 $j = -1$;
3 **repeat**
4 $\quad j = j + 1$;
5 $\quad \mathbf{s}(j) = \arg\min_{s \in \mathcal{C}(G,K)} s^\intercal \nabla f_G(\mathbf{x}(j))$;
6 $\quad \gamma(j) = \arg\min_{\gamma \in [0,1]} f_G(\mathbf{x}(j) + \gamma(\mathbf{s}(j) - \mathbf{x}(j)))$;
7 $\quad$ **if** $f_G(\mathbf{s}(j)) < c_{min}$ **then**
8 $\quad\quad c_{\min} = f_G(\mathbf{s}(j))$;
9 $\quad\quad \mathbf{x}_{\mathrm{FW}} = \mathbf{s}(j)$;
10 $\quad$ **end**
11 $\quad \mathbf{x}_d(j) = \mathrm{BINARIZE}(\mathbf{x}(j))$;
12 $\quad$ **if** $f_G(\mathbf{x}_d(j)) < c_{min}$ **then**
13 $\quad\quad c_{\min} = f_G(\mathbf{x}_d(j))$;
14 $\quad\quad \mathbf{x}_{\mathrm{FW}} = \mathbf{x}_d(j)$;
15 $\quad$ **end**
16 $\quad \mathbf{x}(j + 1) = \mathbf{x}(j) + \gamma(j)(\mathbf{s}(j) - \mathbf{x}(j))$;
17 **until** $[(\mathbf{s}(j) - \mathbf{x}(j))^\intercal(-\nabla f_G(\mathbf{x}(j))) < \epsilon] \vee [j > \mathrm{IMAX}]$;

Frank-Wolfe minimizes the linear approximation of f_G at the current solution $\mathbf{x}(j)$ (Ln. 5 of Alg. 1), resulting in

$\mathbf{s}(j)$. The next iterate $\mathbf{x}(j+1)$ is the vector between $\mathbf{x}(j)$ and $\mathbf{s}(j)$ that minimizes f_G (Ln. 6 and Ln. 16). For the optimal step size $\gamma(j)$ in Ln. 6, we present an efficient algebraic description in Sect. 2.3. The algorithm is stopped in case of a small duality gap $(\mathbf{s}(j) - \mathbf{x}(j))^\intercal(-\nabla f_G(\mathbf{x}(j)))$ or a maximal number of iterations IMAX.

To obtain binary solutions, $\mathbf{x}_{\mathrm{FW}}$ is either a discretized iterate $\mathbf{x}(j)$ (Ln. 11-15), or, $\mathbf{s}(j)$ (Ln. 7-10), as the constraint matrix corresponding to our set $\mathcal{C}(G,K)$ is totally unimodular, so that $\mathbf{s}(j)$ is binary and thus feasible [43, 27].

In order to improve the convergence rate, we use in our implementation a slightly improved variant of the algorithm, that adds so-called away-steps. We refer the interested reader to [30] for further details.

Using $\mathbf{Q}_{\text{pa}} = (q_{u,v})_{u,v\in V} \in \mathbb{R}^{n\times n}$ and $\mathbf{c}_{\text{un}} = (c_v)_{v\in V} \in \mathbb{R}^n$, we define

$$\mathbf{Q} = \mathrm{diag}(\underbrace{\mathbf{Q}_{\text{pa}},\ldots,\mathbf{Q}_{\text{pa}}}_{K \text{ times}}), \quad \mathbf{c} = (\underbrace{\mathbf{c}_{\text{un}}^\intercal,\ldots,\mathbf{c}_{\text{un}}^\intercal}_{K \text{ times}})^\intercal. \quad (10)$$

Then, we obtain f_G in matrix-vector form:

$$f_G(\mathbf{x}) = 0.5\mathbf{x}^\intercal\mathbf{Q}\mathbf{x} + \mathbf{c}^\intercal\mathbf{x}. \quad (11)$$

Due to design of our problem $\mathrm{BQP}(G,K)$, we can run Alg. 1 without the need of storing the huge $\mathbf{Q}$ matrix or the $\mathbf{c}$ vector. Instead, all computations of Alg.1 can be deduced from the upper triangle matrix of $\mathbf{Q}_{\text{pa}}$ and from $\mathbf{c}_{\text{un}}$. Therefore, our approach is memory efficient.

BINARIZE: We seek for binary, feasible vectors according to $\mathcal{C}_b(G,K)$. Hence, we discretize an iterate $\mathbf{x}(j)$ by selecting the closest feasible point $\overline{\mathbf{x}}$ w.r.t $\mathcal{C}_b(G,K)$, using the euclidean distance. To this end, let $\mathbf{1}$ be the vector with all entries equal to 1. It is straightforward to show that

$$\overline{\mathbf{x}} = \operatorname*{arg\,min}_{\mathbf{x}\in\mathcal{C}_b(G,K)} ||\mathbf{x}(j) - \mathbf{x}||_2^2 \quad (12)$$

$$= \operatorname*{arg\,min}_{\mathbf{x}\in\mathcal{C}_b(G,K)} (-2\mathbf{x}(j) + \mathbf{1})^\intercal\mathbf{x}. \quad (13)$$

Now problem (13) is linear in $\mathbf{x}$. Furthermore, the constraint matrix corresponding to $\mathcal{C}(G,K)$ is totally unimodular, so that we can efficiently solve the relaxation of (13) and obtain the exact solution of (12), see, e.g. [43, Chapter 19].

2.3. Computing the Optimal Step Size γ

The step size γ in Ln. 6 can be computed via line search [5]. However, we derive a new algebraic computation, being faster and still optimal.

Let $\mathbf{u}(j) \coloneqq \mathbf{s}(j) - \mathbf{x}(j)$ and let $\Omega(\gamma) \coloneqq f_G(\mathbf{x}(j) + \gamma\mathbf{u}(j))$. Then, since f_G is a quadratic, the only root of Ω' is

$$\overline{\gamma} \coloneqq [-\mathbf{u}(j)^\intercal\nabla f_G(\mathbf{x}(j))][\mathbf{u}(j)^\intercal\mathbf{Q}\mathbf{u}(j)]^{-1}$$

and $\delta \coloneqq \Omega''(\overline{\gamma}) = \mathbf{u}(j)^\intercal\mathbf{Q}\mathbf{u}(j)$. For $\delta \neq 0$, we obtain the minimum of Ω in $[0,1]$, denoted by γ, via

$$\gamma = \begin{cases} \bar{\gamma}, & \text{if } \delta > 0 \text{ and } \bar{\gamma} \in [0,1], \\ 0, & \text{if } (\delta > 0 \text{ and } \bar{\gamma} \leq 0) \text{ or } (\delta < 0 \text{ and } \bar{\gamma} \geq 1), \\ 1, & \text{if } (\delta > 0 \text{ and } \bar{\gamma} \geq 1) \text{ or } (\delta < 0 \text{ and } \bar{\gamma} \leq 0), \\ \operatorname*{arg\,min}_{\gamma\in\{0,1\}} \Omega(\gamma), & \text{if } \delta < 0 \text{ and } \bar{\gamma} \in (0,1). \end{cases}$$

A line search is needed only if $\delta = 0$, making the execution of Ln. 6 very efficient. In contrast to previous works [31, 15], our solution to Ln. 6 contains all cases that may occur.

2.4. Regularization of the Objective Function

Since our cost function is non-convex, Frank-Wolfe delivers only a local optimum [29]. Given $u \neq 0$, our next proposed improvement is to replace the objective function f_G by

$$f_u(\mathbf{x}) = f_G(\mathbf{x}) + u\sum_i (\mathbf{x}_i^2 - \mathbf{x}_i). \quad (14)$$

Then for $\mathbf{x} \in \{0,1\}^{nK}$, we have $f_u(\mathbf{x}) = f_G(\mathbf{x})$. Using $u < 0$ has the effect of pushing the Frank-Wolfe algorithm towards discrete solutions, as $-(x_i^2 - x_i)$ has its minimum at 0 and 1, within $[0,1]$. For $u > 0$, we achieve the opposite, rewarding a distance to binary solutions. We observed better results for positive u, as it shows better behavior in staying out of local optima. Moreover, for a value u sufficiently large, f_u becomes convex [6, 9, 24]. On the other hand, a high u value will bring the optimal solution too close to the constant $(\frac{1}{2})$ vector. For $\omega \coloneqq \max\{\sum_j |\mathbf{Q}_{i,j}| : i \in [nK]\}$, we set $u_0 = \sqrt{\omega}$ and $u_i = 2^{-i}u_0$. Starting with $u = u_0$, we compute $\mathrm{QP}(G,K)$, using f_u in Alg. 1. Empirically, we observed that a short number of iterations corresponds to a too strong convexification term, resulting in a bad local optimum. Thus, if Alg. 1 terminates in too few steps (which we set to 10), we set $i = i+1$, $u = u_i$ and run Alg. 1 again with the updated objective function f_u. In all our experiments, an appropriate weight was found in at most two u iterations. In Sect. 4, we demonstrate the impact of using the modified cost function, with the solver we call $\mathbf{FW} + \mathbf{u}$.

2.5. Hierarchical Solving Scheme

Since $FW + u$ delivers only a local optimum, we propose a new hierarchical solving scheme that enhances the solution of $FW + u$ by removing, correcting and connecting clusters, thus resulting in an improved objective value. Our approach is computationally efficient and continues optimizing problem $\mathrm{BQP}(G,K)$.

In the following, we present all parts of our proposed solving scheme and present a pseudo-code in Alg. 2.

ContractRefinement: Let $\mathbf{x}^{(t)} \in \mathbb{R}^{nK}$ be the current best labeling of G. Initially, we obtain $\mathbf{x}^{(0)}$ using $FW + u$. We replace all nodes labeled k by one new representative

node $v(k)$. These nodes are used to build a new graph $\underline{G} = (\underline{V}, \underline{E})$, which we call the *contracted graph* (see Fig. 3 middle). We set $\underline{V} := \{v(k) \mid k \in [nK]\}$ and $\underline{E}$ connects any two different vertices. We apply a local refinement strategy that corrects obvious errors within the clusters that may have been introduced due to the rounding or local optimality. To this end, a node $v \in V$ labeled k with $0 < \sum_{u \in V} (\mathbf{x}^{(t)})_u^k q_{u,v}$ is removed from $v(k)$ and assigned to a new cluster node $v(k') \notin \underline{V}$, thereby extending $\underline{V}$, resulting in an updated graph $\underline{G}$ (see Fig. 3 middle). Also for nodes that were rejected by the current labeling $\mathbf{x}^{(t)}$, we create a new cluster node $v(k') \notin \underline{V}$, so that we obtain a new complete graph G_{t+1}. Accordingly, we create a new labeling vector $\mathbf{x}_{\text{ref}}$ with the updated clustering, so that ${\mathbf{x}_{\text{ref}}}_v^k = 1$ iff node $v \in V$ is represented by $v(k) \in V(G_{t+1})$.

LabelExpand: Given the current clustering encoded by $\mathbf{x} := \mathbf{x}_{\text{ref}}$, grouping the nodes $V(G)$ into J clusters. To build the labeling BQP on G_{t+1}, we define the unary costs

$$\underline{c}_{v(k)} := \sum_{v \in \{v \in V \mid x_v^k = 1\}} c_v + \sum_{\{u,v\} \in \{\{u,v\} \in E \mid x_u^k x_v^k = 1\}} q_{u,v} \tag{15}$$

and pairwise costs

$$q_{v(k),v(k')} := \sum_{\{u,v\} \in \{\{u,v\} \in E \mid x_u^k = 1, x_v^{k'} = 1\}} q_{u,v}. \tag{16}$$

Define $\hat{\mathbf{x}} \in \{0,1\}^{JJ}$ s.th. all nodes of G_{t+1} get a unique ID, except for the nodes removed in $\mathbf{x}(t)$:

$$\hat{\mathbf{x}}_{v(k)}^a = \begin{cases} 1, & \text{if } a = k \text{ and } v(k) \text{ is not a deactivated node} \\ & \text{in } G, \text{ according to } \mathbf{x}^{(t)}, \\ 0, & \text{otherwise.} \end{cases}$$

Then, $f_{G_{t+1}}(\hat{\mathbf{x}})$ sums up only the unary costs (15), which equal $f_G(\mathbf{x}^{(t)})$ or are improved by the refinement. Hence,

$$f_{G_{t+1}}(\hat{\mathbf{x}}) \leq f_G(\mathbf{x}^{(t)}). \tag{17}$$

Now solving BQP(G_{t+1}, J) results in a solution $\mathbf{x}_H \in \mathcal{C}_b(G_{t+1}, J)$ with

$$f_{G_{t+1}}(\mathbf{x}_H) \leq f_{G_{t+1}}(\hat{\mathbf{x}}) \leq f_G(\mathbf{x}^{(t)}). \tag{18}$$

The result $\mathbf{x}_H$ is transformed to a labeling $\mathbf{x}^{(t+1)} \in \mathcal{C}_b(G, K)$ by graph expansion: If a node $v(k)$ of G_{t+1} is labeled p according to $\mathbf{x}_H$, then all nodes $v \in V$ that were represented by $v(k)$ are assigned the label p, see also Fig. 3. Then, we obtain the improved solution, since

$$f_G(\mathbf{x}^{(t+1)}) = f_{G_{t+1}}(\mathbf{x}_H) \leq f_G(\mathbf{x}^{(t)}). \tag{19}$$

The graph contraction reduces the dimensionality significantly: There are $J \ll n$ nodes to be labeled using at most J labels, w.r.t. BQP(G_{t+1}, J). In most cases, we can solve BQP(G_{t+1}, J) quickly to optimality using Gurobi [23]. Otherwise we use the $FW + u$ solver. The algorithm is stopped once no new clusters are merged.

We demonstrate the effect of the hierarchical solving scheme $\mathbf{FW} + \mathbf{u} + \mathbf{h}$ in Sect. 4.

Algorithm 2: Hierarchical solving scheme

Data: Graph labeling $\mathbf{x}^{(0)}$, graph G
Result: Solution vector $\mathbf{x}^{(t)}$

```
repeat
    c_min = f_G(x^(t));
    (G_{t+1}, x_ref) = ContractRefinement(x^(t), G);
    x^(t+1) = LabelExpand(G_{t+1}, x_ref);
    t = t + 1;
until f_G(x^(t)) = c_min;
```

3. Regression Training

For our tracking system, we consider two input sources: (i) head and (ii) full-body detections (see also Fig. 2). As explained in Sect. 2.1, we train a logistic regression model, with weights θ for the spatial and temporal costs, using [17].

In the following, we introduce spatial affinities, which describe how likely two detections within the same image could belong to the same person. Accordingly, the temporal affinities compare two detections between different frames.

Head detections. To obtain accurate head detections, we employ [45] based on Convolutional Neural Networks and fine-tune it on the MOT16 training set [36].

Full-body detections. We use the full-body detections [19] as provided by the MOT16 challenge [36].

Relative positioning. In order to obtain meaningful features between differently sized boxes, features have to be formulated respecting the different scales.

To this end, given $p \in \mathbb{R}^2$ and a bounding box b, we use its lower left corner, denote by c_1^b, and the upper corners, denoted by c_2^b and c_3^b, to obtain the barycentric coordinates $(\lambda_1, \lambda_2, \lambda_3)$ of p w.r.t. b (see Fig. 4), so that $p_b := p = \sum_i \lambda_i c_i^b$. We fix a rectangle r. Then, p is mapped to $p_r := \sum_i \lambda_i c_i^r$ in r, keeping the relative position as in b. Now, all subsequent distance measurements are computed in r using the mapped position and the euclidean norm.

Spatial affinities. We introduce two features that set the position of the head in relation to the full-body box.

For a pair of head and full-body detection, we mirror the head detection to the left half side of the bounding box, resulting in p_b and making the position robust against different orientations of the person. From the MOT16 training data, we learned the mean relative position m_{h} of a head to a full-body detection, if both belong to the same person, in r. Then, we compute the distance $\|p_r - m_h\|_2$ between the detected and expected position in the fixed rectangle r. The second feature uses the angle between expected and detected position, with the anchor at the box's center (see Fig. 4). We set the spatial affinity between detections from the same detector to a constant high value.

Temporal affinities. We define temporal affinities via correspondences of pixels between two frames. DeepMatching

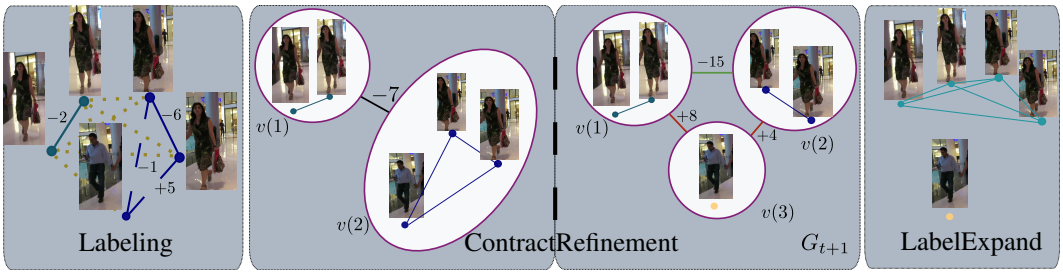

Figure 3. One iteration of our hierarchical solver. We present an illustrative example with pairwise costs on the edges. **Left**: A labeling, computed by Alg. 1. Dotted edges indicate removed links. Nodes connected by edges of the same color are grouped, resulting in two clusters. Dashed edges indicate wrong connections. **Middle**: Each cluster is replaced by a new node (purple circles). In addition, wrongly connected nodes (the men's node has positive costs of $+4$ to its connected nodes) are separated. The green edge indicates the right assignment, whereas red edges indicate that clusters do not belong together. **Right**: The problem BQP(G_{t+1}, J) is solved w.r.t. G_{t+1} and $J = \{1, 2, 3\}$ labels. Since J is small, we can solve BQP(G_{t+1}, J) quickly and optimal, using Gurobi.

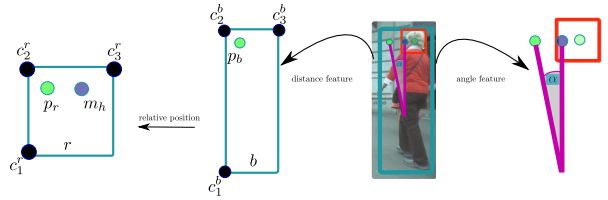

Figure 4. Distance and angle between the expected (blue node) and the observed (mirrored) head position (green node). Barycentric coordinates are computed w.r.t. black corners.

Table 1. Solver comparison: While our solver quickly terminates, Gurobi is not able to finish after 1000 seconds. Entries in brackets denote the results of Gurobi after that time.

Method	Iters	Time[sec]	Obj Value	MOTA
FW	**16**	**0.7**	-3060	14.2
$FW + u$	676	27	-5481	26.8
$FW + u + h$	-	27+0.5	**-5925**	**27.5**
Gurobi	-	1000	(-5531)	24.9
Gurobi bound	-	1000	(-5973)	-

[49] (DM) provides such assignments, which are more reliable than spatio-temporal affinities, see [47]. Given rectangles u and v, DM samples dm_u and dm_v many pixels in u and v, respectively. Let $\text{co}_{u,v}$ denote the number of correspondences, found by DM. Comparing two heads or two full-body detections, we use the features $\frac{\text{co}_{u,v}}{\text{dm}_u}$, $\frac{\text{co}_{u,v}}{\text{dm}_v}$ and $\frac{\text{co}_{u,v}}{0.5*(\text{dm}_u+\text{dm}_v)}$, as in [47]. As head detections are significantly smaller than full-body detections, we only use the temporal head to full-body feature $\frac{\text{co}_{u,v}}{\text{dm}_u}$, where u denotes the head detection and v the full-body detection. From the MOT16 training data, we learned the mean ratios r^w_{m} and r^h_{m} between a head and body detection, w.r.t. width and height, respectively, if both belong to the same person. Then, we obtain features $\|r^w_{\text{m}} - r^w_{\text{det}}\|_2$ and $\left\|r^h_{\text{m}} - r^h_{\text{det}}\right\|_2$, for the observed ratios r^w_{det} and r^h_{det} w.r.t. width and height, respectively, given a pair of detected head and full-body detection.

4. Experimental Results

In this section, we first analyze the gain both in speed as well as in tracking performance by our proposed solver. Next, we investigate the impact of multiple detector fusion on the tracking performance, using the training sequences of the challenging MOT16 benchmark [36]. This benchmark consists of 7 sequences for training and 7 for testing, with footage of crowded scenes. In the last experiment, we show our performance on the test set of the benchmarks *MOT16* and *MOT17*, where we achieve state-of-the-art performance. We evaluate our experiments using well-established tracking metrics [4, 35].

4.1. Implementation Details

In our implementation, we set the temporal costs of two nodes being more than 9 frames apart to zero. The maximal number of labels K is fixed to 70. We process a sequence in batches containing no more than 1800 nodes. We stop the Frank-Wolfe iterations of Alg.1 in case that the duality gap is below 10^{-4} or 750 iterations are reached.

4.2. Frank-Wolfe Optimization

Our first experiment analyzes the impact of our modifications on the Frank-Wolfe optimization. To this end, we choose a representative batch of 41 frames from the MOT16-13 training sequence and perform tracking using full-body detections only. It consists of 403 detections, so that we have 28210 decision variables. In Tab. 1 we show the number of iterations performed by the solver until the duality gap is below the defined threshold, the runtime, the final objective value of f_G as well as the corresponding Multiple Object Tracking Accuracy (MOTA).

Our proposed modification $FW+u+h$ improves the objective value considerably compared to the standard Frank-Wolfe algorithm FW. This naturally translates to almost

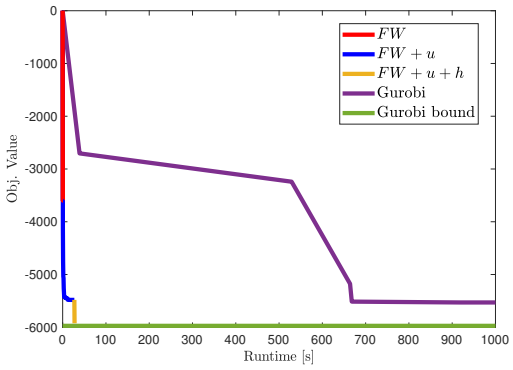

Figure 5. Minimization performance of each BQP solver and the bound as given by Gurobi for each moment.

double MOTA accuracy, 14.2% vs 27.5%. Note also that the objective value comes very close to the global optimum. The commercial solver Gurobi [23], which uses the branch-and-bound algorithm is still far away from the global optimum after 1000 seconds, while we obtain a much better energy after only 27.5 seconds. While Gurobi was not able to compute the global optimum in the given time span[2], it delivers at each time step a lower bound (Gurobi bound) on the optimal value, showing that the optimal solution to the BQP has an objective value ≥ -5973.

The energy evolution of the different solvers is plotted in Fig. 5. Here we clearly see where FW stops (red line), how our modification $FW + u$ improves the energy by a large margin (blue line), and how finally $FW + u + h$ (yellow line) comes even closer to the estimated lower bound (green line), as provided by Gurobi. In contrast, Gurobi (purple line) has a much slower convergence.

4.3. Tracking with Two Detectors

We analyze how our tracking system benefits from two detectors. To this end, we apply our tracker to all MOT16 training sequences and report the evaluation in Tab. 2, with full-body detections only (*Body*) against body and head detections (*Body+Head*). For the evaluation of a MOT16 training sequence, we train the head detector and the regression model on all other MOT16 training sequences. By using the two detectors, the system significantly improves almost all relevant tracking metrics, justifying our tracking framework. Due to the coupling of head detections with full-body detections, the number of false positives (FP) is halved and the system is less prone to partial occlusions, which results in an increase of the number of mostly tracked (MT) trajectories. Overall, the MOTA score increases by more than 5%.

[2]In fact, we let it run for 10 hours, without any improvement, but more than 200 GB RAM consumption.

Table 2. Feature evaluation on MOT16 training sequences.

$FW + u + h$						
Feature	MOTA	MT	ML	FP	FN	IDs
Body	33.0	76	**219**	11949	**61603**	378
Body+Head	**38.2**	**86**	220	**4972**	62935	**372**

4.4. Benchmark Evaluation

We evaluate the tracking performance of *Body+Head* on the benchmarks *MOT16* and *MOT17* with the full-body detections as given by the benchmarks. The corresponding results for the three best performing trackers are provided in Tab. 3, as shown on the website[3]. Our proposed tracker performs on-pair with state-of-the-art in terms of tracking accuracy on *MOT16* and sets a new state-of-the-art on *MOT17*, thereby winning the *MOT17* challenge. Due to the detector fusion, we have the lowest ML (mostly lost) score within all trackers in both benchmarks. This demonstrates that we can recover more trajectories than any other tracker. Also our MT score is highest on the MOT16 benchmark and ranks second on the MOT17 benchmark, demonstrating that we obtain long and consistent trajectories.

5. Conclusion

We presented a global formulation for multi-target tracking, that integrates head and full-body detectors. The problem is casted into a quadratic program, which is solved efficiently via the Frank-Wolfe algorithm. We improved the solver in three ways; (i) regarding time by providing complete and efficient computation of the optimal step size and (ii) regarding minimization by a reformulation of the objective function, resulting in better discrete solutions. Finally (iii), we showed that our hierarchical solving scheme improves a feasible solution, often close to optimality and yet is easy to integrate and fast.

The detector fusion delivered superior results when compared to single detector tracking, thus proving the benefits of our formulation. The overall performance on two tracking benchmarks showed state-of-the-art results.

[3]See `https://motchallenge.net/results/MOT16/` and `https://motchallenge.net/results/MOT17/`.

Table 3. The three best performing trackers of *MOT16* and *MOT17*.

Method	MOTA↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	ID↓	FM↓
MOT16								
LMP [48]	**48.8**	**79.0**	18.2	40.1	6654	86245	**481**	**595**
Ours	47.8	75.5	**19.1**	**38.2**	8886	**85487**	852	1534
NLLMPa [34]	47.6	78.5	17.0	40.4	**5844**	89093	629	768
MOT17								
Ours	**51.3**	77.0	21.4	**35.2**	**24101**	247921	2648	4279
MHT_DAM[28]	50.7	**77.5**	20.8	36.9	22875	252889	2314	**2865**
EDMT17[11]	50.0	77.3	**21.6**	36.3	32279	**247297**	**2264**	3260

References

[1] A. Alahi, V. Ramanathan, and L. Fei-Fei. Socially-aware large-scale crowd forecasting. In *CVPR*, 2014.

[2] S. M. Assari, H. Idrees, and M. Shah. Re-identification of humans in crowds using personal, social and environmental constraints. *arXiv preprint arXiv:1612.02155*, 2016.

[3] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *TPAMI*, 2011.

[4] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008.

[5] D. P. Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999.

[6] A. Billionnet, S. Elloumi, and A. Lambert. Extending the qcr method to general mixed-integer programs. *Mathematical programming*, 131(1):381–401, 2012.

[7] W. Brendel, M. Amer, and S. Todorovic. Multiobject tracking as maximum weight independent set. In *CVPR*, 2011.

[8] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010.

[9] S. Burer and A. N. Letchford. Non-convex mixed-integer nonlinear programming: A survey. *Surveys in Operations Research and Management Science*, 17(2):97–106, 2012.

[10] V. Chari, S. Lacoste-Julien, I. Laptev, and J. Sivic. On pairwise costs for network flow multi-object tracking. In *CVPR*, 2015.

[11] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong. Enhancing detection model for multiple hypothesis tracking. In *CVPRW*, 2017.

[12] S. Chen, A. Fern, and S. Todorovic. Multi-object tracking via constrained sequential labeling. In *CVPR*, 2014.

[13] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*, 2015.

[14] A. Dehghan, S. Assari, and M. Shah. Gmmcp-tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In *CVPR*, 2015.

[15] A. Dehghan and M. Shah. Binary Quadratic Programing for Online Tracking of Hundreds of People in Extremely Crowded Scenes. *TPAMI*, 2017.

[16] P. Dolĺar, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *TPAMI*, 2014.

[17] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 9, 2008.

[18] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *TPAMI*, 2010.

[19] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010.

[20] K. Fragkiadaki and J. Shi. Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement. In *CVPR*, 2011.

[21] K. Fragkiadaki, W. Zhang, G. Zhang, and J. Shi. Two-granularity tracking: mediating trajectory and detections graphs for tracking under occlusions. In *ECCV*, 2012.

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

[23] I. Gurobi Optimization. *Gurobi Optimizer Reference Manual*. 2015.

[24] P. L. Hammer and A. A. Rubin. Some remarks on quadratic programming with 0-1 variables. *Revue française d'informatique et de recherche opérationnelle. Série verte*, 1970.

[25] R. Henschel, L. Laura Leal-Taixé, and B. Rosenhahn. Efficient multiple people tracking using minimum cost arborescences. In *GCPR*, 2014.

[26] H. Jiang, S. Fels, and J. Little. A linear programming approach for multiple object tracking. In *CVPR*, 2007.

[27] A. Joulin, K. Tang, and F. F. Li. Efficient image and video co-localization with frank-wolfe algorithm. In *ECCV*, 2014.

[28] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited: Blending in modern appearance model. In *ICCV*, 2015.

[29] S. Lacoste-Julien. Convergence rate of frank-wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*, 2016.

[30] S. Lacoste-Julien and M. Jaggi. On the global linear convergence of frank-wolfe optimization variants. *NIPS*, 2015.

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

[32] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society*, 1960.

[33] L. Leal-Taixé, M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese. Learning an image-based motion context for multiple people tracking. In *CVPR*, 2014.

[34] E. Levinkov, J. Uhrig, S. Tang, M. Omran, E. Insafutdinov, A. Kirillov, C. Rother, T. Brox, B. Schiele, and B. Andres. Joint graph decomposition & node labeling: Problem, algorithms, applications. In *CVPR*, 2017.

[35] Y. Li, C. Huang, and R. Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *CVPR*. IEEE, 2009.

[36] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[37] A. Milan, L. Leal-Taixé, K. Schindler, and I. Reid. Joint tracking and segmentation of multiple targets. In *CVPR*, 2015.

[38] J. E. Mitchell. Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, pages 65–77, 2002.

[39] P. M. Pardalos and S. A. Vavasis. Quadratic programming with one negative eigenvalue is np-hard. *Journal of Global Optimization*, 1(1):15–22, 1991.

[40] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011.

[41] S. Ren, R. G. K. He, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*, 2015.

[42] S. Sahni. Computationally related problems. *SIAM Journal on Computing*, 1974.

[43] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[44] G. Seguin, P. Bojanowski, R. Lajugie, and I. Laptev. Instance-level video segmentation from object tracks. In *CVPR*, 2016.

[45] R. Stewart, M. Andriluka, and A. Y. Ng. End-to-end people detection in crowded scenes. In *CVPR*, 2016.

[46] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph decomposition for multi-target tracking. In *CVPR*, 2015.

[47] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Multi-person tracking by multicuts and deep matching. In *ECCV Workshops - Benchmarking Multi-Target Tracking*, 2016.

[48] S. Tang, M. Andriluka, B. Andres, and B. Schiele. Multi people tracking with lifted multicut and person re-identification. In *CVPR*, 2017.

[49] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *ICCV*, 2013.

[50] F. Yang, W. Choi, and Y. Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *CVPR*, 2016.

[51] A. Zamir, A. Dehghan, and M. Shah. Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs. In *ECCV*, 2012.

[52] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.