

Fusion of Head and Full-Body Detectors for Multi-Object Tracking

Roberto Henschel¹ Laura Leal-Taixé² Daniel Cremers² Bodo Rosenhahn¹

¹Leibniz Universität Hannover ²Technische Universität München

{henschel,rosenhahn}@tnt.uni-hannover.de {leal.taixe,cremers}@tum.de

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. Consequently, this work demonstrates how to fuse two detectors into a tracking system. To obtain the trajectories, we propose to formulate 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. Evaluation on pedestrian tracking is provided for multiple scenarios, showing superior results over single detector tracking and standard QP-solvers. Finally, our tracker ranks 2nd on the MOT16 benchmark and 1st on the new MOT17 benchmark, outperforming over 90 trackers.

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 [15, 16, 37, 57]. 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 [14, 21, 44] have proposed to use other image features aside from full-body detections, with the main goal of recovering partially occluded pedestrians.

In this paper, we present a framework for offline multiple object tracking using two detector types, namely, full-body detections together with head detections, since heads can be detected very accurately, as they are barely prone to pose variations or occlusions. This is especially useful in crowded scenarios: Fig. 1 shows a heavily occluded pedestrian. While the full-body detector is unable to detect that person, due to the occlusion, its head is still visible so that our tracker localizes that pedestrian correctly.

Our tracking formulation ensures long-term temporal consistency by taking *all* detections assigned to a person (we denote detection to person assignments as labelings) into account. Therefore, our clustering concept shares similarities to correlation clustering approaches [16, 56, 57, 63], but we propose a very efficient labeling formulation that avoids the exponential growth in the constraints. Due to our powerful solver, we are able to optimize our problem globally on the input detections without the need of potentially 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.

We propose to use the Frank-Wolfe algorithm (FW) to solve the relaxation of the BQP. 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 in terms of the objective value and the tracking performance, as we show in Section 4.2. At the same time, the proposed algorithm is much faster than the standard branch&bound approach. 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.



Figure 1. MOT16-09 sequence (from left to right: frame 10,12,15). **Top row:** Body detections (orange) and head detections (red). The body detector misses the person depicted by the arrow until frame 15. **Bottom row:** The result of our tracker. The false positive is removed as it does not have a corresponding head detection. The tracker recovers the heavily occluded pedestrian due to the presence of head detections.

1.1. Contributions

To summarize, our contribution is three-fold:

- We propose a novel detector fusion 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

Data association models. 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 [18, 19, 48, 62] 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 [5, 9, 25, 47, 54, 64], or BLPs [12, 28].

Most of these trackers were derived from a Markov chain model [64]. Recent systems utilize correlation clustering based formulations that ensure consistency within all links of a trajectory [16, 44, 56, 57, 58, 59, 63]. Thereby, simplified models were used initially, which created trajectories iteratively, computing one best clique [63] or dominant set [59]

corresponding to exactly one person and then removing the respective detections from the loop. This concept has been extended [16, 44] to obtain trajectories in a global manner, for all persons at the same time. However, the inference relies on potentially error-prone initial tracklets to keep the approach computationally feasible. In contrast, our solver is fast and accurate enough to optimize directly on the detections, thereby avoiding error propagation that might have been introduced by the tracklets. Further progress has been made by computing the correlation clustering directly on the input detections [56, 57], using a huge set of clique constraints in a BLP, that has exponential growth. Accordingly, a heuristic solver has to be applied. In contrast, our formulation needs only very few constraints, making it capable for the usage of many detections.

BQP Optimization. Tracking methods that need to solve a BQP have been rare so far, due to the computational challenge, although many advanced tracking models are naturally expressed as a BQP. For instance, the Markov model [64] can be augmented by one additional detector [12], resulting in a BQP. While this problem can be solved by rewriting the BQP as an equivalent BLP, we show in our experiments, that this simple trick is not applicable to our more demanding correlation clustering based model, due to the problem size of our BQP. Another work [17] formulates online tracking via a BQP and solves it using the Frank-Wolfe algorithm, which is also the basis for our solver. While [17] 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, 17, 35]. 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 [29] or object segmentation [53].

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 [14], the optimization is done via greedy propagation, while in [44], supervoxel labeling is formulated as CRF.

There are several works that use dense point tracks (DPT) [10] or KLT [42, 60] together with detections to improve tracking performance. In [4], corner features are tracked using KLT to obtain a motion model between detections. In [20], multi-target tracking is tackled by clustering DPTs and further combined with detection-based track-

lets in a two-step approach in [21]. Further improvement is achieved using a globally optimal fusion formulation [26].

In [12], 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 [53] to motion segmentation using superpixels, the per-person consistency is not considered.

2. Detector Fusion for Multi-Target Tracking

In this section, we describe the data association that couples multiple detectors and detections in a correlation clustering fashion to ensure long-term temporal consistency. As correlation clustering is NP-Complete [3], we rely on finding a good approximation to the solution. We propose to use a BQP formulation for the clustering problem that can be well approximated using the Frank-Wolfe [22] solver. In particular, we compute the relaxed solution of the BQP first, and perform a rounding step afterwards. Frank-Wolfe is well suited for continuous quadratic problems with linear constraints, as each iteration step involves solving a computationally efficient linear optimization problem. The binary solution is then obtained by an efficient rounding step.

When applied to a non-convex problem, like our model, the Frank-Wolfe algorithm delivers only a local optimum [33]. 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 [36, 45] to find the globally optimal solution.

2.1. Joint Data Association

We cast the data association using two detectors as a graph labeling problem: Consider a weighted complete graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$, where the vertex set \mathcal{V} consists of all

input detections. We set $n = |\mathcal{V}|$. Each node $v \in \mathcal{V}$ has costs $c_v \in \mathbb{R}$ reflecting the likelihood of v being a correct detection. An edge $e = \{u, v\} \in \mathcal{E}$ encodes a possible linking of two detections to the same person. The nodes u, v are labeled k , if u and v belong to person k . Likewise, $q_{u,v} \in \mathbb{R}$ reflects how likely u and v belong to the same person.

Finally, the goal of the data association problem is then to find the labeling for all detection nodes that minimizes the total costs.

Hence, for each node $v \in \mathcal{V}$, consider a decision variable x_v^k that equals 1, if node v has label k , and 0 otherwise. For P being an upper bound on the number of persons, let $[P] := \{1, \dots, P\}$. Then, the vector $\mathbf{x} \in [0, 1]^{nP}$ stacks all decision variables in a vector.

Given the unary and pairwise potentials

$$\text{un}_{\mathcal{G}}(\mathbf{x}) := \sum_{v \in \mathcal{V}, k \in [P]} c_v x_v^k \quad (1)$$

and

$$\text{pa}_{\mathcal{G}}(\mathbf{x}) := \sum_{\{u, v\} \in \mathcal{E}, k \in [P]} q_{u,v} x_u^k x_v^k, \quad (2)$$

we define the cost function

$$f_{\mathcal{G}}(\mathbf{x}) = \text{un}_{\mathcal{G}}(\mathbf{x}) + \text{pa}_{\mathcal{G}}(\mathbf{x}). \quad (3)$$

Finally, our tracking model BQP(\mathcal{G}, P) is described by the labeling problem:

$$\text{BQP}(\mathcal{G}, P) := \arg \min_{\mathbf{x} \in \mathcal{C}_b(\mathcal{G}, P)} f_{\mathcal{G}}(\mathbf{x}), \quad (4)$$

where $\mathcal{C}_b(\mathcal{G}, P) := \{0, 1\}^{nP} \cap \mathcal{C}(\mathcal{G}, P)$ and

$$\mathcal{C}(\mathcal{G}, P) := \{\mathbf{x} \in [0, 1]^{nP} \mid \sum_{k \in [P]} x_v^k \leq 1, \forall v \in \mathcal{V}\}. \quad (5)$$

The constraints (5) ensure that each detection is assigned to at most one label $k \in [P]$, i.e. to at most one person. Note that BQP(\mathcal{G}, P) 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 BQP(\mathcal{G}, P), if the costs are defined via the logit function, see [56]. Therefore, we set the unary costs as

$$c_v := \log((1 - p_v)p_v^{-1}) \quad (6)$$

with p_v denoting the probability of detection v (inferred from the detection's score). For the pairwise costs, we learn model parameters θ to obtain probabilities

$$p_{u,v} := p(x_u^k = 1 \wedge x_v^k = 1 | \mathcal{F}, \theta), \quad (7)$$

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

$$q_{u,v} := \log((1 - p_{u,v})p_{u,v}^{-1}). \quad (8)$$

In Sect. 3, we describe the model features \mathcal{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 $q_{u,v} := 0$. This strategy effectively sparsifies the graph \mathcal{G} and keeps the proposed approach memory and computationally efficient.

2.2. Frank-Wolfe Optimization

Solving $\text{BQP}(\mathcal{G}, P)$ is a challenging task due to the fact that it belongs to the NP-hard problems [51] and that our domain space is very high-dimensional. We thus follow a common practice and consider the relaxed problem:

$$\text{QP}(\mathcal{G}, P) := \arg \min_{\mathbf{x} \in \mathcal{C}(\mathcal{G}, P)} f_{\mathcal{G}}(\mathbf{x}). \quad (9)$$

However, even the relaxation is still NP-hard to solve [46], as $f_{\mathcal{G}}$ is non-convex, in general. Thus even for commercial quadratic solvers like Gurobi [23], solving $\text{BQP}(\mathcal{G}, P)$ or $\text{QP}(\mathcal{G}, P)$ is computationally very expensive.

This paper proposes to use the Frank-Wolfe algorithm to approximate $\text{QP}(\mathcal{G}, P)$, 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 $f_{\mathcal{G}}$, feasible point $\mathbf{x}(0)$, IMAX, ϵ
Result: Solution vector \mathbf{x}_{FW}

```

1  $f_{\min} = \infty;$ 
2  $j = -1;$ 
3 repeat
4    $j = j + 1;$ 
5    $\mathbf{a}(j) = \arg \min_{\mathbf{a} \in \mathcal{C}(\mathcal{G}, P)} \mathbf{a}^T \nabla f_{\mathcal{G}}(\mathbf{x}(j));$ 
6    $\gamma(j) = \arg \min_{\gamma \in [0, 1]} f_{\mathcal{G}}(\mathbf{x}(j) + \gamma(\mathbf{a}(j) - \mathbf{x}(j)));$ 
7   if  $f_{\mathcal{G}}(\mathbf{a}(j)) < f_{\min}$  then
8      $| f_{\min} = f_{\mathcal{G}}(\mathbf{s}(j));$ 
9      $| \mathbf{x}_{\text{FW}} = \mathbf{a}(j);$ 
10    end
11     $\mathbf{x}_b(j) = \text{BINARIZE}(\mathbf{x}(j));$ 
12    if  $f_{\mathcal{G}}(\mathbf{x}_b(j)) < f_{\min}$  then
13       $| f_{\min} = f_{\mathcal{G}}(\mathbf{x}_b(j));$ 
14       $| \mathbf{x}_{\text{FW}} = \mathbf{x}_b(j);$ 
15    end
16     $\mathbf{x}(j+1) = \mathbf{x}(j) + \gamma(j)(\mathbf{a}(j) - \mathbf{x}(j));$ 
17 until  $[(\mathbf{a}(j) - \mathbf{x}(j))^T (-\nabla f_{\mathcal{G}}(\mathbf{x}(j))) < \epsilon] \vee [j > \text{IMAX}];$ 

```

Frank-Wolfe minimizes the linear approximation of $f_{\mathcal{G}}$ at the current solution $\mathbf{x}(j)$ (Ln. 5 of Alg. 1), resulting in $\mathbf{a}(j)$. The next iterate $\mathbf{x}(j+1)$ is the vector between $\mathbf{x}(j)$ and $\mathbf{a}(j)$ that minimizes $f_{\mathcal{G}}$ (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 $(\mathbf{a}(j) - \mathbf{x}(j))^T (-\nabla f_{\mathcal{G}}(\mathbf{x}(j)))$ or a maximal number of iterations IMAX.

The binary solution \mathbf{x}_{FW} equals either a binarized iterate $\mathbf{x}(j)$ (Ln. 11-15), or, $\mathbf{a}(j)$ (Ln. 7-10), as the constraint matrix corresponding to our set $\mathcal{C}(\mathcal{G}, P)$ is totally unimodular, so that $\mathbf{a}(j)$ is already binary and thus feasible [29, 52].

In order to enhance 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 [34] for further details.

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

$$\mathbf{Q} = \underbrace{\text{diag}(\mathbf{Q}_{\text{pa}}, \dots, \mathbf{Q}_{\text{pa}})}_{P \text{ times}}, \quad \mathbf{c} = \underbrace{(\mathbf{c}_{\text{un}}^T, \dots, \mathbf{c}_{\text{un}}^T)^T}_{P \text{ times}}. \quad (10)$$

Then, we obtain $f_{\mathcal{G}}$ in matrix-vector form:

$$f_{\mathcal{G}}(\mathbf{x}) = 0.5 \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x}. \quad (11)$$

Due to design of our problem $\text{BQP}(\mathcal{G}, P)$, 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}_{pa} and from \mathbf{c}_{un} . Therefore, our approach is memory efficient.

BINARIZE: In order to obtain feasible, binary vectors, we discretize an iterate $\mathbf{x}(j)$ by selecting the closest feasible point $\mathbf{x}_b(j)$ in $\mathcal{C}_b(\mathcal{G}, P)$ w.r.t. euclidean distance. To this end, let $\mathbf{1}$ be the vector with all entries equal to 1. It is straightforward to show that

$$\mathbf{x}_b(j) = \arg \min_{\mathbf{x} \in \mathcal{C}_b(\mathcal{G}, P)} \|\mathbf{x}(j) - \mathbf{x}\|_2^2 \quad (12)$$

$$= \arg \min_{\mathbf{x} \in \mathcal{C}_b(\mathcal{G}, P)} (-2\mathbf{x}(j) + \mathbf{1})^T \mathbf{x}. \quad (13)$$

Now problem (13) is linear in \mathbf{x} and the constraint matrix corresponding to $\mathcal{C}(\mathcal{G}, P)$ is totally unimodular. Thus, we can efficiently solve the relaxation of (13) and obtain the exact solution of (12), see, e.g. [52, Chapter 19].

2.3. Computing the Optimal Step Size γ

The step size γ in Ln. 6 of Alg. 1 can be computed via line search [7]. However, we derive a new algebraic computation, being faster and still optimal.

Let $\mathbf{d}(j) := \mathbf{a}(j) - \mathbf{x}(j)$ and $\Omega(\gamma) := f_{\mathcal{G}}(\mathbf{x}(j) + \gamma \mathbf{d}(j))$. Then, since $f_{\mathcal{G}}$ is a quadratic, the only root of Ω' is γ_* , with

$$\gamma_* := [-\mathbf{d}(j)^T \nabla f_{\mathcal{G}}(\mathbf{x}(j))] [\mathbf{d}(j)^T \mathbf{Q} \mathbf{d}(j)]^{-1}$$

and $\delta := \Omega''(\gamma_*) = \mathbf{d}(j)^T \mathbf{Q} \mathbf{d}(j)$. Now if $\delta \neq 0$, the minimum $\gamma(j) = \arg \min_{\gamma \in [0, 1]} \Omega(\gamma)$ is given by

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

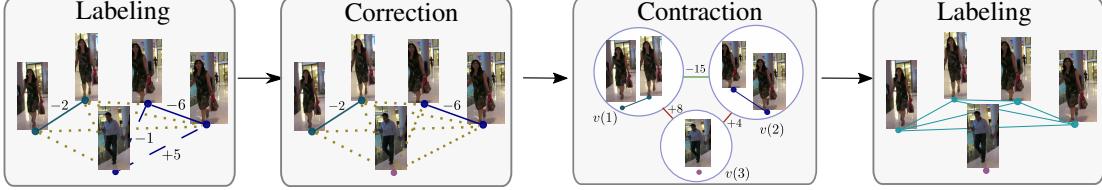


Figure 2. One iteration of our hierarchical solver. We present an illustrative example with pairwise costs on the edges. **Left:** A labeling, computed by Alg. 1. Dotted edges indicate removed links. Nodes connected by edges of the same color are grouped, resulting in two clusters. Dashed edges indicate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

A line search is needed only if $\delta = 0$, making the execution of Ln. 6 very efficient. In contrast to previous works [17, 35], 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 [33]. Given $r \neq 0$, our next proposed improvement is to replace the objective function $f_{\mathcal{G}}$ by

$$f_r(\mathbf{x}) = f_{\mathcal{G}}(\mathbf{x}) + r \sum_i (x_i^2 - x_i). \quad (14)$$

For $\mathbf{x} \in \{0, 1\}^{nP}$, we have $f_r(\mathbf{x}) = f_{\mathcal{G}}(\mathbf{x})$. Using $r < 0$ has the effect of pushing the FW algorithm towards discrete solutions, as $-(x_i^2 - x_i)$ has its minimum at 0 and 1, within $[0, 1]$. For $r > 0$, we observed better behavior in staying out of local optima, as for a value r sufficiently large, f_r becomes convex [8, 11, 24]. On the other hand, a high r value brings the optimal solution too close to the constant $(\frac{1}{2})$ vector. For $\omega := \max\{\sum_j |\mathbf{Q}_{i,j}| : i \in [nP]\}$, we set $r_0 = \sqrt{\omega}$ and $r_i = 2^{-i}r_0$. Starting with $r = r_0$, we compute $\text{QP}(\mathcal{G}, P)$, using f_r in Alg. 1. Empirically, we observed that a short number of iterations of Alg. 1 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 $i = i + 1$, $r = r_i$ and run Alg. 1 again with the updated function f_r . In all our experiments, an appropriate r was found in at most two calls of Alg. 1. In Sect. 4, we demonstrate the impact of using the modified cost function, with the solver we call **FW + r**.

2.5. Hierarchical Solving Scheme

Since $\text{FW} + r$ delivers only a local optimum, we propose a new hierarchical solving scheme that enhances the solution of $\text{FW} + r$ by removing, correcting and connecting clusters, thus resulting in an improved objective value. Our approach is computationally efficient and continues optimizing problem $\text{BQP}(\mathcal{G}, P)$. Compared to other hierarchical approaches like [27] that define specific parameter changes in each iteration, our formulation is generic and can

be applied to many clustering problems without the need of heuristically set parameter update rules.

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

CorrectionContraction: Let $\mathbf{x}^{(t)} \in \mathbb{R}^{nP}$ be the current best labeling of \mathcal{G} . Initially, we obtain $\mathbf{x}^{(0)}$ using $\text{FW} + r$. We apply a relabeling strategy that corrects obvious errors within the clusters that may have been introduced due to the rounding or local optimality. For $v \in \mathcal{V}$, let $\mathcal{N}(v, \mathbf{x}^{(t)})$ be the set of all adjacent nodes that have the same label as v . If $\sum_{u \in \mathcal{N}(v, \mathbf{x}^{(t)})} q_{u,v} > 0$, or, if $c_v > 0$ and $(\mathbf{x}^{(t)})_v^j = 0, \forall j \in [P]$, we assign a new and unique label to v (see Fig. 2 middle). Let $v(k)$ be the set comprising all nodes labeled k . We build a *contracted graph* $\mathcal{G}_{t+1} = (\mathcal{V}_{t+1}, \mathcal{E}_{t+1})$ by using these virtual, new nodes: We set $\mathcal{V}_{t+1} := \{v(k) \mid k \in [P]\}$ and \mathcal{E}_{t+1} connects any two different vertices. Accordingly, we obtain the stacked decisions variables $\mathbf{x}_{\text{contr}}$ for the current labeling of \mathcal{G}_{t+1} .

LabelExpand: Let the current labeling result in J clusters. To compute the optimal labeling on \mathcal{G}_{t+1} according to $\text{BQP}(\mathcal{G}_{t+1}, J)$, we define the unary costs

$$c_{v(k)} := \sum_{v \in v(k)} c_v + \sum_{\{u,v\} \in \mathcal{E} \cap v(k) \times v(k)} q_{u,v} \quad (15)$$

and pairwise costs

$$q_{v(k), v(k')} := \sum_{\{u,v\} \in \mathcal{E} \cap v(k) \times v(k')} q_{u,v}. \quad (16)$$

Consider the stacked decision variables $\hat{\mathbf{x}} \in \{0, 1\}^{JJ}$ where $\hat{x}_{v(k)}^s$ equals 1, if $s = k$ (and thus $(\mathbf{x}_{\text{contr}})_{v(k)}^s = 1$) and if $v(k)$ was not a rejected node by $\mathbf{x}^{(t)}$; and 0 otherwise. Then, $\hat{\mathbf{x}}$ assigns each node of \mathcal{G}_{t+1} a unique label, except for nodes that have been rejected by $\mathbf{x}^{(t)}$. Therefore, $f_{\mathcal{G}_{t+1}}(\hat{\mathbf{x}})$ sums up only the unary costs (15), which equal $f_{\mathcal{G}}(\mathbf{x}^{(t)})$ or are improved by the refinement, implying

$$f_{\mathcal{G}_{t+1}}(\hat{\mathbf{x}}) \leq f_{\mathcal{G}}(\mathbf{x}^{(t)}). \quad (17)$$

Furthermore, solving $\text{BQP}(\mathcal{G}_{t+1}, J)$ results in a solution

Algorithm 2: Hierarchical solving scheme

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

```

1 repeat
2    $f_{\min} = f_{\mathcal{G}}(\mathbf{x}^{(t)})$ ;
3    $(\mathcal{G}_{t+1}, \mathbf{x}_{\text{contr}}) = \text{CorrectionContraction}(\mathbf{x}^{(t)}, \mathcal{G})$ ;
4    $\mathbf{x}^{(t+1)} = \text{LabelExpand}(\mathcal{G}_{t+1}, \mathbf{x}_{\text{contr}})$ ;
5    $t = t + 1$ ;
6 until  $f_{\mathcal{G}}(\mathbf{x}^{(t)}) = f_{\min}$ ;
```

$\mathbf{x}_{\mathcal{H}} \in \mathcal{C}_b(\mathcal{G}_{t+1}, J)$ with

$$f_{\mathcal{G}_{t+1}}(\mathbf{x}_{\mathcal{H}}) \leq f_{\mathcal{G}_{t+1}}(\hat{\mathbf{x}}) \leq f_{\mathcal{G}}(\mathbf{x}^{(t)}). \quad (18)$$

The result $\mathbf{x}_{\mathcal{H}}$ is converted to a labeling $\mathbf{x}^{(t+1)} \in \mathcal{C}_b(\mathcal{G}, P)$ by *graph expansion*: All nodes $v \in v(k)$ are assigned the new label of $v(k)$, according to $\mathbf{x}_{\mathcal{H}}$, see also Fig. 2. Thus, the hierarchical step can improve the last solution, since

$$f_{\mathcal{G}}(\mathbf{x}^{(t+1)}) = f_{\mathcal{G}_{t+1}}(\mathbf{x}_{\mathcal{H}}) \leq f_{\mathcal{G}}(\mathbf{x}^{(t)}). \quad (19)$$

The graph contraction reduces the dimensionality significantly: There are $J \ll n$ nodes to be labeled using at most J labels, w.r.t. $\text{BQP}(\mathcal{G}_{t+1}, J)$. If J is small enough, we can solve $\text{BQP}(\mathcal{G}_{t+1}, J)$ quickly to optimality using Gurobi [23]. Otherwise we use the *FW+r* solver. The algorithm is stopped once no new clusters are merged. We demonstrate the effect of the hierarchical solving scheme **FW + r + h** in Sect. 4.

3. Regression Training

In the following, we introduce spatial and temporal costs, which describe how likely two detections within the same and between different frames belong to the same person, respectively. For each cost type, we train a logistic regression model to obtain weights θ , as described in Sect. 2.1.

For our tracking system, we consider two input sources: (i) head and (ii) full-body detections (see also Fig. 1).

Head detections. To obtain accurate head detections, we employ [55] based on Convolutional Neural Networks and fine-tune it on the MOT16 training set [43].

Full-body detections. We use the full-body detections [19] as provided by the MOT16 challenge [43].

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

To this end, consider a person detection box d with the positions of lower left, upper left and upper right corners $d_{\text{ll}}, d_{\text{ul}}$ and d_{ur} , respectively and $\Delta(d) := (d_{\text{ll}}, d_{\text{ul}}, d_{\text{ur}})^T$. For a pixel $\mathbf{p} \in \mathbb{R}^2$, we obtain barycentric coordinates $\lambda_d = (\lambda_1, \lambda_2, \lambda_3)^T$ of \mathbf{p} w.r.t. $\Delta(d)$, so that $\mathbf{p} = \lambda_d^T \Delta(d)$ (see Fig. 3). We fix a standard box d_{std} . Then, \mathbf{p} is mapped to

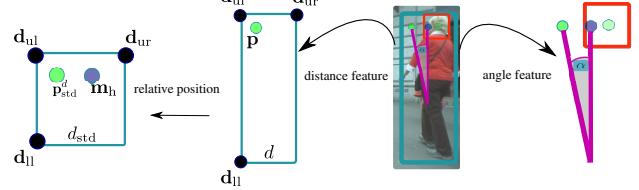


Figure 3. Distance and angle between the expected (blue node) and the observed (mirrored) head position (green node). Barycentric coordinates are computed w.r.t. black corners.

$\mathbf{p}_{\text{std}}^d := \lambda_d^T \Delta(d_{\text{std}})$, keeping the relative position as in d . Now, all subsequent distance measurements are computed using the mapped position w.r.t. d_{std} .

Spatial costs. 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 detection box d , resulting in the pixel \mathbf{p} , thereby making the position robust against different orientations of the person. From the MOT16 training data, we learned the expected relative position \mathbf{m}_h of a head w.r.t. the standard detection d_{std} , corresponding to a full-body detection of the same person. Finally, we obtain the feature $\|\mathbf{p}_{\text{std}}^d - \mathbf{m}_h\|_2$, measuring distance between the detected and expected position. We introduce a second feature which uses the angle between expected and detected position, with the anchor at the box's center (see Fig. 3).

We set the spatial costs between detections from the same detector to a constant high value.

Temporal costs. Temporal costs are defined via correspondences of pixels between two frames. DeepMatching [61] (DM) provides such assignments, which are more reliable than spatio-temporal affinities, see [57]. Given rectangles u and v , DM samples \mathbf{dm}_u and \mathbf{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 [57]. 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 ϕ_m^w and ϕ_m^h between a head and body detection, w.r.t. width and height, respectively, if both belong to the same person. Then, we obtain features $\|\phi_m^w - \phi_{\text{det}}^w\|_2$ and $\|\phi_m^h - \phi_{\text{det}}^h\|_2$, for the observed ratios ϕ_{det}^w and ϕ_{det}^h 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.

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
<i>FW</i>	16	0.7	-3060	14.2
<i>FW + r</i>	676	27	-5481	26.8
<i>FW + r + h</i>	-	27+0.5	-5925	27.5
Gurobi	-	1000	(-5531)	24.9
Gurobi bound	-	1000	(-5973)	-

Next, we investigate the impact of the detector fusion on the tracking performance, using the training sequences of the challenging MOT16 benchmark [43]. 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 [6, 41, 49].

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 P 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 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+r+h* improves the objective value considerably compared to the standard Frank-Wolfe algorithm *FW*. This naturally translates to almost 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, 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

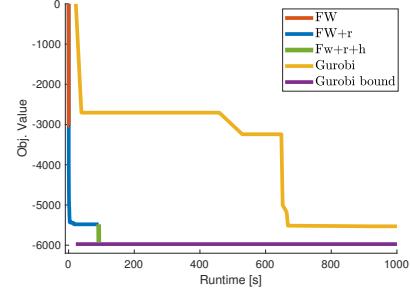


Figure 4. Minimization performance of each BQP solver and the bound as given by Gurobi for each moment.

Fig. 4. Here we clearly see where *FW* stops (red line), how our modification *FW + r* improves the energy by a large margin (blue line), and how finally *FW + r + h* (green line) comes even closer to the estimated lower bound (purple line), as provided by Gurobi. In contrast, Gurobi (yellow line) has a much slower convergence.

To separate the quality of our solver from the detections, we further evaluate the performance on ground-truth person detections for 40 frames of each MOT16 training sequence in Tab.4.2, where we also report the (relative) duality gap to the optimal solution (GAP). The results show a consistent and huge improvement by the hierarchical concept over *FW+r*. At the same time, the solutions are close to optimality w.r.t. to the objective value and w.r.t. to tracking performance. The sequences MOT16-05 and MOT16-11 both contain many partial occlusions that makes it difficult for the DM features to be correct in any situation, thus resulting in lower tracking scores. However this shows that a second type of detections (head detections) is necessary for high quality tracking results. On the other hand, the solver reaches the perfect result on MOT16-09 (which has far less occlusions), thereby justifying our solver.

Table 2. FW+r versus FW+r+h (on GT detections).

Seq	FW+r				FW+r+h				GAP	
	IDF1	ID	FM	MOTA	GAP	IDF1	ID	FM	MOTA	
02	87.4	5	1	84.0	6.424	90.9	3	0	90.8	0.428
04	85.0	5	0	73.2	7.506	92.4	0	0	85.8	0.120
05	57.4	10	8	74.2	9.130	70.1	8	7	75.1	0.071
09	80.6	3	0	98.9	5.353	100.0	0	0	100.0	0.000
10	82.0	10	6	80.4	7.410	87.0	7	6	89.4	0.638
11	76.8	13	2	78.2	12.846	89.4	5	3	96.3	0.084
13	87.2	10	2	85.3	10.332	96.3	2	3	96.9	0.434

4.3. Ablation studies on head and body detections

We analyze how our formulation exploits the information from two detectors. For this experiment, we use all MOT16 training sequences with the full-body detections only (*B*) against body and head detections (*B+H*). We use the body detections provided by the benchmark while we train the head detector and the regression model on MOT16 training sequences in a leave-one-out fashion.

In Tab. 3, we report several ablation results with: (i) different inputs (body and heads) and (ii) different solvers, namely our proposed FW+r+h (*Ours*) is compared to: tracking heads and bodies independently and then using our solver to fuse them. The tracklets are computed by our system (*Ours-fusion*) and from LP2D [39] (LP2D-fusion). We use the affinities as defined in Sect.3, but set the spatial and temporal costs between two tracklets that originate from the same detector to a constant high value, as the tracklets are already separating the persons (Sect.3*). We further provide the quality of the head trajectories, which we evaluated on the head ground truth boxes.

Table 3. Ablation experiments on MOT16 training sequences.

Feature	Affinities	Solver	SolverID	MOTA	MT	FP	FN	IDs
H	2D dist	<i>LP2D</i>	1	14.9	70	14829	50991	472
H	Sect.3	<i>Ours</i>	2	16	70	14168	50959	331
B	2D dist	<i>LP2D</i>	3	31.7	44	3557	71332	467
B	Sect.3	<i>Ours</i>	4	33.0	76	11949	61603	378
B	[16]	GMMCP [16]	5	33.7	46	4053	68675	499
B+H	Sect.3*	<i>LP2D – fusion</i>	6	33.0	54	3501	70163	358
B+H	Sect.3*	<i>Ours – fusion</i>	7	34.2	87	11852	60401	376
B+H	Sect.3	<i>FW</i>	8	31.1	75	5315	69563	1207
B+H	Sect.3	<i>FW + r</i>	9	33.4	82	6497	66238	807
B+H	Sect.3	<i>Ours</i>	10	38.2	86	4972	62935	372
B+H	Sect.3	NLLMPa [40]	11	37.4	86	4954	63831	336

Our system performs comparable on full-body detections to the SolverID 5, using their defined affinities. By using the two detectors, our system significantly improves almost all relevant tracking metrics, justifying our tracking framework (SolverID 10 vs 4). 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 5pp (percentage points). Performing the fusion directly on the input detections is clearly more effective than using initial tracklets. SolverID 6 and 7 use our solver and precomputed trajectories from SolverID 3 and 4, where the gain is no more than 1.3pp, justifying our fusion concept. Using another heuristic solver [40] (SolverID 11) performed worse on the fusion than FW+r+h, using exactly the same graph. The comparison SolverID 8-10 show the improvement on MOT16train due to the regularizer and the hierarchical step (up to 7.1 pp on the MOTA score).

4.4. Benchmark Evaluation

We evaluate the tracking performance of our formulation with body and heads on the benchmarks *MOT16* and *MOT17* with the full-body detections provided by the benchmarks. Due to space constraints, we show some of the best performing published trackers in Tab. 4, as well as the worst performing tracker. For the full table of results, please visit the benchmark’s website.

Our system creates slightly higher identity switches. This can be resolved in future work with more advanced features that include a foreground/background mask in each

detection or in a post-processing step where tracklet consistency is checked, though this is beyond the scope of this paper. However, our proposed tracker performs on par with state-of-the-art in terms of tracking accuracy on *MOT16* and sets a new state-of-the-art on *MOT17*. Furthermore, the tracker won, together with [30], the MOT 2017 Tracking challenge at the CVPR 2017¹. Note that the MOTA metric is regarded as the most representative metric [38]. With our proposed formulation, we have the lowest ML (mostly lost) score within all trackers in both benchmarks, showing 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 recover very long trajectories. In contrast, the GMMCP model approach is not able to produce long-term consistent trajectories possibly due to erroneous initial tracklets, that could not be connected (we used the official code of [16] to produce the results). We note that the LMP tracker uses very advanced and stable convolutional neural network image features that can reliably link boxes over 200 image frames, thus resulting in a better MOTA score.

5. Conclusion

We presented a global formulation for multi-detector multi-target tracking, and showed its state-of-the-art performance with head and full-body detectors. We proposed to cast the problem 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 challenging tracking benchmarks showed state-of-the-art results.

¹https://motchallenge.net/MOT17_results_2017_07_26.html

Table 4. Public tracking results on *MOT16* and *MOT17*.

Method	Rank	MOTA↑	IDFI↑	MOT16		FP↓	FN↓	ID↓
				MT↑	ML↓			
MOT17								
LMP [58]	1	48.8	51.3	18.2	40.1	6654	86245	481
Ours	2	47.8	47.8	19.1	38.2	8886	85487	852
NLLMPa [40]	3	47.6	47.3	17.0	40.4	5844	89093	629
AMIR [50]	4	47.2	46.3	14.0	41.6	2681	92856	774
NOMT [15]	5	46.4	53.3	18.3	41.4	9753	87565	359
GMMCP [16]	15	38.1	35.5	8.6	50.9	6607	105315	937
DP_NMS [47]	23	26.6	31.2	4.1	67.5	3689	130557	365
MOT17								
Ours	1	51.3	47.6	21.4	35.2	24101	247921	2648
MHT.DAM [31]	2	50.7	47.2	20.8	36.9	22875	252889	2314
EDMT17 [13]	3	50.0	51.3	21.6	36.3	32279	247297	2264
GMMPHD.KCF [32]	6	30.5	35.7	9.6	41.8	107802	277542	6774

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] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.
- [4] B. Benfold and I. Reid. Stable multi-target tracking in real-time surveillance video. In *CVPR*, 2011.
- [5] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *TPAMI*, 2011.
- [6] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008.
- [7] D. P. Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999.
- [8] A. Billionnet, S. Elloumi, and A. Lambert. Extending the qcr method to general mixed-integer programs. *Mathematical programming*, 131(1):381–401, 2012.
- [9] W. Brendel, M. Amer, and S. Todorovic. Multiobject tracking as maximum weight independent set. In *CVPR*, 2011.
- [10] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010.
- [11] 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.
- [12] V. Chari, S. Lacoste-Julien, I. Laptev, and J. Sivic. On pairwise costs for network flow multi-object tracking. In *CVPR*, 2015.
- [13] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong. Enhancing detection model for multiple hypothesis tracking. In *CVPRW*, 2017.
- [14] S. Chen, A. Fern, and S. Todorovic. Multi-object tracking via constrained sequential labeling. In *CVPR*, 2014.
- [15] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*, 2015.
- [16] A. Dehghan, S. Assari, and M. Shah. Gmmcp-tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In *CVPR*, 2015.
- [17] A. Dehghan and M. Shah. Binary Quadratic Programming for Online Tracking of Hundreds of People in Extremely Crowded Scenes. *TPAMI*, 2017.
- [18] P. Dollar, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *TPAMI*, 2014.
- [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] R. Henschel, L. Leal-Taixé, B. Rosenhahn, and K. Schindler. Tracking with multi-level features. *arXiv preprint arXiv:1607.07304*, 2016.
- [27] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *European Conference on Computer Vision*, pages 788–801. Springer, 2008.
- [28] H. Jiang, S. Fels, and J. Little. A linear programming approach for multiple object tracking. In *CVPR*, 2007.
- [29] A. Joulin, K. Tang, and F. F. Li. Efficient image and video co-localization with frank-wolfe algorithm. In *ECCV*, 2014.
- [30] M. Keuper, S. Tang, Y. Zhongjie, B. Andres, T. Brox, and B. Schiele. A multi-cut formulation for joint segmentation and tracking of multiple objects. *arXiv preprint arXiv:1607.06317*, 2016.
- [31] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited: Blending in modern appearance model. In *ICCV*, 2015.
- [32] T. Kutschbach, E. Bochinski, V. Eiselein, and T. Sikora. Sequential sensor fusion combining probability hypothesis density and kernelized correlation filters for multi-object tracking in video data. *IEEE AVSS Workshop*, 2017.
- [33] S. Lacoste-Julien. Convergence rate of frank-wolfe for non-convex objectives. *arXiv preprint arXiv:1607.00345*, 2016.
- [34] S. Lacoste-Julien and M. Jaggi. On the global linear convergence of frank-wolfe optimization variants. *NIPS*, 2015.
- [35] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher. Block-Coordinate Frank-Wolfe Optimization for Structural SVMs. In *ICML*, 2013.
- [36] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society*, 1960.
- [37] 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.
- [38] L. Leal-Taixé, A. Milan, K. Schindler, D. Cremers, I. Reid, and S. Roth. Tracking the trackers: an analysis of the state of the art in multiple object tracking. *arXiv preprint arXiv:1704.02781*, 2017.
- [39] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. *ICCV. 1st Workshop on Modeling, Simulation and Visual Analysis of Large Crowds*, 2011.
- [40] 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.
- [41] Y. Li, C. Huang, and R. Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *CVPR*, 2009.
- [42] B. D. Lucas, T. Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.
- [43] 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.
- [44] A. Milan, L. Leal-Taixé, K. Schindler, and I. Reid. Joint tracking and segmentation of multiple targets. In *CVPR*, 2015.
- [45] J. E. Mitchell. Branch-and-cut algorithms for combinatorial optimization problems. *Handbook of applied optimization*, pages 65–77, 2002.
- [46] 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.
- [47] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011.
- [48] S. Ren, R. G. K. He, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*, 2015.
- [49] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV*, 2016.
- [50] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *ICCV*, 2017.
- [51] S. Sahni. Computationally related problems. *SIAM Journal on Computing*, 1974.
- [52] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.
- [53] G. Seguin, P. Bojanowski, R. Lajugie, and I. Laptev. Instance-level video segmentation from object tracks. In *CVPR*, 2016.
- [54] F. Solera, S. Calderara, and R. Cucchiara. Learning to divide and conquer for online multi-target tracking. In *CVPR*, 2015.
- [55] R. Stewart, M. Andriluka, and A. Y. Ng. End-to-end people detection in crowded scenes. In *CVPR*, 2016.
- [56] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph decomposition for multi-target tracking. In *CVPR*, 2015.
- [57] 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.
- [58] S. Tang, M. Andriluka, B. Andres, and B. Schiele. Multiple people tracking by lifted multicut and person reidentification. In *CVPR*, 2017.
- [59] Y. T. Tesfaye, E. Zemene, M. Pelillo, and A. Prati. Multi-object tracking using dominant sets. *IET computer vision*, 10(4):289–297, 2016.
- [60] C. Tomasi and T. Kanade. Detection and tracking of point features. 1991.
- [61] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *ICCV*, 2013.
- [62] 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.
- [63] A. Zamir, A. Dehghan, and M. Shah. Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs. In *ECCV*, 2012.
- [64] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.