

# Tracking with multi-level features

Roberto Henschel, Laura Leal-Taixé, Bodo Rosenhahn, Konrad Schindler

**Abstract**—We present a novel formulation of the multiple object tracking problem which integrates low and mid-level features. In particular, we formulate the tracking problem as a quadratic program coupling detections and dense point trajectories. Due to the computational complexity of the initial QP, we propose an approximation by two auxiliary problems, a temporal and spatial association, where the temporal subproblem can be efficiently solved by a linear program and the spatial association by a clustering algorithm. The objective function of the QP is used in order to find the optimal number of clusters, where each cluster ideally represents one person. Evaluation is provided for multiple scenarios, showing the superiority of our method with respect to classic tracking-by-detection methods and also other methods that greedily integrate low-level features.

**Index Terms**—Multiple People Tracking, Feature Integration, Quadratic cost function, Linear Programming, Spectral Clustering



## 1 INTRODUCTION

DETECTING and tracking people in a video is an important task of computer vision, and its output is used in many applications such as autonomous driving, video surveillance or activity recognition. A common approach to recover 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 [1], [2], [3]. But, at the same time, the dependence on detection results – typically bounding boxes – is also a main limitation. State-of-the-art object detectors [4], [5], [6] perform well in not too crowded environments, but they still consistently fail in the presence of significant occlusions.

Although “connecting the dots” supplied by a pedestrian detector is convenient, a lot of potentially important information is lost along the way. In particular, the tracker does not use the actual image data, except sometimes in the form of rather weak appearance models to discriminate different people. Recently, a number of approaches have proposed to step away from the standard paradigm [7], [8], [9] and instead tackle the tracking problem more directly, going straight from low-level image cues to trajectories. Instead of using detections they base their processing on low- or mid-level information such as dense point tracks [10] or space-time supervoxels [11], thus also moving closer to the related problem of motion segmentation.

In this paper, we propose a principled *global formulation to integrate low- and mid-level features for multi-target tracking*. We cast the coupled problems of (i) temporally

linking features into feature tracks, (ii) grouping them into individual moving targets (persons), as one single quadratic program (QP) with linear constraints. Solving the QP directly is computationally and memory infeasible, therefore we approximate the problem by solving two subproblems: The temporal linking can be solved efficiently and optimally by linear programming relaxation, whereas the grouping of features into individual persons is found with spectral clustering. In this paper, we use detections as mid-level features and dense point tracklets (DPTs) as low-level features, since both are commonly used together in the literature, and therefore we can directly compare with state-of-the-art methods. Low-level features generally provide very accurate motion information, while mid-level features like detections provide the necessary structure information. To summarize, the **contribution** of the present paper is three-fold:

- We propose a global formulation for the integration of mid- and low-level features for multi target tracking. The problem is cast as a quadratic program (QP) with linear constraints, coupling the problems of (i) linking features into tracks and (ii) grouping them into individual moving targets (persons).
- Given that the initial QP is NP-hard and computationally too expensive to solve, we propose an approximation using a decomposition into an efficient linear program and a clustering step.
- We propose to use the QP objective function to robustly determine the number of clusters, which is always a delicate step in other approaches.

### 1.1 Related work

**Tracking-by-detection.** Multiple people tracking is a key problem for tasks such as surveillance or activity recognition. Tracking-by-detection has become the standard way of solving it. The problem is split into two steps: object detection and data association. In crowded environments, where occlusions are common, even state-of-the-art detectors [4], [12], [13] suffer from many false alarms and/or missed

- R. Henschel and B. Rosenhahn are with the TNT group at Leibniz University of Hannover, Germany.  
E-mail: {henschel,rosenhahn}@tnt.uni-hannover.de
- L.Leal-Taixé is with Computer Vision Group at Technical University Munich, Germany.  
E-mail: leal.taixe@tum.de
- K.Schindler is with the Photogrammetry and Remote Sensing Group, ETH Zürich, Switzerland.  
E-mail: konrad.schindler@geod.baug.ethz.ch

Manuscript received July, 2016

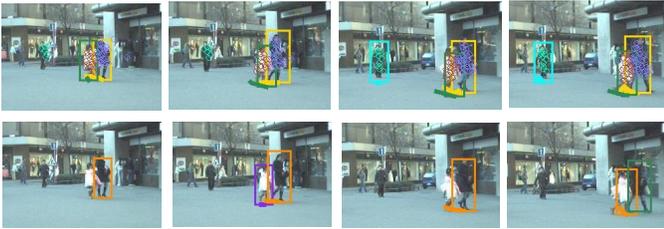


Fig. 1. Tracking results on ETH-Jelmoli over 18 frames. *Top row*: The proposed TbX tracker is able to generate stable trajectories. *Bottom row*: LP2D is unable to track both pedestrians consistently, creating ID switches and missed detections.

detections. The focus of data association is to overcome these detection failures, by filtering out false alarms and filling in the gaps due to false negatives. While the data association step can be solved on a frame-by-frame basis [14], or one track at a time [15], recent work has shown that it is beneficial to jointly solve it over all tracks and all frames. This is usually done in discrete space, using Linear Programming (LP) [16], [17], [18], [19] or graph-theoretic methods like generalized clique graphs [20] or maximum weight-independent sets [21], although there are also continuous formulations [2]. Most of these methods can be solved optimally, though some not efficiently, as they require the use of expensive decomposition methods [22], [23].

All the aforementioned works are tracking-by-detection methods and rely on a fixed set of detections as input. This has the drawback that much of the image information is discarded during the non-maxima suppression step built into any detector, potentially ignoring semi-occluded objects. There have been some attempts to include additional evidence, by starting from weaker candidate detections and coupling detection and trajectory estimation [24], using individual part responses of DPM [25], creating dedicated detector for occluded people [26], or learning frequent occlusion patterns [27]. Still, the basic problem remains, namely that the tracker is completely dependent on the detector output, and has no access to the potentially much richer image data. We argue that tracking should be based on both mid-level features, such as a detector output, and low-level image information.

**Tracking from low-level features.** In the recent literature, several works have started incorporating low-level 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 [9], supervoxels are labelled according to target identities with greedy propagation, starting from manually initialised segmentation masks. Greedy propagation tends to fail in crowded scenarios, leading to long trajectories that often switch from person to person or from person to background. Furthermore, the method needs manual initial segmentation masks. In [7], supervoxel labeling is formulated as CRF inference, where the targets are modelled as volumetric “tubes” through the sequence. Here, we propose to assemble the tracking solution from the linking and clustering problem which is much more efficient, and can integrate motion information over much

longer time windows than [7]. In [28], the fusion of head and people detections to improve tracking performance is discussed. To this end, a quadratic program is used to model non-maxima suppression as well as a simple overlap consistency between the different features. In contrast to our method, their model is designed to consider co-occurrences of active features only. Our formulation directly models the grouping of features to different persons, which is more appropriate for the tracking task, allowing to ensure consistency within each cluster. Note, that the input size of features and number of constraints in [28] is much smaller, allowing them to solve the relaxed version of the quadratic problem. Since we leverage DPTs, it is computationally not feasible to use their solver. Also in the extension [29] to motion segmentation using superpixels, the per-person consistency is not considered. There are several works that use dense point tracks or KLT tracks together with detections to improve tracking performance. Close to our work is [30], where authors use the KLT feature tracker [31] as motion model to guide detection tracks frame-by-frame. The main difference to our method is that the treatment of both features is different, since each one has a specific purpose within their tracking framework. In contrast, we propose a holistic formulation, where both feature types can equally influence trajectory extraction. Let us consider the case where a bounding box is wrongly surrounding two people. In [30], the KLT features would just provide conflicting information, but would not be able to generate two trajectories out of one bounding box at that point in time. In contrast, the DPTs in time would be clustered into two different clusters, therefore creating two trajectories. Another related work is that of [32], where multi-target tracking is formulated as a clustering of dense feature tracks [10]. This method suffers from two main problems: (i) the automatic choice of the number of clusters on the basis of the eigengap is unstable, and (ii) individual tracks are typically very short on moving objects and have a low temporal overlap among each other, which destabilises the clustering and makes it impossible to recover from occlusion. Follow up work was presented in [8], where dense point tracklets are combined with detection-based tracklets in a two-step approach. In contrast, we propose a global optimization formulation to integrate low- and mid-level features, so as to take full advantage of the strength of both. The fact that we link the dense point tracklets across time to obtain longer, but nevertheless reliable tracks, makes it possible for us to track even through occlusions, unlike [8]. Recently, [33] used interest point trajectories to create a more accurate affinity measure to associate detections.

## 2 TRACKING WITH MULTI-LEVEL FEATURES

In this section, we present a formulation for the multi-target tracking problem which uses as input not only mid-level features such as detections, but also low-level features, e.g. dense point tracklets (DPTs) [10]. Our goal is to define a common framework in which both features can be coupled to obtain a single set of trajectories that leverages the information of both feature channels. Dense point tracks [10] are capable of following a pedestrian even under partial

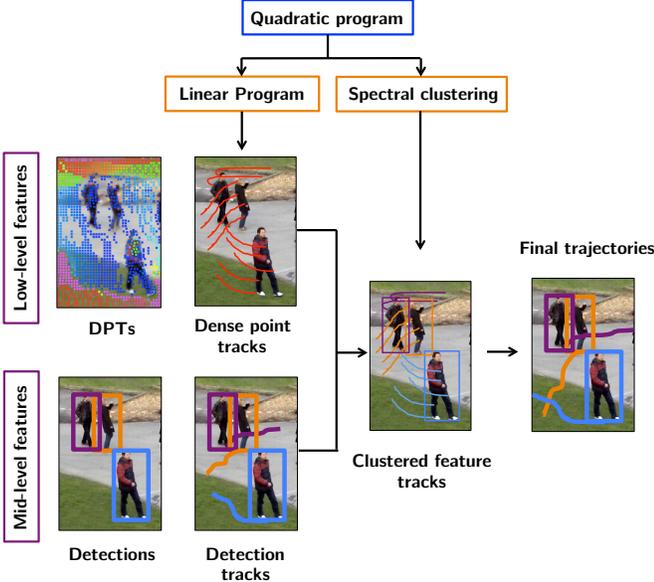


Fig. 2. Diagram of our approach.

occlusion, while detections give us the necessary structured information to distinguish pedestrians walking with a similar motion. Fig. 1 demonstrates the advantage of this concept.

We formulate the multi-target tracking problem as a quadratic program (QP) where we couple: (i) temporal association of low- and mid-level features into *feature tracks*, (ii) spatial clustering of low-level features in accordance to the structure provided by the mid-level features (ideally one cluster per person). Since such QP is NP-hard, we propose a decomposition that allows to find an approximate solution much more efficiently.

## 2.1 Temporal association with linear programming

We first focus on the temporal association of features into feature tracks. Let  $\mathfrak{M}_c = \{m_i^c\}$  be a set of features of a category  $c$ . By way of example, we propose to use  $c \in \mathcal{C} = \{\text{low}, \text{mid}\}$ . For DPTs, i.e.  $c = \text{low}$ ,  $m_i^c = (\mathbf{p}_i^t, \mathbf{a}_i^t)$ . At time  $t = t_{ij}$ ,  $\mathbf{p}_i^t = (x, y)^T$  denotes its 2D image position,  $\mathbf{a}_i^t$  the mean color value of a patch around  $\mathbf{p}_i^t$ , respectively. For  $c = \text{mid}$ , or detections in our case, features are defined as  $m_j^c = (\mathbf{p}_j^t, w_j, h_j, t_j)$ . At time  $t = t_j$ , detection  $j$  is centered at  $\mathbf{p}_j^t = (x, y)^T$  and has width  $w_j$  and height  $h_j$ .

A feature track (of category  $c$ ) is defined as an ordered list  $T_n^c \subset \mathfrak{M}_c$  of features, sorted in time and without a temporal overlap. Our goal is now to find the set of tracks  $\mathcal{T}^{c*} = \{T_n^c\}$  that best explain the feature evidence.

This can be formulated as a minimization of the following objective function:

$$\begin{aligned} \mathcal{T}^{c*} &= \underset{\mathbf{f}_{\text{LP}}^c}{\text{argmin}} \quad c_{\text{LP}}^c \mathbf{f}_{\text{LP}}^c \\ &= \underset{\mathbf{f}_{\text{LP}}^c}{\text{argmin}} \sum_i c_{\text{in}}^c(i) f_{\text{in}}^c(i) + \sum_i c_{\text{out}}^c(i) f_{\text{out}}^c(i) \\ &\quad + \sum_i c_d^c(i) f_d^c(i) + \sum_{i,j} c_t^c(i,j) f_t^c(i,j) \end{aligned} \quad (1)$$

subject to edge capacity constraints, flow conservation at the nodes (3b), (3c) and exclusion constraints. See [17] for further details.

The flags  $\mathbf{f}_{\text{LP}}^c$  take values in  $\{0, 1\}$ , indicating whether a particular feature connection is taken into the solution ( $f = 1$ ) or not ( $f = 0$ ). The start/end costs  $c_{\text{in}}^c$  and  $c_{\text{out}}^c$  define how probable it is for a track to start or end. These are learned from training data and kept the same for all the experiments in Section 4. For detections,  $c_d^{\text{mid}}$  will be proportional to the score given by the detector, so that only confident detections will be matched into tracks. For DPTs, on the other hand, we constraint  $f_d^{\text{low}} = 1$  and set the costs  $c_d^{\text{low}} = 0$ , so that they will all be matched.

The cost of a link edge  $c_t^c(i, j)$  measures the affinity between features  $m_i^c, m_j^c$ , based on motion, appearance (in case of DPTs; for detections, using appearance is not beneficial) and temporal separation:

$$c_t^c(i, j) = \frac{\|\mathbf{p}_j^{t+\Delta t} - \mathbf{p}_i^t\|}{V_{\text{max}} \Delta t} + \frac{\|\mathbf{a}_j^{t+\Delta t} - \mathbf{a}_i^t\|}{A_{\text{max}}} + \frac{\Delta t}{F_{\text{max}}} \quad (2)$$

where  $V_{\text{max}}$  is the maximum speed of a pedestrian in pixels,  $A_{\text{max}}$  is the maximum appearance distance we allow, and  $F_{\text{max}}$  is the maximum time gap. Eq. (1) is the classic integer linear programming (ILP) formulation, which has been extensively used in multiple object tracking with detections as features. After relaxation to a linear program it can be efficiently solved using Simplex [17] or  $k$ -shortest paths [19].

## 2.2 Quadratic terms for spatial association

For detections, the formulation of Eq. (1) already provides suitable pedestrian trajectories. Nonetheless, since we also work with low-level features such as DPTs, a straightforward application of Eq. (1) would yield many DPTs per pedestrian. We therefore need to impose further constraints, so as to obtain larger clusters of coherent tracks which coincide with the detections. Roughly speaking, we want to combine DPT tracks which are close to each other and run in parallel, meaning they likely follow the same person. At the same time, we want to allow two DPT tracks to belong to different clusters, if they only approach each other for a short period of time, e.g., when two people cross. Note, that it is in this step where the different features are coupled. We encode these conditions in the form of a quadratic term in the objective function, and an extra set of linear constraints.

The goal now is to find the set of full trajectories  $\mathcal{S}^* = \{S_m\}$  which cluster tracks  $T$  into pedestrians. This is expressed by the following quadratic problem:

$$\mathcal{S}^* = \underset{\mathbf{f}}{\text{argmin}} \quad \mathbf{c}^T \mathbf{f} + \mathbf{f}^T \mathbf{Q} \mathbf{f}, \quad (3a)$$

subject to

$$f_{\text{in}}^c(i) + \sum_j f_t^c(j, i) = f_d^c(i) \quad \forall c \in \mathcal{C} \quad (3b)$$

$$f_d^c(i) = f_{\text{out}}^c(i) + \sum_j f_t^c(i, j) \quad \forall c \in \mathcal{C} \quad (3c)$$

$$\sum_{k=1 \dots N_{cl}} f_{\text{QP}}(i, k) = f_d^c(i) \quad \forall c \in \mathcal{C} \quad (3d)$$

$$f_t^c(i, j) + f_{\text{QP}}(i, k) - f_{\text{QP}}(j, k) \leq 1 \quad \forall c \in \mathcal{C} \quad (3e)$$

$$f_t^c(i, j) - f_{\text{QP}}(i, k) + f_{\text{QP}}(j, k) \leq 1 \quad \forall c \in \mathcal{C}. \quad (3f)$$

The aforementioned constraints are created for all indices  $i, j$  in the corresponding index sets, which we omitted for clarity. Let  $d_{LP}$  and  $d_{QP}$  define the number of linear and quadratic costs, respectively. The flag vector  $\mathbf{f}$  is decomposed into the flags  $\mathbf{f}_{LP}^{[c]} \in \{0, 1\}^{d_{LP}}$  for the temporal association and  $\mathbf{f}_{QP} \in \{0, 1\}^{d_{QP}}$  for the spatial association, so  $\mathbf{f} = \begin{pmatrix} \mathbf{f}_{LP}^{[c]} \\ \mathbf{f}_{QP} \end{pmatrix}^T$ . Hereby we fixed an order  $\{c_1, \dots, c_{|c|}\} = \mathcal{C}$  and set  $\mathbf{f}^{[c]} := (\mathbf{f}^{c_1}, \dots, \mathbf{f}^{c_{|c|}})$ .

Using the costs defined in (1), we set  $\mathbf{c} = \begin{pmatrix} \mathbf{c}_{LP}^{[c]} & \mathbf{0} \end{pmatrix}^T$ . Now  $\mathbf{Q}$  provides the costs for two features' spatial compatibility. We derive it from an affinity matrix  $\mathbf{W}_{\text{Spatial}}$  that is described in detail in Sec. 3. Given  $\mathbf{W}_{\text{Spatial}}$ , we transform it into  $\mathbf{Q}_{\text{Spatial}} := -2\mathbf{W}_{\text{Spatial}} + \mathbf{1}$ , so that  $Q_{\text{Spatial}}(i, j) \in [-1, 1]$ , for all  $i, j$ . Finally, we construct the complete quadratic cost matrix via

$$\mathbf{Q} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_{\text{Spatial}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{Q}_{\text{Spatial}} \end{pmatrix}.$$

Note that  $Q(i, j) = 0$  for all  $i, j \leq d_{LP}$ . Furthermore, we have  $N_{cl}$  copies of  $\mathbf{Q}_{\text{Spatial}}$  on the diagonal,  $N_{cl}$  is an upper bound on the number of clusters. Accordingly, we have  $\mathbf{f}_{QP} \in \{0, 1\}^{N_{cl}F^2}$ , where  $F$  denotes the number of all feature tracks.

Now, for each  $k \in \{0, \dots, N_{cl} - 1\}$  and  $i, j \in \{kF + 1, \dots, (k + 1)F\} + d_{LP}$ , the entry  $Q(i, j)$  is the cost of assigning both nodes  $i$  and  $j$  to cluster  $k$ .

**Coupling LP and QP terms through the constraints.** The constraint (3d) guarantees that each active tracklet  $i$  (i.e.,  $f_d^c(i) = 1$ ) is assigned to exactly one cluster. Whenever tracklet  $i$  and  $j$  both appear in the same cluster  $k$ , the variables  $f_{QP}(i, k)$  and  $f_{QP}(j, k)$  will both be active, and the cost  $Q(i, j)$  will be applied. If only one of the tracklets is assigned to the cluster, e.g.  $f_{QP}(i, k) = 0$  and  $f_{QP}(j, k) = 1$ , the resulting cost will be zero. This definition of costs will drive the optimization towards a solution which is temporally consistent according to the temporal costs in  $\mathbf{c}^T \mathbf{f}$ , and at the same time will try to cluster similar tracklets, which are likely to belong to the same moving object.

Eqs. (3e) and (3f), on the other hand, guarantee the temporal consistency of the clusters. That is, if tracklets  $i$  and  $j$  are linked in time by edge  $f_t^c(i, j)$ , then both tracklets belong to the same cluster  $k$ . Note that the linear and quadratic terms are coupled through the three constraints (3d), (3e), and (3f).

**Final trajectories.** Finally we create a pedestrian trajectory  $S_m$  for each cluster if it contains features from both categories (detections and DPTs). DPTs on the background are grouped into outlier clusters (since no detections are associated with them) which are then removed. Details are described in Sec. 3.3. Conceptually, DPTs help to distinguish multiple people (with different motion) inside the same bounding box and detections help in distinguishing people who have the same motion pattern but are inside different detections. Due to the constraints (3d)-(3f) consistency between the feature categories is ensured.

## 2.3 Approximation of the problem

The program defined in Eq. 3 has one practical drawback: as a quadratic program it is NP-hard. We can solve it with branch-and-bound, but this is computationally inefficient and unfeasible even if we process a sequence in small batches. Alternatively, one can convert each quadratic constraint into three linear constraints [34], bringing the program to linear form. However, the corresponding constraint matrix is not totally unimodular (unlike the constraints of the LP (1)), which means that its LP-relaxation is not tight, and one is again faced with a NP-hard ILP. One can still relax the integrality constraint, solve the problem, and later apply a rounding scheme as in [35]. Nonetheless, even in that case, the initial linearized problem has millions of constraints even if we track only few pedestrians in few frames, which means finding a solution is computationally infeasible. Finally, one could resort to decomposition methods like dual decomposition [34] or Dantzig-Wolfe [22], but overall the problem remains unfeasible memory and computation-wise.

**Proposed solution.** We propose to take advantage of the particular structure of our program and divide the optimization into two steps as shown in Fig. 2. In the first step, we solve the initial LP of Eq. (1), which will result in a set of low- and mid-level feature tracks  $\mathcal{T}^c = \{T_n^c\}$ .

In a second step, we approximate the minimization of the quadratic part of the cost function (3), given the linked tracklets by the linear solution. This part can be considered as a correlation clustering problem [36], which we tried to solve directly. However, the solvers did not terminate even after days of computation, even though applicability on a large scale is suggested [37].

Since correlation clustering seems to be too expensive for our task, we use a sample-based approach. We generate clusters using spectral clustering [38], [39], which is known to provide good quality results, and evaluate it using the quadratic cost function defined by  $\mathbf{Q}$ .

A key aspect of most clustering methods is the choice of the number of clusters  $k$ , which varies from one problem instance to the next, and thus needs to be determined in a data-driven fashion. We propose to choose the best  $k$  by computing the set of trajectories for several values of  $k$ , and selecting the best of these candidate sets, based on the cost (3) of the full quadratic model. The function (3) describes our complete tracking problem, and is thus more suitable as a quality metric than more heuristic measures for the goodness of a given clustering. In fact, the proposed strategy bears some similarity with "re-ranking" strategies in the field of recognition and detection, which also replace a sophisticated, but computationally costly model with a simpler proxy to obtain a list of candidate solutions, and then rescore those candidates with the full model. In our case decomposing the problem has a further advantage: once feature tracks with high temporal overlap have been found, one can easily extract rich longer-term motion descriptors and consequently obtain a better clustering than with only short feature tracklets [10] or single detections. Using that information is intractable in the integrated model (3), since it would introduce a huge (combinatorial) number of higher-

order terms.

### 3 CLUSTERING

Using the temporal associations from the decomposed linear program, the tracking formulation (3) simplifies to solving the quadratic costs given the constraints (3d)-(3f). This however can be seen as a clustering problem that is well-approximated by standard clustering algorithms, e.g., normalized cuts [38], [39], where the number of clusters  $k$  has to be known a priori. The current state-of-the-art tracking solution [8] that follows a similar clustering approach, is lacking of a good strategy to compute this parameter. A heuristic approach is used, namely several normalized cuts are computed with varying parameters and combine various clusters from different n-cut results to construct the final clusters.

In contrast, we follow a systematic approach with a reduced search space compared to [8]. We propose a computationally reasonable approach to find a proper  $k$ , that is derived directly from our problem formulation (3), thus providing a rational parameter inference. Moreover, the number of detection tracklets can be used as a good initial guess for the parameter  $k$ . Ideally, each cluster will correspond to a trajectory, composed of several low-level features which will group into the structure provided by the mid-level features. In the following, we discuss our solution in more detail.

#### 3.1 Defining the distance matrix

Between any two features tracks  $T_i$  and  $T_j$ , we compute an affinity  $W_{\text{Spatial}}(i, j) \in [0, 1]$  according to their motion and spatial consensus. While [8] takes detection tracks as tracking units, we keep their temporal association as a soft constraint, thereby avoiding error propagation from the feature tracks obtained by the LP formulation, i.e. the clustering can implicitly compensate from temporal linking errors.

There are three different types of affinities that we consider:  $\mathbf{W}_{\text{PP}}$  between dense point tracks (detection overlap, distance, speed and angle),  $\mathbf{W}_{\text{PD}}$  between dense point tracks and detection tracks (intersection, distance, speed, angle) and  $\mathbf{W}_{\text{DD}}$  between detections tracks (intersection and DPT overlap). Note that in consistence with the temporal linking, we use only those detections that have not already been removed by the LP in Eq. (1). The entire affinity matrix is then given by

$$\mathbf{W}_{\text{Spatial}} = \begin{pmatrix} \mathbf{W}_{\text{PP}} & \mathbf{W}_{\text{PD}} \\ \mathbf{W}_{\text{PD}}^T & \mathbf{W}_{\text{DD}} \end{pmatrix}. \quad (4)$$

It is worth mentioning that  $\mathbf{W}_{\text{Spatial}}$  can be easily expanded to include more feature categories in a similar manner. Special care has to be taken to define values of  $\mathbf{W}_{\text{Spatial}}$  when the information in the image data is insufficient to define a reasonable affinity value. If we would simply set the affinity to 0, it would mean that the affinity in the absence of any information is lower than if the two trajectories have strongly incoherent clues. Our compromise for such cases is to assign the value 0.5, with the idea that it should not have a tendency for either being clustered or separated in different clusters. The details of the affinity definition can be found in the Appendix.

#### 3.2 Cluster evaluation

Once we have a properly set affinity matrix, we can apply normalized cuts to obtain the spatial links. The last question that remains is how to set the number of clusters  $k$  correctly. There are heuristics to automatically estimate  $k$ , such as [40], [41], but these seem to be somewhat problem-specific and did not work well in our case. Instead, we propose to go back to the original quadratic program (3), and use the actual objective of our tracking task to determine the cluster number. The affinity matrix  $\mathbf{W}_{\text{Spatial}}$  is transformed into costs via  $\mathbf{Q}_{\text{Spatial}} = -2\mathbf{W}_{\text{Spatial}} + \mathbf{1}$  so that it has costs in  $[-1, 1]$ . As minimizing the quadratic costs defined by  $\mathbf{Q}_{\text{Spatial}}$  results in a correlation clustering problem that is hard to solve, we generate cluster samples using spectral clustering in order to minimize the cost function. In particular this means we compute clustering results for different  $\hat{k}$  using spectral clustering. Each result with cluster number  $\hat{k}$  corresponds to a decision vector  $\mathbf{f}_{\hat{k}}$ . The optimal number  $k$  together with the cluster is then found according to the original objective function via

$$k := \arg \min_{\hat{k}=1, \dots, N_{\text{cl}}} \mathbf{f}_{\hat{k}}^T \mathbf{Q}_{\text{Spatial}} \mathbf{f}_{\hat{k}}. \quad (5)$$

#### 3.3 Trajectory extraction

In order to generate the final trajectories, we first connect the clustered detections. We use the clustered dense point tracks to obtain reliable interpolations between detections and extrapolations, as long as we have reliable dense point tracks in a cluster, indicating the same direction, i.e., having a low variance in their direction. Furthermore, clusters which do not contain detections are considered as outlier clusters, i.e., background DPTs, and are thus ignored.

## 4 EXPERIMENTAL RESULTS

We evaluate the performance of the proposed algorithm in three parts: (i) the correctness of the dense point tracklets obtained with LP, (ii) evaluation of the different parts of our method, and (iii) the tracking performance in several publicly available datasets, as well as the challenging MOTChallenge benchmark [42] which contains 11 sequences for testing.

**Implementation details.** We process the videos in batches of 50 frames. For each batch, we compute the n-cut result using [38], [39]. Since n-cut involves a random process, we run the clustering for each cluster parameter 50 times and take the best result, using our cluster evaluation function. All experiments are performed with:  $\sigma_{\text{Dist}} = 0.4$ ,  $\mu_{\text{Dist}} = 0.05$  and  $\sigma_{\text{angle}} = 50$ . For performance reasons, we filter out a DPT if it does not indicate any motion or if the response of the detection's confidence map indicates that it lies on the background. We used  $V_{\text{max}} = 25 \text{ pixel/s}$ ,  $F_{\text{max}} = 15$  and  $A_{\text{max}} = 20$ .

**Performance evaluation.** For the subsequent tracking performance evaluation, we use the popular CLEAR MOT metrics [43] that provide two complementary measures: tracking accuracy (TA), which incorporates missing recall, false alarms and identity switches; and tracking precision

(TP) which is a measure of the localization error (overlap of bounding boxes if the evaluation is done in 2D or distance between detections). The overlap to consider a detection a match in the ground truth is 50%. We also quote four popular metrics proposed in [44]. The first two reflect the temporal coverage of true trajectories by the tracker: mostly tracked (MT,  $> 80\%$  overlap with ground truth) and mostly lost (ML,  $< 20\%$ ), while the third and fourth are the well-known Recall and Precision measures.

#### 4.1 Dense point tracks

Firstly, we evaluate the correctness of the dense point tracks obtained by solving the LP and compare them to the initial DPTs of [10]. For this, we use the Figment dataset [32], which contains 18 sequences of basketball players and ground truth masks every 7 frames. Making this experiment on a sequence with segmentation masks instead of bounding boxes allows us to make sure that we are not counting as moving object the part of the background that is inside the bounding box.

We present three measures: (i) IDsw/traj: that is how many tracks switch from background to person (or vice-versa) or between two different players, averaged per track; (ii) Avg. length: average length of a track, measured as a percentage of the total length of the sequence; (iii) Overlap: average number of frames of overlap between tracks.

As we can see in Table 1, the tracks that we generate are much longer than the ones from [10], with an average length of almost 80% of the sequence. This comes at the expense of a few identity switches (or leaking as referred to in [8]), showing that the proposed tracks are robust enough for clustering. More importantly, the overlap between tracks is increased from 0.7 to 5.16 frames, meaning the affinities computed from these tracks are much more meaningful.

TABLE 1  
Comparison of feature tracks vs DPTs.

Method	IDsw/traj	Avg. length	Overlap
[10]	$8.5 \times 10^{-4}$	30.15	0.74
Proposed	$1.0 \times 10^{-1}$	<b>79.94</b>	<b>5.16</b>

#### 4.2 Analysis of the proposed method

In these experiments, we analyze the performance of each of the components of the proposed method. We first perform experiments with the  $k$  obtained from the QP objective function, and analyze the contribution of each of the affinities described in the Appendix. We consider several baselines that consists of the following parts:

- LP2D: Using only the standard LP-approach (1) on detections.
- Dist: Using detections and DPTs. As affinities between DPT's the spatial distance (Eq. 8) is considered.
- Det: Using detections and DPTs. As affinities between DPT's the box-driven distance (Eq. 6) is considered.

- Speed: Using detections and DPTs. As affinities between DPT's the speed affinity (Eq. 12) is considered.
- Angle: Using detections and DPTs. As affinities between DPT's the angle affinity (Eq. 13) is considered.

Furthermore, we compare to [8] (Greedy), which also incorporates detections and dense points tracks, but computes the number of clusters in a greedy fashion. As datasets we use 6 sequences: TUD-Stadtmitte, PETS09-S2L1, S2L2, S2L3, S1L1-2, S1L2-1 and report the average performance over the 6 sequences given as MOTA score in Table 2. As we can see, our search approach for the optimal number of clusters clearly outperforms the greedy one, by more than 15%. Furthermore our coupling formulation of low- and mid-level features can successfully integrate information from both channels, and thus improves over a tracking system that is based on detections only (LP2D) by more than 3%. Finally, Table 2 shows that clustering feature tracks leverages from the combination of all defined affinities, i.e. we can successfully group DPTs belonging to the same person by taking distance and motion information into consideration.

**Choosing the best  $k$  for spectral clustering.** We also evaluated if we can correctly find the best number of clusters with our QP formulation. To this end, we vary the number  $k'$  of clusters around the number  $k$ , chosen by our method and compute the averaged MOTA score on the 6 sequences. As shown in Fig. 3, our method automatically selects the number of clusters that provides the best tracking accuracy.

#### 4.3 Multiple people tracking

Finally, we evaluate the proposed method against state-of-the-art trackers on 35 sequences. For all experiments, we use only publicly available detections and ground truth, and use the evaluation scripts from [42].

TABLE 2  
Comparison to baseline

Affinity	MOTA
Greedy (detections+DPT) [8]	22.3
LP2D (detections)	34.8
Dist	35.8
Det	36.5
Dist+Det	36.5
Dist+Det+Speed	36.3
Dist+Det+Angle	36.6
Dist+Det+Angle+Speed	<b>37.9</b>

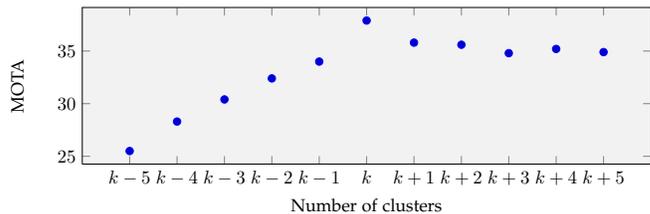


Fig. 3. Variation around the computed number of clusters on the 6 sequence dataset.

We compare to several state-of-the-art methods, drawing special attention to:

- Comparison with [8], [30]: both methods use detections as well as low-level trajectories (DPTs and KLT feature tracks, respectively).
- Comparison to LP2D: trajectories obtained by using only the Linear Program part of our algorithm, which links detections using image (2D) information.

The first set of sequences of our experiments is the Urban dataset [8]. This contains 17 urban crowded scenes filmed from a pedestrian’s viewpoint, which creates complex occlusions and pedestrians of various sizes cross the field of view of the camera. This is why standard detectors [45] struggle to find the pedestrians in these sequences, and why accuracy is in general low. We convert the ground truth of [8] from segmentation masks to bounding boxes, so that we can compare with the aforementioned metrics. The results are shown in Table 3. Note, that even though both methods presented use low- and mid-level features, we outperform the method in all measures, specially reducing the identity switches from 173 to 28.

TABLE 3  
Results on URBAN dataset.

Method	TA	TP	IDsw	Frag
Fragkiadaki et al. [8]	5.0	73.2	173	180
Proposed	<b>9.1</b>	<b>74.2</b>	<b>28</b>	<b>41</b>

Next we test 6 sequences: TUD-Stadtmitte, PETS09-S2L1, S2L2, S2L3, S1L1-2, S1L2-1. As shown in Table 4, top part, we outperform all methods in tracking accuracy (TA). We track significantly more trajectories and have less identity switches than competing methods. Note that [8], that also uses low-level features, has less ID switches, but its tracking accuracy is 15% lower than our method and it is able to track less than half of our trajectories (MT), proving that our proposed formulation is much superior at combining low- and mid-level features. We also compare our method on AVG-TownCentre [30], a sequence of a busy city center filmed from a high viewpoint, with detections from [30]. As we can see in Table 4, we outperform [30], a method that uses low-level features in the form of KLT tracks, by 6 percentage points in accuracy. The precision of the method is higher for [30] because they do bounding box position refinement, while we directly use the bounding boxes output by the detector. We also improve over tracking-by-detection methods like LP2D in accuracy, precision and identity switches.

We also test on the recent MOTChallenge benchmark [42], which contains 11 sequences for training and 11 for testing. Sequences vary in viewpoint, density of pedestrians as well as moving/static cameras, which makes it extremely hard for trackers to work well in all scenarios. For many of these sequences, pedestrians are very small, a very hard scenario for our tracker as there are very few dense tracks per pedestrian. In Table 5, we provide the results as shown on the website. We are one of the best performing published algorithms in terms of tracking accuracy. Note, that unlike competing methods, we do not use any type of trajectory post-processing. We want to especially point out that we

TABLE 4  
Evaluation in image space. *Top*: Averaged over six sequences: PETS09-S2L1, S2L2, S2L3, S1L1-2, S1L2-1, TUD-Stadtmitte. *Bottom*: Results on AVG-TownCentre.

6 sequences							
Method	TA	TP	Rcll	Prcn	MT	ML	IDsw
Fragkiadaki et al. [8]	22.3	<b>72.1</b>	35.5	74.1	8.4	44.1	<b>216</b>
Pellegrini et al. [46]	28.5	65.2	48.6	75.1	15.8	31.7	1162
Yamaguchi et al. [47]	29.0	65.1	48.8	74.8	16.3	<b>31.2</b>	987
Leal-Taixé et al. [48]	31.9	65.7	50.4	75.5	19.8	34.2	608
LP+2D	34.8	65.4	50.0	79.5	18.3	<b>31.2</b>	663
Proposed	<b>37.9</b>	65.5	<b>50.9</b>	<b>81.4</b>	<b>20.3</b>	32.2	414
AVG-TownCentre							
Method	TA	TP	Rcll	Prcn	MT	ML	IDsw
Yamaguchi et al. [47]	52.8	76.6	77.3	79.5	50.4	7.5	328
Pellegrini et al. [46]	55.2	76.6	78.6	80.7	54.0	<b>7.1</b>	324
Benfold et al. [30]	58.6	<b>83.6</b>	<b>79.0</b>	82.2	<b>59.7</b>	10.6	236
LP2D	61.6	77.2	73.9	<b>89.3</b>	48.7	12.4	245
Proposed	<b>64.2</b>	79.1	76.9	<b>89.3</b>	51.8	9.3	<b>231</b>

TABLE 5  
Results on the MOTChallenge test set.

Method	TA	TP	MT	ML	IDsw	FP
NOMT [33]	33.7	71.9	12.2	44.0	442	7762
TDAM [49]	33.0	72.8	13.3	39.1	464	10064
MHT-DAM [50]	32.4	71.8	16.0	43.8	435	9064
MDP [51]	30.3	71.3	13.0	38.4	680	9717
TbX (proposed)	27.5	70.6	10.4	45.8	759	7968
LP-SSVM [52]	25.2	71.7	5.8	53.0	849	8369
ELP [53]	25.0	71.2	7.5	43.8	1396	7345
JPDA-m [54]	23.8	68.2	5.0	58.1	365	6373
MotiCon [1]	23.1	70.9	4.7	52.0	1018	10404
SegTrack [7]	22.5	71.7	5.8	63.9	697	7890
LP2D (baseline)	19.8	71.2	6.7	41.2	1649	11580
DCO-X [55]	19.6	71.4	5.1	54.9	521	10652
CEM [2]	19.3	70.7	8.5	46.5	813	14180
RMOT [56]	18.6	69.6	5.3	53.3	684	12473
SMOT [57]	18.2	71.2	2.8	54.8	1148	8780
ALEXTRAC [58]	17.0	71.2	3.9	52.4	1859	9233
TBD [59]	15.9	70.9	6.4	47.9	1939	14943
TC-ODAL [60]	15.1	70.5	3.2	55.8	637	12970
DP-NMS [18]	14.5	70.8	6.0	40.8	4537	13171
LDCT [61]	4.7	71.7	11.4	32.5	12348	14066

outperform LP2D by almost 8 percentage points in accuracy and less than half the identity switches, clearly showing the strength of combining low- and mid-level features.

## 5 CONCLUSION

We presented a global formulation for the integration of mid- and low-level features for multi-target tracking. The problem is cast into a quadratic program, which is then decomposed for efficiency into temporal associations with linear programming and spatial associations with spectral clustering. The full objective function of the QP is still used to choose the optimal number of clusters for spectral clustering, which is always a delicate step in other approaches. We showed superior results when compared to tracking-by-detection methods and methods that also use low-level features, thus proving the benefits of our formulation. As future work, we plan on exploring the integration of other features. Since the number of affinities grows quadratically with the number of features, we will focus on metric learning for the affinity computation.

## REFERENCES

- [1] L. Leal-Taixé, M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese, "Learning an image-based motion context for multiple people tracking," *CVPR*, 2014.
- [2] A. Milan and S. R. R. Schindler, "Continuous energy minimization for multitarget tracking," *TPAMI*, 2014.
- [3] A. Zamir, A. Dehghan, and M. Shah, "Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs," *ECCV*, 2012.
- [4] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *TPAMI*, 2014.
- [5] M. Sadeghi and D. Forsyth, "30hz object detection with dpm v5," *ECCV*, 2014.
- [6] S. Ren, R. G. K. He, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Neural Information Processing Systems (NIPS)*, 2015.
- [7] A. Milan, L. Leal-Taixé, K. Schindler, and I. Reid, "Joint tracking and segmentation of multiple targets," in *CVPR*, 2015.
- [8] K. Fragkiadaki, W. Zhang, G. Zhng, and J. Shi, "Two-granularity tracking: mediating trajectory and detections graphs for tracking under occlusions," *ECCV*, 2012.
- [9] S. Chen, A. Fern, and S. Todorovic, "Multi-object tracking via constrained sequential labeling," *CVPR*, 2014.
- [10] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," *ECCV*, 2010.
- [11] C. Xu, C. Xiong, and J. Corso, "Streaming hierarchical video segmentation," *ECCV*, 2012.
- [12] J. Gall, A. Yao, N. Razavi, L. van Gool, and V. Lempitsky, "Hough forests for object detection, tracking and action recognition," *TPAMI*, 2011.
- [13] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *TPAMI*, 2010.
- [14] Z. Khan, T. Balch, and F. Dellaert, "Mcmc-based particle filtering for tracking a variable number of interacting targets," *TPAMI*, 2005.
- [15] J. Berclaz, F. Fleuret, and P. Fua, "Robust people tracking with global trajectory optimization," *CVPR*, 2006.
- [16] H. Jiang, S. Fels, and J. Little, "A linear programming approach for multiple object tracking," *CVPR*, 2007.
- [17] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," *CVPR*, 2008.
- [18] H. Pirsivavash, D. Ramanan, and C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," *CVPR*, 2011.
- [19] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua, "Multiple object tracking using k-shortest paths optimization," *TPAMI*, 2011.
- [20] A. Dehghan, S. Assari, and M. Shah, "Gmmcp-tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking," *CVPR*, 2015.
- [21] W. Brendel, M. Amer, and S. Todorovic, "Multiobject tracking as maximum weight independent set," *CVPR*, 2011.
- [22] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn, "Branch-and-price global optimization for multi-view multi-object tracking," *CVPR*, 2012.
- [23] A. Butt and R. Collins, "Multi-target tracking by Lagrangian relaxation to min-cost network flow," *CVPR*, 2013.
- [24] B. Leibe, K. Schindler, and L. van Gool, "Coupled detection and trajectory estimation from multi-object tracking," *ICCV*, 2007.
- [25] H. Izadinia, I. Saleemi, W. Li, and M. Shah, "(mp)2t: Multiple people multiple parts tracker," *ECCV*, 2011.
- [26] C. Wojek, S. Walk, S. Roth, and B. Schiele, "Monocular 3d scene understanding with explicit occlusion reasoning," *CVPR*, 2011.
- [27] S. Tang, M. Andriluka, A. Milan, K. Schindler, S. Roth, and B. Schiele, "Learning people detectors for tracking in crowded scenes," *ICCV*, 2013.
- [28] V. Chari, S. Lacoste-Julien, I. Laptev, and J. Sivic, "On pairwise costs for network flow multi-object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5537–5545.
- [29] G. Seguin, P. Bojanowski, R. Lajugie, and I. Laptev, "Instance-level video segmentation from object tracks," in *Proc. CVPR*, 2016.
- [30] B. Benfold and I. Reid, "Stable multi-target tracking in real-time surveillance video," *CVPR*, 2011.
- [31] C. Tomasi and T. Kanade, "Detection and tracking of point features," CMU-CS-91-132, Tech. Rep., 1991.
- [32] K. Fragkiadaki and J. Shi, "Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement," *CVPR*, 2011.
- [33] W. Choi, "Near-online multi-target tracking with aggregated local flow descriptor," *ICCV*, 2015.
- [34] L. Torresani, V. Kolmogorov, and C. Rother, "Feature correspondence via graph matching: models and global optimization," *ECCV*, 2008.
- [35] V. Chari, S. Lacoste-Julien, I. Laptev, and J. Sivic, "On pairwise costs for network flow multi-object tracking," *CVPR*, 2015.
- [36] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Machine Learning*, vol. 56, no. 1-3, pp. 89–113, 2004.
- [37] S. Bagon and M. Galun, "Large scale correlation clustering optimization," *arXiv preprint arXiv:1112.2903*, 2011.
- [38] J. Shi and J. Malik, "Normalized cuts and image segmentation," *TPAMI*, vol. 22, no. 8, pp. 888–905, 2000.
- [39] T. Cour, S. Yu, and J. Shi, "Normalized cut segmentation code. copyright 2004 university of pennsylvania," *Computer and Information Science Department*, 2004.
- [40] Z. Li, J. Liu, S. Chen, and X. Tang, "Noise robust spectral clustering," in *JCCV*. IEEE, 2007, pp. 1–8.
- [41] L. Zelnik-Manor and P. Perona, "Self-tuning spectral clustering," in *Advances in neural information processing systems*, 2004, pp. 1601–1608.
- [42] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "Motchallenge 2015: Towards a benchmark for multi-target tracking," *arXiv:1504.01942*, 2015.
- [43] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, M. Boonstra, V. Korzhova, and J. Zhang, "Framework for performance evaluation for face, text and vehicle detection and tracking in video: data, metrics, and protocol," *TPAMI*, vol. 31, no. 2, 2009.
- [44] Y. Li, C. Huang, and R. Nevatia, "Learning to associate: hybrid boosted multi-target tracker for crowded scene," *CVPR*, 2009.
- [45] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *PAMI*, 2014.
- [46] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool, "You'll never walk alone: modeling social behavior for multi-target tracking," *ICCV*, 2009.
- [47] K. Yamaguchi, A. Berg, L. Ortiz, and T. Berg, "Who are you with and where are you going?" *CVPR*, 2011.
- [48] 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.
- [49] M. Yang and Y. Jia, "Temporal dynamic appearance modeling for online multi-person tracking," *Computer Vision and Image Understanding*, 2016.
- [50] C. Kim, F. Li, A. Ciptadi, and J. Rehg, "Multiple hypothesis tracking revisited: Blending in modern appearance model," *ICCV*, 2015.
- [51] Y. Xiang, A. Alahi, and S. Savarese, "Learning to track: Online multi-object tracking by decision making," *ICCV*, 2015.
- [52] S. Wang and C. Fowlkes, "Learning optimal parameters for multi-target tracking," *BMVC*, 2015.
- [53] N. McLaughlin, J. M. D. Rincon, and P. Miller, "Enhancing linear programming with motion modeling for multi-target tracking," *WACV*, 2015.
- [54] H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, and I. R. A. Dick, "Joint probabilistic data association revisited," *ICCV*, 2015.
- [55] A. Milan, K. Schindler, and S. Roth, "Multi-target tracking by discrete-continuous energy minimization," *TPAMI*, 2016.
- [56] J. Yoon, H. Yang, J. Lim, and K. Yoon, "Bayesian multi-object tracking using motion context from multiple objects," *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015.
- [57] C. Dicle, O. Camps, and M. Sznajder, "The way they move: Tracking targets with similar appearance," *ICCV*, 2013.
- [58] A. Bewley, L. Ott, F. Ramos, and B. Upcroft, "Alextrac: Affinity learning by exploring temporal reinforcement within association chains," *ICRA*, 2016.
- [59] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3d traffic scene understanding from movable platforms," *TPAMI*, 2014.
- [60] S. Bae and K. Yoon, "Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning," *CVPR*, 2014.
- [61] F. Solera, S. Calderara, and R. Cucchiara, "Learning to divide and conquer for online multi-target tracking," *ICCV*, 2015.

## APPENDIX

In this appendix we detail the computation of the affinities between detections and dense point trajectories.

Note that we threshold very low affinities in order to speed the convergence of the clustering algorithm.

### A1. Affinities between dense point tracks

The affinity matrix  $\mathbf{W}_{PP}$  is constructed considering motion and spatial proximity of dense point tracks. In particular, let  $T_i, T_j$  be two dense point tracks. We define two affinities regarding the spatial distance and two affinities regarding their motion. Let  $\mathcal{F}(i, j)$  denote the set of common frames between  $T_i$  and  $T_j$ . We consider two cases:  $\mathcal{F}(i, j) = \emptyset$  and  $\mathcal{F}(i, j) \neq \emptyset$ . Now, let  $\mathcal{F}(i, j) = \emptyset$ . The optimal linking of dense point tracks in time has already been computed by the LP in Eq. (1). During clustering we want to especially enforce coupling between different feature categories, hence we set  $W_{PP}(i, j) = 1$  if there is a detection track connecting  $T_i$  with  $T_j$ . Otherwise we set its value to 0.

Now consider the case that  $\mathcal{F}(i, j) \neq \emptyset$ . We compute the affinity using 4 different aspects that we compare:

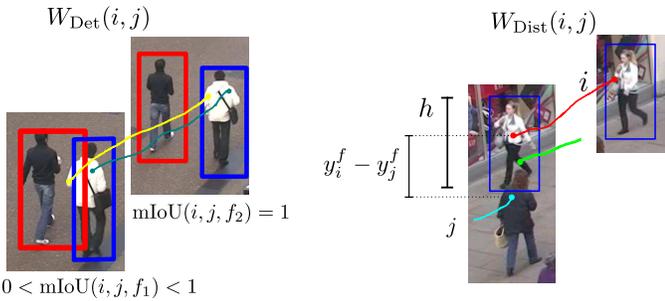


Fig. 4. *Left*: Illustration of the  $\mathbf{W}_{Det}$  definition. *Right*: the  $\mathbf{W}_{Dist}$  definition.

- The *detection* affinity matrix

$$W_{Det}(i, j) = \frac{\text{mIoU}(i, j)}{|\mathcal{F}(i, j)|} \quad (6)$$

measures how similar are the detections crossed by the dense point tracks, where

$$\text{mIoU}(i, j) = \sum_{f \in \mathcal{F}(i, j)} \text{mIoU}(i, j, f), \quad (7)$$

is defined as the maximal intersection over union (IoU) between any two detections that intersect  $T_i$  and  $T_j$ . If  $T_i$  or  $T_j$  do not have a detection at time stamp  $f$ , we set  $\text{mIoU}(i, j, f) = 0.5$ , as there is no scale available from which we can judge proximity. The left part of Figure 4 explains the idea: the green and yellow dense points are close to each other but not in the same box, yet the affinity of being clustered together gets a non-zero value.

- The *distance* affinity matrix  $\mathbf{W}_{Dist}$  measures directly the spatial distance of two dense tracks and weights it by the scale given by the detections. We equally

weight information from width and height, hence we set  $D(i, j) := 0.5(D^H(i, j) + D^W(i, j))$ . Then

$$W_{Dist}(i, j) = \begin{cases} D(i, j) & D^H(i, j), D^W(i, j) \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

Thereby  $D^H$  measures the spatial proximity with respect to the height and  $D^W$  with respect to the width, accordingly. More specifically, for the height we define

$$D^H(i, j) := \text{med}\{D_j^H(i), D_i^H(j)\}. \quad (9)$$

Thereby,  $D_i^H(j)$  judges whether  $T_j$  should be clustered with  $T_i$  on the basis of the detection height observed by  $T_i$ . Let  $y_i^f$  be the  $y$ -position of  $T_i$  at time  $f$ . We set

$$\hat{D}_i^H(j) = \text{med} \left\{ \frac{y_i^f - y_j^f}{\text{med}_H(i)} \mid f \in \mathcal{F}(i, j) \right\}, \quad (10)$$

where  $\text{med}_H(i)$  is the median height of all boxes that intersect  $T_i$ . Now

$$D_i^H(j) = \begin{cases} 1 & \text{if } \hat{D}_i^H(j) \leq \mu_{Dist} \\ \frac{\mathcal{N}(\hat{D}_i^H(j), \mu_{Dist}, \sigma_{Dist})}{\mathcal{N}(\mu_{Dist}, \mu_{Dist}, \sigma_{Dist})} & \text{otherwise} \end{cases}, \quad (11)$$

where  $\mathcal{N}(x, \mu_{Dist}, \sigma_{Dist})$  denotes the normal probability density function evaluated at  $x$  with mean value  $\mu_{Dist}$  and variance  $\sigma_{Dist}$ .

The width affinities are defined accordingly, by replacing the  $y$ -coordinate with the  $x$ -coordinate and the median height by the median width.

- Dense point tracks belonging to a person should have the same speed. Thus we define a *speed* affinity matrix

$$W_{Speed}(i, j) = \text{med} \left\{ \frac{\min(v_i^f, v_j^f)}{\max(v_i^f, v_j^f)} \right\}, \quad (12)$$

where  $f \in \mathcal{F}(i, j)$ , and  $v_i^f$  denotes the magnitude of the 2D velocity of  $T_i$  at time stamp  $f$ .

- Apart from the same speed, the dense point tracks should follow a similar direction. Hence, we compare angles between the velocity vectors of the tracks. We define the *angular* affinity as

$$W_{Angle}(i, j) = \frac{\mathcal{N}(\angle(i, j), 0, \sigma_{angle})}{\mathcal{N}(0, 0, \sigma_{angle})}, \quad (13)$$

where  $\angle(i, j)$  denotes the median angle between the velocity vectors of  $T_i$  and  $T_j$  at each common time stamp.

Finally, we combine speed and angle affinities with equal weight to create the velocity  $\mathbf{W}_{Velocity}$ . Since 2D velocities can be noisy, we reduce the weight of this affinity by transforming it linearly to the interval  $[0.5, 1]$ .

Having defined these affinities we finally set

$$W_{PP}(i, j) = W_{\text{Velocity}}(i, j) \times \frac{1}{2}(W_{\text{Dist}}(i, j) + W_{\text{Det}}(i, j)). \quad (14)$$

## A2. Affinities between a dense point track and a detection

Given a dense point track  $T_i$  and a detection  $d_j$  of a detection track  $T_j$ , we compare them in two ways. First, we check for spatial intersection: If  $\mathcal{F}(i, j) \neq \emptyset$ , we set

$$W_{\text{PD-intersect}}(i, j) = \begin{cases} 1 & \text{if } \mathbf{p}_i^f \text{ lies in detection } d_j \\ 0 & \text{otherwise.} \end{cases} \quad (15)$$

For the case  $\mathcal{F}(i, j) = \emptyset$ , we set the affinity to 0.5 for the reasons discussed above. Furthermore, we compute  $W_{\text{PD-Link}}(i, j)$  using the same terms as described for  $\mathbf{W}_{PP}$ . Finally, we combine the two terms:

$$\mathbf{W}_{PD} = \frac{1}{2}(\mathbf{W}_{\text{PD-intersect}} + \mathbf{W}_{\text{PD-Link}}). \quad (16)$$

## A3. Affinities between detections

Comparison between detections is driven by the detection tracks as well as intersecting dense point tracks. Let  $\mathcal{T}(d_i)$  be the set of dense point tracks intersecting  $d_i$  and

$$r(i, j) = \frac{|\mathcal{T}(d_i) \cap \mathcal{T}(d_j)|}{|\mathcal{T}(d_i)|}.$$

Then

$$W_{\text{DD-t}}(i, j) = 0.5 * \begin{cases} r(i, j) + r(j, i) & |\mathcal{T}(d_i)|, |\mathcal{T}(d_j)| > 0 \\ 1 & \text{otherwise.} \end{cases} \quad (17)$$

A second term compares the whole span of the tracks:

$$W_{\text{DD-l}}(i, j) = \begin{cases} 1 & \text{if } d_i, d_j \text{ are in the same track} \\ 0 & \text{otherwise.} \end{cases} \quad (18)$$

Finally, we set  $\mathbf{W}_{DD} = \mathbf{W}_{\text{DD-t}} \mathbf{W}_{\text{DD-l}}$ .