Non-Markovian Globally Consistent Multi-Object Tracking

Andrii Maksai[1], Xinchao Wang[2], François Fleuret[3], and Pascal Fua[1]

[1]Computer Vision Laboratory, EPFL, Lausanne, Switzerland, {firstname.lastname}@epfl.ch
[2]Beckman Institute, UIUC, Illinois, USA {firstname}@illinois.edu
[3]IDIAP Reseach Institute, Martigny, Switzerland, {firstname.lastname}@idiap.ch

Abstract

Many state-of-the-art approaches to multi-object tracking rely on detecting them in each frame independently, grouping detections into short but reliable trajectory segments, and then further grouping them into full trajectories. This grouping typically relies on imposing local smoothness constraints but almost never on enforcing more global ones on the trajectories.

In this paper, we propose a non-Markovian approach to imposing global consistency by using behavioral patterns to guide the tracking algorithm. When used in conjunction with state-of-the-art tracking algorithms, this further increases their already good performance on multiple challenging datasets. We show significant improvements both in supervised settings where ground truth is available and behavioral patterns can be learned from it, and in completely unsupervised settings.

1. Introduction

Multiple Object Tracking (MOT) has a long tradition for applications such as radar tracking [18]. These early approaches gradually made their way into vision community for object tracking purposes. They initially relied on Gating, Kalman Filtering [17, 64, 36, 89, 57] and later on Particle Filtering [32, 78, 66, 45, 90, 60, 19]. Because of their recursive nature, when used to track objects in crowded scenes, they are prone to identity switches and trajectory fragmentations, which are difficult to recover from.

With the recent improvements of object detectors [27, 8], the Tracking-by-Detection paradigm [4] has now become the preferred way to address MOT. In most state-of-the-art approaches [80, 23, 61, 88], this involves detecting objects in each frame independently, grouping detections into short but reliable trajectory segments, or tracklets, and then further grouping those into full trajectories.

While effective, existing tracklet-based approaches tend to only impose local smoothness constraints on the trajectories. These are Markovian in nature as opposed to being global ones that stem from behavioral patterns. For example, a person entering a building via a particular door can be expected to head to a specific set of rooms. Similarly, a pedestrian emerging on the street from a shop will often turn left or right to follow the sidewalk. Such patterns are of course not absolutes because people sometimes do the unexpected but they should nevertheless inform the tracking algorithms. We know of no existing technique that imposes this kind of global non-Markovian constraints in a globally optimal fashion.

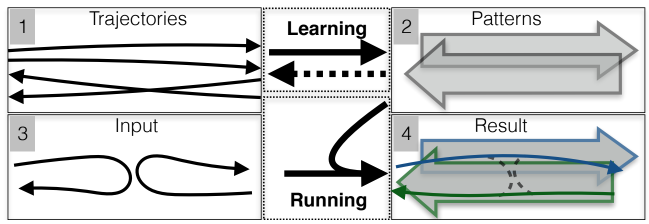


Figure 1. **Top row.** At training time, our procedure alternates between learning global patterns from trajectories and improving the trajectories on the basis of these patterns. When the initial trajectories come from annotated ground truth data, the patterns are simply learned without further iterations. **Bottom row.** At run time, the learned patterns are used to improve trajectories produced by state-of-the-art algorithms.

Our first contribution is an energy function that relates behavioral patterns to trajectories assigned to them. We use it to infer global patterns and to guide a multi-target tracking algorithm in a non-Markovian fashion.

Our second contribution is an unsupervised training scheme. Given input trajectories from any source, it iterates between learning patterns that maximize our energy function, and improving the trajectories by linking the detections that were the part of the original ones in a potentially different way so as to maximize the same energy. When the original trajectories come from annotated ground truth data, the patterns are simply learned for them without further iterations. The top row of Fig. 1 depicts this process. At run-time, previously leared patterns are used to improve the trajectories produced by the original algorithm or any other, as depicted by the bottom row of Fig. 1. We show that this approach consistently improves performance on multiple challenging datasets by 7% and 5% on average in supervised and unsupervised fashion respectively. This is mostly attributable to the reduction in identity switches between objects following different patterns. Our code is made publicly available [1].

2. Related Work

We briefly review data association and behavioral modeling techniques and refer the interested reader to [86, 55] for more details. We also discuss the metrics we use for MOT evaluation and their sensitivity to identity switches.

2.1. MOT as Data Association

Finding the right trajectories linking the detections, or data association, has been formalized using various models. For real-time performance, data association often relies either on matching locally between existing tracks and new targets [28, 53, 6, 23, 62] or on filtering techniques [65, 75]. The resulting algorithms are fast but often perform less well than batch optimization methods, which use a sequence of frames to associate the data optimally over a whole set of frames, rather than greedily in each following frame.

Batch optimization can be formulated as a shortest path problem [14, 70], network flow problem [96], generic linear programming [39], integer or quadratic programming [52, 20, 83, 73, 26, 94, 59]. A common way to reduce the computational burden is to group reliable detections into short trajectory fragments known as tracklets and then reason on these tracklets instead of individual detections [41, 77, 56, 50, 11].

However, whether or not tracklets are used, making the optimization problem tractable when looking for a global optimum limits the class of possible objective functions. They are usually restricted to functions that can be defined on edges or edge pairs in a graph whose nodes are individual detections or tracklets. In other words, such objective functions can be used only to impose relatively local constraints. To impose global constraints, the objective functions have to involve multiple objects and long time spans. They are optimized using gradient descent with exploratory jumps [63], inference with a dynamic graphical model [23], or iterative groupings of shorter tracklets into longer trajectories [49, 31, 5]. However, this comes at the cost of losing any guarantee of global optimality.

By contrast, our approach is designed for batch optimization and finding the global optimum, while using an objective function that is rich enough to express the relation between global trajectories and non-linear motion patterns. The method of [24] advocates the same philosophy but for the very different activity recognition task.

2.2. Using Behavioral Models

A number of works incorporate human behavioral models into tracking algorithms to increase their reliability. For example, the approaches of [68, 2] model collision avoidance behavior to improve tracking, the one of [92] uses behavioral model to predict near future target locations, and the one of [71] encodes local velocities into the affinity matrix of tracklets. These approaches boost the performance but only account for very local interactions, instead of global behaviors that influence the *whole* trajectory.

Many approaches to inferring various forms of global patterns have been proposed over the years [72, 42, 58, 69, 95, 35, 87, 21, 47, 54]. However, the approaches of [13], [3], [48], and [7] are the only ones we know of that attempt to use these global patterns to guide the tracking. The method of [13] is predicated on the idea that behavioral maps describing a distribution over possible individual movements can be learned and plugged into the tracking algorithm to improve it. However, even though the maps are global, they are only used to constrain the motion locally without enforcing behavioral consistency over the whole trajectory. In [7], an E-M-based algorithm is used to model the scene as a Gaussian mixture that represents the expected size and speed of an object at any given location. While the model can detect global motion anomalies and improve object detection, the motion pattern information is not used to improve the tracking explicitly. In [48], modeling the optical flow helps to detect anomalies but only when the crowd is dense enough. In [3], global behavioral patterns are learned as vector fields on the floor. However, when used for tracking in high-density crowds, they are converted to to local Markovian transition probabilities, thereby loosing their global nature.

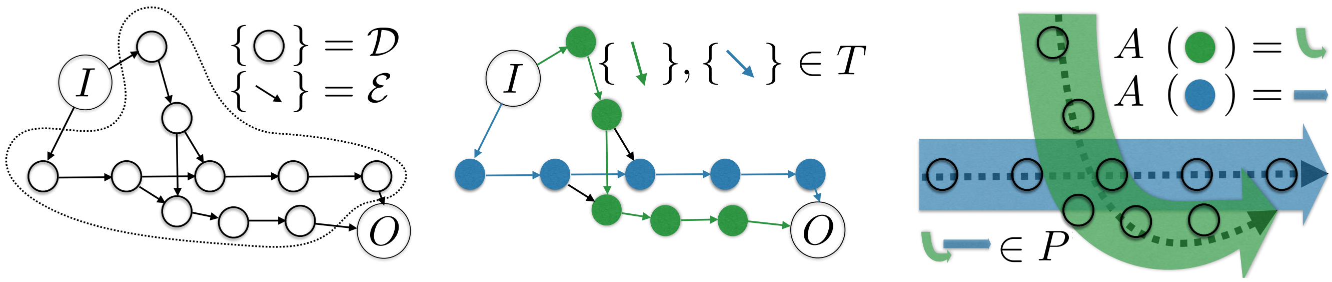
Vehicle motion is more structured than the human kind and behavioral models often take into account speed limits or states of the traffic lights [97, 43, 34, 38, 82]. Nevertheless, they retain enough similarities with human motion that we can represent patterns in the same way for both.

2.3. Quantifying Identity Switches

In this paper, we aim for globally consistent tracking by preventing identity switches along reconstructed trajectories, for example when trajectories of different objects are merged into one or when a single trajectory is fragmented into many. We therefore need an appropriate metric to gauge the performance of our algorithms.

The set of CLEAR MOT metrics [15] has become a *de-facto* standard for evaluating tracking results. Among these,

[1] https://github.com/maksay/ptrack_cpp



(a) (b) (c)

Figure 2. **(a)** Given a set of high-confidence detections $\mathcal{D}$, and a set of allowed transitions $\mathcal{E}$, we seek to find: **(b)** trajectories of the objects, represented by transitions from T; **(c)** a set of behavioural patterns P, which define where objects behaving in a particular way are likely to be found; an assignment A of each individual detection to a pattern, specifying which pattern did the object in this detection follow.

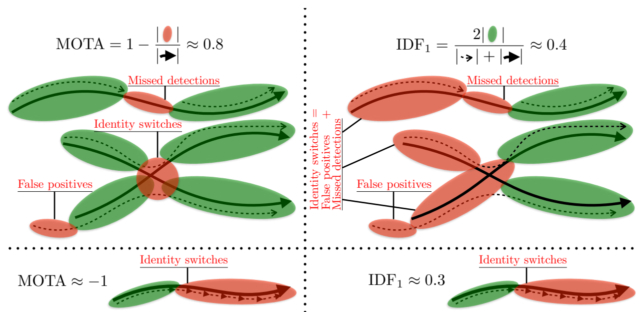


Figure 3. Effect of identity switches on the tracking metrics. The thick lines represent ground-truth trajectories and the thin dotted ones recovered trajectories. The trajectory fragments that count positively are shown in green and those that count negatively in red. The formulas at the top of the figure depict graphically how the **MOTA** and **IDF**$_1$ scores are computed. **Top**: Three ground-truth trajectories, with the bottom two crossing in the middle. The four recovered trajectories feature an identity switch where the two real trajectories intersect, missed detections resulting in a fragmented trajectory and therefore another identity switch at the top, and false detections at the bottom left. When using **MOTA**, the identity switches incur a penalty but only very locally, resulting in a relatively high score. By contrast, **IDF**$_1$ penalizes the recovered trajectories over the *whole* trajectory fragment assigned to the wrong identity, resulting in a much lower score. **Bottom:** The last two thirds of the recovered trajectory are fragmented into individual detections that are not linked. **MOTA** counts each one as an identity switch, resulting in a negative score, while **IDF**$_1$ reports a more intuitive value of 0.3.

Multiple Object Tracking Accuracy (MOTA) is the one that is used most often to compare competing approaches. However, it has been pointed out that MOTA does not properly account for identity switches [10, 94, 12], as depicted on the left side of Fig. 3. More adapted metrics have therefore been proposed. For example, **IDF**$_1$ is computed by matching trajectories to ground-truth so as to minimize the sum of discrepancies between corresponding pairs [74]. Unlike **MOTA**, it penalizes switches over the *whole* trajectory fragments assigned to the wrong identity, as depicted by the right side of Fig. 3. Furthermore, unlike Id-Aware metrics [94, 10], it does not require knowing the true identity of the objects being tracked, making it more widely applicable. In Section 6.4, we report results both in terms of **MOTA** and **IDF**$_1$, to highlight the drop in identity switches our method brings about.

3. Formulation

In this section, we formalize the problem of discovering and using behavioral patterns to impose global constraints on a multi-object tracking algorithm. In the following sections we will use it to estimate trajectories given the patterns and to discover the patterns given ground-truth trajectories.

3.1. Detection Graph

Given a set of high-confidence detections $\mathcal{D} = \{1, \ldots, L\}$ in consecutive images of a video sequence, let $\mathcal{V} = \mathcal{D} \cup \{I, O\}$, where I and O denote possible trajectory start and end points and each node $v \in \mathcal{D}$ is associated with a set of features that encode location, appearance, or other important properties of a detection. Let $\mathcal{E} \subset \mathcal{V}^2$ be the set of possible transitions between the detections. $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ can then be treated as a *detection graph* of which the desired trajectories are subgraphs. As depicted by Fig. 2, let

- $T \subset \mathcal{E}$ be a set of edges defining objects' trajectories.
- P be a set of patterns, each defining an area where objects behaving in a specific way are likely to be found, plus an empty pattern $\emptyset$ used to describe unusual behaviors. Formally speaking, patterns are functions that associate to a trajectory made of an arbitrary number of edges a score that denotes how likely it is to correspond to that specific pattern, as discussed in Section 3.3.
- A be a set of assignments of individual detections in $\mathcal{D}$ into patterns, that is, a mapping $A : \mathcal{D} \to \{1, \ldots, N_p\}$, where N_p is the total number of patterns.

Each trajectory $t \in T$ must go through detections via allowable transitions, begin at I, and end at O. Here we abuse the notation $t \in T$ to express that all edges $(I, t_1), (t_1, t_2), \cdots, (t_{|t|}, O)$ from trajectory $t = (t_1, \cdots, t_{|t|})$ belong to T. Furthermore, since we only consider high-confidence detections, each one must belong to **exactly** one trajectory. In practice, this means that potential false positives end up being assigned to the empty behavior $\emptyset$ and can be removed as a post-processing step. Whether to do this or not is governed by a binary indicator R_e that is learned. In other words, the edges in T must be such that for each detection there is exactly one selected edge coming in and one going out, which we can write as

$$\forall j \in \mathcal{D}, \exists! i \in \mathcal{V}, k \in \mathcal{V} : (i,j) \in T \wedge (j,k) \in T \ . \tag{1}$$

Since all detections that are grouped into the same trajectory T must be assigned to the same pattern, we must have

$$\forall (i,j) \in T : (i \in \mathcal{D} \wedge j \in \mathcal{D}) \Rightarrow A(i) = A(j) \ . \tag{2}$$

In our implementation, each pattern $p \in P \backslash \emptyset$ is defined by a trajectory that serves as a centerline and a width, as depicted by Fig. 2(c) and 4. However, the optimization schemes we will describe in Sections 4.1 and 4.2 do not depend on this specific representation and can be replaced by any other.

3.2. Building the Graph

To build the graph we use trajectories produced by another algorithm, as input. We want to improve these trajectories, therefore we build a graph so that we can obtain new trajectories and recover from identity switches, fragmentations, and incorrectly merged input trajectories.

We take the set of detections along these input trajectories to be our high-confidence detections $\mathcal{D}$ and therefore the nodes of our graph. We take the edges $\mathcal{E}$ to be pairs of nodes that are either *i)* consecutive in the original trajectories, *ii)* within ground plane distance D_1 of each other in successive frames, *iii)* the endings and beginnings of input trajectories within distance D_2 and within D_t frames, *iv)* or whose first node is I or second node is O.

3.3. Objective Function

Our goal is to find the most likely trajectories formed by transitions in T^*, patterns P^*, and mapping linking one to the other A^*, given the image information and any *a priori* knowledge we have. In particular, given a set of patterns P^*, we look for the best set of trajectories that match these patterns. Conversely, given a set of known trajectories T^*, we learn a set of patterns, as discussed in Section 4.

To formulate these searches in terms of an optimization problem, we introduce an objective function $C(T,P,A)$ that reflects how likely it is to observe the objects moving along the trajectories defined by T, each one corresponding to a pattern from $P = \{p_1 \cdots, p_{N_p}\}$ given the assignment A. Ideally, C should be the proportion of trajectories that correctly follow the assigned patterns. To compute it in practice, we take our inspiration from the **MOTA** and **IDF**$_1$ scores described in Section 2.3. They are written in terms of ratios of the lengths of trajectory fragments that follow the ground truth to total trajectory lengths. We therefore take our objective function to be a similar ratio, but instead of ground truth trajectories we use patterns. More formally:

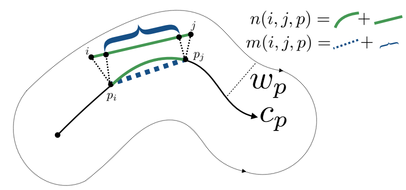


Figure 4. For a pattern p defined by centerline c_p, shown as a thick black line, with width w_p, and an edge (i,j), we compute functions $n(i,j,p)$ and $m(i,j,p)$ introduced in Section 3.3 and shown in green and blue, respectively, as follows: $n(i,j,p)$ is the total length of the edge and the corresponding length of the pattern centerline, measured between the points p_i and p_j, which are the points on the centerline closest to i and j. If both i and j are within the pattern width w_p from the centerline, we take $m(i,j,p)$ to be the sum of two terms: the length in the pattern along the edge, that is, the distance between p_i and p_j, plus the length in the edge along the pattern, that is, the length of the projection of (p_i, p_j) onto the line connecting i and j. Otherwise $m(i,j,p) = 0$ to penalize the deviation from the pattern.

$$\begin{aligned} C(T,P,A) &= \frac{\sum_{t \in T} M(t, p_{A(t_1)})}{\sum_{t \in T} N(t, p_{A(t_1)})}, \\ N(t,p) &= n(I, t_1, p) + n(t_{|t|}, O, p) + \sum_{1 \le j \le |t|-1} n(t_j, t_{j+1}, p), \\ M(t,p) &= m(I, t_1, p) + m(t_{|t|}, O, p) + \sum_{1 \le j \le |t|-1} m(t_j, t_{j+1}, p), \end{aligned} \tag{3}$$

where $n(i,j,p)$ is the sum of the total length of edge (i,j) and of the length of the corresponding pattern centerline, while $m(i,j,p)$ is the sum of lengths of aligned parts of the pattern and the edge. Fig. 4 illustrates this computation and we give the mathematical definitions of m and n in the supplementary material.

As a result, $N(t,p)$ is the sum of the lengths of trajectory and assigned pattern while $M(t,p)$ measures the length of parts of trajectory and pattern that are aligned with each other. Note that the definition of Eq. (3) is very close to that of the metric **IDF**$_1$ introduced in Sec. 2.3. It is largest when each person follows a single pattern for as long as possible. This penalizes identity switches because the trajectories that are erroneously merged, fragmented, or jump between objects are unlikely to follow any specific pattern.

In Eq. (3), we did not explicitly account for the fact that the first vertex i of some edges can be the special entrance vertex, which is not assigned to any behavior. When this happens we simply use the pattern assigned to the second vertex j. From now on, we will replace $A(i)$ by $A(i,j)$ to denote this behavior. We also adapt the definitions of m and n accordingly to properly handle those special edges.

4. Computing Trajectories and Patterns

In this section, we describe how we use the objective function C of Eq. (3) to compute trajectories given patterns and patterns given trajectories. The resulting procedures will be the building blocks of our complete MOT algorithm, as described in Section 5.

4.1. Trajectories

Let us assume that we are given a precomputed set of patterns P^*, then we look for trajectories and corresponding assignment as

$$T^*, A^* = \arg\max_{T,A} C(T, P^*, A) \; . \tag{4}$$

To solve this problem, we treat the motion of objects through the detection graph $\mathcal{G}$ introduced in Section 3.1 as a flow. Let $o_{ij}^p \in \{0, 1\}$ be the number of objects transitioning from node i to j in a trajectory T assigned to pattern $p \in P^*$. It relates to P^* and T according to:

$$o_{ij}^p = \mathbb{I}(((i,j) \in T) \wedge (P^*_{A(i,j)} = p)) \; . \tag{5}$$

Using these new binary variables, we reformulate constraints (1) and (2) as

$$\begin{aligned} \forall i \in \mathcal{D} \cup O \sum_{(i,j)\in\mathcal{E}, p\in P^*} o_{ij}^p &= 1 \; , \\ \forall j \in \mathcal{D}, p \in P^* \sum_{(i,j)\in\mathcal{E}} o_{ij}^p &= \sum_{(j,k)\in\mathcal{E}} o_{jk}^p \; . \end{aligned} \tag{6}$$

This lets us rewrite our cost function as

$$C(T, P^*, A) = \frac{\sum\limits_{(i,j)\in T, p\in P^*} m(i,j,p) o_{ij}^p}{\sum\limits_{(i,j)\in T, p\in P^*} n(i,j,p) o_{ij}^p} \; , \tag{7}$$

which we maximize with respect to the flow variables o_{ij}^p subject to the two constraints of Eq. (6). This is an integer-fractional program, which could be transformed into a Linear Program [22]. However, solving it would produce non-integer values that would need to be rounded. To avoid this we propose a scheme based on the following observation: Maximizing $\frac{a(x)}{b(x)}$ with respect to x when $b(x)$ is always positive can be achieved by finding the largest α such that an x satisfying $a(x) - \alpha b(x) \geq 0$ can be found. Furthermore, α can be found by binary search. We therefore take a to be the numerator or Eq. (7), b its denominator, and x the vector of o_{ij}^p variables. In practice, given a specific value of α, we do this by running a Integer Linear Program solver [33] until it finds a feasible solution. When α reaches its maximum possible value, that feasible solution is also the optimal one. We provide implementation details in the supplementary.

4.2. Patterns

In the previous section, we assumed the patterns known and used them to compute trajectories. Here, we reverse the roles. Let us assume we are given a set of trajectories T^*. We learn the patterns and corresponding assignments as

$$\begin{aligned} P^*, A^* &= \arg\max_{P,A} C(T^*, P, A) \; , \\ \text{subject to} \quad & P \subset \mathcal{P}, |P| \leq \alpha_p, \sum_{p\in P} M(p) \leq \alpha_c \; , \end{aligned} \tag{8}$$

where α_c, α_p are thresholds and $M : P \to \mathbb{R}^+$. The purpose of the additional constraints is to limit both the number of patterns being used by α_p and their spatial extent by α_c, to prevent over-fitting. In our implementation, we take $M(p) = l_p w_p$, where l_p is the length of the pattern centerline and w_p is its width. $\mathcal{P}$ is a set of all admissible patterns, which we construct by combining all possible ground-truth trajectories as centerlines with each width from a predefined set of possible pattern widths.

To solve the problem of Eq. (8), we look for an assignment between our known ground truth trajectories T^* and all possible patterns $\mathcal{P}$ and retain only patterns associated to at least one trajectory. To this end, we introduce auxiliary variables a_{tp} describing the assignment $A^* : T^* \to \mathcal{P}$, and variables b_p denoting if at least one trajectory is matched to pattern p. Formally, this can be written as

$$\begin{aligned} a_{tp} &\in \{0, 1\} \; , \forall t \in T^*, p \in \mathcal{P} \; , \\ b_p &\in \{0, 1\} \; , \forall p \in \mathcal{P} \; , \\ \sum_{p\in\mathcal{P}} a_{tp} &= 1 \; , \forall t \in T^* \; , \\ a_{tp} &\leq b_p \; , \forall t \in T^*, p \in \mathcal{P} \; . \end{aligned} \tag{9}$$

Given that C is defined as the fraction from Eq. (3), we use an optimization scheme similar to the one described at the end of Sec. 4.1, where we perform binary search to find the optimal value of α such that there exists a feasible solution for constraints of Eq. (9) as well as:

$$\begin{aligned} &\sum_{t\in T^*} \sum_{p\in\mathcal{P}} (m(t,p) - \alpha n(t,p)) a_{tp} \geq 0 \; , \\ &\sum_{p\in\mathcal{P}} b_p \leq \alpha_p \; , \quad \sum_{p\in\mathcal{P}} b_p M(p) \leq \alpha_c \; . \end{aligned} \tag{10}$$

5. Non-Markovian Multiple Object Tracking

Given that we can learn patterns from a set trajectories, we can now enforce long-range behavioral patterns when linking a set of detections. This is in contrast to approaches enforcing local smoothness constraints, that is, Markovian.

If annotated ground-truth trajectories T^* are available, we use them to learn the patterns as described in Sec. 4.2. Then, at test time, we use the linking procedure of Sec. 4.1.

If no such training data is available, we can run an E-M-style procedure, very similar to the Baum-Welch algorithm [37]: We start from a set of trajectories computed using a standard algorithm, use them to compute a set of patterns, then use the set of patterns to improve trajectories, and iterate. In practice, this yields results that are very similar to the supervised case in terms of accuracy but much slower because we have to run through many iterations.

Name	Annotated length, s	FPS	Trajectories
Duke	5100	60	7000+
Town	180	2.5	246
ETH	360	4.16	352
Hotel	390	2.5	175
Station	3900	1.25	12362
Rene	30 of 300	30	27

Table 1. Dataset statistics. The number of trajectories is calculated as a total sum of number of trajectories in each test set on which we evaluated. All test sets were approximately 1min long.

This alternate optimization is the key to making the computation tractable and making its components replaceable.

More specifically, each iteration of our unsupervised approach involves *i)* finding a set of patterns P^i given a set of trajectories T^{i-1}, *ii)* finding a set of trajectories T^i given a set of patterns P^i, as described in Sec. 4.2 and 4.1.

In practice, for a fixed maximum number of patterns α_c, this scheme converges after few iterations. Since the optimal α_c is unknown *a priori*, we start with a small α_c, perform 5 iterations, increase α_c, and repeat until we reach a predefined maximum number of patterns. To select the best trajectories without reference to ground truth, we define

$$\widetilde{\textbf{IDF}_1}(T^i) = \frac{1}{2}(C(T_1^i, P_2^i, A_{T_1^i \to P_2^i}) + C(T_2^i, P_1^i, A_{T_2^i \to P_1^i}))\ ,$$

where T_1^i and T_2^i are time-disjoint subsets of T^i, P_1^i and P_2^i are patterns learned from T_1^i and T_2^i. $A_{T_1^i \to P_2^i}$ and $A_{T_2^i \to P_1^i}$ are such assignments of trajectories to the patterns learned on another subset that maximize $\widetilde{\textbf{IDF}_1}(T^i)$.

In effect, $\widetilde{\textbf{IDF}_1}$ is a valid proxy for $\textbf{IDF}_1$ due to the many similarities between our cost function and $\textbf{IDF}_1$ outlined in Sec. 3.3. In the end, we select the trajectories that maximize $\widetilde{\textbf{IDF}_1}$. Using such cross-validation to pick the best solution in E-M models is justified in [1].

6. Evaluation

In this section, we demonstrate the effectiveness of our approach on several datasets, using both simple and sophisticated approaches to produce the initial trajectories, which we then improve as discussed in Section 5.

In the remainder of this section, we first describe the datasets and the tracking algorithms we rely on to build the initial graphs. We then discuss the experimental protocol. Finally, we present our experimental results.

6.1. Datasets

We use **Duke** [74], **Town** [51, 9], **Station** [98], **MOT16** [61], **ETH** and **Hotel** [67] datasets for people tracking. We use **Rene** [40] for vehicle-tracking, and provide additional results on [79] data. Textual description of the datasets is available in supplementary materials. Dataset statistics are shown in Table 1. These datasets share several characteristics that make them well suited to test our approach in challenging conditions. First, they feature real-life behaviors as opposed to random and unrealistic motions acquired in lab settings. Second, many of them feature frame rate below 5 frames per second, which is representative of outdoor surveillance setups but makes tracking more difficult.

6.2. Baselines

As discussed in Section 3.2, we use as input to our system trajectories produced by recent MOT algorithms. In Section 6.4, we will show that imposing our pattern constraints systematically results in an improvement over the numerous baselines listed below.

On various datasets we compare to the following approaches: two highest-ranking approaches of 2DMOT2015 [51] with publicly available implementation at the time of writing, namely **MDP** [88] and **SORT** [16]; ECCV 2016 MOT Challenge winner **DM** [80, 81]; various other 2DMOT2015 top scoring methods [23, 76, 84, 44, 91, 46, 85, 93] to which we will refer by the name that appears in the official scoreboard [51]. Finally, we use **RNN** [62] and **KSP** [14] as simple baselines that do not use appearance information, and compare with **BIPCC** [74] as a baseline provided for **Duke** dataset. We provide the textual description in supplementary materials.

Top scoring methods from the 2DMOT2015 benchmark on the **Town** dataset rely on a people detector that is not always publicly available. We therefore used their output to build the detection graph, and report their results only on **Town**. For all others, the available code accepts a set of detections as input. To compute them, we used the publicly available POM algorithm of [30] to produce probabilities of presence in various ground locations and we kept those with probability greater than 0.5. This proved effective on all our datasets. For comparison purposes, we also tried using SVMs trained on HOG features [25] and deformable part models [29]. While their performance was roughly similarly to that of POM on **Town**, it was much worse when the people are far away or seen from above. For cars, we used background subtraction followed by blob detection.

6.3. Experimental Protocol

The data is split one minute long validation and test sequences, and the rest is used for training. Results are averaged for all test intervals which we select in a leave-one-out fashion. We follow this protocol for most of the sequences since the shortest sequence is only 3 minutes long. Two exceptions are **Duke**, in which we trained and validated using provided training data, and evaluated on the whole test sets of 10 and 25 minutes in batch mode to show the ability of our approach to handle long sequences, and **Rene**, in which we had 30 seconds of annotated data. Training data

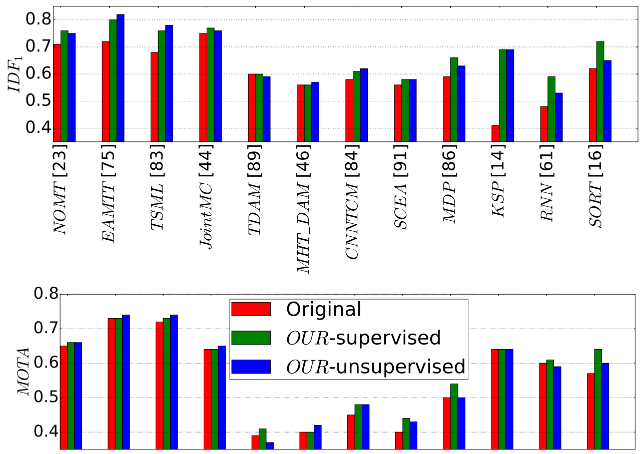


Figure 5. **IDF**$_1$ and **MOTA** scores for various methods on the **Town** dataset. Our approach almost always improves **IDF**$_1$. We provide the actual numbers in the supplementary material.

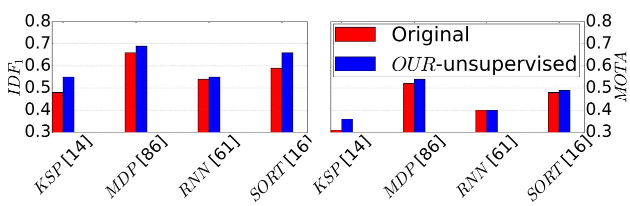


Figure 6. **IDF**$_1$ (left) and **MOTA** (right) scores on the **Rene** dataset.

trajectories were used to learn the patterns of Section 4.2. Validation data trajectories were used to optimize values of the hyperparameters D_1, D_2, D_t, R_e, α_c, α_p introduced in Sections 4.1, 4.2, using coordinate ascent.

For the sake of fairness, we trained **MDP** and **RNN**, the trainable baselines of Section 6.2, similarly and using the same data. However, for **RNN** we obtained better results using the provided model, pre-trained on the 2DMOT2015 training data, and we report these results.

Since for some approaches we only had results in the form of bounding boxes and had to estimate the ground plane location based on that, this resulted in large errors further away from the camera. For this reason, we evaluated **MOTA** and **IDF**$_1$ assuming that a match happens when the reported location is at most at 3 meters from the ground truth location. We also provide results for the traditional 1 meter distance in the supplementary material and they are similar in terms of method ordering. For the **Station** and **Rene** datasets, we did not have the information about the true size of the floor area, as we only estimated the homography between the image and ground plane. That is why we used a distance that is 10% of the size of the tracking area.

6.4. Results

IDF$_1$ **and MOTA.** Here we report summarized results for multiple approaches and datasets. Detailed breakdown and

Approach	Δ**IDF**$_1{}^s$	Δ**IDF**$_1{}^u$	Δ**MOTA**^{s}	Δ**MOTA**^{u}
KSP	0.16	0.15	-0.01	-0.01
MDP	0.05	0.02	0.03	-0.01
RNN	0.04	0.03	0.00	-0.02
SORT	0.04	0.02	0.06	0.00

Table 2. **IDF**$_1$ and **MOTA** improvement, delivered by our approach, averaged over all datasets. The 2nd and 4th columns correspond to the supervised case, the 3rd and 5th to the unsupervised one. Since **IDF**$_1$ scores range from 0 to 1, these represent significant improvements.

additional results on the [79] dataset is available in supplementary materials. Comparison on **Duke** and **MOT16** is also available on MOTChallenge benchmark [51].

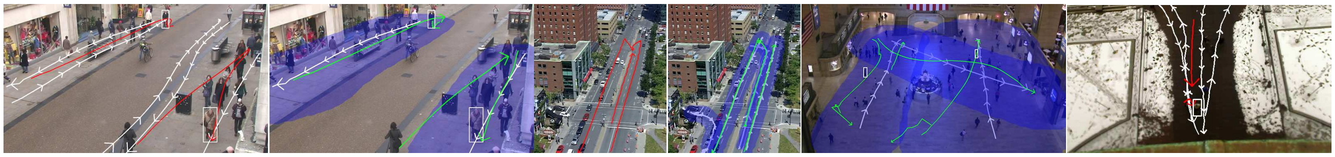
For **Duke** dataset, our supervised approach achieves +1.1% IDF on all Easy sequences combined, with improvements on 7 out of 8 sequences up to 3.7%, and one drop of 0.5%. It achieves +0.5% IDF on all Hard sequences combined, with improvements on 7 out of 8 sequences up to 8%, and one drop of 0.2%. The unsupervised approach achieves +0.9% IDF on all "trainval-mini" sequences combined, with improvements on 7 out of 8 sequences up to 4.2%, and one drop of 0.1%. Improvements are shown with respect to [74]. Examples of learned patterns are shown in Fig. 9.

Fig. 5 shows results of methods with published results on the **Town** sequence. For the 4 methods for which there is a publicly available implementation— **KSP**, **MDP**, **RNN**, **SORT**—we computed trajectories on various datasets and evaluated the improvement brought by our approach. These results are reported in Table 2 for people and Fig. 6 for cars.

As shown in Fig. 5, our supervised method improves all the tracking results in **IDF**$_1$ terms on **Town** except one that remains unchanged. The same can be said of the unsupervised version of our method except for one that it degrades by 0.01. Recall that **IDF**$_1$ ranges from 0 to 1. A 0.01 improvement is therefore equivalent to a 1% improvement and our algorithm delivers a significant performance increase. Similarly, Fig. 6 depicts original and improved car-tracking results on **Rene**, but only in the unsupervised case owing to the short length of the manually annotated sequence, which we needed for evaluation purposes.

In Tab. 2, we average improvement in people-tracking results brought by our approach for four baselines. We observe a consistent improvement in **IDF**$_1$ terms in both the supervised and unsupervised cases. As could be expected, the improvement is much less clear in **MOTA** terms because our method modifies the set of input detections minimally while **MOTA** is more sensitive to the detection quality than to identity switches. Fig. 7 depicts some of the results.

Finally, we used the output of **DM** on the two **MOT16** sequences as input to the supervised and unsupervised versions of our algorithm, as discussed above. We obtained a 37% and 25% drop in identity switches, 4% and 1% drop in number of fragmented trajectories, and 0.1% and 4% in-

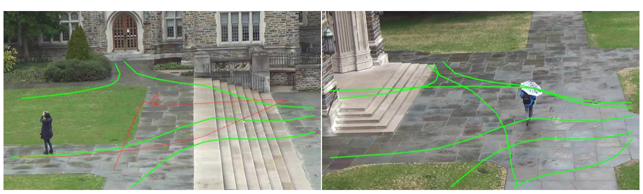


(a) (b) (c) (d) (e) (f)

Figure 7. Examples of learned patterns, denoted by their centerline in white, with some erroneous trajectories found by various baselines in red. White bounding boxes for people following the trajectories are shown. Improved trajectories found by our approach in green. Area in blue shown pattern widths, helping understand to which patterns trajectories are assigned. **(a)** **Town** dataset, **EAMTT** [76] merges trajectories going in opposite directions, but **(b)** correct pattern assignment helps to fix that; **(c)** Using only affinity information, **KSP** is prone to multiple identity switches of cars going in different directions; **(d)** Our approach correctly recovers all trajectories, including one with the turn; **(e)** On **Station** dataset our approach recovers mostly correct trajectories, but trajectories of two different people in the lower left corner going in the same general direction are merged; **(f)** **ETH** dataset, due to low visibility using flow and feature point tracking is hard, and **MDP** fragments a single trajectory into two, but our approach fixes that (not shown). Best viewed in color.

(a) (b) (c)

Figure 8. Example of unsupervised optimization. **(a)** Four people are tracked using **KSP**. Trajectories are shown as solid black lines, bounding boxes are white. Tracks feature several identity switches. **(b)** First, alternating scheme finds a single pattern, in white, that explains as many trajectories as possible, it is the leftmost trajectory. Given this pattern, next step is the tracking. Trajectories in blue are the ones assigned to this pattern, trajectories in red are assigned to no pattern. One identity switch is fixed. **(c)** After several iterations, we look for the best two patterns. Rightmost trajectory is picked as the second pattern. Fitting trajectories to the best two patterns allows to fix the remaining fragmented trajectory. Trajectories assigned to the second pattern in green.



(a) (b)

Figure 9. Examples of learned patterns on **Duke** dataset shown in green. **(a)** Some sequences contain highly non-linear patterns with turns, and our method successfully recovers them. An example of trajectory assigned to no pattern is shown in red. **(b)** A sequence with high number of patterns - each pattern goes in both directions. In such cases our model can incorrectly split an unexpected trajectory into two parts, each of which follows one pattern.

crease in **MOTA**, compared to the published results. Unfortunately, MOT'16 benchmark does not provide the **IDF**$_1$ numbers which is why we don't report them for **DM**.

Component evaluation and computational burden evaluation are described in more details in the supplementary material. In the first experiment, we measured the importance of having a non-Markovian model and learning the patterns. To do so, we replaced our learned patterns, which are often relatively straight, by a pencil of lines traversing the scene in all directions. This clearly degraded the results but not as much as replacing our patterns by a simple local smoothness term. In other words, the non-Markovian global constraints provided by the straight lines were still more powerful than the Markovian smoothness term.

Second, we assessed the influence of various terms on our method's runtime. All people tracking results reported in Figs. 5, 6 and Tab. 2 ran at an average speed of 0.906 fps for the supervised case on a 4 core 2.5Hz machine. The unsupervised computation is much slower, requiring hours for dataset of containing several hundred trajectories. However, this remains practical, as it can be run overnight, and once the patterns have been learned, the system can run in the supervised mode that can be sped up limiting the density of the graph through parameter D_1 and/or decreasing the number of binary search iterations. Using 5 instead of 10 didn't affect the IDF$_1$ by more than 1% in our experiments.

7. Conclusion

In this work we have proposed an approach to tracking multiple objects under global, non-Markovian behavioral constraints. It allows us to estimate global motion patterns using input trajectories, either annotated ground truth or ones from any sources, to guide tracking and improve upon a wide range of state-of-the-art approaches.

Our optimization scheme is generic and allows for a wide range of definitions for the patterns, beyond the ones we have used here. In the future, we plan to work with more complex patterns, account for appearance, and handle correlations between objects' behavior.

8. Acknowledgements

Andrii Maksai was supported in part by the Swiss National Science Foundation grant CRSII2-147693 "Tracking in the Wild".

References

[1] Electronic Statistics Textbook. Finding the Right Number of Clusters in k-Means and EM Clustering: v-Fold Cross-Validation. Technical report, 2010.

[2] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-fei, and S. Savarese. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *Conference on Computer Vision and Pattern Recognition*, 2014.

[3] S. Ali and M. Shah. Floor Fields for Tracking in High Density Crowd Scenes. In *European Conference on Computer Vision*, 2008.

[4] M. Andriluka, S. Roth, and B. Schiele. People-Tracking-By-Detection and People-Detection-By-Tracking. In *Conference on Computer Vision and Pattern Recognition*, June 2008.

[5] A. Andriyenko, K. Schindler, and S. Roth. Discrete-Continuous Optimization for Multi-Target Tracking. In *Conference on Computer Vision and Pattern Recognition*, pages 1926–1933, June 2012.

[6] S.-H. Bae and K.-J. Yoon. Robust Online Multi-Object Tracking Based on Tracklet Confidence and Online Discriminative Appearance Learning. In *Conference on Computer Vision and Pattern Recognition*, 2014.

[7] A. Basharat, A. Gritai, and M. Shah. Learning Object Motion Patterns for Anomaly Detection and Improved Object Detection. In *Conference on Computer Vision and Pattern Recognition*, 2008.

[8] R. Benenson, O. Mohamed, J. Hosang, and B. Schiele. Ten Years of Pedestrian Detection, What Have We Learned? In *European Conference on Computer Vision*, pages 613–627, 2014.

[9] B. Benfold and I. Reid. Guiding visual surveillance by tracking human attention, booktitle = cvpr, year = 2011.

[10] H. BenShitrit, J. Berclaz, F. Fleuret, and P. Fua. Tracking Multiple People Under Global Apperance Constraints. In *International Conference on Computer Vision*, 2011.

[11] H. BenShitrit, J. Berclaz, F. Fleuret, and P. Fua. Multi-Commodity Network Flow for Tracking Multiple People. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1614–1627, 2014.

[12] J. Bento. A Metric for Sets of Trajectories That is Practical and Mathematically Consistent. *arXiv Preprint*, 2016.

[13] J. Berclaz, F. Fleuret, and P. Fua. Multi-Camera Tracking and Atypical Motion Detection with Behavioral Maps. In *European Conference on Computer Vision*, October 2008.

[14] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple Object Tracking Using K-Shortest Paths Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):1806–1819, 2011.

[15] K. Bernardin and R. Stiefelhagen. Evaluating Multiple Object Tracking Performance: the Clear Mot Metrics. *EURASIP Journal on Image and Video Processing*, 2008, 2008.

[16] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple Online and Realtime Tracking. In *International Conference on Image Processing*, 2016.

[17] J. Black, T. Ellis, and P. Rosin. Multi-View Image Surveillance and Tracking. In *IEEE Workshop on Motion and Video Computing*, 2002.

[18] S. Blackman. *Multiple-Target Tracking with Radar Applications*. Artech House, 1986.

[19] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Online Multi-Person Tracking-By-Detection from a Single Uncalibrated Camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[20] W. Brendel, M. Amer, and S. Todorovic. Multiobject Tracking as Maximum Weight Independent Set. In *Conference on Computer Vision and Pattern Recognition*, 2011.

[21] S. Calderara, U. Heinemann, A. Prati, R. Cucchiara, and N. Tishby. Detecting anomalies in people's trajectories using spectral graph analysis. *Computer Vision and Image Understanding*, 2011.

[22] A. Charnes and W. Cooper. Programming with linear fractional functionals. *Naval Research logistics quarterly*, 1962.

[23] W. Choi. Near-Online Multi-Target Tracking with Aggregated Local Flow Descriptor. In *International Conference on Computer Vision*, 2015.

[24] W. Choi and S. Savarese. A Unified Framework for Multi-Target Tracking and Collective Activity Recognition. In *European Conference on Computer Vision*, 2012.

[25] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005.

[26] A. Dehghan, S. M. Assari, and M. Shah. Gmmcp Tracker: Globally Optimal Generalized Maximum Multi Clique Problem for Multiple Object Tracking. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[27] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012.

[28] L. Fagot-bouquet, R. Audigier, Y. Dhome, and F. Lerasle. Improving Multi-Frame Data Association with Sparse Representations for Robust Near-Online Multi-Object Tracking. In *European Conference on Computer Vision*, pages 774–790, October 2016.

[29] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[30] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua. Multi-Camera People Tracking with a Probabilistic Occupancy Map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):267–282, February 2008.

[31] K. Fragkiadaki, W. Zhang, G. Zhang, and J. Shi. Two-Granularity Tracking: Mediating Trajectory and Detection Graphs for Tracking Under Occlusions. In *European Conference on Computer Vision*, 2012.

[32] J. Giebel, D. Gavrila, and C. Schnorr. A Bayesian Framework for Multi-Cue 3D Object Tracking. In *European Conference on Computer Vision*, 2004.

[33] I. Gurobi Optimization. Gurobi optimizer reference manual, 2016.

[34] R. A. Hadi, G. Sulong, and L. E. George. Vehicle Detection and Tracking Techniques: A Concise Review. *arXiv Preprint*, 2014.

[35] W. Hu, T. Tan, L. Wang, and S. Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man, and Cybernetics*, 2004.

[36] S. Iwase and H. Saito. Parallel Tracking of All Soccer Players by Integrating Detected Positions in Multiple View Images. In *International Conference on Pattern Recognition*, pages 751–754, August 2004.

[37] F. Jelinek, L. Bahl, and R. Mercer. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 1975.

[38] H. Jeong., Y. Yoo., K. Yi, and J. Choi. Two-Stage Online Inference Model for Traffic Pattern Analysis and Anomaly Detection. *Machine vision and applications*, 2014.

[39] H. Jiang, S. Fels, and J. Little. A Linear Programming Approach for Multiple Object Tracking. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[40] J.-P. Jodoin, G.-A. Bilodeau, and N. Saunier. Urban Tracker: Multiple Object Tracking in Urban Mixed Traffic. In *IEEE Winter Conference on Applications of Computer Vision*, 2014.

[41] S. W. Joo and R. Chellappa. A Multiple-Hypothesis Approach for Multiobject Visual Tracking. *IEEE Transactions on Image Processing*, 2007.

[42] M. Kalayeh, S. Mussmann, A. Petrakova, and M. Lobo, N.and Shah. Understanding Trajectory Behavior: A Motion Pattern Approach. *arXiv preprint arXiv:1501.00614*, 2015.

[43] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, M. Boonstra, V. Korzhova, and J. Zhang. Framework for Performance Evaluation of Face, Text, and Vehicle Detection and Tracking in Video: Data, Metrics, and Protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):319–336, 2009.

[44] 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.

[45] Z. Khan, T. Balch, and F. Dellaert. MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1918, 2005.

[46] C. Kim, F. Li, A. Ciptadi, and J. Rehg. Multiple Hypothesis Tracking Revisited. In *International Conference on Computer Vision*, 2015.

[47] J. Kim and K. Grauman. Observe locally, infer globally: a space-time MRF for detecting abnormal activities with incremental updates. In *Conference on Computer Vision and Pattern Recognition*, 2009.

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

[49] C.-H. Kuo, C. Huang, and R. Nevatia. Multi-Target Tracking by On-Line Learned Discriminative Appearance Models. In *Conference on Computer Vision and Pattern Recognition*, 2010.

[50] C.-H. Kuo and R. Nevatia. How Does Person Identity Recognition Help Multi-Person Tracking? In *Conference on Computer Vision and Pattern Recognition*, 2011.

[51] L. Leal-taixe, A. Milan, I. Reid, S. Roth, and K. Schindler. Motchallenge 2015: Towards a Benchmark for Multi-Target Tracking. In *arXiv Preprint*, 2015.

[52] B. Leibe, K. Schindler, and L. Van Gool. Coupled Detection and Trajectory Estimation for Multi-Object Tracking. In *International Conference on Computer Vision*, October 2007.

[53] P. Lenz, A. Geiger, and R. Urtasun. Followme: Efficient Online Min-Cost Flow Tracking with Bounded Memory and Computation. In *International Conference on Computer Vision*, pages 4364–4372, December 2015.

[54] W. Li, V. Mahadevan, and N. Vasconcelos. Anomaly detection and localization in crowded scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014.

[55] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. Hengel. A Survey of Appearance Models in Visual Object Tracking. *ACM Transactions on Intelligent Systems and Technology*, 2013.

[56] Y. Li, C. Huang, and R. Nevatia. Learning to Associate: Hybridboosted Multi-Target Tracker for Crowded Scene. In *Conference on Computer Vision and Pattern Recognition*, June 2009.

[57] D. R. Magee. Tracking Multiple Vehicles Using Foreground, Background and Motion Models. *Image and Vision Computing*, 22(2):143–155, February 2004.

[58] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos. Anomaly detection in crowded scenes. In *Conference on Computer Vision and Pattern Recognition*, 2010.

[59] A. Maksai, X. Wang, and P. Fua. What Players Do with the Ball: A Physically Constrained Interaction Modeling. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[60] T. Mauthner, M. Donoser, and H. Bischof. Robust Tracking of Spatial Related Components. In *International Conference on Pattern Recognition*, 2008.

[61] A. Milan, L. Leal-taixe, I. Reid, S. Roth, and K. Schindler. Mot16: A Benchmark for Multi-Object Tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[62] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler. Online Multi-Target Tracking using Recurrent Neural Networks. In *American Association for Artificial Intelligence Conference*, 2017.

[63] A. Milan, S. Roth, and K. Schindler. Continuous Energy Minimization for Multitarget Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36:58–72, 2014.

[64] A. Mittal and L. Davis. M2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene. *International Journal of Computer Vision*, 51(3):189–203, 2003.

[65] S. Oh, S. Russell, and S. Sastry. Markov Chain Monte Carlo Data Association for Multi-Target Tracking. *IEEE Transactions on Automatic Control*, 2009.

[66] K. Okuma, A. Taleghani, N. de Freitas, J. Little, and D. Lowe. A Boosted Particle Filter: Multitarget Detection

and Tracking. In *European Conference on Computer Vision*, May 2004.

[67] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool. You'll Never Walk Alone: Modeling Social Behavior for Multi-Target Tracking. In *International Conference on Computer Vision*, 2009.

[68] S. Pellegrini, A. Ess, and L. Van Gool. Improving Data Association by Joint Modeling of Pedestrian Trajectories and Groupings. In *European Conference on Computer Vision*, 2010.

[69] C. Piciarelli, G. Foresti, and L. Snidaro. Trajectory clustering and its applications for video surveillance. In *Advanced Video and Signal Based Surveillance. IEEE Conference on*, 2005.

[70] H. Pirsiavash, D. Ramanan, and C. Fowlkes. Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects. In *Conference on Computer Vision and Pattern Recognition*, pages 1201–1208, June 2011.

[71] Z. Qin and C. Shelton. Improving Multi-Target Tracking via Social Grouping. In *Conference on Computer Vision and Pattern Recognition*, pages 1972–1978, June 2012.

[72] M. Ravanbakhsh, M. Nabi, H. Mousavi, E. Sangineto, and N. Sebe. Plug-And-Play CNN for Crowd Motion Analysis: An Application in Abnormal Event Detection. *arXiv preprint arXiv:1610.00307*, 2016.

[73] S. H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid. Joint Probabilistic Data Association Revisited. In *International Conference on Computer Vision*, 2015.

[74] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking. *arXiv preprint arXiv:1609.01775*, 2016.

[75] M. Rodriguez, I. Laptev, J. Sivic, and J. Audibert. Density-Aware Person Detection and Tracking in Crowds. In *International Conference on Computer Vision*, pages 2423–2430, 2011.

[76] R. Sanchez-matilla, F. Poiesi, and A. Cavallaro. Online Multi-Target Tracking with Strong and Weak Detections. In *European Conference on Computer Vision*, 2016.

[77] V. K. Singh, B. Wu, and R. Nevatia. Pedestrian Tracking by Associating Tracklets Using Detection Residuals. *IEEE Workshop on Motion and Video Computing*, pages 1–8, 2008.

[78] K. Smith, D. Gatica-Perez, and J.-M. Odobez. Using Particles to Track Varying Numbers of Interacting People. In *Conference on Computer Vision and Pattern Recognition*, 2005.

[79] R. systems s.r.o. Data from Sky User Guide, March 2017.

[80] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph Decomposition for Multi-Target Tracking. In *Conference on Computer Vision and Pattern Recognition*, pages 5033–5041, 2015.

[81] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Multi-Person Tracking by Multicut and Deep Matching. In *European Conference on Computer Vision*, 2016.

[82] J. Varadarajan, R. Emonet, and J. Odobez. A Sequential Topic Model for Mining Recurrent Activities from Long Term Video Logs. *International Journal of Computer Vision*, 2013.

[83] B. Wang, G. Wang, K. L. Chan, and L. Wang. Tracklet Association with Online Target-Specific Metric Learning. In *Conference on Computer Vision and Pattern Recognition*, 2014.

[84] B. Wang, G. Wang, K. L. Chan, and L. Wang. Tracklet Association by Online Target-Specific Metric Learning and Coherent Dynamics Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[85] B. Wang, L. Wang, B. Shuai, Z. Zuo, T. Liu, K. L. Chan, and G. Wang. Joint Learning of Convolutional Neural Networks and Temporally Constrained Metrics for Tracklet Association. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[86] X. Wang. Intelligent Multi-Camera Video Surveillance: A Review. *Pattern Recognition*, 2013.

[87] X. Wang et al. *Learning motion patterns using hierarchical bayesian models*. PhD thesis, 2009.

[88] Y. Xiang, A. Alahi, and S. Savarese. Learning to Track: Online Multi-Object Tracking by Decision Making. In *Conference on Computer Vision and Pattern Recognition*, 2015.

[89] M. Xu, J. Orwell, and G. Jones. Tracking Football Players with Multiple Cameras. In *International Conference on Image Processing*, pages 2909–2912, October 2004.

[90] C. Yang, R. Duraiswami, and L. Davis. Fast Multiple Object Tracking via a Hierarchical Particle Filter. In *International Conference on Computer Vision*, 2005.

[91] M. Yang and Y. Jia. Temporal Dynamic Appearance Modeling for Online Multi-Person Tracking. *Computer Vision and Image Understanding*, 2016.

[92] S. Yi, H. Li, and X. Wang. Pedestrian Behavior Understanding and Prediction with Deep Neural Networks. In *European Conference on Computer Vision*, pages 263–279, October 2016.

[93] J. H. Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon. Online Multi-Object Tracking via Structural Constraint Event Aggregation. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[94] S. Yu, D. Meng, W. Zuo, and A. Hauptmann. The Solution Path Algorithm for Identity-Aware Multi-Object Tracking. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[95] E. Zelniker, S. Gong, and T. Xiang. Global abnormal behaviour detection using a network of CCTV cameras. In *The Eighth International Workshop on Visual Surveillance*, 2008.

[96] L. Zhang, Y. Li, and R. Nevatia. Global Data Association for Multi-Object Tracking Using Network Flows. In *Conference on Computer Vision and Pattern Recognition*, 2008.

[97] Y. Zheng. Trajectory Data Mining: An Overview. *ACM Transactions on Intelligent Systems and Technology*, 2015.

[98] B. Zhou, X. Wang, and X. Tang. Understanding Collective Crowd Behaviors: Learning a Mixture Model of Dynamic Pedestrian-Agents. In *Conference on Computer Vision and Pattern Recognition*, 2012.