

Contents lists available at [SciVerse ScienceDirect](#)

# Computer Vision and Image Understanding

journal homepage: [www.elsevier.com/locate/cviu](http://www.elsevier.com/locate/cviu)

## Tracking with a mixed continuous-discrete Conditional Random Field<sup>☆</sup>

Stefano Pellegrini<sup>\*</sup>, Luc Van Gool

Computer Vision Laboratory, ETH Zurich, Switzerland

### ARTICLE INFO

#### Article history:

Received 28 February 2012

Accepted 11 September 2012

Available online xxx

#### Keywords:

Tracking

Social interaction

Particle Belief Propagation

Sequential sampling

### ABSTRACT

Tracking algorithms are an indispensable prerequisite for many higher-level computer vision tasks, ranging from surveillance to animation to automotive applications. A complete tracker is a complex system with many modules that need to cooperate. It is important to exploit all the sources of information, such as the appearance, the physical constraints and, though less commonly used, social factors like the walking patterns of people that belong to the same group. Given this complexity, a tracker often resorts to ad hoc solutions and scene specific customizations to improve the performance. We propose here a multi-target tracking model that succeeds in uniformly including the mentioned sources of information and is amenable to further extensions. We build our model within the Conditional Random Field framework. As the model cannot be globally optimized, we adopt an approximate inference strategy. Therefore we use a recently published sampling-based inference method that we customize to our needs and show the effectiveness of the choice in the experimental results.

© 2012 Elsevier Inc. All rights reserved.

### 1. Introduction

Tracking algorithms are an indispensable prerequisite for many higher-level computer vision tasks, including surveillance, animation and automotive applications. Advances in observation models, such as object detectors or classification-based appearance models, have enabled tracking in previously infeasible scenarios. Still, tracking remains a challenging problem, especially in crowded environments. Indeed, the performance of a tracker is greatly affected by the target density in the scene. We can call a scene *sparse*, when the low target density causes very few occlusions lasting a short period of time. Targets in a sparse scene move freely toward their destination and few intra-target interactions can be observed. On the other end of the spectrum, we have a *crowded* scene, where no subject is fully visible and each target motion is constrained by the motion of other targets. Our work focuses on the gray area between these two extremes. Our goal is to tackle those scenarios where there is more than a single dominant scene motion pattern and the targets interact with one another while moving towards their destinations. We call this scene a *busy* scene.

A multi-target tracker is a complex system. Among the many modules that usually collaborate in a successful tracker, we have:

- an observation model, to search the images for likely target positions based on what we know about the target appearance,

- a motion model, to exploit the temporal correlation among successive target positions,
- a way to associate the available observations to the multiple targets.

Furthermore, a fully autonomous tracker should be able to initialize and terminate tracks for targets entering and leaving the scene, respectively. Given the need to employ such heterogeneous components, a tracker often resorts to ad hoc solutions and scene specific customizations to improve the performance. In contrast, we propose here a single model that contains most of the tracker components just introduced. We aim to define simple functions, each of which models a particular aspect of the tracker. One such function, for instance, models the appearance similarity of a target with a certain portion of the image. Another function models the temporal correlation of the tracks. More interestingly, we define functions to model the interactions among targets, based on position and velocity configurations. The challenge that we face is how to combine these many functions. To do this in a principled way, we use the graphical model formalism as our *language*. One of the contributions of this work is the construction of an easily configurable, extensible and modular tracking model. At design time, the model has been built mostly disregarding inference feasibility, as the focus has been kept on making it accurate and rich with features. However, a functional tracker needs to provide results in a reasonable time. In a second contribution, starting from a general inference technique, we propose a customization that exploits the application structure to achieve better performance.

As a final contribution, we show the effectiveness of the proposed method challenging sequences with both human and non-human targets.

<sup>☆</sup> This paper has been recommended for acceptance by JKA Celebration issue Guest Editors, Ph.D.

<sup>\*</sup> Corresponding author.

E-mail address: [pellegrinistefano@gmail.com](mailto:pellegrinistefano@gmail.com) (S. Pellegrini).

The paper is structured as follows. After describing the related work in Section 2, we show how to model a tracker in a probabilistic framework in Section 3. While describing the modeling phase, we pay little attention to the computational constraints and focus only on modeling accurately the main aspects of a tracker. In Section 4 however, when describing the inference phase, we relax some model constraint in order to achieve feasibility. We do this in a principled way, trying to keep as much as possible of the original model. Finally, we show the experimental results in Section 5 and conclude in Section 6.

## 2. Related work

Tracking is one of the core problems in the computer vision community (see [61] for a survey on object tracking). Fostered by recent progress in object detection, there is an impressive body of work in people tracking-by-detection [8,14,34,39,58,62,4]. All propose different ways of handling the data association problem [12], but do not take advantage of any social factors beyond spatial exclusion principles. In their “space–time event-cone tracking” [32], the authors explicitly model physical exclusion between subjects in world coordinates, however, this is restricted to the selection of the best trajectory hypotheses only—the important step of creating these hypotheses is done independently and does not cater for interactions. On the contrary, in Khan et al. [27], the authors account for interactions within the motion model. The work proposes a joint Particle Filter, where the avoidance behavior is modeled with a Markov Random Field that penalizes overlapping targets configurations, i.e. a reactive repulsion based on distance only. Our motivation is similar, but our model is different in that we use not only avoidance interactions, but also grouping ones. The grouping variables are made explicit in order to enforce other constraints on them, similar to [7]. Furthermore, in this work, rather than using a Markov Chain Monte Carlo sampling strategy, we resort to a more efficient sequential sampling scheme. Another difference is that we show the effectiveness of our approach also on the challenging and especially important task of tracking multiple people.

Target interactions are not always necessary. If a single target is to be tracked in the scene, it is indeed meaningless to take into account social interactions and tracker performance is often addressed by employing multiple heterogeneous components. In this context, [49] alleviate the drift problem of appearance based trackers by combining template matching, online random forests and optical flow in a cascade. Another way of combining trackers has been proposed in [28]. In this work the uncertainty of the tracker is accounted for with a sampling method that samples both the trackers and the states of the targets.

When the task is to track multiple targets, some researchers use more complex motion models. A first solution is that of using scene specific information. The work of [1] in crowded scenarios is an interesting example of how scene constraints can help a tracking application. While this is directed at scenarios with a single dominant motion pattern, the work of [46] propose an elegant solution to exploit the multiple patterns of motion in the scene. In both these works the inferred motion patterns are tied to a specific scene. More recently, [46] propose using a video database of crowd behaviors to learn people motion priors.

Other researchers have focused on modeling the interactions independently of the particular scene. We have already seen an example in the work of [27], where the core interaction is the distance based repulsion between targets. This is already an effective principle, the characteristics of which have been studied in the social sciences [23,16] and employed in simulation applications

[24,50], path-planning [55], for unusual event detection [37] and tracking [35].

The physical exclusion principle is not the only studied aspect of human motion behavior. Another important aspect is that of avoidance anticipation, that is people anticipate obstacles and start the avoidance maneuver earlier than what is predicted by distance based repulsion models [43]. These findings have found applications in several disciplines, among which we mention computer graphics [43,40], robotics [57,22] and computer vision, where they have also been employed for tracking [2,41,51].

More relevant for this paper is the modeling of the group interactions among pedestrians [36]. Recently, some works have shown the benefits of exploiting grouping interactions for tracking [17,42,11,20,30]. In particular the work in [11], published contemporaneously with our previous work [42], is closely related to our approach. Among the differences that separate the two works, we propose a model that propagates the grouping interactions and uses a different inference strategy. Albeit with a different motivation, the work of [21] exploits the correlation among features to predict the position of possibly invisible targets. A Generalized Hough Transform is used to let these correlated features vote for the target of interest. Similarly, [17] exploits the correlation among targets by having one target using the motion parameters of another target with probability proportional to the correlation between the two. Some other authors propose to combine data association and grouping. Data association is usually carried out at the level of the single target. In [20] the authors instead describe an extension of the Joint Probabilistic Data Association [15] based on group of individuals. Group merging and splitting are also handled by the method. Linear dynamics are still assumed and false measurements and missed detections are not taken into account. With the same motivation, [30] extend the Multiple Hypothesis Tracker algorithm [45] by hypothesizing both over data association and interaction events, like group splitting or merging. Both works carry out clustering of the observations at each time step. Yamaguchi et al. [59] build a prediction model that explicitly exploits grouping and destination estimates. They show how the destination prediction accuracy does not improve significantly when more frames from the past are available, while an improvement is observed in the group prediction accuracy.

Recently several authors tackled the multi-target tracking using a graph formulation of the problem, by representing single or sequence of detections as graph nodes and modeling interactions through, possibly weighted, edges. In this context [9] build a graph with tracklets as nodes and edges connecting two tracklets that have at least one detection in common. The set of nodes that (approximately) solve the maximum weighted independent set problem on this graph is then used to carry out multi-target tracking. Interactions such as grouping or avoidance are not explicitly modeled, but tracklets that rapidly change the velocity correlation over time are penalized by connecting them with a weighted edge. [60] also cast the multi-target tracking problem in a graphical model framework. Differently from our approach, their nodes represent pairs of tracklets built on the output of a detector. The learned pairwise terms among the nodes encode both motion and occlusion dependencies. Finally, the problem is solved using simulated annealing, starting from an initial solution obtained by applying the Hungarian algorithm on a unary-only instantiation of the graph. Single tracklets are modeled instead by nodes in [52], while the edges associate pairs of tracklets. In this paper, the prediction in motion and feature space of one tracklet is used to compute a similarity measure with other tracklets. This similarity is then used as a weight for the graph edges and the optimal solution is computed using the Hungarian algorithm. Incorrect associations are compensated for by using a graph evolution strategy that adapts the weights of the graph based on long-term con-

sistency of the connected tracklet features. Other authors [62,31,44] use a minimum cost-flow algorithm. Leal-Taixé et al. [31] take into account both avoidance, similarly to the Social Force model, and some form of grouping behavior. These approaches, differently from what we propose in this work, solve for the Maximum A Posteriori (MAP) estimate of their equivalent probabilistic formulation of the problem.

Tracking is often used as a preprocessing step to carry out scene analysis. The knowledge of people tracks is an effective input for understanding people interaction and analyzing group composition. [19] use a bottom up clustering algorithm to discover the groups in the crowd while in [47] the group detection is carried out jointly with group activities discovery. In [13] the groups are detected using the head orientation in a voting scheme. A probabilistic group relationship is instead used in [10] to recognize group behaviors. In this paper we show how we can extract and use similar information *while* tracking, and not *after*.

As already mentioned, this paper builds on our previous work [42], however it strongly differentiates from it in several aspects:

- The model comprises of continuous state variables for the targets, repeated at each time step, rather than only discrete ones representing candidate trajectories provided by an external tracker module. In this work the tracking is therefore done inside the proposed framework.
- We propose a fully autonomous tracker, capable of initialization and destruction of targets, in contrast with a ground truth based initialization and a lack of termination mechanism of our previous approach.
- As a consequence of the presence of continuous variables, we need to employ a different inference method.
- We still perform, albeit differently, inference within a time window, but we propose a way to concatenate the results in a fully functional tracker and we show human intervention free results on minutes of video.

Pellegrini et al. [42] is mostly a data-association method, that does not allow tracking a whole sequence unless other components of the tracker are provided, such as a scheme to concatenate the results of the window inference. Inspired by that work, this work proposes a complete tracker that can be used on arbitrarily long sequences and an extensible framework for exploiting interactions and potentially other priors in a uniform way.

### 3. Model

In this paper we use graphical models, and in particular factor graphs [6], to write as many tracker components as possible, including the observation model, the motion model, the interaction model and the termination procedure. We use a log-linear Conditional Random Field [29]

$$p(\mathbf{a}|\mathbf{D};\theta) = \frac{1}{Z(\theta)} e^{-\sum_k f^k(\mathbf{a}_k)\cdot\theta_k}, \quad (1)$$

where  $\mathbf{a}$  is the vector that concatenates all the model variables,  $\mathbf{D} = [D^0, \dots, D^T]$  is the vector of the evidence,  $\theta$  is the vector of model parameters,  $Z(\theta)$  is the partition function and  $f^k$  are the feature functions on a subset  $\mathbf{a}_k$  of variables, with the corresponding subset of parameters  $\theta_k$ .

Casting the problem in a well known and studied framework makes it easy to exploit the techniques and state of the art methods that have been developed for that framework. Furthermore, although sometimes underestimated, re-usability, extensibility and modularity are positively affected by this choice.

The effort in writing the tracker as a graphical model consists of defining the variables of interest  $\mathbf{a}$  (the nodes of the graph) and specifying feature functions  $f_k$  (the links of the graph) to model the properties of individual, and group of, variables.

#### 3.1. Model variables

The core variable of interest is the position at timestep  $t$  for a target  $i = 1, \dots, N$ . In this paper a target is represented by the 2D projection on the plane of a reference point  $\mathbf{p}^t : [p_x^t, p_y^t]$ . We do not use a scale parameter to model the appearance, but this extension is possible and straightforward. In order to better exploit temporal consistency, it is very useful to also include in the state an estimate of the target velocity  $\mathbf{v}^t : [v_x^t, v_y^t]$ .

We want to account for false positive initialization and in general of tracker mistakes in a coherent way. Once again, we would like to keep this reasoning within the graphical model framework. In order to do so, we define a binary variable  $u^t$  for each target. This variable takes the value 1 when the estimate, for any reason, is believed to be *unreliable*. All the subject specific variables can be concatenated in the mixed continuous-discrete state vector

$$\mathbf{s}^t = [\mathbf{p}^t \ \mathbf{v}^t \ u^t]. \quad (2)$$

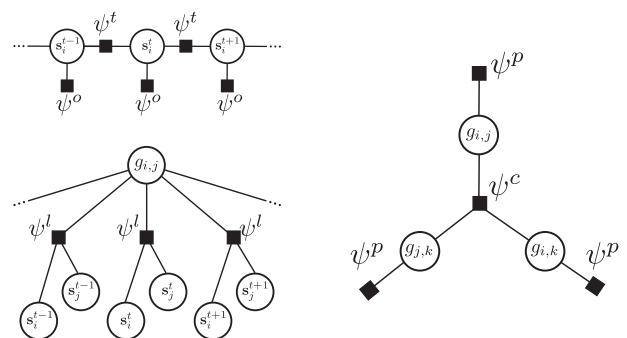
Note that the positions and velocities, albeit continuous, are assumed to be bounded by a limited region of the world and by a maximum reasonable speed, respectively.  $\mathbf{s}_i^t$  is represented as a node in the graphical model, as can be seen in Fig. 1.

The other variable that we are interested in estimating is the *grouping* variable. The group membership among two targets  $i$  and  $j$  can be represented with a binary variable  $g_{ij}$  that takes value 1 if the two subjects are in the same group and 0 otherwise. In this paper, we say that two people belong to the same group if they walk or stand together.

We can finally concatenate all the state and group variables to write  $\mathbf{a} = [\mathbf{s}_i^0, \dots, \mathbf{s}_i^T, \dots, \mathbf{g}_{ij}]$ , with  $i, j = 1, \dots, N$ .

#### 3.2. Motion model

For each target, a pair of temporally consecutive state variables are strongly correlated. This correlation is usually exploited in a tracker by means of a motion model. The most common choice of a motion model in this regard is represented by a constant velocity model with some added noise, often normally distributed. Some authors have shown that using a more complex motion model can



**Fig. 1.** The model sub-structures. Upper left: the chain portion of the graph that represents a target  $i$ . The  $\psi^o$  factors implement the observation model, while the  $\psi^t$  ones implement the motion model. Lower left: the portion of the graph of the deals with modeling the interactions between two targets  $i$  and  $j$ . Note that all the interactions factors  $\psi^l$  are connected with the unique time independent variable  $g_{ij}$ . Right: the portion of the graph that handles the consistency among the group assignments with the transitivity factor  $\psi^c$  together with the prior  $\psi^p$  on the group assignment.

**Table 1**  
Detailed description of the energy functions used to model the various building blocks of a tracker. First part.

Motion functions	
$f_p^m(\mathbf{s}_i^t, \mathbf{s}_i^{t+1}) = \begin{cases} \ \mathbf{p}_i^{t+1} - (\mathbf{p}_i^t + \Delta \mathbf{v}_i^t)\ ^2 & \text{if } u_i^t = u_i^{t+1} = 0 \\ \kappa & \text{if } u_i^{t+1} = 1 \\ \infty & \text{else} \end{cases}$	These motion functions assign a cost to the deviation from the constant velocity model $\begin{bmatrix} \mathbf{p}_i^{t+1} \\ \mathbf{v}_i^{t+1} \end{bmatrix} = \begin{bmatrix} \mathbf{p}_i^t + \Delta \mathbf{v}_i^t \\ \mathbf{v}_i^t \end{bmatrix}$ , while an infinite cost is assigned for the transition from ( $u_i^t = 0$ to $u_i^{t+1} = 1$ ). The transition to the unreliable state has a fixed cost
$f_v^m(\mathbf{s}_i^t, \mathbf{s}_i^{t+1}) = \begin{cases} \ \mathbf{v}_i^{t+1} - \mathbf{v}_i^t\ ^2 & \text{if } u_i^t = u_i^{t+1} = 0 \\ \kappa & \text{if } u_i^{t+1} = 1 \\ \infty & \text{else} \end{cases}$	
Observation functions	
$f_a^o(\mathbf{s}_i^t) = \begin{cases} \text{app}(\mathbf{p}_i^t) & \text{if } u_i^t = 0 \\ 0 & \text{else} \end{cases}$	An appearance based online classifier $\text{app}(\cdot)$ . Lower energy are assigned to locations that are likely to contain the object
$f_d^o(\mathbf{s}_i^t) = \begin{cases} \text{det}(\mathbf{p}_i^t) & \text{if } u_i^t = 0 \\ 0 & \text{else} \end{cases}$	The detector function $\text{det}(\cdot)$ assign low energy to input which correspond to positions in the image with high detector score
$f_u^o(\mathbf{s}_i^t) = u_i^t$	The cost of being in the <i>unreliable</i> state
Group functions	
$f^c(\mathbf{g}_{ij}, \mathbf{g}_{jk}, \mathbf{g}_{ki}) = \begin{cases} 1 & \text{if } \mathbf{g}_{ij} + \mathbf{g}_{jk} + \mathbf{g}_{ki} = 2 \\ 0 & \text{else} \end{cases}$	This returns a cost for those configurations of grouping that violate the transitivity property
$f^p(\mathbf{g}_{ij}) = \mathbf{g}_{ij}$	A fixed prior on the grouping variable $\mathbf{g}_{ij}$

lead to better performance in certain circumstances [2,41]. For example, a more complex motion model can include the avoidance behavior and thus represent a more informative prior during occlusions. In this work we separate the motion model from the interaction model. For the motion model, we use a constant velocity model in the form of pairwise energy functions  $f^m(\mathbf{s}^t, \mathbf{s}^{t+1}) \rightarrow \mathbb{R}$ , that assign low energy to pairs of arguments that deviate little from a constant velocity assumption (see Table 1 for more details). We also need to deal with the case of unreliable estimates. If at time  $t$  a target  $i$  is believed unreliable ( $u_i^t = 1$ ), we forbid a possible recovery by assigning an infinite cost to the transition  $u_i^t = 1 \rightarrow u_i^{t+1} = 0$ . This is done in order to avoid modeling the target motion when a track is lost, which is particularly complex, since we make no assumptions about the circumstances that caused the tracking loss. We prefer to re-initialize the track for the target as a means to recover from failure.

The graphical model factor that represents the motion model is defined as

$$\psi^m(\mathbf{s}_i^t, \mathbf{s}_i^{t+1}) = e^{-\mathbf{f}^m \cdot \theta^m} \tag{3}$$

with

$$\mathbf{f}^m = \begin{bmatrix} f_p^m(\mathbf{s}_i^t, \mathbf{s}_i^{t+1}) \\ f_v^m(\mathbf{s}_i^t, \mathbf{s}_i^{t+1}) \end{bmatrix}, \quad \theta^m = \begin{bmatrix} \theta_p^m \\ \theta_v^m \end{bmatrix},$$

where  $f_p^m$  and  $f_v^m$  are defined in Table 1. The factor  $\psi^m$  is shown in Fig. 1.

### 3.3. Observation model

A crucial part of every tracker is the observation model. This component scores state hypotheses by evaluating feature correspondences between the target model and the data. As the observation model operates independently for each subject and each timestep, we represent it with unary energy functions of the form  $f^o(\mathbf{s}^t) \rightarrow \mathbb{R}$ . In particular, we use a target-class specific detector trained offline [18] and then wrapped in a trained logistic function. The detector function,  $f_d^o$ , assigns low energy when the detector output is high. Also, we model the appearance by encapsulating an on-line classifier [48] in another feature function,  $f_a^o$ . Finally, with the function  $f_u^o$ , we model within the observation model the cost of being in the unreliable state (see Table 1).

The graphical model factor that represents the observation model is defined as

$$\psi^o(\mathbf{s}_i^t | \mathbf{D}) = e^{-\mathbf{f}^o \cdot \theta^o} \tag{4}$$

with

$$\mathbf{f}^o = \begin{bmatrix} f_a^o(\mathbf{s}_i^t) \\ f_d^o(\mathbf{s}_i^t) \\ f_u^o(\mathbf{s}_i^t) \end{bmatrix}, \quad \theta^o = \begin{bmatrix} \theta_a^o \\ \theta_d^o \\ \theta_u^o \end{bmatrix},$$

where the  $f^o$  functions are defined in Table 1.

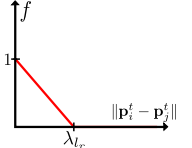
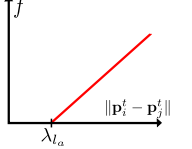
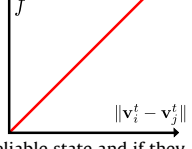
### 3.4. Interaction model

The model described so far is a simple chain graph (Fig. 1, upper left), where each target is modeled independently from all the others. As we are interested in multi-target tracking, it is reasonable to consider what kind of interactions we might exploit. One first kind of interaction is the physical one that forbids two targets to occupy the same position at the same time. But there is more to the simple exclusion, as we already mentioned in Section 1. In particular, it is reasonable to expect that the interaction is different when the two subjects belong to the same group than when they do not know each other [36]. We define a set of energy functions  $f^l(\mathbf{s}_i^t, \mathbf{s}_j^t, \mathbf{g}_{ij}) \rightarrow \mathbb{R}$  that assign a low value to more likely configurations of state and grouping variables. As mentioned we are interested in modeling the physical avoidance or repulsion ( $f_r^l$ ), but also the attraction when two subjects belong to the same group ( $f_a^l$ ) and the fact that two people in the same group have similar velocity ( $f_s^l$ ). See again Table 2 for the specific choices of the feature functions.

Note that the interaction functions all return a fixed cost when one of the interacting arguments is in the unreliable state. The reason for this choice is that the interaction, while propagating useful information from one variable to the neighboring ones, can lead to wrong estimates when erroneous information is propagated during inference. Say, for instance, that there is a target that is estimated to occupy a certain position  $\mathbf{p}_i^t$ . According to the interaction model that we are describing, we want to forbid any other target to get too close to the same region of space in the same time. Now, if the estimate  $\mathbf{p}_i^t$  is unreliable and there is actually no real target in that location, this limitation is a mistake. We want therefore to *turn off* this interaction whenever we infer that an estimate is unreliable. Assigning a fixed cost  $\kappa$  in the interaction functions whenever one of the two arguments is in the unreliable state allows to achieve the desired result.

**Table 2**

Detailed description of the energy functions used to model the various building blocks of a tracker. Second part.

Interaction functions	
$f_r^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) = \begin{cases} \max\left(0, 1 - \frac{\ \mathbf{p}_i^t - \mathbf{p}_j^t\ }{\lambda_{lr}}\right) & \text{if } u_i^t = u_j^t = 0 \\ \kappa & \text{else} \end{cases}$	<p>This models a repulsion effect when the targets distance is less than <math>\lambda_{lr}</math>. It does not depend on whether the two targets are in the same group or not (<math>g_{ij}</math> is not used)</p> 
$f_a^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) = \begin{cases} \max\left(0, \frac{\ \mathbf{p}_i^t - \mathbf{p}_j^t\ }{\lambda_{la}} - 1\right) g_{ij} & \text{if } u_i^t = u_j^t = 0 \\ \kappa & \text{else} \end{cases}$	<p>This models the attraction between two targets of the same group. It is zero if the targets distance is less than <math>\lambda_{la}</math>. If targets <math>i</math> and <math>j</math> are not in the same group, the function has no effect</p> 
$f_s^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) = \begin{cases} \frac{\ \mathbf{v}_i^t - \mathbf{v}_j^t\ }{\kappa} g_{ij} & \text{if } u_i^t = u_j^t = 0 \\ \kappa & \text{else} \end{cases}$	<p>A pair of targets in the same group have similar velocity. A linear cost is assigned to the norm of the velocity difference</p> 
$f_b^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) = \begin{cases} g_{ij} & \text{if } u_i^t = u_j^t = 0 \\ \kappa & \text{else} \end{cases}$	<p>This function acts as a bias term. It returns one if the two targets are not in the unreliable state and if they are in the same group</p>

The graphical model factor (see Fig. 1) that represents the interaction term is defined as

$$\psi^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) = e^{-\mathbf{f}^l \cdot \boldsymbol{\theta}^l} \quad (5)$$

with

$$\mathbf{f}^l = \begin{bmatrix} f_r^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) \\ f_a^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) \\ f_s^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) \\ f_b^l(\mathbf{s}_i^t, \mathbf{s}_j^t, g_{ij}) \end{bmatrix} \quad \boldsymbol{\theta}^l = \begin{bmatrix} \theta_r^l \\ \theta_a^l \\ \theta_s^l \\ \theta_b^l \end{bmatrix}$$

We choose not to integrate out the grouping variable  $g_{ij}$  in order to be able to explicitly model some relational properties on it, similarly to other works [7,53]. In our definition, grouping is an equivalence relation, i.e. it fulfills reflexivity, symmetry, and transitivity. While reflexivity and symmetry are enforced by the graph construction, the transitivity is not. We therefore define a function  $f^c(-g_{ij}, g_{jk}, g_{ki}) \rightarrow \{0, 1\}$  that assigns zero energy to triplet of arguments that respect the transitivity property. In other words, when  $i$  and  $j$  are assigned to the same group (so that  $g_{ij}$  takes the value 1) and  $j$  and  $k$  belong to the same group ( $g_{jk}$  takes values 1), then  $k$  and  $i$  must also belong to the same group (therefore  $g_{ik}$  should be 1).

The graphical model factor that encodes this constraint is defined as

$$\psi^c(g_{ij}, g_{jk}, g_{ki}) = e^{-f^c \theta^c} \quad (6)$$

Finally, we also define a function  $f^p(g_{ij}) \rightarrow \{0, 1\}$  that acts as a prior on the grouping variable. The graphical model factor for the group prior is defined as

$$\psi^p(g_{ij}) = e^{-f^p \theta^p} \quad (7)$$

### 3.5. Connectivity

So far, we deliberately decided not to specify which pairs of subjects nodes are connected to each other (and to the corresponding grouping variable). We could indeed use link  $\psi^l$  to connect all the pair of subjects that co-exist in the same time step (as a conse-

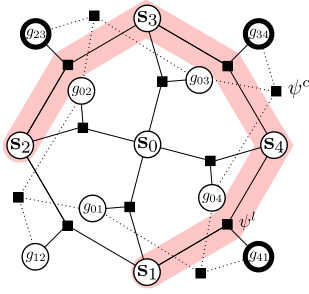
quence, we could connect all the possible triplets of nodes  $g_{ij}$ ). Doing so would mean that the interactions occur regardless of the distance and the reciprocal visibility of a pair of targets. Furthermore, the increase in connectivity, would increase the inference computational requirements. We instead assume that a target  $i$  interacts only with the neighboring targets  $j$ . Let us define the set  $\mathcal{D}^t(\mathbf{P}^t)$  as the set of pairs of targets  $i, j$  that are connected after a Delaunay triangulation on the target positions  $\mathbf{P}^t := [\mathbf{p}_1^t, \dots, \mathbf{p}_N^t]$ . We can now define the set  $\mathcal{N}^t$  of neighboring targets at time  $t$  as

$$\{i, j\} \in \mathcal{N}^t \quad \text{if } \{i, j\} \in \mathcal{D}^t \wedge d(\mathbf{p}_i^t, \mathbf{p}_j^t) \leq \lambda_{con}, \quad (8)$$

where  $d(\cdot, \cdot)$  is the Euclidean distance function and  $\lambda_{con}$  is a threshold parameter. We also choose to apply the transitivity constraint locally. In particular we add a transitivity function for a triplet of subjects reciprocally connected by the Delaunay triangulation. While this solution still enforces the property locally, it does not model global transitivity, as Fig. 2 shows. We could prevent this violation by increasing the number of  $\psi^c$  factors to connect more and more grouping nodes. However, we believe that the local transitivity is a sufficient constraint and further complexity would offer little benefit.

### 4. Inference

In Section 3 we focused on building a model that could properly represent the main component of a multi-target tracker. In this section we deal with how to estimate the model variables. We did not concern ourselves with the computational feasibility issues so far. It turns out that the model that we described is inherently complex. One first issue is the connectivity of the graph. In the general case, the graph contains loops. Also, no assumption was made on any particular structure for the feature functions. Another, more important obstacle for exact inference lays in the fact that some of the variables of interest are, at least partially, continuous, albeit bounded. This is the case for the position  $\mathbf{p}$  and the velocity  $\mathbf{v}$  variables. Discretizing this variables jointly would require choosing a proper resolution parameter and, above all, would probably have unfeasible memory requirements. Finally, there is a problem due to the size of the problem itself. Multiple targets can be present



**Fig. 2.** The grouping nodes with bold borders are assigned a value 1, meaning they assign the two neighboring target nodes, e.g.  $s_1$  and  $s_4$ , to the same group. The others are assigned a value of zero. Dotted lines connect grouping nodes to transitivity factors  $\psi^c$ . Note that  $g_{1,2}$  is assigned a zero value, implying that the subjects 1 and 2 do not belong to the same group. However, we could as well infer that subjects 1 and 4 belong to the same group ( $g_{1,4} = 1$ ), that subjects 4 and 3 belong to the same group ( $g_{3,4} = 1$ ) and that subjects 3 and 2 belong to the same group ( $g_{2,3} = 1$ ). Therefore, assuming a transitivity property for group membership, we would conclude that also subjects 1 and 2 belong to the same group. However this is not true in our model. Global transitivity is not enforced, only local transitivity (by means of the  $\psi^c$  factors) is. This example was built with the purpose of showing this aspect of the model, but such situations are in fact extremely rare.

in the scene for a long time. For each target we have a node at each timestep. We tackle all these problems in this section.

#### 4.1. Belief propagation with continuous variables

A common strategy to adopt when dealing with loopy graph, is Loopy Belief Propagation (LBP) [38]. Although in the general case it gives no optimality guarantees, this method and its variants have led to good results in the literature. In our case, as already mentioned, we have the additional problem that some the variables are continuous. Let us consider, as an example, the messages  $m$  exchanged between a state node of a target  $i$  and one of the neighboring interactions link  $\psi^l$  at a certain time. The message passing adopted in belief propagation strategies could be generalized to the presence of continuous variables as follows

$$m_{i \rightarrow \psi^l}(\mathbf{s}_i) = \prod_{\psi \in ne(i)/\psi^l} m_{\psi \rightarrow i}(\mathbf{s}_i), \quad (9)$$

$$m_{\psi^l \rightarrow i}(\mathbf{s}_i) = \sum_{g_{ij} \in \{0,1\}} \int \psi^l(\mathbf{s}_i, \mathbf{s}_j, g_{ij}) m_{j \rightarrow \psi^l}(\mathbf{s}_j) m_{g_{ij} \rightarrow \psi^l}(g_{ij}) d\mathbf{s}_j, \quad (10)$$

where  $ne(i)$  is the set of neighboring factors of the state node of target  $i$ . The integral in Eq. (10) is intractable in the general case. One solution is to resort to sampling in alternation with belief propagation [26,54]. We use the Particle Belief Propagation (PBP) [26] for our purposes. The idea behind this method is to approximately solve the integral in Eq. (10) with importance sampling. Given an importance function  $q(\mathbf{s}_i)$  we can derive the sampled approximation to Eq. (10)

$$m_{\psi^l \rightarrow i}(\mathbf{s}_i) \approx \sum_{g_{ij} \in \{0,1\}} \sum_{\mathbf{s}_j \in S_j} \psi^l(\mathbf{s}_i, \mathbf{s}_j, g_{ij}) \frac{m_{j \rightarrow \psi^l}(\mathbf{s}_j)}{q(\mathbf{s}_j)} m_{g_{ij} \rightarrow \psi^l}(g_{ij}), \quad (11)$$

where  $S_j$  is the set of samples on the state of target  $j$  drawn from  $q(\mathbf{s}_j)$ . Note that the elements of the summation are now divided by the importance weight  $q(\mathbf{s}_j)$ . Also, note that the message can be evaluated on any value in the continuous range of the variable  $\mathbf{s}_i$ .

We have therefore a way to compute the messages  $m$ , but we did not specify the importance distribution  $q$ . We discuss this in the next subsection.

#### 4.2. Sampling strategies

In [26], the authors show that the best choice of importance distribution  $q(\mathbf{s}_i)$  is the marginal distribution  $b(\mathbf{s}_i)$  for the variable  $\mathbf{s}_i$ . This distribution is not available however, and the best approximation is only available at the end of the inference process. An increasingly more accurate approximation  $\tilde{b}(\mathbf{s}_i)$  to this marginal distribution is however available at each iteration of the belief propagation, simply obtained by multiplying the incoming messages at each node

$$\tilde{b}(\mathbf{s}_i) = \prod_{\psi \in ne(i)} m_{\psi \rightarrow i}(\mathbf{s}_i). \quad (12)$$

As shown in [26] this is an effective choice. Note that this product is a continuous function.

Sampling from the product of incoming messages, in the continuous case, is not a trivial task. A possible solution is to use a Markov Chain Monte Carlo (MCMC) sampling method, such as Metropolis–Hastings.

While perfectly valid, this strategy requires using MCMC to sample from each target node at each iteration of the algorithm. This is significantly demanding in terms of computation. We are not forced in principle to use the product of messages as the sampling distribution. The MCMC sampling works in the general framework of PBP, but it does not exploit all the structure of the specific problem, in our case, the tracker application. It was already mentioned that there is a strong correlation among two consecutive state nodes for each target. We exploit this temporal correlation to propagate the samples from one to the next target nodes. We therefore devise an importance sampling scheme, similar to Briers Arnaud et al. [3], but within the PBP framework.

The nodes that we need to sample are the target nodes. The grouping node indeed represents a binary variable that needs no sampling. The idea here is to sample from the target nodes sequentially. Let us assume a directed graphical model like the one in Fig. 3 for each target. This model does not need to be identical to the one in Fig. 1, top-left. Let us define the distribution from which we would like to sample as

$$q^*(\mathbf{s}^t | D^{0,\dots,t}), \quad (13)$$

where  $D^{0,\dots,t}$  is the vector of the available evidence up to the current timestep  $t$ . Assuming the model of Fig. 3, we can rewrite the distribution as follows:

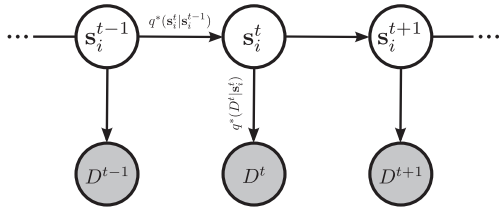
$$q^*(\mathbf{s}^t | D^{0,\dots,t}) = \int q^*(\mathbf{s}^t, \mathbf{s}^{t-1} | D^{0,\dots,t}) d\mathbf{s}^{t-1}, \quad (14)$$

$$= \frac{q^*(D^t | \mathbf{s}^t)}{q^*(D^t)} \int q^*(\mathbf{s}^t | \mathbf{s}^{t-1}) q^*(\mathbf{s}^{t-1} | D^{0,\dots,t-1}) d\mathbf{s}^{t-1}. \quad (15)$$

If, for the moment, we assume that samples  $\mathbf{s}_j^{t-1}$  are available from the distribution  $q^*(\mathbf{s}^{t-1} | D^{0,\dots,t-1})$ , then we can write

$$q^*(\mathbf{s}^t | D^{0,\dots,t}) \approx \frac{q^*(D^t | \mathbf{s}^t)}{q^*(D^t)} \sum_{\mathbf{s}^{t-1} \in S_j} q^*(\mathbf{s}^t | \mathbf{s}^{t-1}). \quad (16)$$

Eq. (16) has the form of a mixture model. Note that the denominator  $q^*(D^t)$  is not dependent on  $\mathbf{s}^t$  and therefore it can be treated as a constant. Sampling from Eq. (16) therefore reduces to a weighted sampling of the propagation density  $q^*(\mathbf{s}^t | \mathbf{s}^{t-1})$ . We see how to this below. Now, to finish showing that we can sequentially sample from this distribution, we need to show that we can sample from  $q^*(\mathbf{s}^0 | D^0)$ . We can assume that the first node of the chain for each target, is a discrete node, therefore we can always sample from it. In Section 4.4 we see that the target initialization provides us with a state from which we can easily sample.



**Fig. 3.** The directed graph model for sequential sampling. Rather than sampling from the individual belief estimates as in the PBP framework, we propose to use sequential sampling to exploit the sequential structure of the problem. The chain in the figure is used for this purpose. Here  $q^*(s_i^t | s_i^{t-1})$  is the time propagation density and is defined in Eq. (17), while  $q^*(D^t | s_i^t)$  is the observation density and is defined in Eq. (18).

Finally, we need to define the propagation density  $q^*(s_i^t | s_i^{t-1})$  and the observation density  $q^*(D^t | s_i^t)$ . We keep in mind that our goal is to have the importance function as similar as possible to the marginal for  $s_i^t$  [26]. For the propagation function, we use a function similar to the motion function described in Table 1. These energy functions implement a motion model from which we cannot directly sample, because of the presence of the unreliable state. We use instead the following propagation function:

$$q^*(s_i^t | s_i^{t-1}) = \alpha \delta(s_i^t - \hat{s}^{t-1}) + (1 - \alpha) \begin{bmatrix} \mathcal{N}\left(\begin{bmatrix} \mathbf{p}_i^t \\ \mathbf{v}_i^t \end{bmatrix}; \boldsymbol{\mu}_i^t, \boldsymbol{\Sigma}\right) \\ \delta(u_i^t - u_i^{t-1}) \end{bmatrix} \quad (17)$$

with

$$\hat{s}^{t-1} = \begin{bmatrix} \mathbf{p}_i^{t-1} \\ \mathbf{v}_i^{t-1} \\ 1 \end{bmatrix}, \quad \boldsymbol{\mu}_i^t = \begin{bmatrix} \mathbf{p}_i^{t-1} + \Delta \mathbf{v}_i^{t-1} \\ \mathbf{v}_i^{t-1} \end{bmatrix},$$

$$\boldsymbol{\Sigma} = \frac{1}{2} \begin{bmatrix} (\theta_p^m)^{-2} & 0 & 0 & 0 \\ 0 & (\theta_p^m)^{-2} & 0 & 0 \\ 0 & 0 & (\theta_v^m)^{-2} & 0 \\ 0 & 0 & 0 & (\theta_v^m)^{-2} \end{bmatrix},$$

where  $\delta(\cdot)$  has value zero everywhere except when its argument is 0,  $\alpha$  is a parameter that regulates the random switch from the reliable to the unreliable state and the variance parameters  $\boldsymbol{\Sigma}$  are the same parameters that are used to multiply the motion functions in the motion factor  $\psi^m$ . Notice that in the propagation function of Eq. (17), the  $u_i$  can only change to 1 and never to 0, in accordance with the motion model energy functions. We can sample from Eq. (17) by randomly choosing one of the two terms: we choose the first term with probability  $\alpha$  and the second term with probability  $(1 - \alpha)$ . If the first term is chosen, there is only one choice for the sample of the state  $s_i^t$ , i.e.  $\hat{s}^{t-1}$ . If the second term is chosen, position and velocity are sampled from a multivariate normal density function, while the  $u_i^t$  keeps the old value  $u_i^t$ .

The observation function  $q^*(D^t | s_i^t)$  acts as a weight for the samples from Eq. (16). As we already described for the MH sampling strategy, at each iteration of the belief propagation algorithm, we have a set of messages being sent to all the nodes. It seems therefore reasonable to use the messages coming from the neighboring factors of node  $s_i^t$ , with the exclusion of the time factors  $\psi^m$  (that are already accounted for in the propagation function), to implement the observation function

$$q^*(D^t | s_i^t) = \prod_{\psi \in ne(i) / \psi^m} m_{\psi \rightarrow i}(s_i) \quad (18)$$

Here, slightly abusing the notation, we included the messages in the available evidence. In this way the weights convey the information coming not only from the images, but also from the interacting targets.

### 4.3. Splitting the inference

So far we have seen how to adapt LBP to a graph with continuous variables. There are two further considerations. The first is that performing inference for all the graph at the same time might be prohibitive, especially in terms of memory requirements. The tracker might be used for long sequences, depending on the task, and the amount of data (samples and images) necessary for the inference might be too big to fit in memory. The other consideration is that, sometimes, tracking applications require an estimate as soon as new data becomes available. Therefore waiting for the end of the sequence to perform inference would not be appropriate in these situations.

The solution we employ is to perform the inference in temporal windows. The idea is to split the graph into partially overlapping subgraphs, and perform inference on each of these. Fig. 4 shows an illustrative example. For the sake of clarity we show only the portion of the graph relative to a single target ( $i$ ) for the first three time steps. The graph is split in such a way that the all the target chains end at a certain time (time 1 in the figure) and restart with the same node in the next subgraph. Note that the grouping nodes, like  $g_{ij}$ , are repeated in all the subgraphs. The splitting shown in Fig. 4 detaches the first graph slice from the whole graph. The procedure can be repeated (on the rightmost subgraph in figure) to obtain the other time slices.

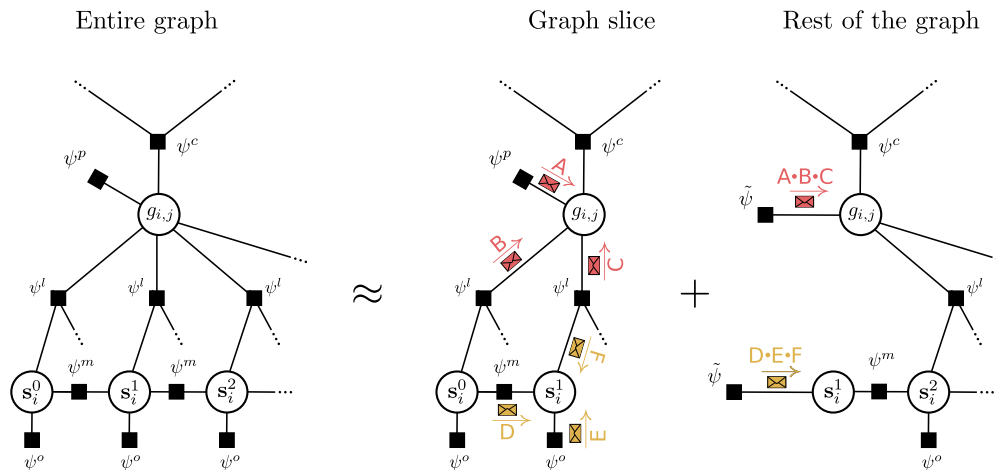
The inference is carried out for each graph slice in chronological order. To exploit the information coming from previous slices of the graph, the messages to overlapping nodes ( $g_{ij}$  and  $s_i^t$  in the figure) coming from factors  $\psi$  that are not repeated in the successive slice are multiplied and collected in special connection factors  $\tilde{\psi}$ . This is not the same as performing the inference with the whole graph at the same time. The main difference is that the information can propagate only from one graph slice to the following ones, but not backwards. Also, the overlapping target nodes are sampled only in the first graph slice in which they appear. At the end of the inference for that slice, the samples and the sampling weights are stored. When the overlapping nodes are used again in the following graph slice, they are not resampled, but the stored values are re-used.

The sequential sampling scheme introduced in Section 4.2 is also adapted to cope with the graph splitting. Since the samples and the sampling weights of the overlapping target nodes ( $s_i^t$  in figure) are stored for the PBP on the next graph slice, they can also readily be used for sequential sampling.

We employed this approximation also to be able to provide an estimate at the end of each graph slice. Sometimes, for example for visualization or for evaluation purposes, a hard decision for each variable is necessary. Rather than a max-marginal estimate for each node, we use dynamic programming for each target chain within the graph slice to obtain smoother results. The dynamic programming uses Eq. (18) to compute the unary costs for each chain node, thus accounting for observation model and interactions. The transition costs from one node to the next are evaluated using the  $\psi^m$  factor of the model. In particular, if the estimate returns the unreliable state for a target, that target is not propagated to the next graph slice.

### 4.4. External modules

In this subsection we discuss the tracker components that we did not manage to include in the graphical model framework.



**Fig. 4.** Graph slicing. The undirected graphical model on the left shows a section of the graph built for an entire sequence (see Section 3). Both because of the limited computational resources and to allow the method to return results as new data becomes available, we extract and compute the result for one slice at a time (center). This slice contains all the subject nodes for a certain time window and all the grouping nodes. In this figure, the slice contains only two consecutive time-steps, but this need not be the case in general. As the rest of the graph (right) is not used for the current computation, it does not even need to be known. This is the reason why we can apply the method sequentially as new data becomes available. Once the inference stage is complete, we repeat the same slicing operation on the remaining graph. The slices are not treated completely independently however. The result of the inference on previous slices is propagated to the following ones, as illustrated by the A, B, C (for the grouping variables) and D, E, F (for the subject variables) messages that are sent from one slice to the next. Note, however, that the information does not travel backwards.

One external component is initialization, that is, the component that decides whether one or more targets have to be initialized for tracking. We have already seen that the inference is not carried out in one single step for all the graph, but rather it is performed in a temporal window. When a new inference window is initialized, new targets might have entered the scene. We use an external module that, given the detector output and the target estimates until the current time step, infers whether there are new targets to be initialized. This is done by looking at the local maxima of the detector image, and while visiting them in order of decreasing detector score, accepting only those that do not overlap with more than a certain threshold to those that have already been initialized. This process is performed only at the beginning of each inference window. Since from the single detector image we have no estimate of the velocity of the newly initialized target, we initialize the first state samples for each target with the position given by the detector and the velocity sampled uniformly in all directions and within a reasonable speed. The first node can be considered therefore a discrete node, with all the discrete choices equally likely and therefore easy to sample from, as required in Section 4.2. Finally, to be robust with respect to poor initialization, the target for the first inference window is not linked by interaction factors  $\psi^l$  to the other targets. In other words, for the duration of the first inference window, it is tracked independently.

The other component that we could not cast into the framework is the one responsible for updating the appearance model. Once a target is initialized, the appearance model is built by using as a positive example the initialized position, and as negative examples positions taken at random in the vicinity of the positive example. The appearance model needs to be updated however, because of the well known appearance change that targets undergo. These updates are carried out using the same strategy employed for building the appearance model, but as positive examples we use samples with high detector score, similarly to [8].

#### 4.5. Implementation remarks

Finally, in this section we account for some technical non-trivial caveats that are necessary to properly implement the tracker.

With regard to propagation to the unreliable state (Eq. (17)), when a certain number of samples are propagated to the unreliable

state, the effective number of samples that represent the position and velocity part of the state decreases. Therefore the accuracy of the representation decreases and it becomes more likely for the samples to be propagated to the unreliable state. This process can favor the premature termination of a target. In order to avoid this, while resampling in Eq. (17), we try to keep the number of samples that are in the reliable state equal to their initial number, by increasing the number of output samples as new unreliable samples are drawn. However, we do not sample more than twice the initial number of samples.

The sampling procedure described in the previous section, both in the MCMC form and in the sequential sampling one, are parallelizable. In the former case, each node can be sampled independently, while in the latter this is true for each target chain. In the same way, the caching of the energy function values, as described above, can be carried out in parallel. With the widespread growth of parallel computing, and seen the resource requirement of the tracker presented in this work, this becomes necessary, rather than an option.

## 5. Experiments

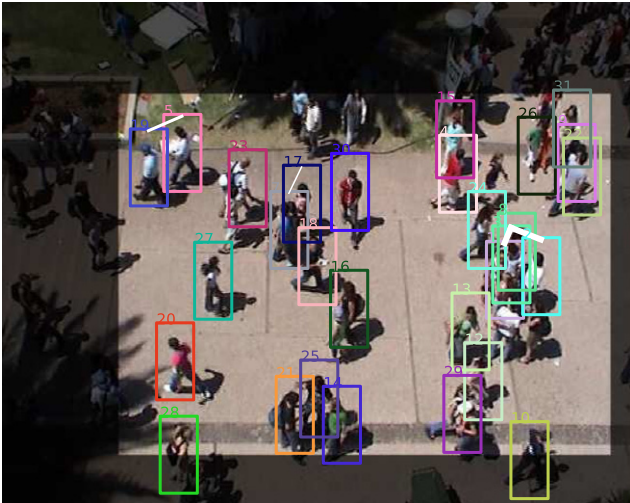
In the following we show the experiments we carried out in order to evaluate the tracker. For the evaluation, we need data that can challenge all the major aspects of a multi-target tracker. In particular, the tracker should work on a scene with several naive targets. Instructing targets to walk in various configurations might fail in reproducing those interaction patterns that we want to model. Furthermore, since we need many targets to be visible in the scene at the same time, we chose sequences with cameras set high enough to capture a big portion of the scene.

### 5.1. Datasets and setup

#### 5.1.1. Students

This is an outdoor sequence provided by a third party [33] that we manually annotated to collect ground truth. The sequence is particularly challenging due to the high number of subjects in the scene, the multiple patterns of motion, the compression artifacts and the strong shadows. Although the background is static,





**Fig. 5.** A result frame from the Students sequence. The white line connecting the bounding boxes shows the group relationship estimate. The thicker the line, the stronger is the estimate that the two connected subjects belong to the same group.

we do not use this information and rely on a person detector [18]. Given the position of the camera, the detector was trained on the same scene from a small set of frames (50). We used another small subset of the frames (300) to choose the best parameters  $\theta$ ,  $\kappa$  and  $\lambda$ . To choose the parameters we used a simple gradient descent with fixed step size, one coordinate at the time, optimizing for accuracy. After removing all the frames used for training, we were left with 4400 frames at 25 fps (i.e., about 3 min of video) that we used for testing. Fig. 5 shows a screenshot of the tracker results. The darker area in the image has not been used for tracking. This was done both to avoid border effects and to avoid dark shadows and stairs (for the top and left part of the images).

### 5.1.2. BIWI-Walking

These sequences have been made available by Pellegrini et al. [41]. They have been captured with almost top-view cameras and show people walking in a busy street (Fig. 6) and at the entrance of a public building (Fig. 7). We kept the parameters used for the Students sequence. The detector has been re-trained for the busy street sequence on a portion of the data while for the

building entrance, because of low resolution and compression artifacts, a simple background subtraction has been employed.

### 5.1.3. Mycoplasma

We also used a video sequence from Ref. [56], showing moving bacteria. As a detector we used a logistic function (trained on a single frame) on the intensity on the image. The bacteria intensity is substantially different from the background and provides a simple but effective detector. Like before, we cut out the borders of the image to overcome border effects. No ground truth was available for this sequence. For the parameter settings, we used the same parameters from the Students sequence, with the only exception of the velocity parameter  $\theta_v^m$  that was reduced to account for more irregular target motion.

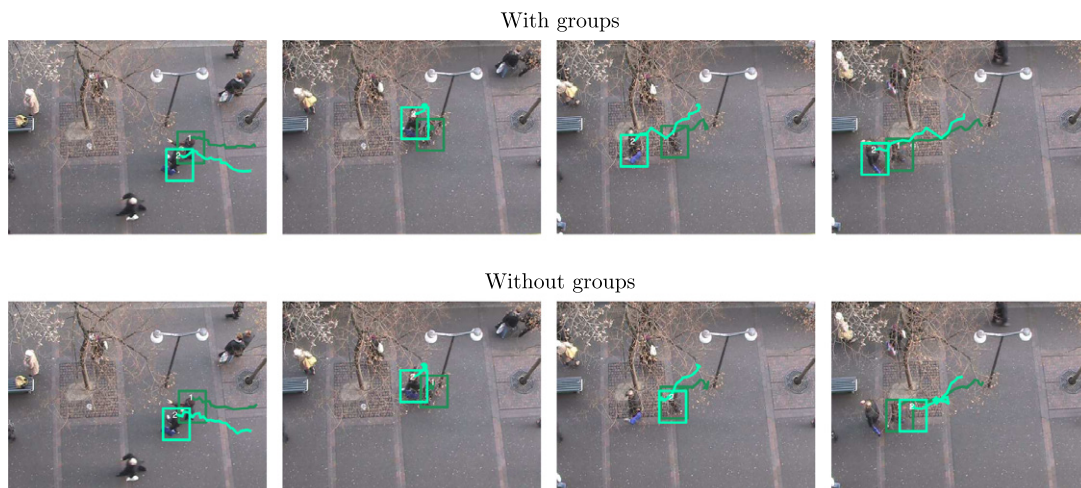
For the following experiments, we set the size of the inference window to 5, as a compromise between the computational requirements, the need of producing results as soon as new data becomes available and the desire of exploiting the temporal correlation. We use a fixed aspect ratio and scale within each sequence for all the targets, as these quantities do not vary much in the images. Finally, we use 100 samples per target and we set the interaction threshold  $\lambda_{con}$  from Eq. (8) to 2 m.

### 5.2. Grouping influence

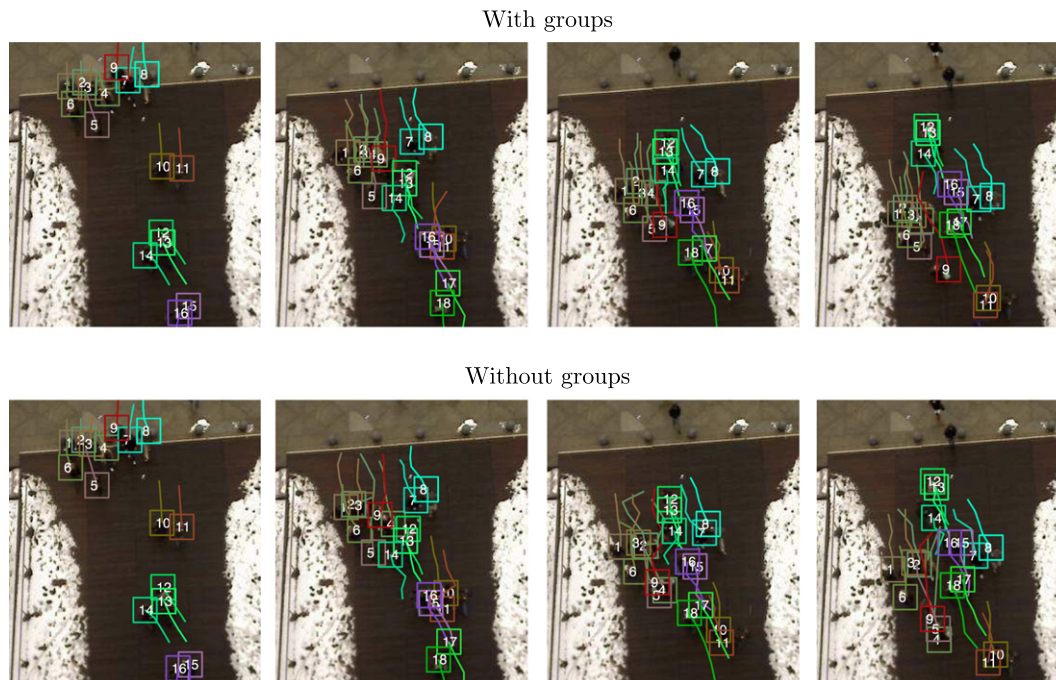
To highlight the importance of the grouping component of the tracking, we show the output of the system when tracking groups with two different setups. In the first we manually set  $g_{ij} = 1$  for all the pair of subjects  $i$  and  $j$  within the same group. In the second, we instead manually set all the groups relations to 0, that is  $g_{ij} = 0$  for all  $i$  and  $j$  in the scene.

A first example is shown in Fig. 6. The sequence is extracted from the BIWI-Walking dataset. Note that subject 1 is tracked properly in both setups, while 2 is not. The observation model is indeed weaker because of the partial occlusion of the tree branches. In the setup with the group correctly initialized the failure is avoided because the velocity of subject 2 is affected by the velocity of subject 1, that, as stated, is well tracked and pushes 2 forward.

Another example is shown in Fig. 7. This sequence is extracted from the same dataset as the previous one and shows the entrance of a public building. The two tracker settings are the ones used in the previous experiment, with the difference that now several



**Fig. 6.** Two subjects belonging to the same group. Two setups are compared: the one with the group correctly initialized (top row) and the one in which the two subjects do not belong to the same group (see text for more details). Note that subject 1 is properly tracked in both setups. The grouping component of the tracking uses the velocity of subject 1 to estimate the velocity of subject 2 when the observation model is weak due to the partial occlusion of the tree branches.



**Fig. 7.** Several groups interacting while walking in opposite directions. Subjects within the same group are assigned similar colors and ID number. In particular, we have the following groups: {1, 2, 3, 4, 5}, {7, 8}, {9}, {10, 11}, {12, 13, 14}, {15, 16} and {17, 18}. Two tracker settings are compared (see text and Fig. 6 for more details). Note that subject 9, in the second image, is incorrectly estimated going through the left-most group in the setup without groups. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

groups are extracted. The figure shows the comparison of the two tracker setups. While in this sequence, because of the poor quality of the background subtraction, the trajectory precision is much lower, the difference in accuracy still favors the setup with the groups correctly initialized. Note how the group prior favors the cohesion of the trajectories.

### 5.3. Group formation and splitting

Although our modeling of groups is only static and no merging or splitting are explicitly defined, we observe similar dynamics emerging from the inference strategy that we use. In particular, by doing inference on a time window basis, we actually update the estimate of the group relation as new evidence becomes available. Fig. 8 shows the progressive estimate of the grouping relation among few subjects. Note that eventually the local transitivity property holds for all the triplets of subjects in the figure. The group relationship is a result by itself that can be used, for example, for scene understanding [10].

Group splitting is also not directly modeled in our framework. However group sometimes split and the model should not hallucinate members walking together. If the observation model is reliable enough, this does not happen. Fig. 10 shows a case in the Students sequence. This is also an example of recovery from wrong

initialization. One major reason for false positives in the tracker is given by wrong initialization, as the detector fires often on shadows and backpacks (see Fig. 9). The tracker is anyhow capable of alleviating this problem by not trusting completely the initializer. The inference is indeed able to assign the unreliable state to those tracks that violate the interaction model and/or do not have much support from the observation model.

### 5.4. Avoidance

While the appearance model provides already a means for disambiguation among targets, when the targets look similar and the resolution is relatively low, the appearance model is not enough to prevent the stronger observation explaining multiple tracks. In Fig. 11 we compare the result of the tracker when using the interaction terms as previously described and when not using them.

In Fig. 12 we show frames from the result of the bacteria tracking experiment. In the right part of the figure we follow a single bacterium from the initialization to when it leaves of the scene. Each frame reports an interaction with other bacteria. Even if they come very close to each other, the id of the targets are preserved. Note that there is no group formation in the bacteria experiment. This happens because these kind of bacteria rarely move together.



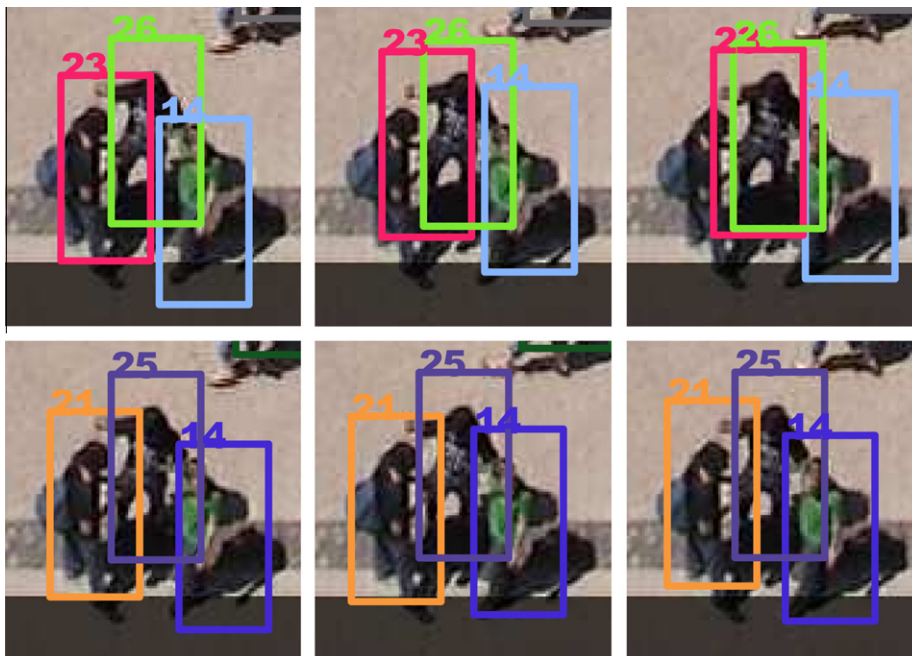
**Fig. 8.** Group formation. The thickness of the white line is proportional to the estimated probability of the two subjects being in the same group. The frames being shown are, from left to right, 4000, 4100, 4300, 4320 and 4350.



**Fig. 9.** Tracker errors. Some of the tracker errors are caused by a wrong initialization. In the left figure, the shadow is detected instead of the person. Furthermore, the shadow moves indeed like a human, and this makes it harder for the inference to infer that there is a failure. The right figure shows another mistake. This time is the bag that is being recognized as a person and is also being assigned the same group as the other two subjects.



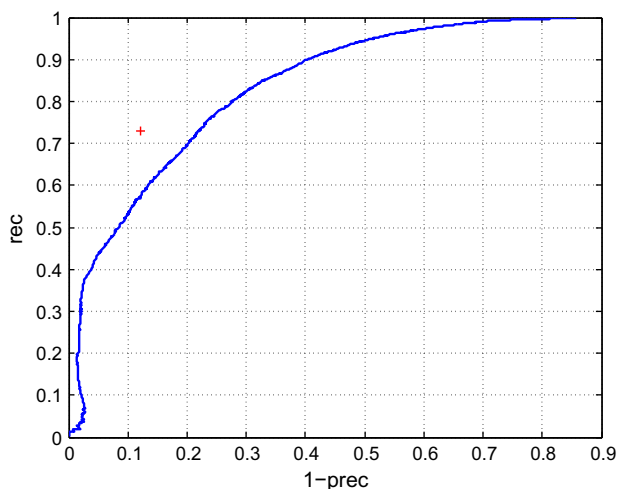
**Fig. 10.** Tracker failure detection. Sometimes, the initialization or the tracker itself lead to a wrong estimate. The leftmost figure shows such a case, where for the two subjects with ID 28 and 35, there is another pair of wrong estimates, namely 50 and 46. The tracks of interest are represented with a shaded bounding box. The tracker finds out that the estimates are wrong in the following frames. This happens because the wrong tracks violate the avoidance behaviors, since they overlap with other tracks and lack of a strong observation support. We show frames, from left to right 1175, 1200 and 1250.



**Fig. 11.** Top: results when using no interaction terms. When the appearance difference is not strong enough, as it happens for the two similarly looking subjects, the tracks overlap as the one with the stronger observation support (ID 26, in this figure) attracts the other (23). Bottom: this problem is solved using an interaction term, in this case specifically the avoidance term.



**Fig. 12.** The figure on the left shows a screenshot of the tracker applied to the Mycoplasma dataset. The red bounding box identifies the bacterium with ID 78 at the moment of initialization. The bacterium is followed in the right side in the smaller images. Note that the bacterium approaches other almost identical bacteria. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



**Fig. 13.** Recall/(1-precision) curve for the detector. The red cross shows the result of the tracker output when treated as a detector. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

### 5.5. Quantitative results

The output of the tracker strongly relies on the quality of the detector. In Fig. 13 we plot the precision and accuracy for the detector alone for the Students sequence. We also evaluate the tracker as a detector by using the best estimate (see Section 4.3) of tracked targets at each frame. The results show that our system is capable of keeping track of the targets also when these are not clearly visible to the detector.

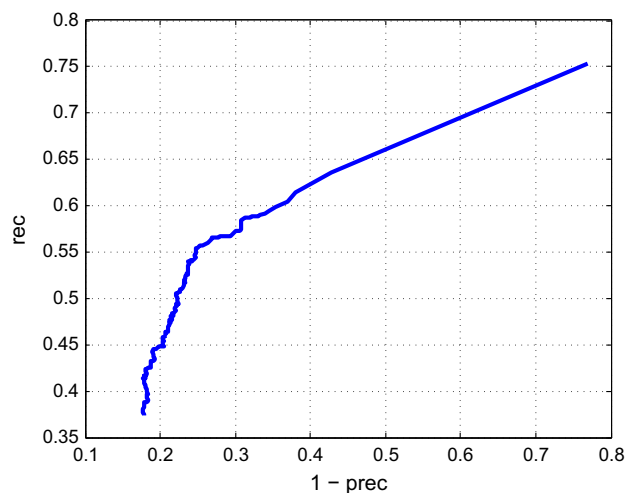
We finally evaluate the performance of the tracker on the Students sequence, using the CLEARMOT metric [5]. We compare our method with a baseline instantiation of the tracker that does not use the interaction model. The tracker relies therefore on the detector and on the online classifier to disambiguate multiple tracks. We also use an instantiation of the tracker that assigns all the subjects to separate groups, so that group interactions are not used. The results are shown in Table 3. As discussed already in the previous subsection, interactions, especially the avoidance behavior, are indeed necessary to avoid multiple tracks being explained by the same observation. Grouping offers a marginal improvement, due mostly to a reduction in false negatives, while it produces more false positives.

In our previous work [42], we present a data association method that we apply on short (2 s) sub-sequences of the Students

**Table 3**

CLEARMOT [5] evaluation, showing accuracy (MOTA), precision (MOTP), false negative rate (FN), false positive rate (FP) and the mismatches rate (MM). We compare the results of the full tracker presented in this chapter (“Full”), with an instantiation that does not use grouping (“No Group”) and with one that does not make use of interactions at all (“No Int.”).

Method	MOTA (%)	MOTP (%)	FN (%)	FP (%)	MM (%)
Full	<b>67.3</b>	75	21.7	9.4	1.6
No Group	65.3	74	26.2	6.9	1.7
No Int.	52.5	70	20.2	25.2	2.1



**Fig. 14.** Recall/(1-precision) curve for the group classification. To produce this result, the tracker has been run on 40 consecutive sub-sequences of the Student sequence, each lasting 2 s. The targets were initialized using the ground truth.

dataset. Although the tracker we presented is capable of automatic initialization and can cope with longer sequence, we use the same initialization from ground truth and the same experiment length in order to compare the two performances. Using the same test set (2000 frames, from frame 1000 to frame 3000) we initialize the tracker at each of the 40 sub-sequences using the ground truth annotation. As the sequences are short there is no need of using the unreliable state in the state variables for the subject. Using our approach we achieve  $\sim 79\%$  correctly predicted trajectories<sup>1</sup>

<sup>1</sup> As in [42], a trajectory is correctly predicted when the tracked position at the end of the 2 s is within a threshold of 0.5 m from the ground truth.

when interactions are not used and  $\sim 90\%$  with interactions. This offers a considerable improvement over the  $\sim 70\%$  result reported in [42].

As for the grouping results, we plot the precision and recall in Fig. 14. We achieve an Equal Error Rate of  $\sim 62\%$ , compared to the  $\sim 46\%$  precision and  $\sim 82\%$  recall reported in [42]. For the same sequence, the authors of [31] report 80.5% precision and 77% recall. When comparing the results we should note that in our case, grouping variables are only present where a link of the Delaunay triangulation is present. Therefore, we cannot achieve 100% recall no matter what threshold we use. Furthermore, in our case grouping and tracking are carried out jointly.

## 6. Conclusions

In this work we presented a principled model for a tracker that unifies an observation model, motion model and interaction model within the same framework. We cope with initialization errors and with tracker failures in the same way by means of an additional state variable. The target tracks and the group memberships are estimated jointly, so that each of them can propagate information to the other during the inference. The interactions themselves, namely the grouping relationship, are also propagated by means of an additional layer of connections that models local transitivity.

The work can be seen as a generalization of Particle Filters, in that it uses a sequential sampling scheme. However it is different from a Particle Filter solution in many aspects, including the use of Particle Belief Propagation and the interaction factors.

One of our goals was to propose a model that is modular, suitable to extensions and modifications. It should be easy, as a result, to add other functions to represent more properties. In this work we used a relatively small number of functions, and training the parameters has been carried out with a simple search on a validation set. While adding many other functions is possible, with the increasing number of parameters, a different, more efficient learning strategy must be planned. Since a log-linear model has been used, learning strategies like Contrastive Divergence [25] might be exploited.

Another advantage of modeling the whole tracker as a graphical model, is that it can readily deal with externally provided information, such as user annotation. This allows to easily extend the tracker to an interactive one. The user annotation forces the node variables to a particular state and the rest of the inference process stays the same. We developed such an interactive interface and we are currently using it to assist the tracker.

The tracker presented in this work is not capable of real-time performance. The computational time depends on the number of samples used, but also on the number and density of targets in the scene. As an example, for the Students sequence, with about 30 people per frame in the scene, the full tracker requires about 3 s per frame. Code optimization and more parallel computation would alleviate this limitation.

## Acknowledgement

The authors acknowledge financial support from EC project TANGO (FP7-ICT-249858) and the NCCR project IM2, sponsored by the Swiss National Foundation.

## References

- [1] S. Ali, M. Shah, Floor fields for tracking in high density crowd scenes, in: *ECCV*, 2008.
- [2] G. Antonini, S.V. Martinez, M. Bierlaire, J.P. Thiran, Behavioral priors for detection and tracking of pedestrians in video sequences, *IJCV* 69 (2006) 159–180.
- [3] Mark Briens Arnaud, Arnaud Doucet, Sumeetpal S. Singh, Sequential auxiliary particle belief propagation, in: *International Conference on Information Fusion*, 2005, pp. 705–711.
- [4] Ben Benfold, Ian Reid, Stable multi-target tracking in real-time surveillance video, in: *CVPR*, June 2011, pp. 3457–3464.
- [5] Keni Bernardin, Rainer Stiefelhof, Evaluating multiple object tracking performance: the clear mot metrics, *J. Image Video Process.* 2008 (2008) 1:1–1:10.
- [6] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, first ed., Springer, 2006.
- [7] Michael J. Black, Anand Rangarajan, The outlier process: unifying line processes and robust statistics, in: *Conference on Computer Vision and Pattern Recognition*, 1994, pp. 15–22.
- [8] Michael D. Breitenstein, Fabian Reichlin, Bastian Leibe, Esther Koller-Meier, Luc Van Gool, Online multiperson tracking-by-detection from a single, uncalibrated camera, *IEEE Trans. Pattern Anal. Mach. Intell.* 33 (2011) 1820–1833.
- [9] William Brendel, Mohamed R. Amer, Sinisa Todorovic, Multiobject tracking as maximum weight independent set, in: *CVPR*, 2011, pp. 1273–1280.
- [10] Ming-Ching Chang, Nils Krahnstoeber, Weina Ge, Probabilistic group-level motion analysis and scenario recognition, in: *IEEE International Conference on Computer Vision (ICCV)*, November 2011, p. 8.
- [11] Wongun Choi, Silvio Savarese, Multiple target tracking in world coordinate with single, minimally calibrated camera, in: *European Conference on Computer Vision*, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 553–567.
- [12] Ingemar J. Cox, *A Review of Statistical Data Association Techniques for Motion Correspondence*, 1993.
- [13] M. Cristani, L. Bazzani, G. Paggetti, A. Fossati, D. Tosato, A. Del Bue, G. Menegez, V. Murino, Social interaction discovery by statistical analysis of f-formations, in: *Proceedings of the British Machine Vision Conference*, BMVA Press, 2011, pp. 23.1–23.12.
- [14] A. Ess, B. Leibe, K. Schindler, L. Van Gool, A mobile vision system for robust multi-person tracking, in: *CVPR*, 2008.
- [15] T.E. Fortmann, Y. Bar Shalom, M. Scheffe, Sonar tracking of multiple targets using joint probabilistic data association, *JOE* 8 (3) (1983) 173–184.
- [16] Jonathan L. Freedman, *Crowding and Behavior*, 1975.
- [17] Andrew French, *Visual Tracking: From An Individual To Groups Of Animals*, PhD Thesis, 2006.
- [18] J. Gall, V. Lempitsky, Class-specific hough forests for object detection, in: *CVPR*, 2009.
- [19] W. Ge, R.T. Collins, B. Ruback, Automatically detecting the small group structure of a crowd, in: *2009 Workshop on Applications of Computer Vision (WACV)*, December 2009, pp. 1–8.
- [20] G Gennari, G D Hager, Probabilistic data association methods in visual tracking of groups, in: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2004 CVPR 2004*, vol. 2, 2004, pp. 876–881.
- [21] H. Grabner, J. Matas, L. Van Gool, P. Cattin, Tracking the invisible: learning where the object might be, in: *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.
- [22] Stephen J. Guy, Ming C. Lin, Dinesh Manocha, Modeling collision avoidance behavior for virtual humans, in: *AAMAS*, 2010, pp. 575–582.
- [23] E.T. Hall, *The Hidden Dimension*, Garden City, 1966.
- [24] D. Helbing, P. Molnár, Social force model for pedestrian dynamics, *Phys. Rev.* 51 (5) (1995).
- [25] Geoffrey Hinton, Training products of experts by minimizing contrastive divergence, *Neural Comput.* 14 (2000) 2002.
- [26] Alexander Ihler, David McAllester, Particle belief propagation, in: D. van Dyk, M. Welling (Eds.), *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS) 2009*, Clearwater Beach, Florida, 2009, pp. 256–263. *JMLR: W&CP* 5.
- [27] Z. Khan, T. Balch, F. Dellaert, MCMC-based particle filtering for tracking a variable number of interacting targets, *PAMI* 27 (11) (2005) 1805–1819.
- [28] Junseok Kwon, Mu Kyoung Lee, Tracking by sampling trackers, in: *International Conference on Computer Vision*, 2011.
- [29] J.D. Lafferty, A. McCallum, F.C.N. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, in: *ICML*, 2001.
- [30] Boris Lau, Kai Arras, Wolfram Burgard, Multi-model hypothesis group tracking and group size estimation, *Int. J. Soc. Robot.* 2 (2010) 19–30, <http://dx.doi.org/10.1007/s12369-009-0036-0>.
- [31] Laura Leal-Taixé, Gerard Pons-Moll, Bodo Rosenhahn, Everybody needs somebody: modeling social and grouping behavior on a linear programming multiple people tracker, in: *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 1st Workshop on Modeling, Simulation and Visual Analysis of Large Crowds, 2011.
- [32] B. Leibe, K. Schindler, N. Cornelis, L. Van Gool, Coupled detection and tracking from static cameras and moving vehicles, *PAMI* 30 (10) (2008) 1683–1698.
- [33] A. Lerner, Y. Chrysanthou, D. Lischinski, Crowds by example, in: *EUROGRAPHICS*, 2007.
- [34] Y. Li, C. Huang, R. Nevatia, Learning to associate: HybridBoosted multi-target tracker for crowded scene, in: *CVPR*, 2009.
- [35] Matthias Luber, Johannes A. Stork, Gian Diego Tipaldi, Kai O. Arras, People tracking with human motion predictions from social forces, in: *International Conference on Robotics and Automation*, IEEE, 2010, pp. 464–469.
- [36] Clark McPhail, Ronald T. Wohlstein, Using film to analyze pedestrian behavior, *Sociol. Methods Res.* 10 (3) (1982) 347–375.

- [37] R. Mehran, A. Oyama, M. Shah, Abnormal crowd behavior detection using Social Force model, in: CVPR, 2009.
- [38] Kevin P. Murphy, Yair Weiss, Michael I. Jordan, Loopy belief propagation for approximate inference: an empirical study, in: *Proceedings of Uncertainty in AI*, 1999, pp. 467–475.
- [39] K. Okuma, A. Taleghani, N. de Freitas, J. Little, D. Lowe, A boosted particle filter: multitarget detection and tracking, in: ECCV, 2004.
- [40] Jan Ondřej, Julien Pettré, Anne-Hélène Olivier, Stéphane Donikian, A synthetic-vision based steering approach for crowd simulation, in: *ACM SIGGRAPH 2010 papers, SIGGRAPH '10*, ACM, New York, NY, USA, 2010, pp. 123:1–123:9.
- [41] S. Pellegrini, A. Ess, K. Schindler, L. Van Gool, You'll never walk alone: modeling social behavior for multi-target tracking, in: ICCV, 2009.
- [42] Stefano Pellegrini, Andreas Ess, Luc Van Gool, Improving data association by joint modeling of pedestrian trajectories and groupings, in: *European Conference on Computer Vision (ECCV)*, 2010.
- [43] Julien Pettré, Jan Ondřej, Anne-Hélène Olivier, Armel Cretual, Stéphane Donikian, Experiment-based modeling, simulation and validation of interactions between virtual walkers, in: *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '09*, ACM, New York, NY, USA, 2009, pp. 189–198.
- [44] Hamed Pirsiavash, Deva Ramanan, Charles C. Fowlkes, Globally-optimal greedy algorithms for tracking a variable number of objects, in: *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, IEEE Computer Society, Washington, DC, USA, 2011, pp. 1201–1208.
- [45] D.B. Reid, An algorithm for tracking multiple targets, *IEEE Trans. Autom. Control* 24 (6) (1979) 843–854.
- [46] Mikel Rodriguez, Saad Ali, Takeo Kanade, Tracking in unstructured crowded scenes, in: *International Conference on Computer Vision*, 2009, pp. 1389–1396.
- [47] M. Ryoo, J. Aggarwal, Stochastic representation and recognition of high-level group activities, *Int. J. Comput. Vis.* 93 (2011) 183–200, <http://dx.doi.org/10.1007/s11263-010-0355-5>.
- [48] Amir Saffari, Martin Godec, Thomas Pock, Christian Leistner, Horst Bischof, Online multi-class LPBoost, in: *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.
- [49] Jakob Santner, Christian Leistner, Amir Saffari, Thomas Pock, Horst Bischof, Prost: parallel robust online simple tracking, in: *Computer Vision and Pattern Recognition*, 2010, pp. 723–730.
- [50] A. Schadschneider, Cellular automaton approach to pedestrian dynamics – theory, in: *Pedestrian and Evacuation Dynamics*, 2001.
- [51] P. Scovanner, M. Tappen, Learning pedestrian dynamics from the real world, in: ICCV, 2009.
- [52] Bi Song, Ting-Yueh Jeng, Elliot Staudt, Amit K. Roy-Chowdhury, A stochastic graph evolution framework for robust multi-target tracking, in: *Proceedings of the 11th European conference on Computer vision: Part I, ECCV*, 2010, pp. 605–619.
- [53] Andrew N. Stein, Derek Hoiem, Martial Hebert, Learning to find object boundaries using motion cues, in: *International Conference on Computer Vision*, 2007.
- [54] E.B. Sudderth, A.T. Ihler, W.T. Freeman, A.S. Willsky, Nonparametric belief propagation, in: *Computer Vision and Pattern Recognition*, 2003.
- [55] Peter Trautman, Andreas Krause, Unfreezing the robot: navigation in dense, interacting crowds, in: IROS, 2010.
- [56] Atsuko Uenoyama, Makoto Miyata, Gliding ghosts of *Mycoplasma mobile*, *Proc. Nat. Acad. Sci. USA* 102 (36) (2005) 12754–12758.
- [57] Jur P. van den Berg, Ming C. Lin, Dinesh Manocha, Reciprocal velocity obstacles for real-time multi-agent navigation, in: ICR, 2008, pp. 1928–1935.
- [58] B. Wu, R. Nevatia, Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet part detectors, *IJCV* 75 (2) (2007) 247–266.
- [59] Kota Yamaguchi, Alexander C. Berg, Luis E. Ortiz, Tamara L. Berg, Who are you with and where are you going? in: CVPR, 2011, pp. 1345–1352.
- [60] Bo Yang, Chang Huang, Ram Nevatia, Learning affinities and dependencies for multi-target tracking using a CRF model, in: CVPR, 2011, pp. 1233–1240.
- [61] Alper Yilmaz, Omar Javed, Mubarak Shah, Object tracking: a survey, *ACM Comput. Surv.* 38 (2006).
- [62] L. Zhang, Y. Li, R. Nevatia, Global data association for multi-object tracking using network flows, in: CVPR, 2008.