Tracking millions of humans in crowded space in crowded spaces

Alexandre Alahi, Vignesh Ramanathan, Li Fei-Fei

Stanford University

1. Introduction

As Aristotle noted, "man is by nature a social animal". We do not live in isolation. On a daily basis, thousands of individuals walk in terminals, malls or city centers. They consciously or unconsciously interact with each other. They make decisions on where to go, and how to get to their destination. Their mobility is often influenced by their surrounding. Understanding human social dynamics plays a central role in the design of safer and smarter spaces. It enables the development of ambient intelligence, *i.e.*, spaces that are sensitive and responsive to human behavior. For instance, many sites such as train terminals were constructed several years ago to serve an estimated traffic demand. However, this estimated demand is greatly exceeded by forecasted traffic within a span of one decade. Sensing how individuals move through these large spaces provides insights needed to modify the space or design new ones to accommodate increased traffic. This enables reduced congestion and smooth flow of people.

In this chapter, we present the computer vision techniques behind understanding the behavior of more than hundred million individuals in crowded urban spaces. We cover the full spectrum of an intelligent system that detects and tracks humans in high density crowds using a camera network. To the best of our knowledge, we have deployed one of the largest networks of cameras (more than hundred cameras per site) to capture the trajectories of pedestrians in crowded train terminals over the course of two years. At any given time, up to a thousand pedestrians need to be tracked simultaneously (see Fig. 1). The captured dataset is publicly available to enable various research communities, from psychology to computer vision, to dive into a large-scale analysis of human mobility in crowded environments[1]. In the remaining of the chapter, we will share all the technical details that lead to successfully analyze millions of individuals.

While computer vision has made great progress in detecting humans in isolation [1, 2, 3, 4], tracking people in high density crowds is very challenging. Individuals highly occlude each other and their motion behavior is not independent. We present detailed insights on how to address these challenges with sparsity promoting priors, and discrete combinatorial optimization that models social interactions.

Understanding the behavior of pedestrians using a network of cameras is comprised of the following three steps: (i) Human detection in 3D space, (ii) Tracklet generation, and

[1] www.ivpe.com/crowddata.htm

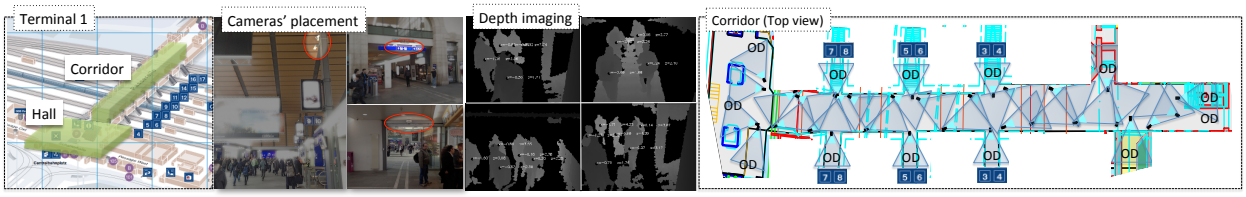


Figure 1: Real-world setup. Illustration of one of the monitored corridors in a train terminal. More than 30 cameras are deployed in the presented corridor, whereas 132 cameras are deployed in the terminal. At any given time, the occupancy of the corridor can reach more than one thousand of pedestrians. The label "OD" represents entry/exit zones.

(iii) Tracklet association. We define *tracklet* as the short trajectory of a human limited to the field-of-view of a single camera. Each camera extracts tracklets corresponding to multiple people. The resulting tracklets are linked across cameras to obtain the long-term trajectories of humans in the full space. In the reminder of this chapter, we expand on each of the three steps and provide more details required to reach real-time performance with high accuracy. First, we cast the human detection problem as an inverse problem with sparse prior, which can be solved in an efficient optimization framework. Then, we formulate the tracking problem as a linear integer program and use the social affinity of individuals to effectively associate tracklets for long-term tracking.

2. Related work

We present an overview of relevant works to solve each of the three steps presented in the introduction: (i) detection, (ii) tracklet generation, and (iii) tracklet association in a camera network.

Human detection. Pedestrians in isolation are accurately detected using a single image and robust classification techniques such as R-CNNs or deformable parts models [1, 2, 3, 4]. Individuals are detected in the image plane as opposed to 3D coordinates of people in the real world. With a calibrated camera, it is possible to map detected bounding boxes to the real world coordinates [5, 6]. Algorithms with high levels of confidence have been proposed to locate crowded people with a single top view or several head-level overlapping field-of-views [7, 8, 9, 10]. For instance, Khan and Shah locate people on the ground by taking the intersection of projected foreground silhouettes on the ground plane. Fleuret *et al.* [10] use a generative model with a probabilistic framework to outperform previous work. Alahi *et al.* in [6] propose a sparsity driven framework to handle noisy observations and reduce the number of false positives. Golbabaee *et al.* [11] propose a real-time solver to the sparsity driven framework inspired by the set cover problem. In the next section, we will present more details on the sparsity driven formulations.

Tracklet generation. Once individuals are located on the ground, various graph-based algorithms can be used to track them. Each node represents a detection and the edges measure the similarity cost to link the detections. It is possible to find the global optimum solution with linear programming to solve the data association problem [12, 13]. It outperforms previous works based on Markov Chain Monte Carlo [14] or inference in Bayesian

networks [15]. The data association problem is expressed as a graph theoretic problem for finding the best pathes/flows over the graph. The main challenge is to find a robust similarity measure. Recently, Xiang *et al.* [16] have shown that tracking multiple humans can be formulated as a Markov Decision Process instead of a graph-based formulation. They learn an appearance-based similarity function to outperform previous ones based on color histograms. Their approach will fail if limited information on the appearance of the pedestrian is available or if all pedestrians look the same (e.g. when the back of their head is only visible). In this chapter, we present the generic graph-based framework to track multiple humans since both simple and complex similarity measure can be modeled.

Tracklet association. A large body of work models visual appearance to link tracklets across cameras [17, 18, 19, 20]. Andriluka *et al.* [21] use person detection as a cue to perform tracking and vice-versa. Javed *et al.* in [22] use travel time and the similarity of appearance features. Song *et al.* in [23] use a stochastic graph evolution strategy. Tracklets extracted by each camera are linked with the Hungarian algorithm [24], MCMC [25], or globally optimal greedy approaches [20]. These approaches have not addressed the linking of tracklets that are dozens of meters away in a highly crowded scene. Alahi *et al.* [26] propose to model social interactions and more precisely social affinities to solve the tracklet association step. In Section 6.1 on this chapter, we will present more details on their method.

Tracking with social prior. Social behavior has recently been incorporated into existing tracking frameworks by modeling the well-known social forces [27] with Kalman filters [28], extended Kalman filters [29], or Linear Programming [13, 30]. Antonini *et al.* [31] use Discrete Choice Models to simulate the walking behavior of people. These approaches improve the operational-level tracking when a few frames are missing (*e.g.,* when given a low-frame rate, or short occlusion cases). They also often model a grouping cue to solve the data association problem [32, 13, 30]. They model it as a set of pedestrians with similar velocities and spatial proximity. Similarly, [33] use grouping cues in a hierarchical framework to identify sports player roles. The grouping cue is typically handled as a binary variable indicating group similarity. However, the key challenge is to use a finer representation to capture group association and integrate it into the problem of tracklet association. Yang *et al.* [34, 32] use a conditional random field framework to jointly estimate group membership and tracks. Leal *et al.* [13] iteratively compute the minimum cost flow for various velocity and grouping assignments until convergence or when a maximum number of iterations is reached. Qin *et al.* [30] use the Hungarian algorithm to jointly group and link tracklets. However, the Hungarian algorithm does not solve the global minimization over the full long-term track, whereas the minimum network flow formulation does. In this chapter, we present more details on a descriptor representing the grouping cue as a feature to efficiently match behavior across pedestrians.

3. System overview

We work with 132 RGB/Thermal/Depth cameras which monitors 20,000 square meters with human density reaching 1 individual per square meter. This introduces new challenges in designing detection, and tracking algorithms which can work at such scale. The camera

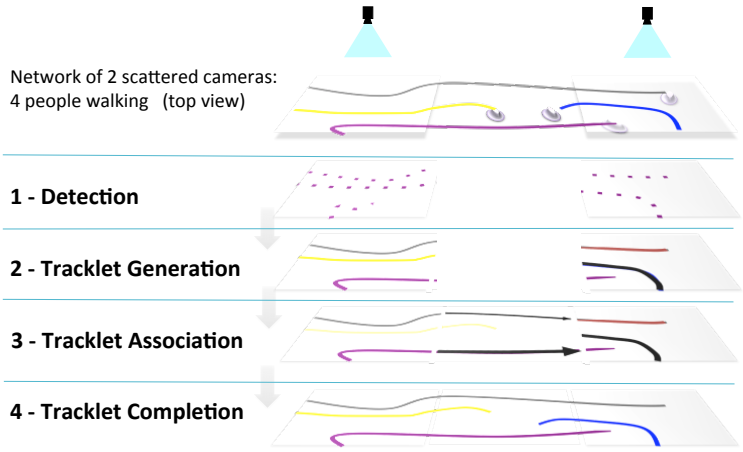


Figure 2: Overview of the system: each camera extracts independently tracklets with high confidence. Then, the tracklets are linked across cameras. For completeness, we illustrate a fourth step: tracklet completion which will be presented in the next chapter. The goal of this step is to predict the detailed trajectories of pedestrians.

network constantly collects large volume of visual data. Here are some facts regarding the dataset: individuals travel time in the monitoring area is 50 seconds and spans 70 meters in average. At a given time, up to 1000 people can be in the same area. Typically, after 1 minute, more than a million people are detected and the detections need to be linked to each other. To handle such a deluge of data, we distribute the processing as follows: first every camera independently locates people on the ground using the method presented in Section 4. Then, detected individuals are tracked within each camera independently given a global optimization framework (solved with Linear Programming) similar to [13] (see Section 5). The resulting tracklets are matched across cameras to obtain the long-term tracks over the full area by modeling social affinities 6.1. Figure 2 illustrates the distributed processing pipe.

4. Human detection in 3D

4.1. Method

The first step in our system involves locating 3D position of people on the ground within the field-of-view of each single camera. The scene geometry needs to be estimated across thermal and optical cameras. Most previous works locating people on the ground (in 3D) use calibration data to map image coordinates to the real-world [7, 8, 10, 6]. A set of 3D coordinates or the cameras extrinsic parameters are needed to estimate the homography matrices. However, when dealing with large-scale setups, it is difficult to obtain calibration data for all cameras. Some works exist to automatically estimate the scene geometry using vanishing lines [35], and additional human poses [36]. However such approaches do not work on thermal images where binary silhouettes are only observed (see Figure 3).

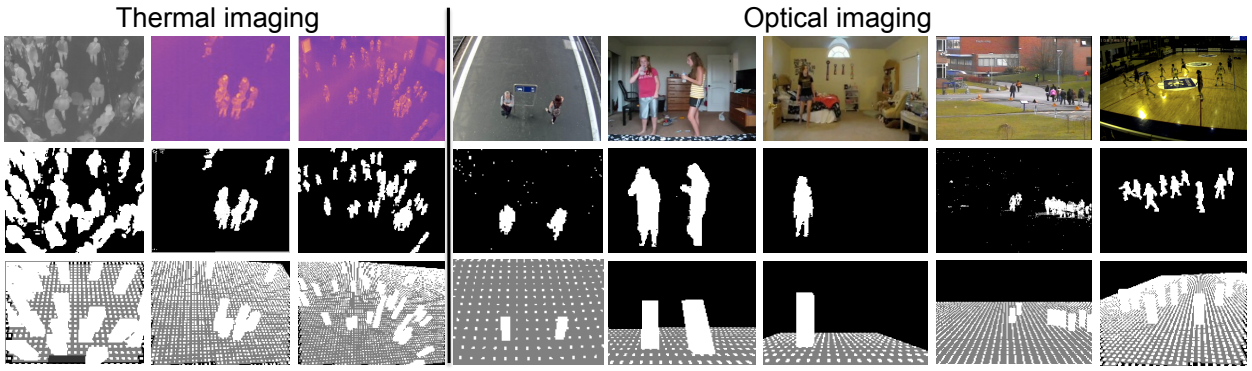


Figure 3: A collection of camera where people are correctly detected in 3D given the observed foreground silhouettes only. The learned ground plane points are also plotted.

We propose to solve the single view scene geometry and 3D localization problem using only extracted binary human silhouettes. As a result, it works across any camera modalitiy (*i.e.,* thermal or optical). We address the problem as a dictionary-based inverse problem where both the dictionary and the occupancy vector are unknown. We use the same sparsity driven formulation as in [6]. An inverse problem is formulated to deduce people location points (*i.e.,* occupancy vector x) given a sparsity constraint on the ground occupancy grid. Let y be our observation vector (*i.e.,* the binary foreground silhouettes), and D the dictionary of atoms approximating the binary foreground silhouettes of a single person at all locations (see Figure 4). We present an algorithm to find the ground occupancy vector x satisfying the following equation when the dictionary D is unknown as opposed to previous work:

$$x, D = \underset{x\in\{0,1\}, D_i \in D^c}{\arg\min} \|y - D_i x\|_2^2 \;\; \text{s.t.} \;\; \|x\|_0 < \varepsilon_p, \tag{1}$$

where D^c is the space of all potential dictionaries and ε_p is the maximum number of people to be detected.

Algorithm 1 illustrates our 'Detection while Learning' algorithm solving a relaxation of Equation 1. We iterate over the space of dictionary $\forall D_i \in D^c$. We solve Equation 1 for a fixed D_i using the 'Set Covering Object Occupancy Pursuit' (SCOOP) algorithm presented in [11]. The latter iteratively recovers one element of the support set. More precisely, at each iteration, it selects the atom a of the dictionary which contributes the most in the signal energy (the most correlated atom with the signal) and fits well the image. The output of SCOOP algorithm is an occupancy vector associated with a residual error on the data fidelity term ($\|y - D_i x\|_2^2$). We compute SCOOP over all dictionaries and select the one with the smallest residual errors. Note that the dictionaries D_i are not random matrix. They lie on a manifold. The space of solutions D^c is much smaller than the dimension of the matrix. It is related to the extrinsic parameters of the cameras, *i.e.* its height to the ground, and its orientation (see Figure 4). A coarse to fine sampling step is used to select potential dictionaries approximating a gradient descent.

Algorithm 1: SCOOP-Learning: Detecting while Learning

Input: signal y, regularization parameter w.
Output: occupancy vector x, and dictionary D
1) **Init :**
$E_{s+1} = E_s = 0, s_{min} = 0$
2) $\forall D_i \in D^c$
- **SCOOP algorithm with a fixed D_i:**
-Initialization:
$\widehat{\mathcal{S}} \Leftarrow \{\}$, $r \Leftarrow y$, $\widehat{y} \Leftarrow \mathbf{0}$, $e_{s+1} = \Big|\text{supp}(y)\Big|$, $e_s = \Big|\text{supp}(y)\Big|$
- Matching pursuit-like process:
while $(e_{s+1} - e_s \leq 0)$ **do**

$$j \Leftarrow \arg\min_{j' \in \mathcal{U}} \Big\{ w \frac{\Big|\text{supp}(r) \backslash \text{supp}(d_{j'})\Big|}{\Big|\text{supp}(r)\Big|} + (1-w) \frac{\Big|\text{supp}(d_{j'}) \backslash \text{supp}(r)\Big|}{\Big|\text{supp}(d_{j'})\Big|} \Big\}$$

- Updates:
Recovered support: $\widehat{\mathcal{S}} \Leftarrow \widehat{\mathcal{S}} \cup \{j\}$
Recovered $\hat{y}$: $\text{supp}(\hat{y}) \Leftarrow \text{supp}(\hat{y}) \cup \text{supp}(d_j)$
Remainder: $\text{supp}(r) \Leftarrow \text{supp}(r) \backslash \text{supp}(d_j)$
Error: $e_s \Leftarrow e_{s+1}$
Error: $e_{s+1} \Leftarrow \Big|\text{supp}(y \oplus \hat{y})\Big|$

end
- Updates:
$E_s = E_{s+1}$
$E_{s+1} = e_i$
if $E_{s+1} \leq E_s$ **then**
$D = D_i$
$x = \widehat{\mathcal{S}}$
end

4.2. Evaluation

First, qualitative results are available in Figure 3. It illustrates the performance of the proposed SCOOP-Learning algorithm in locating people in 3D as well as the scene geometry. Various height and viewing angle are illustrated. Then, quantitative results are shared in Figure 5. It presents the accuracy of the presented algorithm to estimate the camera parameters with respect the estimated heights and angles of various cameras setups. We can see that the cameras parameters are correctly estimated once enough people are present in the scene. Indeed, the more number of silhouettes are used to select the dictionary, the less error-prone the estimations are, since many possible solutions exist with a single silhouette

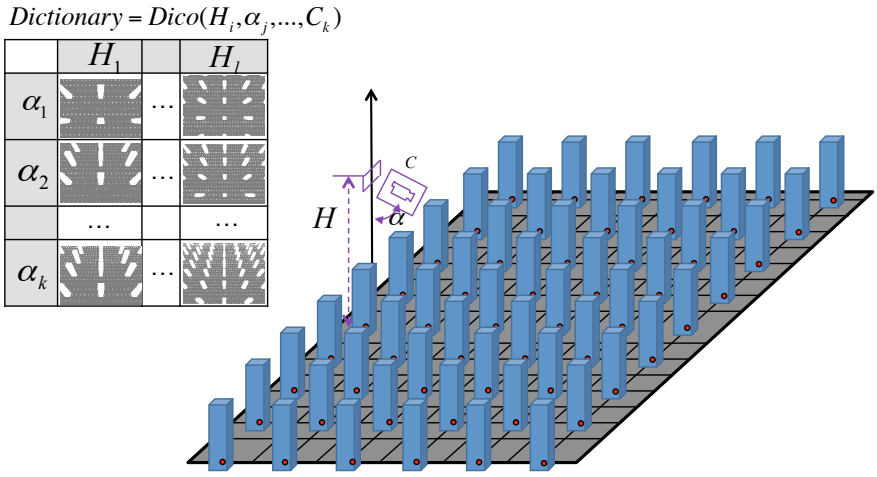


Figure 4: Illustration of the dictionary space. We approximate the ideal silhouettes of people with cuboids on the ground. A dictionary is the collection of atoms representing the observation of the cuboid in the image plane of the camera with a given height and viewing angle.

as opposed to several ones. Note that the error on the 3D estimation is in average less than 50 cm once 3 people are present in the field of view although the camera parameters have still about 15% errors.

Once the camera parameters are found, *i.e.*, the dictionary is known, we can evaluate the performance of human detection in 3D in both low and high density crowds. We refer the readers to [6] and [11] for a detailed analysis on this topic.

5. Tracklet Generation

Once humans are located in the field-of-view of each camera, we need to track them across time (second step in Figure 2). As a reminder, we refer to tracklet as the short trajectories of humans captured by a single camera. We want to find the set of tracklets X, where each tracklet $x \in X$ is represented as an ordered set of detections, (L_x), representing the detected coordinates of humans (using the method described in previous section). Similarly, $L_x = (l_x^{(1)}, \ldots, l_x^{(n)})$ is an ordered set of intermediate detections which are linked to form the tracklets. These detections are ordered by the time of initiation. The problem can be written as a Maximum a-posteriori estimation problem similar to [19, 20]:

$$X^* = \max_X P(L|X)P(X), \tag{2}$$

where $P(L|X)$ is the probability of the detections in L being true positive detection. The probability $P(L|X)$ is:

$$P(L|X) \propto \prod_{x \in X} \prod_{l \in L_x} \frac{P_{tp}(l)}{P_{fp}(l)}, \tag{3}$$

where $P_{tp}(l)$ and $P_{fp}(l)$ are probabilities of the detection being a true positive, and false positive respectively.

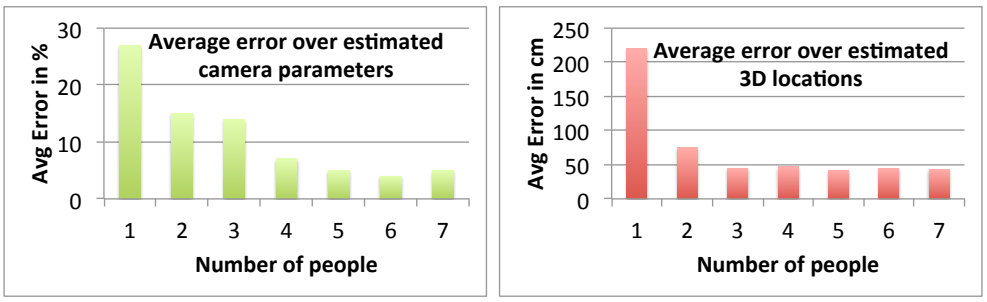


Figure 5: Performance of SCOOP-learning algorithm to estimate the scene geometry. Top graph illustrates the measured error over the estimated camera parameters with respect to the number of people observed in the sequence of images. Bottom graph illustrates the error in the final 3D localization of people on the ground.

Next, similar to [20], we assume a Markov-chain model connecting every intermediate detection $l_x^{(i)}$ in the tracklet X, to the subsequent detection $l_x^{(i+1)}$ with a probability given by $P(l_x^{(i+1)}|l_x^{(i)})$. The tracklet probability $P(X)$ is:

$$\begin{aligned} P(X) &= \prod_{x \in X} P(x), \\ P(x) &= \prod_{i=1}^{n} P\left(l_x^{(i)}|l_x^{(i-1)}\right), \end{aligned} \tag{4}$$

where $n = |L_x|$ is the number of intermediate detections in the tracklet.

The MAP problem from Eq. 7 can now be formulated as a linear integer program:

$$\begin{aligned} \min_f \quad & C(f) \\ C(f) \quad = \quad & \sum_{x_i \in X} \alpha_i f_i + \sum_{x_i, x_j \in X} \beta_{ij} f_{ij} \\ \text{s.t} \quad & f_i, f_{ij} \in \{0, 1\} \\ \text{and} \quad & f_i = \sum_j f_{ij}, \end{aligned} \tag{5}$$

where f_i is the flow variable indicating whether the corresponding detection is a true positive, and f_{ij} indicates if the corresponding detections are linked together. The variable β_{ij} denotes the transition cost given by $\log P\left(l_i|l_j\right)$ for the detection $l_i, l_j \in L$. The local cost α_i is the log-likelihood of an intermediate detection being a true positive. In our case, we suppose that all detections have the same likelihood.

We note that the optimization problem in Eq. 5 is equivalent to the flow optimization problem widely discussed in [20, 19]. Such problems can be solved through k-shortest paths algorithm [20, 12].

The main challenge in solving the tracklet generation step is to define the transition cost β_{ij}. For any two detections, it can be split into several components as shown below:

$$\beta_{ij} = \beta_{ij}^{appearance} + \beta_{ij}^{motion}, \tag{6}$$

where $\beta^{appearance}$ is the cost to ensure similar appearance and β^{Motion} is the cost to ensure motion smoothness in the connected detections. The choice behind these similarity metrics is still on open research. Xiang *et al.* [16] have shown that learning these metrics from training data outperforms hand-designed features such as color histogram or kalman filters. We refer the readers to [16] for a detailed evaluation of their method on the public Multi-Object Tracking (MOT) challenge [37].

6. Tracklet Association

The third step of our intelligent system is to connect tracklets across cameras (see Figure 2). This task becomes even more challenging when cameras are scattered and distant by several dozens of meters. Previous techniques based on appearance and motion similarities are not sufficient since the camera viewpoints might be very different leading to strong appearance changes, and the linear motion assumption is not valid anymore on long distances. In this section, we present a descriptor that models social interactions to reason on the data association step. We show how to use the same graph-based framework presented in previous section to solve the tracklet association step although additional constraints need to be modeled.

6.1. Social Affinity Map: SAM

When walking in crowded environments, humans often have social affinities that remain stable over time.

Definition 1. *We define "social affinity" as the motion affinity of neighboring individuals.*

Social affinities can be consciously formed by friends, relatives or co-workers. However, in crowded environments, subconscious affinities exist. For example, the "*Leader-follower*" phenomenon [38] represents a spontaneous formation of lanes in dense flows, as a result of fast pedestrians, passing slower ones. More formally, the leader-follower pattern captures the behavior of a pedestrian (a follower) who adjusts his/her motion to follow a leader to enable smooth travel. We propose to learn the various social affinities which bind people in a crowded scene through a feature called as Social Affinity Map (SAM).

6.2. The SAM feature

We observed that in public settings, social forces are mostly determined by the proximity of people to each other as noted in previous works [27]. Since, people are more easily influenced by others in their vicinity, we develop a social affinity feature which captures the spatial position of the tracklet's neighbors. As shown in Fig. 6, we achieve this by radially binning the position of neighbouring tracklets.

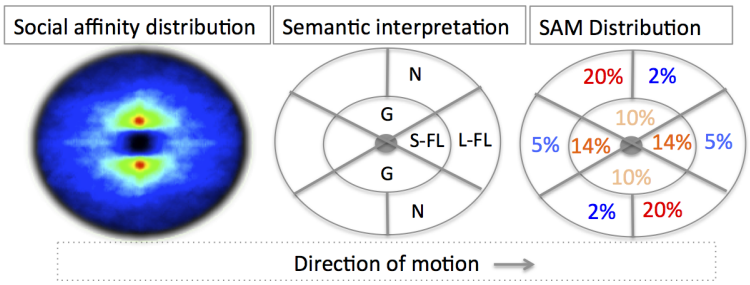


Figure 6: Left hand-side: Heatmap of the relative positions of all neighboring pedestrians across all tracklets. Middle column: it represents the SAM with our semantic description where "G" is the group affinity (such as couples, friends), "S-FL" is the short distance Follow-Leader behavior, "L-FL" is the long distance FL behavior, and "N" can be seen as the comfortable distance to maintain while walking in the same direction. The right hand-side represents the distribution of presented behavior.

We further learn the spatial binning by first clustering the relative position of surrounding individuals over all captured trajectories. We considered relative positions within a limit of $3m$, to avoid outliers. The distribution of the relative positions across the million trajectories is visualized in Fig. 6. We obtain 10 bins as a result of this clustering, as shown in the figure. The percentage of relative positions pooled into this bins is also shown in the figure. It is interesting to point out that the most used bin is the one on far right side ("N" label in Fig. 6). It can be interpreted as the comfortable pattern to walk with respect to other individuals as opposed to the left hand side.

Given a new tracklet, we perform vector quantization (VQ) coding to obtain the SAM feature. We fit a Gaussian Mixture Model to the relative position of its surrounding tracklets. The inferred GMM values within the previously learned spatial bins are discretized to obtain a binary radial histogram, which represents the SAM feature vector. The complete process is illustrated in Fig. 7. Hamming distance is used to compare SAM across tracklets. Note that binary quantization has little impact on the efficacy of the feature, and is only used to speed up the comparison method.

Our SAM feature can differentiate between various configurations of social affinities such as "*couple walking*", or the "*Leader-follower*" behavior. Fig. 8 illustrates the 8 most observed SAM over millions of trajectories. It is worth pointing out that 76% of individuals belong to a group, hence a SAM provides valuable information in crowded settings, motivating the use of these cues in forecasting the mobility of pedestrians.

6.3. Tracklet association method

Often, there is a sparse network of cameras monitoring the transit of people in a public setting like a railway terminal. The terminal has a set of entry points referred to as the *origin*, and exit points referred to as the *destination*. One key motivation behind tracking humans in the terminal is to identify the Origin and Destination (OD) of every person entering and exiting the camera network. We achieve this by identifying the trajectories which connect the tracklets starting at the origin to the tracklets ending at the destination. The number of intermediate tracklets linked to obtain these trajectories decreases with the

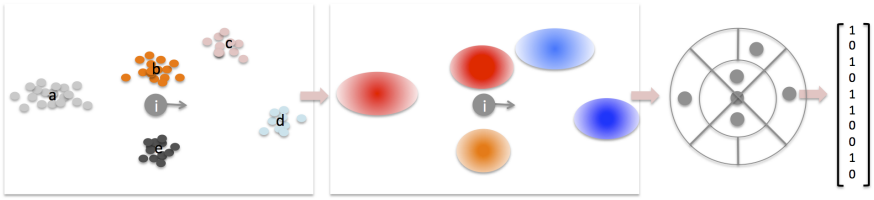


Figure 7: Illustration of a Social Affinity Map extraction (top view). The relative positions of neighboring individuals are clustered into a radial histogram. The latter is one bit quantized.

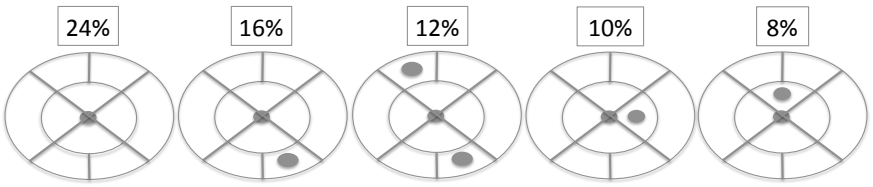


Figure 8: Illustration of the 8 most observed social affinities learned from the data. The above percentage represents the frequency of occurrence of the corresponding SAM.

sparsity of the camera network. Fig. 9 illustrates an extreme case with only origin and destination tracklets.

We have a set of origin tracklets O and an equal number of destination tracklets D. Each tracklet in O is captured at one of the many entrances into the area, and a destination track in D is captured at an exit. We also have a set of intermediate tracklets X obtained by our sparse camera network. We want to find the set of trajectories T, where each trajectory $t \in T$ is represented as an ordered set of tracklets, (o_t, X_t, d_t), with $o_t \in O$ and $d_t \in D$ representing the origin and destination tracklets of the trajectory. Similarly, $X_t = (x_t^{(1)}, \ldots, x_t^{(n)})$ is an ordered set of intermediate tracklets which are linked to form the trajectory. These tracklets are ordered by the time of initiation. The problem can be written as a Maximum a-posteriori estimation problem similar to Section 5:

$$T^* = \max_T P(X|T)P(T), \tag{7}$$

where $P(X|T)$ is the probability of the tracklets in X being true positive tracklets. The probability $P(X|T)$ is:

$$P(X|T) \propto \prod_{t \in T} \prod_{x \in X_t} \frac{P_{tp}(x)}{P_{fp}(x)}, \tag{8}$$

where $P_{tp}(x)$ and $P_{fp}(x)$ are probabilities of the tracklet being a true positive, and false positive respectively.

We define $P_{OD}(o, d)$ as the OD-prior term which states the probability of a person entering at the origin corresponding to o exiting at the destination corresponding to d. Such prior is often neglected and assumed to be uniform. However, in many applications, it is a strong prior, such as avoiding forbidden paths in airports.

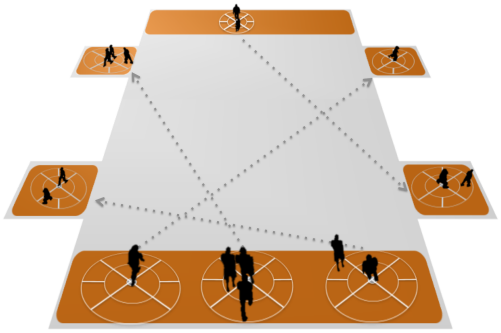


Figure 9: Predicting the behavior of pedestrians given Social Affinity Maps (SAM) with few cameras. Orange regions represent the monitoring areas of cameras. We illustrate the extreme case when cameras are only placed at entrance or exit zones, referred to as OD cameras.

Next, similar to Section 5, we assume a Markov-chain model connecting every intermediate track $x_t^{(i)}$ in the trajectory T, to the subsequent track $x_t^{(i+1)}$ with a probability given by $P(x_t^{(i+1)}|x_t^{(i)})$. The trajectory probability $P(T)$ is:

$$\begin{aligned} P(T) &= \prod_{t \in T} P(t), \\ P(t) &= P_{OD}(o_t, d_t) P\left(x_t^{(1)}|o_t\right) \\ &\quad \prod_{i=2}^{n} P\left(x_t^{(i)}|x_t^{(i-1)}\right) P\left(d_t|x_t^{(n)}\right), \end{aligned} \tag{9}$$

where $n = |X_t|$ is the number of intermediate tracklets in the trajectory.

The MAP problem from Eq. 7 can now be formulated as a linear integer program in a manner similar to [20]:

$$
\begin{aligned}
\min_{f} \quad & C(f) \qquad (10)\\
C(f) \;=\; & \sum_{x_i \in X} \alpha_i f_i + \sum_{x_i, x_j \in X} \beta_{ij} f_{ij} + \\
& \sum_{\substack{x_i \in X,\\ o \in O}} \beta_{oi} f_{oi} + \sum_{\substack{x_i \in X,\\ d \in D}} \beta_{id} f_{id} + \sum_{\substack{o \in O,\\ d \in D}} \gamma_{od} f_{od} \\
\text{s.t} \quad & f_i, f_{ij}, f_{od} \in \{0, 1\} \\
\text{and} \quad & f_i = \sum_j f_{ij} + \sum_d f_{id} = \sum_i f_{ji} + \sum_o f_{oi}, \\
& \sum_{od} f_{od} = |O| = |D|, \\
& \sum_d f_{od} = \sum_i f_{oi}, \\
& \sum_o f_{od} = \sum_i f_{id} \;\; \forall \; x_i, x_j \in X, \; o \in O, \; d \in D,
\end{aligned}
$$

where f_i is the flow variable indicating whether the corresponding tracklet is a true positive, and f_{ij} indicates if the corresponding tracklets are linked together. The variable β_{ij} denotes the transition cost given by $\log P\left(x_i | x_j\right)$ for the tracks $x_i, x_j \in X$. The log-likelihoods β_{oi}, β_{id} are also defined similarly, for the origin track o and destination track d. The local cost α_i is the log-likelihood of an intermediate track being a true positive. Finally, the OD-prior cost is represented as $\gamma od = \log P_{OD}(o, d)$.

We note that the optimization problem in Eq. 10 is equivalent to the flow optimization problem in Equation 5 in the absence of the OD prior term. The addition of the OD-prior term leads to loops in the network-flow problem, and can no longer be solved exactly through shortest path algorithms. Hence, we adopt a heuristic approach to solve Eq. 10, as discussed in Sec. 6.4.

The local cost α_i is proportional to the length of a tracklet. This helps us to remove short tracklets that might represent false positives. The transition cost β_{ij} for any two tracklets is split into two components as shown below.

$$
\beta_{ij} = \beta_{ij}^{SAM} + \beta_{ij}^{M}, \qquad (11)
$$

where β^{SAM} is the social-affinity cost and β^{M} is a cost to ensure smoothness in the connected tracklets.

Social Affinity cost. In our model, we wish to ensure that tracklets moving in similar social groups have a stronger likelihood of being linked to each other. This affinity forms an important component in large scale tracking scenarios like ours, where the appearance of an individual is not very discriminative. The SAM features introduced in Sec. 6.1 are used to measure the social affinity distance between tracklets moving in groups as shown below

$$\beta_{ij}^{SAM} = \mathbf{H}(sam_i, sam_j), \tag{12}$$

where $\mathbf{H}(.)$ denotes the Hamming distance between two binary vectors, and sam_i, sam_j denote the SAM feature vector of the two tracks.

Motion similarity. Another cue β^M, which is used to ensure smoothness in trajectory motion is obtained by measuring the distance between the motion patterns of two tracklets similar to [25, 23]

The OD-prior cost is the log-likelihood of the prior probability of transiting from an origin point to the destination. In most surveillance settings, we can use prior knowledge on the geography of the terminal, as well as rough estimates of the passenger freight to obtain an OD prior. In addition, the OD prior can be used to enforce constraints such that passengers entering a certain entry point would not return to the same location from a parallel entrance. In our experiments in later sections, the OD prior is obtained by a short survey in the location.

6.4. Optimization

As stated before, the optimization in Eq. 10 cannot be trivially solved through existing shortest path algorithms [20] as in the case of traditional tracking. Hence, we adopt a heuristic approach as explained below.

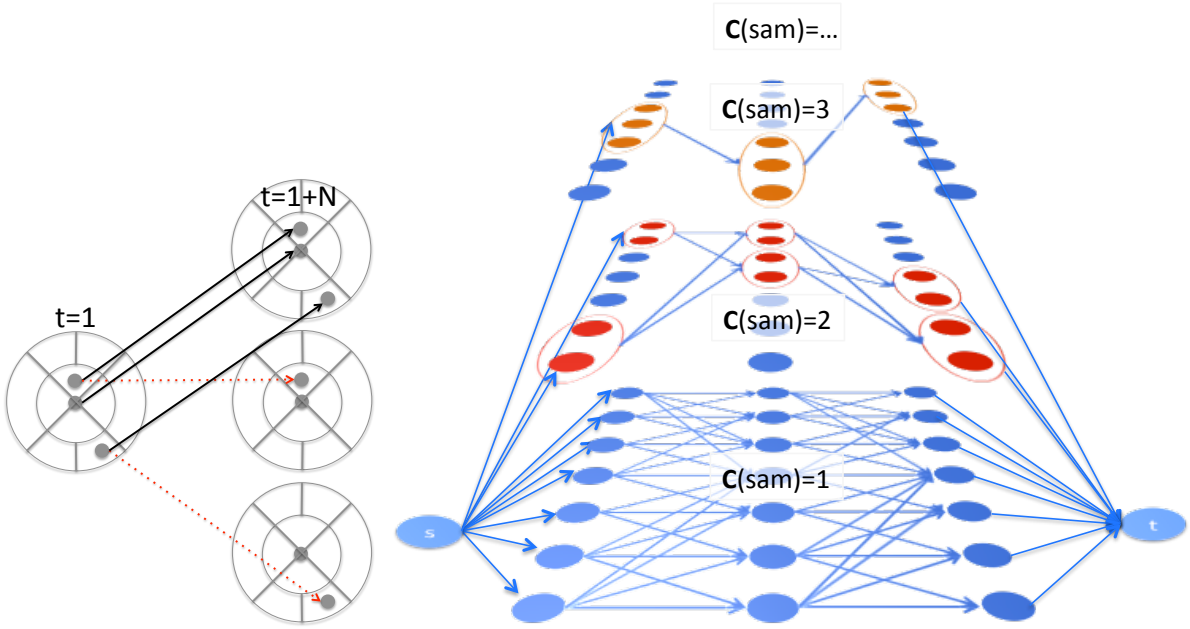
Greedy optimization with OD-prior. We first run a greedy algorithm to identify the low-cost solutions in the graph:

1. Find the shortest path which links an origin tracklet to the destination tracklet in Eq. 10
2. Remove the tracklets which are part of the trajectory obtained in the previous step and repeat.

The greedy algorithm provides an approximate solution to the problem and is computationally efficient. However, it does not solve the global optimization problem. We use a simple heuristic explained below to obtain a better solution.

Optimization with OD re-weighted cost. The solution of the greedy algorithm helps us identify the paths which agree with the OD-prior. Hence, the transition flow variables set by this algorithm provide a rough estimate of the pairwise affinity between tracklets in the presence of OD-prior. We use this intuition to add an additional cost which penalizes the link between tracklets which were not originally connected by the greedy algorithm. While adding this cost, we remove the original OD-prior cost γ_{od}, thus resulting in a network-flow problem which can be solved by k-shortest path approach. The modified cost $\tilde{C}$ is shown below:

$$\begin{aligned} \tilde{C}(f) \;=\; & \sum_{x_i \in X} \alpha_i f_i + \sum_{x_i, x_j \in X} \tilde{\beta}_{ij} f_{ij} + \\ & \sum_{\substack{x_i \in X, \\ o \in O}} \tilde{\beta}_{oi} f_{oi} + \sum_{\substack{x_i \in X, \\ d \in D}} \tilde{\beta}_{id} f_{id}, \end{aligned} \tag{13}$$



(a) Toy example (b) Coarse-to-fine data association

Figure 10: (a) Toy example of 3 tracklets which could be wrongly linked. The dashed red arrows illustrate wrong assignments that are likely to occur without a coarse-to-fine data association. (b) Coarse-to-fine data association given SAM cardinality. Each sub-graph corresponds to the tracklet association problem over tracklet groups of specific cardinalities, denoted by $\mathbf{C}(sam)$ representing the sum of the elements of the SAM feature. The flow variables obtained by solving these sub-problems are used to defined additional transition costs used in the final optimization.

where $\tilde{\beta}$ is the OD-re-weighted cost defined below.

$$\tilde{\beta}_{ij} = \beta_{ij} + \lambda \mathbf{1}(f_{ij}^{greedy} = 1), \tag{14}$$

where f_{ij}^{greedy} is the solution obtained from the greedy algorithm and λ is a parameter indicating the strength of the OD-prior cost. The transition cost is re-weighted for all pairs of tracklets including the origin and destination tracklets.

6.5. Coarse-to-Fine Data Association

The model presented in Sec. 6.3, uses a social affinity cost to ensure that tracklets with similar grouping cues are connected. However, it does not account for the fact that people belonging to groups of different cardinalities (number of people in a group) can still share the same SAM feature. An example is shown in Fig. 10.a, where two tracklets belonging to groups of different cardinalities are wrongly connected (indicated in red) due to similar SAM. However, we want to encourage tracklets from groups of similar sizes to be connected together (black arrows). We account for this by proposing a coarse-to-fine data association method.

We cluster tracklets co-occurring at the same time, into different groups based on the social separation. The cardinality of a tracklet denoted by $\mathbf{C}(x_i)$ is the number of people belonging to the group corresponding to the tracklet x_i. We can imagine that if the clustering is perfect and people moved in the same configuration across the entire camera network, it would suffice to link the tracklet groups instead of the tracklets. This would also solve the problem of tracklets being linked across groups of different cardinalities. However, in practical setting, the grouping is not perfect and people break away from groups. Hence, we link the groups of same cardinality and use the links obtained from this group tracking to define additional transition costs. The complete method is explained in the supplementary document. The method is briefly visualized in Fig. 10.b.

7. Experiments

7.1. Large-scale evaluation

The data collection campaign helps us conduct various experiments in real life setting with a large and dynamic crowd. In this section, we present a set of experiments to address the tracking problem in scattered camera network. We select a subset of cameras in our network and measure the performance of our algorithm to track mobility with only these cameras.

Measurement. In this section, we evaluate the correct estimation of the origin and destination of a person entering the camera network. We have limited the monitoring to 14 origins and destinations leading to 196 possible OD-path for a trajectory. We have clustered the cameras into two groups: cameras belonging to OD locations (*i.e.,* capturing the beginning or ending of long-term tracks), and cameras in-between these locations. We compute the OD error rate as the percentage of wrong predictions out of the total number of people covered by the camera network.

Ground truth. Since Big Data is collected, it is not realistic to label the millions of trajectories. We hence use as labels the output of our detection and tracking algorithm. To reach high level of tracking accuracy, we have installed a dense network of cameras to reduce the blind spots as much as possible and link tracklets that are only a few centimeters away from each other. The trajectories computed from this dense network is used as a baseline (our labels). While the trajectories (and OD) computed from the dense network is not the perfect ground truth, in practice they are less easy and less expensive to obtain than manually annotating trajectories at our scale. The goal of our forecasting algorithm is to reach the same performance as the dense network of cameras while using a sparse network.

7.2. OD forecasting

Figure 11 presents the resulting OD error rates for 7 sparse networks of cameras. The evaluation is carried out at several levels of network sparsity, from 0% to 75% of in-between cameras. For instance, networks N4 and N5 use only half of the cameras available in the corridor (see figure 1). The cameras are selected to heuristically minimize the average distance between them at any given sparsity. At a given sparsity, we also evaluate on different camera configurations such as N4 and N5 for 50% sparsity. In average, tracklets

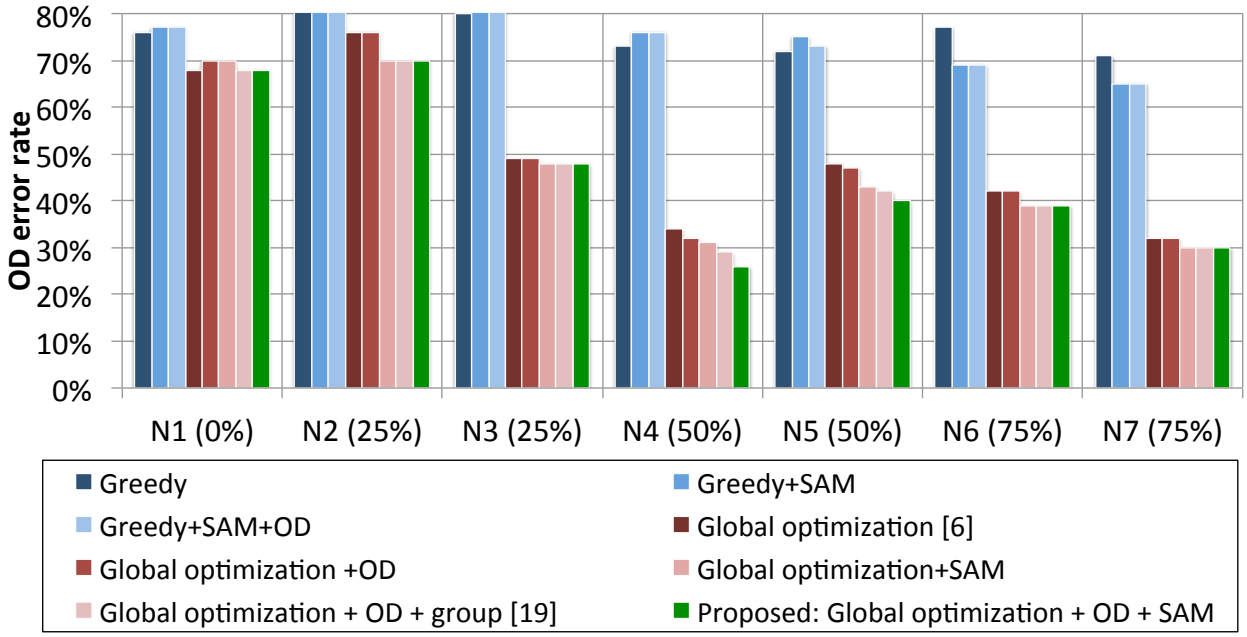


Figure 11: Performance of OD forecasting with different number of in-between cameras. The percentage of in-between cameras are shown in brackets. Seven network configurations are evaluated (referred to as N1 to N7).

from network N1 to N3 are several dozen of meters away from each other, and tracklets from networks N4 to N7 are dozen of meters away from each other. To validate our algorithm, we evaluate the performance of greedy optimization methods against the proposed global one. We measure the impact of using SAM as an additional feature, as well as the impact of modeling the OD prior with coarse to fine tracking.

As expected, the global optimization methods always outperform the greedy methods with and without OD prior. The performance improvement is more than doubled, in the global optimization method. The SAM feature and use of OD-re-weighted cost (use of OD-prior) are both seen to have a positive impact while using global optimization. This justifies our decision to model heuristically model the effect of OD-prior during optimization.

We also compare with the algorithms from [12] and [13]. Our final full model, *i.e.,* "Global optimization + OD + SAM", outperforms these methods when observations are limited to the corridor. Note that the camera placement has an impact on the forecasting. Although the same number of cameras are used by networks N2 and N3, or N4 and N5, the forecasting accuracy differs for these networks. If an in-between camera is strategically placed to capture frequent route choices, it reduces the uncertainty in the linking strategy. This leads to different performance for networks with same number of cameras as shown in Fig. 11

We evaluate the extreme setup when there are no in-between cameras (label as N1), *i.e.,* we only have cameras at entrance and exit zone (OD cameras). In such setup, tracklets are

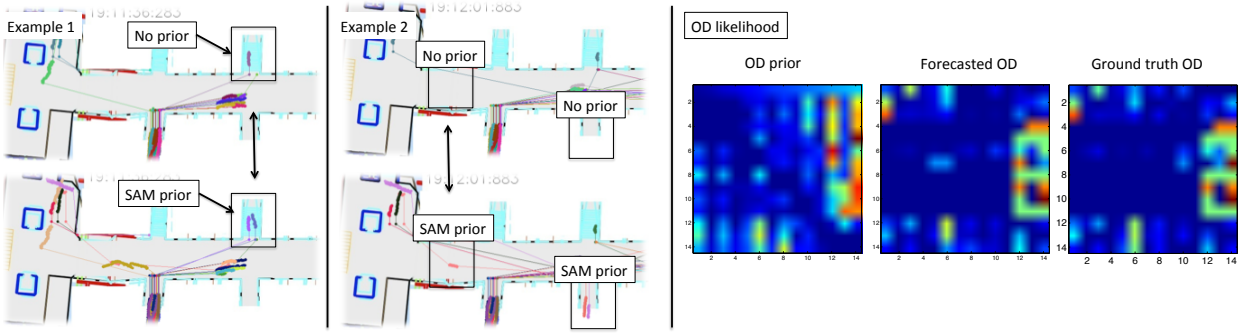


Figure 12: Qualitative results on the linked tracklets within the sparse network 1 where 50% of the in-between cameras within the corridor are not used. Tracklets selected by the method are only shown. The lines illustrate the linked tracklets. On the right side, we illustrate the OD prior as a heatmap, as well as the forecast and ground truth. We can see that although the prior is different, the final result is still similar to the ground truth.

up to 100 m away from each others. Figure 11 presents the resulting drop in performance. The gap between greedy and global optimization is much smaller. In addition, the SAM feature and OD prior do not have an significant impact on such extreme case. These results motivate our future work to handle such extreme case.

Figure 12 illustrates some qualitative results demonstrating the power of SAM. We also plot the OD prior, forecasted OD with a sparse network of cameras with half the number of cameras as the dense network (ground truth).

Impact of SAM We illustrate the tracklet linking achieved by our full method and compare it with a global optimization method which does not use SAM in Fig. 12. As expected, we see that in the absence of SAM, tracklets travelling in similar group configurations are not connected together, leading to erroneous results. On the other hand, SAM helps disambiguate between tracklet choices which are similar to each other, except for the group configuration.

Impact of OD prior In Fig. 12, we present the final OD-matrices estimated by our full model, and compare it with the OD-prior and the ground truth OD (from dense camera network). Clearly, the prior only provides weak cues about the true OD, but helps by down-weighting paths which are highly unfavorable like blocked corridors. The OD-matrix forecasted by our method is close to the ground truth OD matrix obtained from a dense camera network.

8. Conclusions

We have presented an efficient system to detect and track millions of individuals in real-world crowded environments. The first step in the system used a dictionary based sparsity promoting method to detect and track people within the field-of-view of a single camera. These short "tracklets" from multiple cameras were then linked to each other to obtain long-term human trajectories. We showed that social affinities between people can be modeled

in an effective fashion to improve this tracklet association. These affinities were captured through a new powerful SAM descriptor, which empowers tractable global optimization of the tracklet association problem. We also deployed a large network of cameras to enable large-scale analysis of real-world crowd motion. Several hundred thousands trajectories were collected per day leading to more than 100 million trajectories to date. It helps in the development of new motion priors to predict human behavior in crowded scenes. In the next chapter, we will show that it is possible to not just track but also *predict* long-term human behaviors from these millions of trajectories.

References

[1] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, Cascade object detection with deformable part models, in: CVPR, IEEE, 2010.

[2] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: CVPR, IEEE, 2014.

[3] O. Tuzel, F. Porikli, P. Meer, Pedestrian detection via classification on riemannian manifolds, IEEE, 2008.

[4] R. Benenson, M. Mathias, R. Timofte, L. Van Gool, Pedestrian detection at 100 frames per second, in: CVPR, IEEE, 2012.

[5] M. Enzweiler, D. M. Gavrila, Monocular pedestrian detection: Survey and experiments, PAMI.

[6] A. Alahi, L. Jacques, Y. Boursier, P. Vandergheynst, Sparsity driven people localization with a heterogeneous network of cameras, Journal of Mathematical Imaging and Vision.

[7] D. Delannay, N. Danhier, C. D. Vleeschouwer, Detection and recognition of sports(wo)man from multiple views, in: Proc. ACM/IEEE International Conference on Distributed Smart Cameras, Como, Italy, 2009.

[8] R. Eshel, Y. Moses, Homography based multiple camera detection and tracking of people in a dense crowd, in: Proc. IEEE International Conference on Computer Vision and Pattern Recognition, 2008, pp. 1–8.

[9] S. M. Khan, M. Shah, Tracking multiple occluding people by localizing on multiple scene planes, PAMI.

[10] F. Fleuret, J. Berclaz, R. Lengagne, P. Fua, Multicamera people tracking with a probabilistic occupancy map, TPAMI.

[11] M. Golbabaee, A. Alahi, P. Vandergheynst, Scoop: A real-time sparsity driven people localization algorithm, Journal of Mathematical Imaging and Vision 48 (1) (2014) 160–175.

[12] J. Berclaz, F. Fleuret, E. Turetken, P. Fua, Multiple Object Tracking using K-Shortest Paths Optimization, TPAMI.

[13] L. Leal-Taixe, G. Pons-Moll, B. Rosenhahn, Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker, in: ICCV Workshops, 2011.

[14] Z. Khan, T. Balch, F. Dellaert, Mcmc-based particle filtering for tracking a variable number of interacting targets, Pattern Analysis and Machine Intelligence, IEEE Transactions on 27 (11) (2005) 1805–1819.

[15] P. Nillius, J. Sullivan, S. Carlsson, Multi-target tracking-linking identities using bayesian network inference, in: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, Vol. 2, IEEE, 2006, pp. 2187–2194.

[16] Y. Xiang, A. Alahi, S. Savarese, Learning to track: Online multi-object tracking by decision making, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4705–4713.

[17] C. Kuo, C. Huang, R. Nevatia, Inter-camera association of multi-target tracks by on-line learned appearance affinity models, ECCV.

[18] A. Ess, B. Leibe, K. Schindler, L. Van Gool, A mobile vision system for robust multi-person tracking, in: CVPR, IEEE, 2008.

[19] L. Zhang, Y. Li, R. Nevatia, Global data association for multi-object tracking using network flows, in: CVPR, 2008.
[20] H. Pirsiavash, D. Ramanan, C. C. Fowlkes, Globally-optimal greedy algorithms for tracking a variable number of objects, in: CVPR, 2011.
[21] M. Andriluka, S. Roth, B. Schiele, People-tracking-by-detection and people-detection-by-tracking, in: CVPR, 2008.
[22] O. Javed, Z. Rasheed, K. Shafique, M. Shah, Tracking across multiple cameras with disjoint views, in: Proc. IEEE International Conference on Computer Vision, IEEE Computer Society, Washington, DC, USA, 2003, p. 952.
[23] B. Song, T. Jeng, E. Staudt, A. Roy-Chowdhury, A stochastic graph evolution framework for robust multi-target tracking, ECCV.
[24] A. A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, W. Hu, Multi-object tracking through simultaneous long occlusions and split-merge conditions, in: CVPR, 2006.
[25] Q. Yu, G. Medioni, I. Cohen, Multiple target tracking using spatio-temporal markov chain monte carlo data association, in: CVPR, 2007, pp. 1–8.
[26] A. Alahi, V. Ramanathan, L. Fei-Fei, Socially-aware large-scale crowd forecasting, in: CVPR, 2014.
[27] D. Helbing, P. Molnar, Social force model for pedestrian dynamics, Physical review E.
[28] M. Luber, J. Stork, G. Tipaldi, K. Arras, People tracking with human motion predictions from social forces, in: ICRA, 2010, pp. 464–469.
[29] S. Pellegrini, A. Ess, K. Schindler, L. Van Gool, You'll never walk alone: Modeling social behavior for multi-target tracking, in: ICCV, 2009.
[30] Z. Qin, C. R. Shelton, Improving multi-target tracking via social grouping, in: CVPR, IEEE, 2012.
[31] G. Antonini, M. Bierlaire, M. Weber, Discrete choice models of pedestrian walking behavior, Transportation Research Part B.
[32] S. Pellegrini, A. Ess, L. Van Gool, Improving data association by joint modeling of pedestrian trajectories and groupings, in: ECCV, 2010.
[33] T. Lan, L. Sigal, G. Mori, Social roles in hierarchical models for human activity recognition, in: Computer Vision and Pattern Recognition (CVPR), 2012.
[34] B. Yang, C. Huang, R. Nevatia, Learning affinities and dependencies for multi-target tracking using a crf model, in: CVPR, 2011.
[35] V. Hedau, D. Hoiem, D. Forsyth, Recovering the spatial layout of cluttered rooms, in: ICCV, IEEE, 2009, pp. 1849–1856.
[36] D. Fouhey, V. Delaitre, A. Gupta, A. Efros, I. Laptev, J. Sivic, People watching: Human actions as a cue for single view geometry.
[37] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, K. Schindler, MOTChallenge 2015: Towards a benchmark for multi-target tracking, arXiv:1504.01942 [cs]ArXiv: 1504.01942.
URL `http://arxiv.org/abs/1504.01942`
[38] M. Moussaïd, N. Perozo, S. Garnier, D. Helbing, G. Theraulaz, The walking behaviour of pedestrian social groups and its impact on crowd dynamics, PloS one 5 (4) (2010) e10047.