Unsupervised Camera Localization in Crowded Spaces

Alexandre Alahi, Judson Wilson, Li Fei-Fei, Silvio Savarese

***Abstract*— Existing camera networks in public spaces such as train terminals or malls can help social robots to navigate crowded scenes. However, the localization of the cameras is required, i.e., the positions and poses of all cameras in a unique reference. In this work, we estimate the relative location of any pair of cameras by solely using noisy trajectories observed from each camera. We propose a fully unsupervised learning technique using unlabelled pedestrians motion patterns captured in crowded scenes.**

We first estimate the pairwise camera parameters by optimally matching single-view pedestrian tracks using social awareness. Then, we show the impact of jointly estimating the network parameters. This is done by formulating a nonlinear least square optimization problem, leveraging a continuous approximation of the matching function. We evaluate our approach in real-world environments such as train terminals, where several hundreds of individuals need to be tracked across dozens of cameras every second.

I. Introduction

Nowadays, cameras are everywhere: "an average American citizen can be caught on camera more than 75 times a day" [8]. Public places such as train terminals, malls, or retail shops are monitored by dozens of fixed cameras (originally installed for security purposes). These already installed cameras are a valuable source of information to push the limits of automatic perception in the quest to have social robots everywhere [37]. Robots can use these cameras to obtain the locations of humans that might be out of their own sensing range, or occluded from their point of view. In general, these cameras networks can help the robots in their navigation and go beyond perception by predicting what could happen next [29], [13], [39], [3]. For example, it has been demonstrated that traffic/security cameras can assist in the detection of pedestrians from vehicles [1].

However, to fully take advantage of existing camera networks in both indoor and outdoor environments, the knowledge of the cameras' location is required. Unfortunately, in practice, this knowledge is not available across the 210 million of already installed cameras that were originally deployed for video surveillance purposes [38].

In this work, we use a data-driven method to localize cameras from the same scene, i.e., the relative positions and angles between all cameras. Note that cameras are fixed, and targets (mainly humans) are moving on the ground. These assumptions describe most of the public and private spaces (e.g., terminals, malls...) monitored by security cameras where humans navigate through. The input to our method is noisy human motion trajectories extracted by each camera

All authors are from Stanford University in the Computer Science department. Email alahi@stanford.com

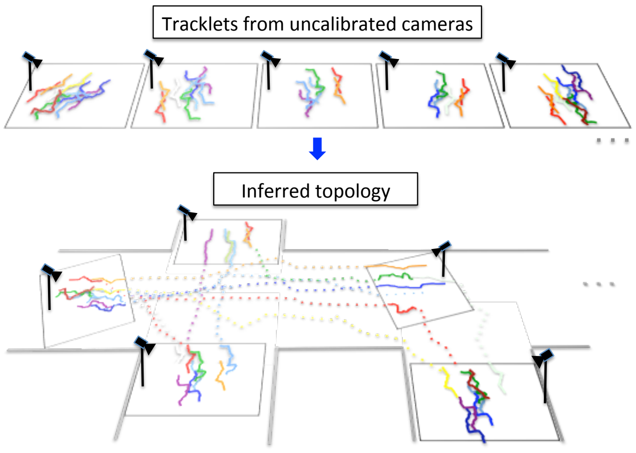


Fig. 1: (Top) Observed tracklets from 5 uncalibrated cameras (top view). These tracklets correspond to pedestrians that are observed to come in and out of the FOV of the cameras. Different color codes correspond to different track IDs. (Bottom) We aim to infer the relative placement of the cameras in the scene.

(see Figure 1). We use the output of a tracking algorithm which is prone to error [2]. The correspondence between the trajectories across cameras (referred to as tracklets in the rest of the paper) is not known.

Pioneering works [19], [27], [24], [4] have shown encouraging results towards our goal of locating cameras in constrained settings. They assume that the correspondence between an observed pedestrian from one view is known across a different view, e.g., a single person moves around the scene. Given the shift in time between the observations, it is possible to estimate the relative camera positions. However, in practice, cameras simultaneously observe dozens of individuals across their fields-of-view and such correspondence is not available. This causes the chicken-and-egg problem: the cameras' relative location can not be inferred without the correspondences and vice versa.

Our model draws inspiration from a number of works in the sensor network community [20], [7], [41]. These works locate the sensors given pairwise distances between the sensors. Our model is based on a non-linear optimization over the distances using Levenberg-Marquardt (LM) algorithm. However, in our camera network setup, we need to extend these techniques to handle the uncertainty in the pairwise estimation of the camera network (cameras are temporally synchronised). The challenges in this pairwise estimation step are the following:

- Tracklet correspondences between cameras are unknown . We need to estimate them from the data.

- Inter-camera observations of a track can be distant enough experiencing strong pedestrians' motions change. We need to forecast realistic motion behavior in non-observed regions.
- Recurrent motion patterns appear across cameras in high density crowds creating track ambiguities promoting several sets of parameters with the same likelihood.
- The tracklet association function between a pair of cameras is not surjective, *i.e.* not every tracklet from the first camera has a correspondence in the next camera.
- Tracklets are noisy since they are the output of a tracking algorithm.

We present in Sec. III-A our method to obtain the pairwise estimates given the aforementioned challenges and our proposed optimization step over a continuous cost function to address the uncertainty in the estimates (in Sect III-B). To the best of our knowledge, it is the first time that such model has been applied to locate cameras in crowded scenes.

The contributions of our paper are summarized as follows: (i) a new problem formulation, i.e., cameras localization solely with extracted tracklets in an unsupervised setting, (ii) a framework that can handle the ill-posed problem of estimating pairwise camera parameters thanks to our proposed joint optimization over a continuous cost function, (iii) an evaluation framework made of real-world crowd datasets to let the community address the same challenge with reproducible results. In Section IV, we show the performance of our method on several public datasets such as [43] and especially the challenging crowd dataset [2].

II. Related work

In this paper, we tackle the problem of locating cameras using tracklets extracted from multiple uncalibrated fixed cameras in crowded settings. The challenges involve extracting and matching tracklets across cameras, as well as calibrating the network. We propose to use pedestrian behavioral models and insights from the sensor network community to address these challenges. We briefly summarize key works in these areas.

Tracklet extraction and matching - Algorithms with high levels of accuracy have been proposed to locate humans with a single top view camera [9], [15], [14], [2]. Once humans are located on the ground, tracking algorithm is used to generate the tracklets by temporally linking located points with various techniques such as Markov Chain Monte Carlo (MCMC) [22], Bayesian networks inference [30], dynamic programming [14], , and more recently linear programming [23]. Once tracklets are extracted, similar frameworks can be used to match them across cameras [33], [44], [34]. Alahi *et al.* in [2] use Social Affinity Map (SAM) as an additional cue to reason on the matching cost.

Calibrating the camera network - First, a large body of work suppose that cameras have overlapping Field-Of-Views (FOV) [10], [28] whereas our work focuses on camera networks with non-overlapping FOVs. Moreover, existing works suppose that either a single person [4] or limited number of humans move around at different time intervals. For instance, they use Mutual information [40], cross correlation and covariance [26], or model visual appearance to learn the topology [19], [31], [42]. Marinakis *et al.* in [27] use tracklets observations of a known number of people with homogeneous behavior. In large-scale crowded environments such as train terminals, we cannot assume that a single person is moving around with homogeneous behavior. Moreover, to the best of our knowledge, none of the previous works have addressed the learning of a network made of several dozens of cameras where hundreds of pedestrians simultaneously enter and exit the FOV of the cameras from any possible location. We therefore propose to address such challenging scenarios using social cue and a non-linear optimization technique.

Behavioral models - A linearly independent human motion behavior assumption in crowded scenes is not realistic [4]. Pedestrian motion models that handle interactions between individuals are better estimates of pedestrian mobility [25], [32], [23], [5]. In crowded environments, pedestrians' behavior is highly influenced by others. D. Hellbing and P. Molnar in [16] proposed the classical social force model. Snape *et al.* [36] propose a motion model coined Optimal Reciprocal Collision Avoidance (ORCA) that works in the velocity space by finding the closest collision-free velocity to the preferred one. The ORCA motion model has the advantage of providing a smooth and collision-free behavior estimate in real-time using linear programming. We will leverage this model to infer our pairwise camera parameters.

Sensor network topology learning - The problem of learning network topology has been widely studied in the sensor network community. The most popular algorithm is the Multidimensional Scaling (MDS-MAP) algorithm, which locates sensors using pairwise distance constraints [20]. The method uses SVD to form a low dimensional projection of a complete set of pairwise distance measurements arranged in a matrix. Missing measurements introduces error although they are estimated by computing the shortest path along the known distances in the network. Biswas and Ye presented a Semidefinite Programming (SDP) relaxation of the non-convex localization problem [7], [41]. The method has potential benefits over MDS–MAP as it does not require complete pairwise distances between every node but it requires three anchor nodes of known location. Non-linear least squares regression (NLS) is another widely used family of algorithms which often produces more accurate results than MDS-MAP without much added computational expense [11]. Unlike MDS-MAP and the SDP relaxation, this algorithm can incorporate angular measurements to aid in placement. It is often solved using the Levenberg-Marquardt algorithm, of which implementations are widely available. We frame our camera network problem as a sensor localization problem using a probabilistic inference. Instead of having hard pairwise constraints, we have soft ones in the form of continuous cost functions. More details are provided in the next sections.

III. Method

We want to locate all cameras within the same scene, i.e., the relative positions and angles between all cameras. As a result, extracted tracklets from each camera can be matched across the network to analyze human mobility in large-scale and help social robots to navigate in such scenes.

We propose a framework made of two steps: i) a pairwise estimation of the camera parameters, and ii) a global optimization to obtain the optimal set of parameters over the full network. The first step is an ill-posed problem. Several solutions can describe the observed flow. Consequently, we propose to solve a joint optimization (step ii) over a continuous cost function describing the cost of each cameras pair. The second step will select the solution for all the cameras pairing in a network that maximize the likelihood of the full network.

A. Pairwise camera parameters estimation with social awareness

For any cameras pairing in a network, we first want to estimate the cameras' relative placements using only the observed tracklets. Formally, the relative pose between two cameras is modeled by 3D translation vector and 3D rotation matrix (or SE(3) groups). We propose an intermediate projection, i.e., projected tracklets on the ground, that will only require a 3 Degree Of Freedom (DOF) to find the cameras relative placements. Many methods exist to extract tracklets from RGB cameras [12], [21], or even RGB-D cameras [35], [17], [6], [2]. The tracklets are projected on the world ground plane (where the camera is the origin of the plane, as illustrated in Figure 1). As a result, our problem is cast into finding the relative 2D translation and rotation between the observed tracklets from one view to the other.

Formally, we intend to estimate 2D pairwise camera constraint parameters tuples $p = (\alpha_m, \alpha_n, r_{m,n})$ given the observed tracklets, where scalar $r_{m,n}$ is the distance between the origins of camera m and n, and α_m and α_n are the angles of cameras m and n (respectively) relative to a line segment between the two camera origins (see Figure 3(a)). Cameras can be far-away from each other, observing high-density crowds.

To estimate the cost associated with a pairwise camera constraint parameters, p, we compute its *match count* M as the number of tracklets that find a match in the second camera by forecasting their trajectory with a collision avoidance motion model (ORCA [36]) and a social affinity descriptor (SAM [2]) given the relative camera placement:

$$M_{\{x^*,y\}}(p) = \sum_{i,j} \mathbb{1}(x_i^*, y_j, p) \tag{1}$$

where $p = (\alpha_m, \alpha_n, r_{m,n})$ is the parameter tuple being evaluated, x_i^* is the forecasted coordinates of tracklet x from camera m using the ORCA motion model [36] presented in Section II, y_j is the observed starting point of a tracklet in camera n, and $\mathbb{1}(\cdot)$ is an indicator function indicating a spatial match of the two points when the cameras are positioned relative to each other according to p.

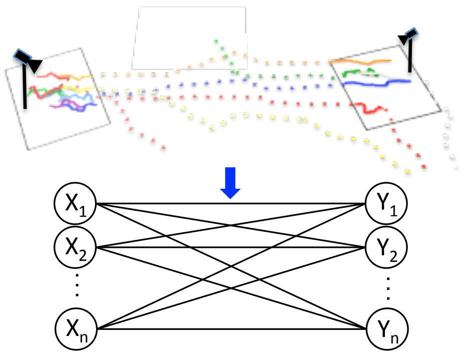


Fig. 2: For a camera pair, we use a bipartite graph to match tracklets with social awareness. Each edge is weighted by the error of matching the forecasted tracklet in one camera to the tracklet in the other camera.

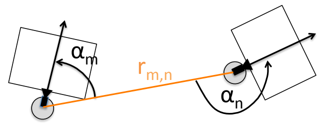
a) Greedy matching: We first consider as a match as long as $\|X_i^* - Y_j\|_2^2 < T$ where $T = 1$ meter. We prune the search space by keeping the candidates that have high likelihood with such simple matching rule, referred to as ORCA augmentation Section IV. We keep the top 10% of the tuple with highest number of match. Next, we use a bipartite graph matching method to fine-tuned the number of matches by finding the optimal assignment given both the forecasting error as well as the Social Affinity Map (SAM) [2].

b) Bi-partite graph matching: For a given pairwise camera parameters tuple $(\alpha_m, \alpha_n, r_{m,n})$, we construct a bipartite graph $G_b = G_b(V_1 \cup V_2, E)$ where vertices V_1 and V_2 represent tracklets X_i from camera 1 and tracklets Y_i from camera 2, respectively (see Figure 2). The weight d_{ij} of an edge $e_{ij} \in E$ represents the forecasting accuracy and Social (group) Affinity Map (SAM) [2] between tracklet i and j:

$$\begin{aligned} d_{ij} &= \beta_{ij}^{\text{Forecasting}} + \beta_{ij}^{\text{SAM}} \\ &= \|X_i^* - Y_j\|_2^2 + H(\text{sam}_i, \text{sam}_j), \end{aligned} \tag{2}$$

where X_i^* is the forecasted coordinates of tracklet i from V_1 at the start time of tracklet Y_j from V_2, H is the hamming distance between the *sam* binary vector of the the two tracklets i and j. The *sam* vector encodes the spatial position of the tracklet's neighbors with a binary radial histogram. It compares the social affinity map of a person with its neighbor, *i.e.* the relative position of your neighbors. The matches corresponding to the parameters set is obtained by applying the minimum weight bipartite matching algorithm presented in [18].

For each point in a discrete range of parameter values $(\alpha_m, \alpha_n, r_{m,n})$, we calculate a score M representing the number of tracklet matches. We assume a higher score M indicates a better relative placement of cameras, and we seek a global placement of all cameras that maximizes the matching tracklets between all cameras. The next section describes our methods for formulating and solving this problem.



(a) Legend of the parameters

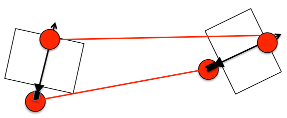

(b) "Two-point per camera" representation

Fig. 3: (a) Illustration of the pairwise camera parameters to estimate from bird view (top view). (b) A camera can be represented by a 2 points formulation see Section 5.4.

B. Global optimization over the full network

Our final goal is to estimate the locations of all cameras in a scene given estimations for pairwise constraints. For a given camera pair, several parameters can share similar match counts. We aim to jointly maximize all the match counts.

We model the pairwise constraints between cameras as a "constraint graph" $G = (N, E)$, where the nodes $N = s_i$ represent each camera and each edge E represents a known tuple of constraints between two cameras, such as the distance and angle constraints. If this graph is connected, the full relative placement may be estimated, either by directly using the constraints of form $(\alpha_m, \alpha_n, r_{m,n})$, or by a "two-point per camera" formulation, where camera position and angle information are captured by representing each camera as a pair of points, as depicted in Figure 3(b).

A greedy approach can iteratively place each camera with respect to a connected neighbor by choosing a minimum subset of the estimated pairwise parameter tuples. A tuple is sufficient to fully constrain the relative placement of two cameras. However, errors in the placement are propagated over the network. It is also possible to use the MDS-MAP algorithm or SDP relaxation mentioned in Section II. However, these methods use a fixed estimate for each pairwise constraint. More details are provided in Section IV.

From our tracklet matching approach, we can estimate a match count for a discrete set of points in the constraint value space. We offer to leverage this using a non-linear least square optimization method.

C. Levenberg-Marquardt nonlinear least squares optimization

We propose to formulate the joint optimization as a nonlinear least squares (NLS) minimization problem:

$$\arg\min_x \sum_i f_i(x)^2 \tag{3}$$

where $f_i(x)$ are nonlinear residual functions. We solve this using the Levenberg-Marquardt (LM) algorithm.

To create a residual or cost function, we borrow concepts from statistics by modeling each tracklet match as a sample at a given value for a specific pairwise camera relative constraint, and seek a solution that is statistically likely. Mean and variance constants are calculated from these constraint-value samples, and the residual functions of the least squares problem are constructed such that the problem is equivalent to the maximum log-likelihood estimation of independent Gaussians. The basic form for the residual for the distance between cameras is:

$$f_{i,j}(r_{i,j}) = \frac{(r_{i,j} - \mu_{i,j})}{\sqrt{2}\sigma_{i,j}} \tag{4}$$

where the variable $r_{i,j}$ is the distance between cameras i and j, and $\mu_{i,j}$ and $\sigma_{i,j}$ are the constant mean and variance values of the samples.

A similar formulation is used to fit a Gaussian distribution representation to the angular constraints, and produce a residual for the camera angles. Because the angular samples are periodic, we applied a window centered at the mean of angles (defined in Section III-D) before calculating the mean and variance parameters.

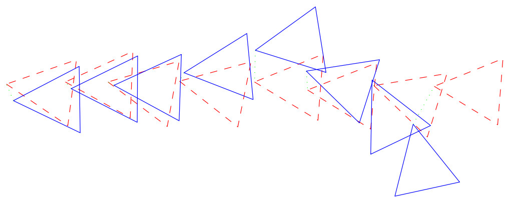
The actual domain of the optimization problem is the absolute locations and angles of the cameras. The residual functions used in practice are the composition of the functions $f_{i,j}$ described above with functions that translate absolute positions and angles to relative distances and angles using basic geometry. The composed functions are nonlinear, and the problem has infinitely many optimal solutions that are equal up to a global rotation.

The LM algorithm can be used with any initialization. However, since it may converge to a local optimum, we use MDS-MAP to generate the initial placement estimate for the descent algorithm.

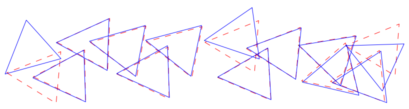
D. Multidimensional Scaling-MAP initialization

First, MDS-MAP is applied as described in the literature [20], using unit weighting for the shortest-path estimates of the missing distances. Second, the angle of each camera is estimated using the $(\alpha_m, \alpha_n, r_{m,n})$ constraints as a separate step. The constraint graph will over-constrain the angular orientation of cameras when edges appear on a cycle. After positions have been determined, each pairwise parameter tuple for a camera infers a specific camera angle. The final angle is estimated by taking the mean-of-angles of the orientation angle implied by each constraint. The mean-of-angles is calculated by assigning a unit vector in the direction of each camera angle implied by each constraint, followed by summing these unit vectors and returning the angle of the resulting vector.

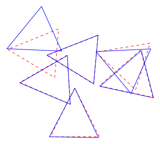
Because MDS-MAP works with distance constraints only, there exists reflectional ambiguity in the solution. We account for this ambiguity by computing the camera angles for both the solved positions and the reflection of the solution, and choosing the placement which minimized the angular cost. The total angular cost is chosen as the sum of the quadratic cost for each constraint angle. The cost associated with the angular component α_i of a camera pairwise constraint $(\alpha_i, \alpha_j, r_{i,j})$ is a squared distance-weighted-error function, where the difference of the constraint angle and the corresponding placement angle is multiplied by the associated distance constraint $r_{i,j}$ and squared. Thus, the overall angular cost is a function of the absolute 2D positions (a_i, b_i) and angles θ_i of the cameras:



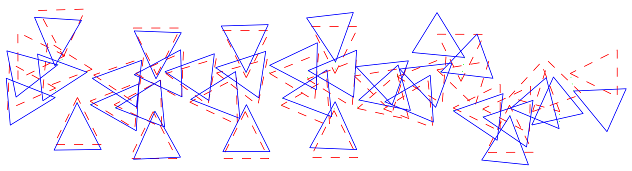
(i) A single linear network arrangement



(ii) A double linear network arrangement



(iii) A non-linear network arrangement



(iv) The full network of 33 cameras

Fig. 4: Illustration of the final topology inferred with our approach for different characteristic camera network arrangements, including the full network of 33 cameras (vi). Solid outlined blue triangles represent the inferred placement, and the dashed red triangles represent the ground truth.

$$C_{\alpha_i}(\{(a, b, \theta)\}) = \left[\left(\alpha_i - (\theta_i - \operatorname{atan2}(b_j - b_i, a_j - a_i))\right) r_{i,j}\right]^2. \quad (5)$$

The distance-weighting term attempts to scale angular error into an appropriate distance error.

The next section will demonstrate the effectiveness of this NLS algorithm when applied to the pairwise constraint estimates.

IV. Experiments

A. *Real-world datasets*

We use several datasets to run our experiment. The most relevant one is the Crowd dataset [2]. In this dataset, more than a hundred cameras have been installed in two train terminals. The cameras are clustered into 5 sites where each site is independently monitored. For our experiments we chose a site with 33 cameras. The collected tracklets are dense (up to one pedestrian$/m^2$) and the dataset contains 42 million trajectories. Several hundred pedestrians may be simultaneously present as illustrated in Figure 6 (a top view of the corridor, 100m long and 8m wide with 14 origin/destination entryways). For the sake of generalization, we have also used trajectories collected from other publicly available datasets such as [43] for the ZARA01, ZARA02, and UCY sequences.

B. *Evaluation protocol*

To quantitatively measure the error in locating cameras, we compute the sum-of-squares error measurements. Positional error is defined as the sum of squared Euclidean distances of the cameras from their calculated location to their ground truth location (provided by the datasets). Angular error is calculated as the sum of squared differences, on the range $[-\pi, \pi)$, between the calculated absolute camera angles and the angles of the ground truth cameras.

We evaluate various network sparsity and camera arrangement grouped as follows: (i) linear arrangement, cameras are placed over a single line (see Figure 4) (ii) dual-linear arrangement, cameras are placed over two lines (iii) non-linear arrangement, cameras do not follow a structure (iv) the full set of 33 cameras from the crowd dataset. Figure 4 presents qualitative results over the suggested grouping.

Several approaches have been previously proposed to locate cameras (often referred to camera network topology). For instance several methods [40], [26] cluster tracklets observed in the FOV of the cameras into few entry/exit zones. Then, they use the departure times and arrival times as two signals. From this, they find the best time offset between the two signals by maximizing correlation. As can be seen in Figure 6, this does not model crowded scenes well. Entry/exit zones tend to spread to the entire outline of the cameras, and pedestrians do not tend to leave a small exit zone and then enter a corresponding small entry zone. We limit our evaluation to methods that are compatible with our challenges/datasets, *i.e.* (i) non-overlapping FOV, (ii) unknown tracklet association between cameras, and (iii) crowd behavior with high pedestrian density.

C. *Evaluation of the pairwise estimation of the camera parameters*

The accuracy of pairwise camera parameters estimation depends on the distance between the cameras and the predictability of the flow of people between the cameras. For instance, uncertainty of pedestrians' paths usually grows with distance traveled between cameras. In Figure 5, we first present the error of inferring the full network with linear motion model, ORCA augmentation, and our proposed bipartite graph matching with social awareness. Note that we use the same greedy optimization to place the cameras to compare the performance of methods for pairwise estimation of the cameras parameters. Various camera networks have been evaluated: Full network of 30 cameras, single linear flow, dual linear flow, and non linear flow 5 have (in average) cameras distant by 3m to 5m; non linear flow 1 & 2 have cameras distant by 7m to 9m; non linear flow 3 & 4 have cameras distant by 14m to 16m.

We briefly discuss the results for these methods:

a) Linear prediction: We evaluate the performance of linear projection of the tracklets to infer the camera's pairwise parameters. The only difference with the ORCA augmentation method is the motion model. We evaluate the performance of a simple linear one that is not avoiding

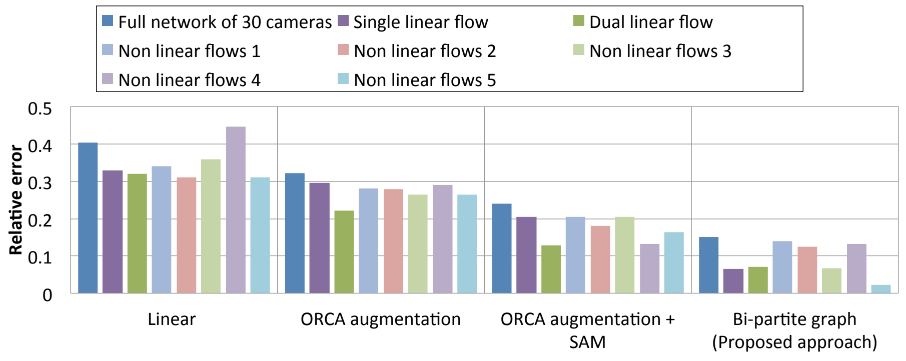


Fig. 5: Quantitative results for various pairwise camera parameter estimation method (in m). We evaluate over several camera networks arrangements as described in 5.2. (linear single arrangement, double, or non linear). The ids (1-5) represents different camera networks types.

collisions or comparing Social Affinity Map. As seen in Figure 5, it performs the worst.

b) ORCA augmentation: This method augments the tracklets as explained in Section 3. The benefit over linear prediction is the capability to avoid collisions and hence forecasting more plausible dynamics. In Figure 5, we can see that such an approach is better than the linear motion model. We use this algorithm to prune the space before evaluating our bipartite graph matching method, as described in Section 3.

c) ORCA augmentation+SAM: We evaluate the impact of verifying whether the Social Affinity Map is the same after forecasting the tracklets from the first camera to the second. We only keep the tracklets that have the same SAM. The performance is slightly improved (see Figure 5).

d) Bipartite graph matching: We can see that our proposed algorithm to estimate the pairwise camera parameters outperforms other methods. It is worth mentioning that in addition to better performance in estimating the distances, the distribution of the estimation can be approximated with a single Gaussian with low variance. Such distribution is used to learn the connectivity of the camera networks.

D. Performance of the full optimization of the network parameters

In this Section, we study the joint optimization to locate the cameras. Figure 7 presents the quantitative results of our approach with respect to the greedy baseline approach as well as other competitive approaches. We used the proposed bipartite graph matching step for all comparisons. We have evaluated the performance of greedy placement, MDS-MAP, SDP-Relaxation, and our proposed LM nonlinear optimization. In addition to solving MDS-MAP and the SDP-Relaxation using position constraints and calculating angle from angle constraints as a second step, we also modeled cameras as a pair of points and associated distance constraints (see Figure 3), implicitly solving for angle without additional steps. We discuss the performance of all the mentioned methods:

a) Greedy method: This method relatively places cameras starting with the pairwise constraint of highest match factor (Equation 1). Additional pairwise constraints are chosen from the remaining set in order of highest match factor, rejecting those pairwise constraints which would make the chosen set overconstrained. The resulting set of constraints implies a unique relative placement of the cameras. The performance of this approach is competitive with other algorithms in various networks. Comparing to the MDS-MAP and SDP formulations, which were both performed using the candidate constraints of highest match factor, shows that there are occasions where the constraint corresponding to the most number of tracklet matches can produce better results than optimizing over multiple overdetermined constraints. In other occasions, the opposite is true. Nevertheless, the greedy approach is particularly susceptible to propagation of error in the constraints.

b) MDS-MAP algorithm: For networks having nearly collinear cameras, such as the 'Single linear flow' networks, MDS-MAP produces much worse errors than other algorithms, possibly because the algorithm cannot properly resolve one dimension. This is especially true when using angular constraints directly rather than the 2-point camera constraint formulation. For the non-collinear networks, the angle constraint formulation usually performs better than the 2-point constraint formulation. One explanation for this is that the 2 points formulation may create new points that are near other points, but without a pairwise constraint between them. The shortest-path estimation step will likely find a much longer distance than the true distance, which introduces large errors.

c) SDP-relaxation localization algorithm: As described in Section II, an SDP-relaxation [41] method can be used as an alternative to MDS-MAP. Since we do not have anchors, we have modified the algorithm to first start with the MDS-MAP solution. Then, 3 random nodes (cameras) are chosen to act as anchors. When angular constraints are used, camera angles are estimated using the same method employed for MDS-MAP. The candidate solution is adopted if it improves a cost function which penalizes the error of the constraints using sum-of-squared-error. If cost does not improve, the candidate solution is rejected. The procedure is repeated, randomly choosing new anchors each iteration, until a fixed

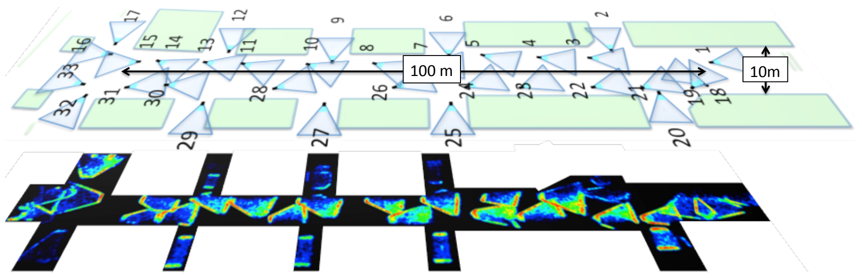
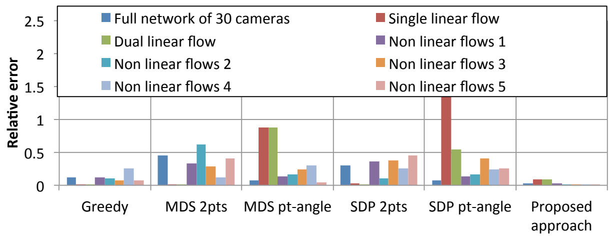
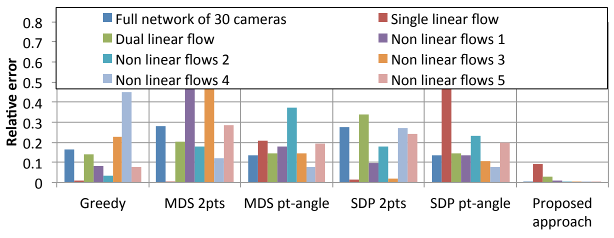


Fig. 6: (Top row) Illustration of a corridor in a train terminal from bird's-eye (top) view where 96 pedestrians are moving around (each colored dot represents a pedestrian). (Middle row) Illustration of the camera network field of view from bird's-eye (top) view. (Bottom row) The heatmap of the tracklet entry/exit of each camera FOV. Each point represents the number of times a pedestrian enter or exit the camera view (red is high, blue is low).



(a) Position error



(b) Angular error

Fig. 7: Quantitative results over various networks of cameras as described in 5.2.: linear single arrangement, double, and non linear ones (in meters). The ids (1-5) represents different camera networks types. We compare to our proposed algorithm with only MDS-MAP [20], and our implementation of the SDP-relaxation [41]. Our proposed nonlinear least square optimization initialized with the MDS-MAP results outperforms the other methods for all networks tested except 'Single linear flow' (in red).

number of iterations occur without improvement in cost.

The overall cost is the sum of individual costs for each distance constraint $r_{i,j}$ and, each angular constraint α_i. The distance constraint cost function for a set of 2D camera positions $\{(a,b)\}$ for distance constraint $r_{i,j}$ is:

$$C_{r_{i,j}}(\{(a,b)\}) = \Big(r_{i,j} - \|(a_i,b_i) - (a_j,b_j)\|\Big)^2. \qquad (6)$$

The angular cost is defined in Section III-D.

Comparing the SDP-relaxation results to the MDS-MAP results, (specifically either the angular camera constraint formulation or the 2 points camera constraint representation), there are instances where error increases and instances where error decreases. It's important to note that cost is no worse for the SDP solution because our method starts with the MDS-MAP solution and only chooses new candidate solutions if the cost improves. This shows that fitting the constraints more accurately than the MDS-MAP solution, as measured by the cost function, does not always reduce our measure of accuracy. The SDP method does not seem to provide added benefit, while greatly increasing implementation complexity and runtime.

d) Proposed LM nonlinear least squares optimization: Overall, our proposed approach produces the best error by at least one or two orders of magnitude in every case, except for one network involving a single linear arrangement of cameras.

In networks containing only a pair of cameras, or any other network that is not overconstrained, the proposed algorithm will place each camera at a relative distance and angle that are exactly equal to the means of the analogous Gaussian distributions that were fit to the match count functions. For most tested cases, the proposed method resulted in an error that was orders of magnitudes lower than the other methods. Considering all of the results together, incorporating the smooth match count function estimate as a cost function produces a better cost measure for solving the overconstrained global network topology, i.e., the relative placement of the cameras in the scene.

V. Conclusions

In this work, we show that by solely having access to pedestrians' observed motion, it is possible to infer the relative placement of the cameras in the scene. We have validated our method over real-world, challenging spaces such as crowded train terminals with thousands of individuals per hour. For future works, we aim to demonstrate the impact of these calibrated camera networks on robot navigation. Robots could reason beyond their sensing range, perceive occluded objects, and generally improve their perception capability. As a results, we envision to deploy social robots in public crowded scenes such as in terminals to assist commuters, or in malls to help elderly people to shop around. We believe that the external network of cameras can play a fundamental role in the success of social robots.

Acknowledgment

We acknowledge the support of ONR-N00014-13-1-0761, MURI - WF911NF-15-1-0479 and Toyota Center grant 1191689-1-UDAWF.

References

[1] A. Alahi, M. Bierlaire, and M. Kunt, “Combination of Fixed and Mobile Cameras for Automatic Pedestrian Detection,” *Transportation Research Part C*.

[2] A. Alahi, V. Ramanathan, and L. Fei-Fei, “Socially-aware large-scale crowd forecasting,” in *CVPR*, 2014.

[3] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social lstm: Human trajectory prediction in crowded spaces,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 961–971.

[4] N. Anjum, “Camera localization in distributed networks using trajectory estimation,” *JECE*, 2011.

[5] G. Antonini, M. Bierlaire, and M. Weber, “Discrete choice models of pedestrian walking behavior,” *Transportation Research Part B*, vol. 40, no. 8, pp. 667–687, 2006.

[6] J. Biswas and M. Veloso, “Depth camera based indoor mobile robot localization and navigation,” in *ICRA*, 2012.

[7] P. Biswas and Y. Ye, “Semidefinite programming for ad hoc wireless sensor network localization,” in *International symposium on Information processing in sensor networks*. ACM, 2004, pp. 46–54.

[8] crimefeed, “Eyes on you: How many times are you caught on surveillance cameras per day?” 2015.

[9] D. Delannay, N. Danhier, and C. D. Vleeschouwer, “Detection and recognition of sports(wo)man from multiple views,” in *ICDSC*, Como, Italy, 30 Aug. - 2 Sep. 2009.

[10] E. Elhamifar and R. Vidal, “Distributed calibration of camera sensor networks,” in *ICDSC*. IEEE, 2009, pp. 1–7.

[11] C. Ellis and M. Hazas, “A comparison of mds-map and non-linear regression,” in *IPIN*. IEEE, 2010, pp. 1–6.

[12] R. Eshel and Y. Moses, “Homography based multiple camera detection and tracking of people in a dense crowd,” in *CVPR*, 2008.

[13] C. Estrada, J. Neira, and J. D. Tardós, “Hierarchical slam: Real-time accurate mapping of large environments,” *Robotics, IEEE Transactions on*, vol. 21, no. 4, pp. 588–596, 2005.

[14] F. Fleuret, J. Berclaz, R. Lengagne, and P. Fua, “Multicamera people tracking with a probabilistic occupancy map,” *PAMI*, no. 2, pp. 267–282, 2008.

[15] W. Ge and R. T. Collins, “Marked point processes for crowd counting,” in *CVPR*. IEEE, 2009, pp. 2913–2920.

[16] D. Helbing and P. Molnar, “Social force model for pedestrian dynamics,” *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[17] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, “Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments,” in *ISER*. Citeseer, 2010.

[18] J. Hopcroft and R. Karp, “An n^5/2 algorithm for maximum matchings in bipartite graphs,” *SIAM Journal on Computing*, vol. 2, no. 4, pp. 225–231, 1973.

[19] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, “Tracking across multiple cameras with disjoint views,” in *ICCV*. Washington, DC, USA: IEEE, 2003, p. 952.

[20] X. Ji and H. Zha, “Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling,” in *INFOCOM 2004*, vol. 4. IEEE, 2004, pp. 2652–2661.

[21] S. M. Khan and M. Shah, “Tracking multiple occluding people by localizing on multiple scene planes,” *PAMI*, vol. 31, no. 3, pp. 505–519, 2009.

[22] Z. Khan, T. Balch, and F. Dellaert, “Mcmc-based particle filtering for tracking a variable number of interacting targets,” *PAMI*, 2005.

[23] L. Leal-Taixe, G. Pons-Moll, and B. Rosenhahn, “Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker,” in *ICCV Workshops*. IEEE, 2011, pp. 120–127.

[24] E. J. Lobaton, P. Ahammad, and S. Sastry, “Algebraic approach to recovering topological information in distributed camera networks,” in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*. IEEE Computer Society, 2009, pp. 193–204.

[25] M. Luber, J. Stork, G. Tipaldi, and K. Arras, “People tracking with human motion predictions from social forces,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 464–469.

[26] D. Makris, T. Ellis, and J. Black, “Bridging the gaps between cameras,” in *CVPR*, vol. 2. IEEE, 2004, pp. II–205.

[27] D. Marinakis, G. Dudek, and D. J. Fleet, “Learning sensor network topology through monte carlo expectation maximization,” in *ICRA*. IEEE, 2005, pp. 4581–4587.

[28] S. Miller, A. Teichman, and S. Thrun, “Unsupervised extrinsic calibration of depth sensors in dynamic scenes,” in *IROS*. IEEE, 2013, pp. 2695–2702.

[29] J. Neira, M. I. Ribeiro, J. D. Tardós, *et al.*, “Mobile robot localization and map building using monocular vision,” in *In The 5th Symposium for Intelligent Robotics Systems*. Citeseer, 1997.

[30] P. Nillius, J. Sullivan, and S. Carlsson, “Multi-target tracking-linking identities using bayesian network inference,” in *CVPR*, vol. 2. IEEE, 2006, pp. 2187–2194.

[31] C. Niu and E. Grimson, “Recovering non-overlapping network topology using far-field vehicle tracking data,” in *ICPR*, vol. 4. IEEE, 2006, pp. 944–949.

[32] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, “You’ll never walk alone: Modeling social behavior for multi-target tracking,” in *ICCV*. IEEE, 2009, pp. 261–268.

[33] A. A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu, “Multi-object tracking through simultaneous long occlusions and split-merge conditions,” in *CVPR*, 2006.

[34] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, “Globally-optimal greedy algorithms for tracking a variable number of objects,” in *CVPR*, 2011.

[35] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, “Real-time human pose recognition in parts from single depth images,” *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[36] J. Snape, J. van den Berg, S. J. Guy, and D. Manocha, “Smooth and collision-free navigation for multiple robots under differential-drive constraints.” in *IROS*, 2010, pp. 4584–4589.

[37] B. Song, C. Ding, A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury, “Distributed camera networks,” *IEEE Signal Processing Magazine*, vol. 28, no. 3, pp. 20–31, 2011.

[38] StorageServers, “”how many video surveillance cameras are there in the world?”,” *on-line*, 2014.

[39] S. Thrun and M. Montemerlo, “The graph slam algorithm with applications to large-scale mapping of urban structures,” *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 403–429, 2006.

[40] K. Tieu, G. Dalley, and W. Grimson, “Inference of non-overlapping camera network topology by measuring statistical dependence,” in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2. IEEE, 2005, pp. 1842–1849.

[41] P. Tseng, “Second-order cone programming relaxation of sensor network localization,” *SIAM Journal on Optimization*, vol. 18, no. 1, pp. 156–185, 2007.

[42] Z. Xiaotao, B. Bir, and R. Amit, “Continuous learning of a multilayered network topology in a video camera network,” *EURASIP Journal on Image and Video Processing*, 2009.

[43] K. Yamaguchi, A. C. Berg, L. E. Ortiz, and T. L. Berg, “Who are you with and where are you going?” in *CVPR*. IEEE, 2011, pp. 1345–1352.

[44] Q. Yu, G. Medioni, and I. Cohen, “Multiple target tracking using spatio-temporal markov chain monte carlo data association,” in *CVPR*, 2007, pp. 1–8.