Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2011, Article ID 604647, 13 pages
doi:10.1155/2011/604647

Research Article

Camera Localization in Distributed Networks Using Trajectory Estimation

Nadeem Anjum

Engineering Research Laboratory, Faculty of Engineering and Applied Sciences, Riphah International University, Sector I-14, Hajj Complex, Peshawar Road, Islamabad 44000, Pakistan

Correspondence should be addressed to Nadeem Anjum, nadeem.anjum@riphah.edu.pk

Received 5 April 2011; Revised 1 July 2011; Accepted 27 July 2011

Academic Editor: Mark Liao

This paper presents an algorithm for camera localization using trajectory estimation (*CLUTE*) in a distributed network of nonoverlapping cameras. The algorithm recovers the extrinsic calibration parameters, namely, the relative position and orientation of the camera network on a common ground plane coordinate system. We first model the observed trajectories in each camera's field of view using Kalman filtering, then we use this information to estimate the missing trajectory information in the unobserved areas by fusing the results of a forward and backward linear regression estimation from adjacent cameras. These estimated trajectories are then filtered and used to recover the relative position and orientation of the cameras by analyzing the estimated and observed *exit* and *entry* points of an object in each camera's field of view. The final configuration of the network is established by considering one camera as a reference and by adjusting the remaining cameras with respect to this reference. We demonstrate the algorithm on both simulated and real data and compare the results with state-of-the-art approaches. The experimental results show that the proposed algorithm is more robust to noisy and missing data and in case of camera failure.

1. Introduction

Over the last few decades, ubiquitous use of large-scale camera networks has been ramping up in a wide range of applications such as visual surveillance of mass transportation sites, calamity watching, and traffic control. These camera networks essentially enable monitoring of extensively large areas and hence detection of interesting activities on a larger scale, which is impossible with the use of single cameras (see [1, 2]). Existing activity detection systems primarily perform manual analysis of the data collected by such networks, which is an extremely tedious job; therefore, the development of automated data analysis and summarization tools are essential to accomplish maximum from these camera networks [3].

Calibration of camera networks is the first and foremost important step in the development of such an automated activity summarization system. The calibration defines the correspondence between points in the image plane and points in the 3D space and can be divided into intrinsic and extrinsic calibration (also labeled as localization). The intrinsic calibration establishes the relationship between camera-centric coordinates and images coordinates and can be performed by acquiring an object with known Euclidean structure, while extrinsic calibration defines relationship between scene-centric coordinate system and camera-centric coordinate system. For overlapping camera networks, the estimation of epipolar geometry is a popular choice for extrinsic calibration, where candidate corresponding points are initially extracted from the scene and then a model is learnt that minimizes the images and their reprojects [4]. However, in many real scenarios, cameras do not have overlapping views (Figure 1). Examples of such scenarios are wide area surveillance of underground train stations and/or subways, and mobile ad-hoc networks of low-cost surveillance cameras. The large number of cameras makes it difficult or expensive to record cameras' locations manually or to equip each camera with a GPS unit [5]. One way of addressing this issue is through the novel paradigm of automated nonoverlapping camera calibration algorithms, which enable cameras to determine their positions and orientations after the placement.

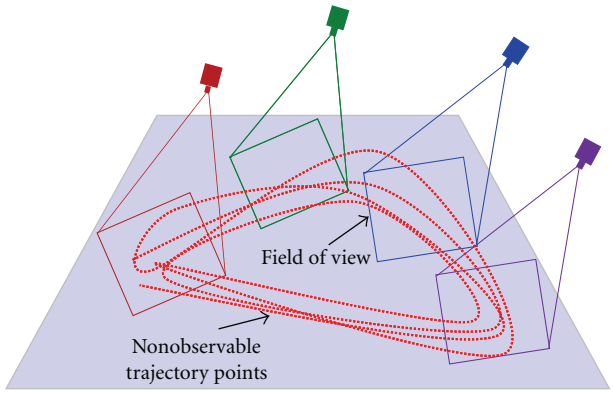


Figure 1: Example of camera network with nonoverlapping fields of view.

In this paper, we present a camera localization algorithm using trajectory estimation (*CLUTE*) for a network of nonoverlapping cameras. The algorithm addresses the problem of recovering the relative position and orientation of multiple cameras whose intrinsic parameters are known. *CLUTE* uses temporal and geometric constraints derived from the available trajectory information to estimate the unobserved trajectory segments, which are then used to position the cameras on the common ground plane. The object motion information also helps in estimating the relative orientation of the cameras. The registration process aligns the cameras with respect to each other in a single camera frame (that of the reference camera). The algorithm is demonstrated on both simulated and real data, and its results are compared with state-of-the-art approaches.

This paper is organized as follows. Section 2 reviews state-of-the-art approaches for camera localization. Section 3 formalizes the problem under consideration. Section 4 provides detailed description of *CLUTE*. Section 5 demonstrates the results and analysis of *CLUTE* on real and simulated data and also compares its performance with two existing techniques. Lastly, Section 6 draws the conclusions.

2. Related Work

Localization is a well-established problem in sensor (camera, audio, RFID, etc.) networks ([6–8]). Algorithms for localization can be categorized into two main classes, namely, fine-grained algorithms (see [9, 10]) and coarse-grained algorithms ([11, 12]). *Fine-grained* methods use timing and/or signal strength for localization. In this class of localization methods, only a few sensors positions are known. These sensors are called *beacons* or *anchors* (see [13–15]). The knowledge of beacons is then propagated across the entire network to find the position of the remaining sensors. The nodes measure the distance to their neighbors whenever possible using hardware ranging techniques such as received signal strength (RSS) and time difference of arrival (TDoA). However, the selection of the anchor nodes is a significant problem and using all anchor nodes does not give the most precise position. Moreover, there is the need to identify criteria for selecting the optimal number of anchor nodes to achieve a more accurate position estimation.

Economic factors and hardware limitations are key motivations for the use of *coarse-grained* localization methods. These methods estimate the proximity of the sensors in a network to an arbitrary reference sensor. Coarse-grained algorithms can further be divided into *nonstatistical* and *statistical* approaches. The first nonstatistical approach is multidimensional scaling (MDS) for localization ([16–18]). MDS arranges the sensors in a lower-dimensional space, whose size depends on the application data. MDS is very accurate in recovering the sensor network configuration when precise distances are available between all sensor pairs. To compute the locations of N cameras on a 2D space using MDS [19], an affinity matrix is constructed based upon pairwise Euclidean distances. Then, the inner product matrix I using the double centering matrix and the affinity matrix is calculated, followed by the computation of the eigenvalues and the eigenvectors of I. After sorting the eigenvalues in descending order, the sensors' locations are calculated using the top p eigenvalues and their corresponding vectors. The major shortcoming of this approach is its dependency on the affinity matrix and the unavailability of a few distances degrades the overall performance significantly.

Another nonstatistical approach is used to calibrate a network of randomly placed cameras with nonoverlapping fields of view using moving scene features in the near and far fields [20]. A strong assumption is made that object motions are deterministic. Distant objects (e.g., stars) enable the recovery of the orientation (rotation) of the cameras, while close objects (e.g., people or cars) enable the recovery of the translation of the cameras up to a scalar multiple. In this approach, the camera parameters are recovered by solving a complex geometry problem, without imposing a probabilistic framework.

Statistical approaches include numerical, motion-based, maximum a posterior (MAP), and velocity extrapolation based approaches. Numerical solutions are iterative methods for network localization, and each iteration contributes to the reduction of the residual errors. Existing approaches are generally variations of the gradient descent or the Gauss-Newton methods. As an example, Taylor et al. [21] use the Newton Raphson method to estimate the network configuration. Although the simplicity of such approaches is a key advantage, they heavily depend on the proper initialization and increment (or decrement) rate to find the global (or local) minimum.

In structure from motion (SFM), the trajectory of a moving camera, and the 3D coordinates of a stationary target are recovered simultaneously from a series of 2D images of a scene. In [22], the focus is on real-time processing of the image data using an extended Kalman filter, whereas the concept of recursive or sequential SFM can be found in [23, 24]. Similar to SFM, simultaneous localization and mapping (SLAM) localizes a moving sensor (robot) and estimates its trajectory using its egomotion and the stationary objects in the scene (see [25, 26]). The performance of SLAM algorithms is affected by noise, as the robots rely on their camera to compute the distance traveled and therefore noisy

Table 1: Summary of the state-of-the-art approaches for sensor localization.

Classes	Approaches	References
Fine-grained localization	Maximum likelihood, trigonometric,	[13, 14, 30],
	Received signal strength, time difference of arrival	[9, 11]
Coarse-grained (nonstatistical localization)	Multidimensional scaling	[16–19]
	Moving scene features	[20]
	Active badge location	[12]
Coarse-grained (statistical localization)	Linear regression and Kalman filter	[31]
	Maximum a posterior probability estimation	[21, 27, 32]
	Simultaneous localization and mapping	[25, 26]
	Structure from motion	[22–24]
	Tracking and camera field of view information	[28]
	Vanishing points and known position	[29]

measurements add up quickly. Environment maps can be helpful in these situations.

Rahimi et al. [27] used the maximum a posterior (*MAP*) framework for simultaneous calibration and tracking. A network of nonoverlapping cameras is localized by using the motion of a target. The *MAP* estimates for the calibration parameters are calculated using the trajectory prior (i.e., the motion model) and the likelihood function, which are constructed from the available observations. The *MAP* approach is highly computationally complex. Furthermore, it is also possible that the solution may place the target inside the field of view of another sensor for which no observations are available at that particular time instance.

Javed et al. [28] use the concept of velocity extrapolation to project the field of view of one camera onto the other. The projection is then used as a tool to find the calibration parameters. However, the approach assumes that people walk in a straight line in the unobserved regions. Finally, Junejo et al. [29] propose an approach in which vanishing points are used to find the relative orientation of the cameras whose positions are already known.

A summary of the state-of-the-art approaches for sensor localization is presented in Table 1.

3. Problem Formulation

Suppose we have a network of N non-overlapping cameras $\psi = \{C^1, C^2, \ldots, C^N\}$, similar to [33]. Let a trajectory $\wp^i$ within C^i be represented as $\wp^i = \{(x^i(j), y^i(j)) : 0 < j < M_i;\ i = 1, \ldots, N\}$, where (x^i, y^i) is the estimated position of the target in the image plane and M_i is the number of target observations from camera C^i.

Table 2: Performance comparison between *CLUTE*, *MAP*, and *MDS* on the three datasets (4-camera network).

	ID	CLUTE		MAP		MDS	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.26	2.36	0.31	5.94	0.57	11.56
	3	0.33	1.60	0.39	2.04	0.41	8.41
	4	0.12	1.18	0.11	2.59	0.27	3.72
	Average	**0.18**	**1.29**	**0.20**	**2.64**	**0.31**	**5.92**
Exp 2	2	0.15	4.36	0.19	1.52	0.23	0.96
	3	0.23	5.36	0.34	6.38	0.72	8.76
	4	0.15	4.65	0.17	8.08	0.23	15.99
	Average	**0.13**	**3.59**	**0.18**	**4.00**	**0.30**	**6.43**
Exp 3	2	0.20	6.42	0.28	11.94	0.39	21.26
	3	0.56	13.06	0.78	18.24	0.89	23.42
	4	0.19	1.98	0.23	4.00	0.57	10.19
	Average	**0.24**	**5.37**	**0.32**	**8.55**	**0.46**	**13.72**

Furthermore, let each observation be generated by a motion model as

$$\begin{bmatrix} x^i(j+1) \\ x^i_v(j+1) \\ y^i(j+1) \\ y^i_v(j+1) \end{bmatrix} = \begin{bmatrix} 1 & a_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & a_y \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^i(j) \\ x^i_v(j) \\ y^i(j) \\ y^i_v(j) \end{bmatrix} + \begin{bmatrix} v_x(j) \\ v_{v,x}(j) \\ v_y(j) \\ v_{v,y}(j) \end{bmatrix}, \quad (1)$$

where (x^i_v, y^i_v) is the velocity of the object. In addition, (a_x, a_y) may change over period of time. Moreover, $\mathbf{v}^j(\mathcal{N}(0, \Sigma_{v(j)}))$ is modeling additive noise with covariance $\Sigma_{v(j)} = \text{diag}([1e^{-3}, 1e^{-3}, 1e^{-3}, 1e^{-3}])$.

Let each camera C^i provides a vertical top-down view of the scene (i.e., its optical axis is perpendicular to the ground plane or the trajectories are preprocessed using a homography transformation [34]). Under this assumption, the number of parameters for the localization of each camera C^i is reduced to two, namely, the camera *position*, $P^i = (p^i_x, p^i_y)$, and the *rotation* angle, ϕ^i, expressed as the relative angle between the camera C^i and the horizontal axis (Figure 2). To summarize, the unknown parameters Θ^i for camera C^i are

$$\Theta^i = \left[p^i_x, p^i_y, \phi^i\right]. \quad (2)$$

If C^i observes the object at a particular time instant t and after $t + \tau$ time intervals the object enters into $C^{i+\eta}$ with $C^i \neq C^{i+\eta}$, then it can be visualized as $C^{i+\eta}$ is viewing the object from the $\phi^{i,i+\eta}\wp^{i,i+\eta}$ position, where $\phi^{i,i+\eta}$ and $\wp^{i,i+\eta}$ are the rotation matrix and the translation vector.

The camera *localization* process estimates $(\wp^{i,i+\eta}, \phi^{i,i+\eta})$ such that the configuration estimation error ε becomes minimum, that is,

$$\begin{aligned} &\left(\phi^{i,i+\eta} + \wp^{i,i+\eta}\right)\left(x^{i+\eta}(t+\tau), y^{i+\eta}(t+\tau)\right) \\ &\quad - \left(\hat{x}^{(i,i+\eta)}(t+\tau), \hat{y}^{(i,i+\eta)}(t+\tau)\right) = \varepsilon \longrightarrow 0, \end{aligned} \quad (3)$$

where $(\hat{x}^{(i,i+\eta)}, \hat{y}^{(i,i+\eta)})$ is the projected estimate of the object's position from C^i at t to $C^{i+\eta}$ at $t + \tau$.

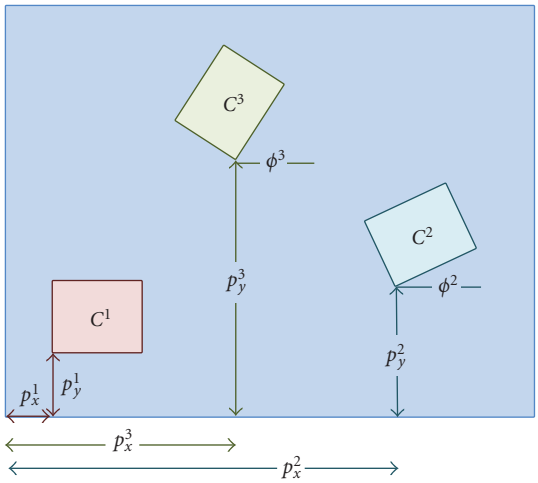


Figure 2: Schematic representation of a scene observed with nonoverlapping cameras (C^i). (p_x^i, p_y^i, ϕ^i) represent the unknown camera location and rotation to be estimated.

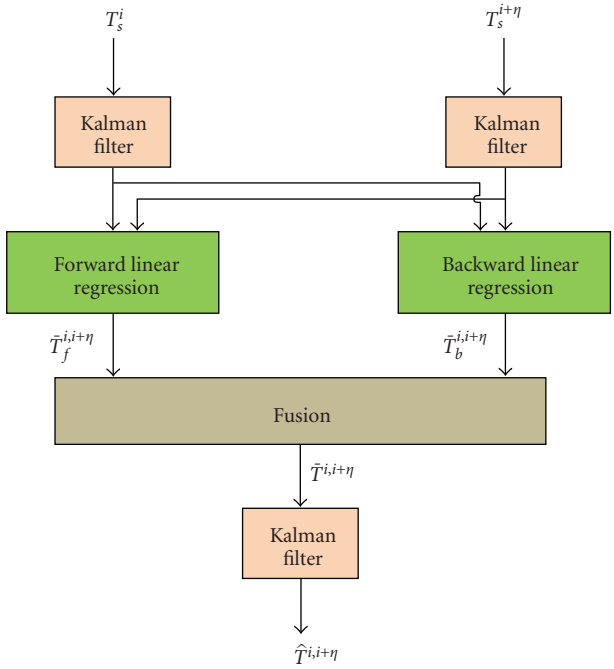


Figure 3: Trajectory estimation process for the unobserved region between camera C^i and $C^{i+\eta}$.

4. Proposed Approach

Let an object move in the environment and be tracked by each camera in the network. To find the rotation matrix, $\phi^{i,i+\eta}$, and the translation vector, $\wp^{i,i+\eta}$, between adjacent cameras, we propose a two-step *iterative* process. Two consecutive batches of measurements from two adjacent cameras form one iteration and the adjacency between cameras are defined by the motion of the object itself. The first step calculates $\wp^{i,i+\eta}$ by estimating the missing trajectory between pairs of adjacent cameras. In the second step, $\phi^{i,i+\eta}$ is calculated by utilizing the trajectory information and the object's *exit* and *entry* points of the fields of view of adjacent cameras. The details of each step are given below.

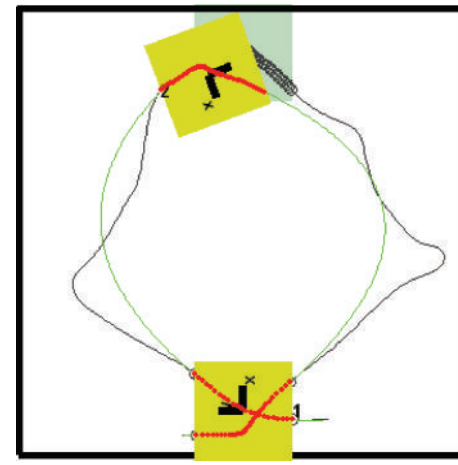


Figure 4: Example of sharp turns modeled with smooth curves in the unobserved regions (*gray*: original trajectory, *green*: estimated trajectory, *gray square*: original camera location (field of view), and *yellow square*: estimated camera location (field of view)).

4.1. Trajectory Estimation in Unobserved Areas. Figure 3 shows the flow diagram of the trajectory estimation process. Unlike [31], each trajectory segment $\wp_s^i$ from each camera C^i is smoothed for intercamera trajectory estimation:

$$\wp_s^i = S\left(\wp^i\right), \tag{4}$$

where S is a smoothing function and $i = 1, \ldots, N$. If C^i and $C^{i+\eta}$ are the two adjacent cameras, then $\hat{\wp}^{i,i+\eta}$ is the estimated trajectory between the camera pair:

$$\hat{\wp}^{i,i+\eta} = H\left(G\left(\wp_s^i, \wp_s^{i+\eta}\right)\right), \tag{5}$$

where G and H are *nonparametric* and a *parametric* functions, respectively, which extrapolate a smoothed trajectory in the unobserved region.

We use the *Kalman filter* (see [35, 36]) as a parametric function (H). The approach is inspired by speech recognition, where the Kalman filter [37] has extensively been used to estimate intervals with missing observations [38]. However, the Kalman filter's innovation signal distorts significantly in the absence of target information. To overcome this problem, we employ *linear regression estimation* as a nonparametric function, as, in the unobserved regions, Kalman filter and the linear regression models exhibit a similar behavior (see the appendix).

Within each camera's field of view, the Kalman filtering is applied to obtain the initialization of the parameters. When the target enters an unobserved region, the linear regression model replaces the Kalman filter to estimate the target position. In order to improve the estimation, we use both *forward* and *backward* motion models. The process

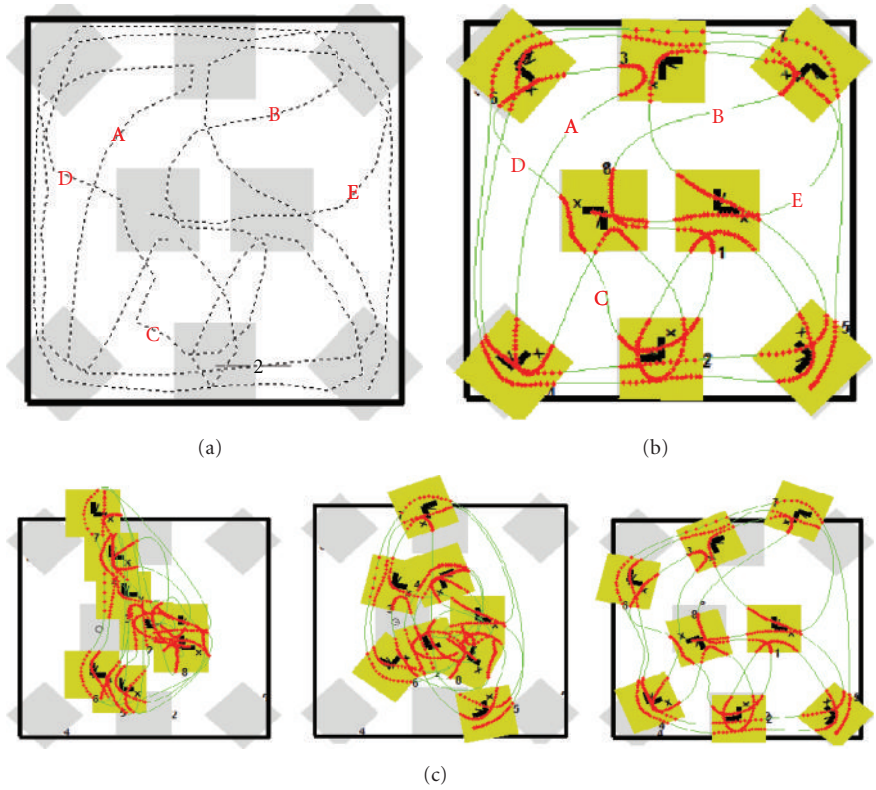


Figure 5: An example of localization of a camera network: (a) a network of cameras along with the original object trajectory (dash-line); (a–d) few nonlinear turns that object take in unobserved regions, (b) estimated trajectory and final localization results with the proposed algorithm, and (c) outputs at intermediate iterations.

consists of trajectory estimation in both the forward ($\overline{\wp}_f^{i,i+\eta}$) and the backward ($\overline{\wp}_b^{i,i+\eta}$) direction between C^i and $C^{i+\eta}$.

The final estimation $\overline{\wp}^{i,i+\eta}$ is obtained as weighted average over $\overline{\wp}_f^{i,i+\eta}$ and $\overline{\wp}_b^{i,i+\eta}$, for each segment k:

$$\overline{\wp}^{i,i+\eta}(k) = (1-\alpha(k))\overline{\wp}_f^{i,i+\eta} + \alpha(k)\overline{\wp}_b^{i,i+\eta}, \tag{6}$$

where $\alpha(k) = k/J$ and $k = 1,\ldots,J$ are J segments of the estimated trajectory. The forward estimation results are given weights that are decreasing with the distance from the border of the camera's fields. The backward estimation results are given higher weights when the object gets closer to the next camera's field of view. The underlying assumption for this approach is that both trajectories contribute to the construction of the estimated trajectory [39]. The linear regression model expands over the uncertainty volume of the region so that when the target is once again visible in a camera, it can immediately reinitialize the Kalman filter. For simplicity, sharp turns are modeled as smooth curves (Figure 4). The process terminates if the difference between five consecutive iterations is smaller than a threshold or when all the available data are utilized.

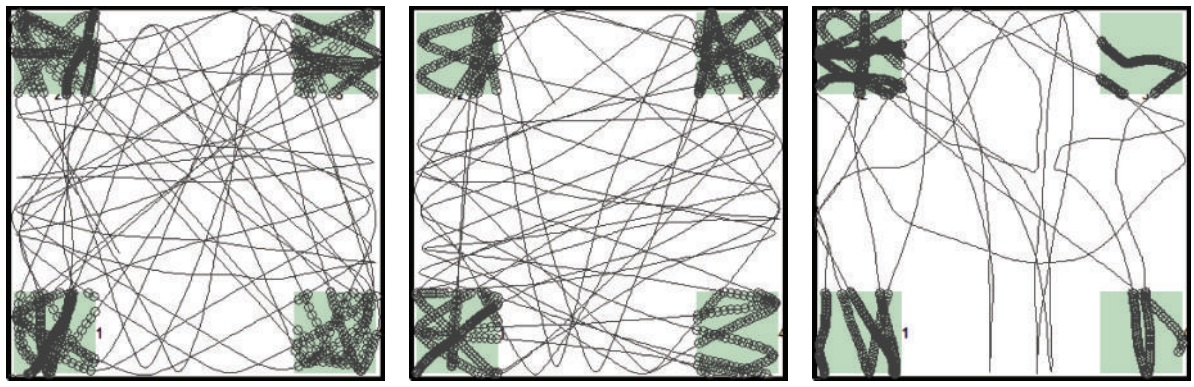
4.2. Orientation Estimation. The relative angle $\phi^{i,i+\eta}$ between two adjacent cameras C^i and $C^{i+\eta}$ is computed by calculating the angle between the observed object position $(x^{i+\eta}, y^{i+\eta})$ in camera $C^{i+\eta}$ and the corresponding estimated object position $(\hat{x}^{(i,i+\eta)}, \hat{y}^{(i,i+\eta)})$ in the same camera by extrapolating the trajectory from C^i:

$$\phi^{i,i+\eta} = \cos^{-1}\left(\frac{(x^{i+\eta}, y^{i+\eta}) \cdot \left(\hat{x}^{(i,i+\eta)}, \hat{y}^{(i,i+\eta)}\right)}{|(x^{i+\eta}, y^{i+\eta})|\,|(\hat{x}^{(i,i+\eta)}, \hat{y}^{(i,i+\eta)})|}\right). \tag{7}$$

Once $\phi^{i,i+\eta}$ is computed for all pairs of adjacent cameras, the final configuration is obtained by rearranging all the cameras C^i $(i = 2,\ldots,N)$ with respect to the reference sensor C^1. An example of the complete localization process of the proposed algorithm in shown is Figure 5. The figure shows the original trajectory segments (linear and nonlinear) as well as reconstructed trajectory segments in unobserved regions. Furthermore, intermediate and final localization results of the proposed algorithm are also provided in the same figure to demonstrate the process of localization over iterations.

5. Experimental Results and Analysis

In this section, we compare the proposed approach (*CLUTE*) with the *MDS* (see [16, 17]) and the *MAP* approaches (see [27, 32]) on both simulated and real data. A 4-camera and a 8-camera network, are tested with simulated data, and a 4-camera network is tested with real data. The performance of the algorithms is evaluated based on the translation



(a) (b) (c)

Figure 6: Three trajectory datasets from a nonoverlapping 4-camera network. *Shaded squares* indicate the field of view of each camera, *circles* denotes the observed object locations, and *lines* show the portions of the trajectories unobserved by the cameras.

and rotation errors in the localization using ground-truth information. The translation error is calculated as Euclidean distance between the true camera position and the estimated camera position. The rotation error is calculated as absolute difference between the true camera orientation (the angle with respect to the reference camera) and the estimated camera orientation. The details about the dataset used and the experimental results are given below.

5.1. 4-Camera Network. In this setup, we first simulate a network of 4 cameras and then test the algorithms on a real camera network. Let an object travel across the network and generate the trajectory. Each camera observes the object within its field of view (in Figure 6, the observations within each field of view are shown with circles). The lines outside the squares are the unobserved track points of the object. We analyze the results of three experiments using the same network configuration, but with three different moving objects. The *CLUTE* results are shown in Figure 8. The translational error at each iteration is shown in Figure 9. The performance of *CLUTE* is evaluated and compared with *MAP* and *MDS* on the original, noisy, and subsampled data. Also, the algorithms are compared in the case of camera failure.

Figure 10 compares the results of *CLUTE*, *MAP*, and *MDS* along with the true positions of the cameras. The visual inspection of the results show that in all three experiments, the performance of the statistical approaches is better than that of nonstatistical approaches (especially for C^2 in the *experiment 1* dataset, for C^3 in the *experiment 2* dataset, and for C^3 and C^4 in the *experiment 3* dataset). The detailed experimental results are shown in Table 2. In the first experiment, the average translation error for *CLUTE* is 0.18 units or 4.5% of the environment area. Furthermore, the average rotation error for *CLUTE* is 1.29°. In comparison with the other two approaches, on average *CLUTE* is more accurate in locating the network. The average translation error for *MAP* and *MDS* is poorer by 0.50% and 3.25%, respectively. Likewise, the average rotation errors for *MAP* and *MDS* are worse by 1.35° and 4.63°, respectively. In the second experiment, the average translation error for *CLUTE* is 0.13 units or 3.25% of the environment area, which is better by 1.25% and 4.25% compared to *MAP* and *MDS*, respectively. Moreover, the average rotation error estimated by *CLUTE* is better by 0.41° and 2.84° than that of *MAP* and *MDS*, respectively. In the third experiment, the average translation error for *CLUTE* is 0.24 units, which are 6% of the environment area. Also, the average rotation error for *CLUTE* is 5.37°. In comparison, the average translation error for *MAP* and *MDS* is 0.32 units and 0.46 units, respectively. Similarly, the average rotation angles for *MAP* and *MDS* approaches are 8.55° and 13.72°. In summary, these experiments show that the performance of *CLUTE* is better by an average of 1.25% for translation and of 1.65° for rotation with respect with *MAP*. In comparison to *MDS*, the performance of *CLUTE* is better by 4.5% and 5.28° for estimating translation and rotation, respectively. In addition to this, the error variance for *CLUTE* is smaller than that for *MAP* and *MDS*. Furthermore, it is noticeable that the estimation results for the statistical approaches (i.e., *CLUTE* and *MAP*) are better than that of the nonstatistical approach (i.e., *MDS*). In general, the main limitation of *MDS* is that it is based on a single parameter (in this case, the shortest Euclidean distance) with an *appropriate* value. Therefore, the parameter calculation is accurate and the approach performs satisfactory only if given enough information.

We also analyzed the robustness of *CLUTE* to reduced sampled trajectory (missing data) and noisy trajectory and in case of camera failure. As *MDS* is poor in estimating the localization, we compare the robustness of *CLUTE* and *MAP* only (Table 3).

For the missing data test, the object's trajectory is downsampled by 2 and by 3 for all the three experiments. The results for downsampling by 2 show that on average *CLUTE* does not suffer for translation estimation in *experiment 1* and *experiment 2* and improves in *experiment 3*. Similarly for the rotation (especially for *experiment 2* and *experiment 3*), a closer look at the results shows that in *experiment 3* the estimation result is quite poor for C^3 on the original data compared to the other cameras. Figure 8 shows that

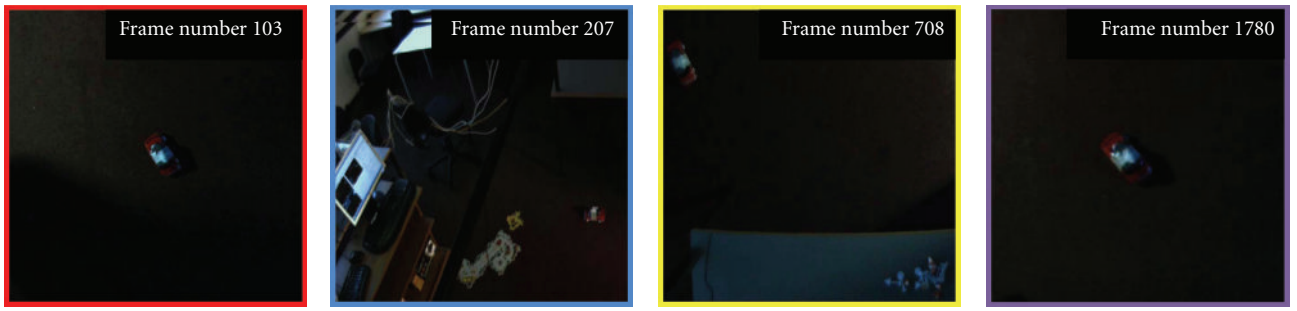


Figure 7: Sequence of images extracted from real dataset consisting of four nonoverlapping cameras. Frame numbers show the motion of the object across the network.

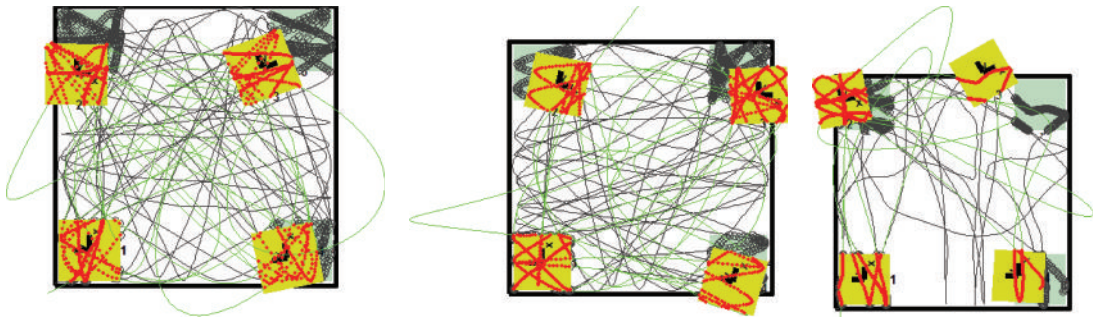


Figure 8: *CLUTE* localization results for the 4-camera network. *Yellow squares* are the estimated localization and *green lines* indicate the e6stimated trajectory in the unobserved areas. (a) Experiment 1 dataset; (b) experiment 2 dataset; (c) experiment 3 dataset.

in most cases the object takes very sharp turns before entering into this camera. These sharp turns cause the degradation of the result, while downsampling reduces the sharpness of the turns and therefore simplifies the modeling. Compared to *MAP*, the average translation error over the three experiments for *CLUTE* is better by 2.75%. Similarly, the rotation estimation is better by 2.58° in favor of *CLUTE*. Also the performance of *MAP* degrades considerably due to downsampling. The reason for this behavior is due to the fact that in *MAP* the track points provide the likelihood function over the trajectories and the camera parameters, and therefore the posterior probability depends directly upon the availability of *enough* reliable object measurements. Downsampling reduces the likelihood probability and hence degrades the overall results. On the other hand, in *CLUTE*, the trajectories are estimated by interpolation of the track points over time. For this reason, as long as the available object observations maintain the shape of the trajectory, the calibration will be performed accurately. For further downsampling by 3, the average translation error over the three experiments for *CLUTE* is still 4.25% better than that of *MAP*. Similarly, the rotation estimation is better by 3.35° for *CLUTE*. Also when downsampled from 2 to 3, the degradation for the position estimation is just 0.02 units for *CLUTE* and 0.08 units for *MAP*. To summarize, *CLUTE* is more robust to reduced sampling rates compared to *MAP*.

In order to analyze the performance of *CLUTE* with noisy observations, we introduced a 5% Gaussian noise in the measurements with variance equal to the 15% of the camera's field of view, for all three datasets. For both *CLUTE* and *MAP*, the results degrade substantially for the position estimation (especially C^3, C^4 in *experiment 1* and C^2, C^4 in *experiment 3*) and orientation (especially C^2 in *experiment 3*). This is due to the fact that the noise not only degrades the quality of the observations (essential for *MAP*), but also changes the shape of the trajectory (essential for *CLUTE*), which are necessary for accurate results from both approaches. However, it is noticed that on average the results for *CLUTE* are degraded by 8% for the translation and 20.02° for the rotation, whereas the degradation for MAP is 8.5% and 20.20°.

In order to analyze the localization performance in the case of camera failure, we removed one camera's observations from the available datasets (1-camera failure case). The complete results for the three experiments are shown in Table 4. On average, the results for *CLUTE* are degraded by 5% for the translation and 22.47° for the rotation, whereas the degradation for *MAP* is 10.25% and 24.81°. The results show that *CLUTE* is more robust in case of camera failure.

In the final experiment for the 4-camera network, we use real data captured indoor (Figure 7). The fields of view of the cameras are squares whose sides are 1.5 meters, and the cameras are between 3 and 4 meters apart. A toy car is

Table 3: Performance comparison between *CLUTE*, *MAP*, and *MDS* on the three datasets for missing and noisy data (4-camera network).

	ID	Downsampled by 2				Downsampled by 3				Noisy			
		CLUTE		MAP		CLUTE		MAP		CLUTE		MAP	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.25	2.48	0.27	5.64	0.22	1.33	0.31	3.43	0.15	1.87	0.15	4.13
	3	0.34	1.70	0.42	2.32	0.32	0.73	0.44	3.39	0.63	10.99	0.63	12.88
	4	0.12	1.11	0.21	4.08	0.14	2.15	0.43	8.70	0.78	13.06	0.78	15.50
	Average	**0.18**	**1.32**	**0.22**	**3.01**	**0.17**	**1.05**	**0.30**	**3.88**	**0.39**	**6.48**	**0.39**	**8.13**
Exp 2	2	0.14	4.27	0.21	4.47	0.15	4.54	0.13	5.96	0.37	11.12	0.39	13.45
	3	0.22	5.02	0.35	6.27	0.24	5.70	0.34	7.88	0.33	7.55	0.43	9.88
	4	0.15	4.45	0.16	7.94	0.16	4.86	0.43	5.41	0.58	19.14	0.71	17.14
	Average	**0.13**	**3.44**	**0.18**	**4.67**	**0.14**	**3.78**	**0.22**	**4.81**	**0.32**	**9.45**	**0.38**	**10.12**
Exp 3	2	0.09	2.95	0.23	13.36	0.03	1.04	0.31	16.54	0.43	168.10	0.57	175.21
	3	0.34	7.16	0.57	12.92	0.35	2.46	0.65	7.63	0.92	4.77	1.32	9.32
	4	0.39	5.61	0.94	8.80	0.64	14.91	1.30	18.96	1.86	41.36	1.90	45.56
	Average	**0.20**	**3.93**	**0.44**	**8.77**	**0.26**	**4.60**	**0.56**	**10.78**	**0.80**	**53.56**	**0.95**	**57.52**

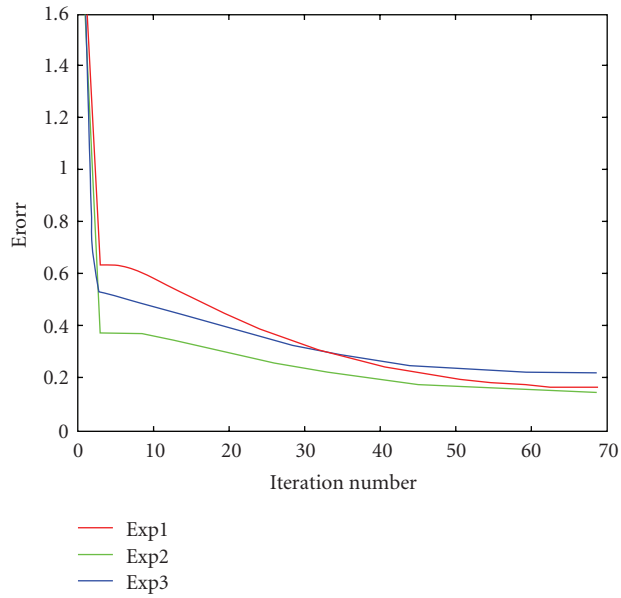


Figure 9: Evolution of the *CLUTE* translation error at each iteration in the three experiments.

moving at varying velocities traces a long trajectory across the cameras. Based on the ground-truth segmentation of the moving object in the field of view of every camera, we compute the relative distance and orientation between the cameras based on trajectory estimation. Figure 11 shows the results obtained with *CLUTE*, and Table 5 compares the results of the three approaches on the real data. For *CLUTE*, the sensors were on average misplaced by 70 cm from the locations measured by hand, with an average orientation error of 10.33°. For the orientation estimation, *CLUTE* performs on average better than *MAP* by 0.33° and better than *MDS* by 5°. For the position estimation, the error of

Table 4: 4-camera network: *CLUTE* and *MAP* comparison for 1-camera failure.

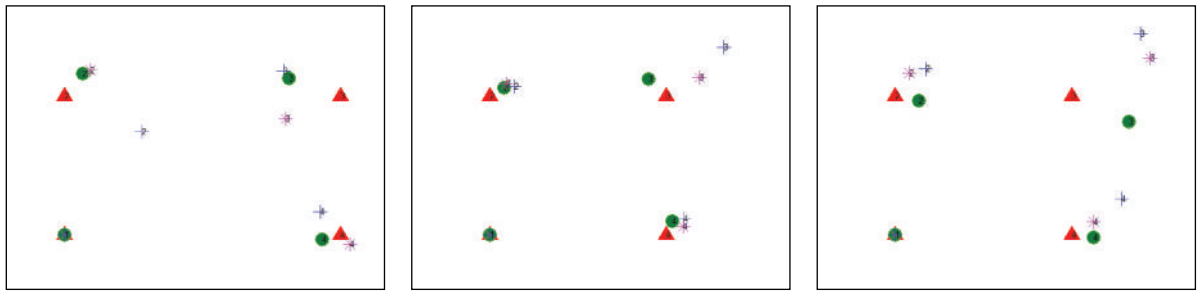
	ID	CLUTE		MAP	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.57	13.32	0.62	16.05
	3	1.06	24.25	1.62	30.53
	Average	**0.54**	**12.52**	**0.75**	**15.53**
Exp 2	2	0.12	3.84	0.25	6.53
	3	0.28	5.80	0.83	12.49
	Average	**0.13**	**3.21**	**0.36**	**6.34**
Exp 3	2	0.40	166.84	0.53	178.73
	3	1.02	18.86	1.92	24.52
	Average	**0.47**	**61.90**	**0.82**	**67.75**

Table 5: The rotation (in degrees) and translation (in meters) errors of *CLUTE* for real data.

ID	CLUTE		MAP		MDS	
	ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r
2	0.80	10.00	0.65	10.50	1.00	12.00
3	0.69	8.00	0.73	8.63	1.12	18.00
4	0.61	13.00	0.59	12.85	1.33	16.00
Average	**0.70**	**10.33**	**0.66**	**10.66**	**1.15**	**15.33**

CLUTE is 0.45 meter smaller than that of *MDS* and poorer than that of *MAP* by 0.04 meters only.

5.2. 8-Camera Network. Three datasets have been generated by simulating three different moving objects in a network of 8 cameras with nonoverlapping fields of view. The results obtained with *CLUTE* on these datasets are shown in Figure 12. Table 6 contains the results obtained in each experiment. *CLUTE* outperformed *MDS* as its average translation and rotation errors are 13.75% and 3.63° lower



(a) (b) (c)

Figure 10: Performance comparison for camera localization using *CLUTE* (*green circle*), *MAP* (*magenta star*), and *MDS* (*blue cross*). The *red triangles* denote the ground-truth position of the cameras. (a) Experiment 1; (b) Experiment 2; (c) Experiment 3.

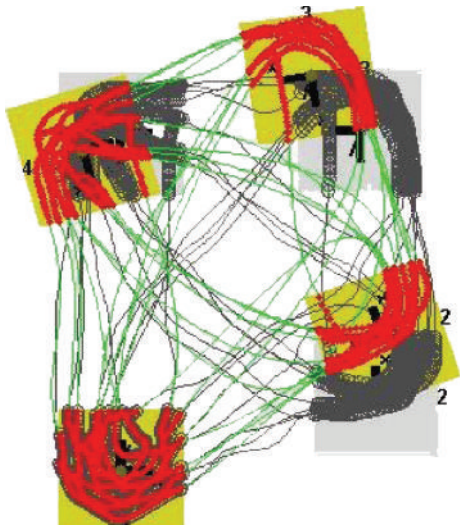


Figure 11: Network localization results using *CLUTE* on real data. (*black*: original trajectory, *green*: estimated object trajectory in the unobserved areas, *gray square*: original camera location (field of view), and*yellow square*: estimated camera location (field of view).

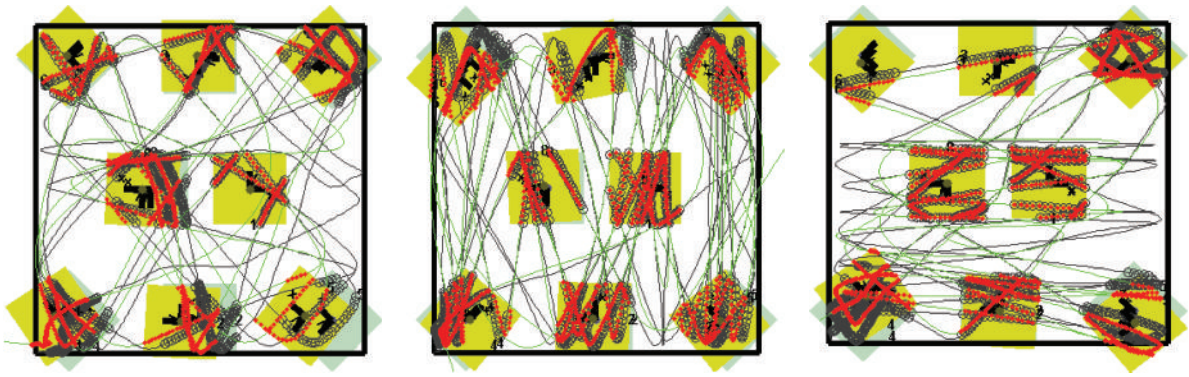
Table 6: Performance comparison between *CLUTE*, *MAP*, and *MDS* on the three datasets (8-camera network).

	ID	CLUTE		MAP		MDS	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.12	0.89	0.12	0.86	1.00	10.54
	3	0.02	0.61	0.02	0.63	1.05	3.48
	4	0.08	1.56	0.08	2.12	0.45	1.82
	5	0.16	1.54	0.16	1.54	1.84	17.29
	6	0.02	0.40	0.02	0.42	2.04	7.13
	7	0.04	0.77	0.04	0.80	0.04	0.67
	8	0.52	2.01	0.05	2.00	0.50	2.52
	Average	**0.14**	**1.11**	**0.07**	**1.20**	**0.99**	**6.21**
Exp 2	2	0.14	0.85	0.24	1.90	0.98	11.01
	3	0.28	1.38	0.28	1.94	0.98	3.47
	4	0.14	3.00	0.26	4.12	0.73	2.92
	5	0.20	0.07	0.22	0.54	1.23	17.29
	6	0.36	1.49	0.35	1.42	1.01	2.13
	7	0.22	1.09	0.22	1.86	0.99	0.67
	8	0.17	1.25	0.14	5.20	0.55	2.52
	Average	**0.19**	**1.14**	**0.22**	**2.12**	**0.81**	**5.00**
Exp 3	2	0.00	0.07	0.00	0.93	0.00	3.07
	3	0.03	0.13	0.03	1.89	0.10	1.87
	4	0.09	8.16	0.10	9.23	0.95	10.16
	5	0.07	2.32	0.06	2.13	0.92	0.68
	6	0.04	0.12	0.06	0.83	0.17	3.12
	7	0.00	0.04	0.03	0.62	0.20	8.96
	8	0.00	0.02	0.04	9.58	0.22	4.69
	Average	**0.03**	**1.36**	**0.04**	**3.15**	**0.32**	**4.07**

than those of *MDS*. Likewise, *CLUTE* outperformed *MAP* in estimating the orientation, as the average rotation error over the three datasets is lower by 0.90% and in estimating the translation *MAP* is better by 0.16%.

We further investigate the robustness of the methods for missing data, noisy trajectories (Table 7), and camera failure. For downsampling by 2, the average translation error over the three experiments for *CLUTE* is 0.03 unit-size better than *MAP*. Similarly, the rotation estimation is better by 2.19°. Also, for downsampling by 3, the average translation error using *CLUTE* over the three experiments is 1.5% better than using *MAP*. Furthermore, the rotation estimation is better by 3.69 for *CLUTE*. Also, when downsampled from 2 to 3, the degradation for the position estimation is 40% better for *CLUTE*, compared to *MAP*. For trajectories contaminated with a 5% Gaussian noise with variance equal to the 15% of the camera's field of view, the average translation error for *CLUTE* over the three datasets is 2.5% better than that of *MAP* and the rotation error is also better by 2.48°.

For the camera failure, we simulated the one, three, and five cameras failure situations by ignoring the measurements coming from one, three, and five cameras, respectively, and



(a) (b) (c)

Figure 12: *CLUTE* localization results with a 8-camera network: (a) experiment 1 dataset; (b) experiment 2 dataset, (c) experiment 3 dataset.

Table 7: Performance comparison between *CLUTE*, *MAP*, and *MDS* on the three datasets for missing and noisy data (8-camera network).

		Downsampled by 2				Downsampled by 3				Noisy			
	ID	CLUTE		MAP		CLUTE		MAP		CLUTE		MAP	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.23	1.01	0.28	2.59	0.28	1.39	0.33	5.05	0.66	33.69	0.96	42.69
	3	0.05	0.63	0.07	0.87	0.06	1.72	0.10	2.93	0.64	18.65	0.74	10.65
	4	0.18	2.87	0.20	3.26	0.21	7.40	0.28	9.36	0.88	17.25	1.68	28.25
	5	0.33	1.46	0.37	1.40	0.40	3.81	0.76	3.74	0.42	15.37	0.82	22.37
	6	0.04	1.20	0.06	0.87	0.07	0.45	0.09	2.86	1.41	58.19	1.61	62.19
	7	0.07	0.72	0.12	1.22	0.05	1.38	0.20	4.25	0.84	23.83	1.34	27.83
	8	0.11	1.96	0.16	2.09	0.12	5.75	0.24	5.18	0.42	15.45	0.52	17.45
	Average	**0.14**	**1.41**	**0.18**	**1.76**	**0.17**	**3.13**	**0.29**	**4.77**	**0.77**	**26.06**	**1.10**	**30.20**
Exp 2	2	0.02	0.78	0.03	3.61	0.02	0.91	0.09	5.91	0.03	0.30	0.04	7.61
	3	0.08	1.47	0.06	0.59	0.08	1.48	0.09	7.52	0.02	0.53	0.05	3.74
	4	0.03	4.38	0.05	6.62	0.05	4.99	0.10	11.01	0.06	1.22	0.08	4.24
	5	0.04	0.06	0.09	5.48	0.04	0.10	0.11	5.62	0.21	0.52	0.23	2.92
	6	0.14	1.68	0.17	5.68	0.13	1.66	0.14	8.85	0.11	0.41	0.13	2.07
	7	0.05	1.06	0.06	3.99	0.06	1.24	0.09	6.59	0.19	3.62	0.20	5.37
	8	0.03	1.25	0.04	4.25	0.03	1.30	0.07	7.31	0.02	0.25	0.06	2.40
	Average	**0.05**	**1.34**	**0.06**	**3.78**	**0.05**	**1.46**	**0.09**	**6.60**	**0.08**	**0.86**	**0.10**	**3.54**
Exp 3	2	0.01	0.30	0.03	2.52	0.01	0.67	0.04	3.39	0.21	0.27	0.28	3.12
	3	0.03	0.12	0.09	3.68	0.04	0.03	0.06	2.89	0.65	6.94	0.86	8.31
	4	0.11	9.13	0.07	11.31	0.10	8.60	0.10	13.55	0.60	22.82	0.93	28.24
	5	0.07	2.31	0.07	2.14	0.06	1.81	0.17	2.00	0.93	28.82	0.99	30.11
	6	0.05	0.32	0.17	5.24	0.05	0.14	0.14	6.69	1.28	40.12	1.32	49.20
	7	0.04	0.05	0.09	4.97	0.03	0.45	0.08	3.53	1.30	30.65	1.33	18.32
	8	0.00	0.01	0.09	12.98	0.00	0.02	0.01	15.78	0.07	3.02	0.08	4.33
	Average	**0.04**	**1.53**	**0.08**	**5.36**	**0.04**	**1.47**	**0.08**	**5.98**	**0.63**	**16.58**	**0.72**	**17.70**

by evaluating the localization accuracy of the remaining sensors. For one camera failure (Table 8), the average translation error (taken over three datasets) for *CLUTE* is 0.16 unit, which is better by 0.25 units compared to *MAP*. The rotation error for *CLUTE* is 6.81° better than *MAP*. Likewise, the translation error for *CLUTE* in case of three (Table 9) and five (Table 10) cameras failing is 12% of the environment size. The translation results obtained by *CLUTE* are better than *MAP* by more than 5% in both cases. Also for the rotation error, the performance of *CLUTE* is better by 2.3° and 9.52° for the three and five cameras missing cases, respectively.

6. Conclusions

We proposed an algorithm to recover the network configuration of a set of cameras with disjoint views. The algorithm finds the position and orientation of each camera on a common ground plane and consists of two main steps: the

Table 8: 8-camera network: comparison between *CLUTE* and *MAP* in case of the failure of 1 camera.

	ID	CLUTE		MAP	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.20	3.12	1.00	9.23
	3	0.06	1.43	1.60	11.23
	4	0.18	14.39	0.83	24.42
	5	0.21	1.61	0.62	14.98
	6	0.08	1.96	0.76	19.22
	7	0.12	2.77	1.12	21.77
	Average	**0.14**	**4.21**	**0.99**	**16.81**
Exp 2	2	0.02	0.71	0.06	3.03
	3	0.09	1.55	0.10	3.89
	4	0.08	4.95	0.13	7.45
	5	0.04	0.07	0.11	2.39
	6	0.18	1.54	0.21	3.88
	7	0.05	1.09	0.08	3.55
	Average	**0.07**	**1.41**	**0.10**	**3.46**
Exp 3	2	0.36	20.39	0.39	21.00
	3	0.03	0.04	0.03	12.31
	4	0.93	29.65	0.96	19.43
	5	0.61	0.35	0.59	20.19
	6	0.02	0.57	0.02	29.05
	7	0.03	0.48	0.03	2.54
	Average	**0.28**	**7.35**	**0.29**	**14.93**

Table 9: 8-camera network: comparison between *CLUTE* and *MAP* in case of the failure of 3 cameras.

	ID	CLUTE		MAP	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.20	2.74	0.76	6.67
	3	0.07	1.67	1.47	5.76
	4	0.13	26.80	0.38	42.81
	5	0.21	1.90	0.87	23.82
	Average	**0.15**	**8.28**	**0.87**	**19.77**
Exp 2	2	0.01	23.14	0.32	3.51
	3	0.03	25.16	0.17	1.99
	4	3.13	30.28	3.44	45.34
	5	0.06	2.33	0.49	7.98
	Average	**0.64**	**16.18**	**0.89**	**11.76**
Exp 3	2	0.40	23.14	0.52	26.25
	3	1.41	25.16	1.31	22.27
	4	0.98	30.28	1.32	38.82
	5	0.60	2.33	1.21	4.18
	Average	**0.68**	**16.18**	**0.87**	**18.30**

estimation of the unobserved trajectory in the regions not covered by the cameras' fields of view and the estimation of the relative orientations of the cameras. Kalman filtering and linear regression are used for the estimation of the trajectory. Forward and backward estimations are used to increase the reliability of the results. The relative orientation of the cameras is obtained by using the *exit* and *entry* point information in each camera's fields of view.

Table 10: 8-camera network: comparison between *CLUTE* and *MAP* in case of the failure of 5 cameras.

	ID	CLUTE		MAP	
		ϵ_t	ϵ_r	ϵ_t	ϵ_r
Exp 1	2	0.65	8.58	0.85	42.51
	3	1.55	2.63	1.63	24.12
	Average	**1.10**	**5.61**	**1.24**	**33.32**
Exp 2	2	0.01	0.35	0.58	4.51
	3	0.01	0.09	0.57	12.51
	Average	**0.01**	**0.14**	**0.38**	**5.67**
Exp 3	2	0.55	30.31	0.75	45.90
	3	1.39	26.42	1.69	24.57
	Average	**0.64**	**18.91**	**0.81**	**23.49**

We have compared the performance of the proposed approach with a statistical approach (*MAP*) and with a nonstatistical approach (*MDS*) on both simulated and real data. The experimental results show that *CLUTE* is more accurate in localizing the network compared to these state-of-the-art approaches. Also, the proposed approach is more robust for missing and noisy data and performs better in case of failure of one or more cameras.

Our current work includes further improvements of the trajectory estimation in unobserved areas by tracking, when available, the audio information captured with stereo microphones coupled with each camera [40].

Appendix

Relationship between Kalman Filter and Linear Regression

To show that the Kalman filter and linear regression exhibit similar behaviors in the unobserved regions, let us define the object state X at time t as $X(t) = [x, \dot{x}, y, \dot{y}]$, where (x, y) is the object position and $(\dot{x}, \dot{y})$ is the object velocity. If A is the model that transforms the object state at time t to the next state at time $t+1$, the state evolution process can be expressed as $X(t+1) = AX(t) + V(t)$, where $V(t)$ is the process additive noise and is assumed to be zero-mean Gaussian noise with covariance Σ_v. The Kalman filter propagates the state using a prediction and an update step. The state prediction equation and error covariance matrix are defined as

$$\begin{aligned} \hat{X}(t+1 \mid t) &= A\hat{X}(t \mid t), \\ \Sigma_v(t+1 \mid t) &= A^T \Sigma_v(t \mid t) A, \end{aligned} \tag{A.1}$$

where $\hat{X}(\cdot)$ is state estimate and the superscript T indicates the transpose of a matrix. The filter is updated by computing the Kalman gain, $K(t)$, as

$$K(t) = \Sigma_v(t \mid t-1) L \left[L^T \Sigma_v(t \mid t-1) L + \Sigma_w(t \mid t) \right]^{-1}, \tag{A.2}$$

where Σ_w is the covariance of the observation noise and L maps the state vector with the measurements. The object state can be updated using

$$\hat{X}(t+1 \mid t) = \hat{X}(t \mid t-1) + K(t)\Big(Z(t) - \Big(L\hat{X}(t \mid t-1)\Big)\Big)$$
$$\Sigma_v(t \mid t) = [I - K(t)L]\Sigma_v(t \mid t-1), \tag{A.3}$$

where Z is the observational model. In the unobserved regions, there is no prior information about the object state and the observation noise covariance is zero. Therefore, (A.2) can be written as

$$K(t) = \Sigma_v(t \mid t-1)L\Big[L^T\Sigma_v(t \mid t-1)L\Big]^{-1}. \tag{A.4}$$

The optimal state estimate in this case can be expressed as

$$\hat{X}(t+1 \mid t) = \left[\left[L^T\Sigma_v(t)L\right]^{-1}L\Sigma_v(t)\right](Z(t))\Big). \tag{A.5}$$

Linear regression finds the optimal estimate of the object state at $t+1$ by minimizing the squared error between the estimate and the observation. Let us consider the generalized weighted sum of the squared residual Γ as

$$\Gamma = \Big(L\hat{X}(t+1 \mid t) - Z(t)\Big)^T\Sigma_w\Big(L\hat{X}(t+1 \mid t) - Z(t)\Big). \tag{A.6}$$

To minimize the squared residual, we take the derivative of (A.6) with respect to optimal state estimate and set it to zero. This results in

$$\hat{X}(t+1 \mid t) = \left[\left(L^T\Sigma_wL\right)^{-1}L^T\Sigma_w\right]Z(t), \tag{A.7}$$

and thus from (A.5) and (A.7) it is possible to notice that both Kalman filter and linear regression exhibit similar behavior for no prior object information.

Acknowledgment

I am deeply indebted to my supervisor Professor Dr. A. Cavallaro from the Queen Mary University of London whose help, stimulating suggestions, and encouragement helped me in completion of this work.

References

[1] T. Kanade, R. Collins, A. Lipton, P. Burt, and L. Wixson, "Advances in cooperative multi-sensor video surveillance," in *Proceedings of the DARPA Image Understanding Workshop*, pp. 3–24, 1998.

[2] R. T. Collins, A. J. Lipton, H. Fujiyoshi, and T. Kanade, "Algorithms for cooperative multisensor surveillance," *Proceedings of the IEEE*, vol. 89, no. 10, pp. 1456–1477, 2001.

[3] B. Keyes, R. Casey, H. A. Yanco, B. A. Maxwell, and Y. Georgiev, "Camera placement and multi-camera fusion for remote robot operation," in *Proceedings of the IEEE International Workshop on Safety, Security and Rescue Robotics*, Gaithersburg, Md, USA, August 2006.

[4] N. Anjum and A. Cavallaro, "Multi-camera calibration and global trajectory fusion," in *Intelligent Video Surveillance: Systems and Technologies*, Y. Ma and G. Qian, Eds., CRC Press, Boca Raton, Fla, USA, 2009.

[5] D. Simon and H. El-Sherief, "Real-time navigation using the global positioning system," *IEEE Aerospace and Electronic Systems Magazine*, vol. 10, no. 1, pp. 31–37, 1995.

[6] J. Hightower, G. Borriello, and R. Want, "SpotON: an indoor 3D location sensing technology based on RF signal strength," CSE Report 2000-02-02, University of Washington, 2000.

[7] A. Savvides, C. Han, and M. R. Strivastava, "Dynamic fine-grained localization in Ad-hoc networks of sensors," in *Proceedings of the 7th annual International Conference on Mobile Computing and Networking (MobiCom '01)*, pp. 166–179, Rome, Italy, July 2001.

[8] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman, "Infrastructure tradeoffs for sensor networks," in *Proceedings of the 1st International Workshop on Wireless Sensor Networks and Applications (WSNA '02)*, pp. 49–58, Atlanta, Ga, USA, September 2002.

[9] A. Ward, A. Jones, and A. Hopper, "A new location technique for the active office," *IEEE Personal Communications*, vol. 4, no. 5, pp. 42–47, 1997.

[10] A. M. Ladd, K. E. Bekris, A. Rudys, G. Marceau, L. E. Kavraki, and D. S. Wallach, "Robotics-based location sensing using wireless ethernet," in *Proceedings of the International Conference on Mobile Computing and Networking*, pp. 227–238, Atlanta, Ga, USA, January 2002.

[11] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices," Tech. Rep. 00-729, University of Southern California, Computer Science Department, 2000, http://isi.edu/ johnh/PAPERS/Bulusu00a.pdf.

[12] R. Want, A. Hopper, V. Falco, and J. Gibbons, "The active badge location system," *ACM Transactions on Information Systems*, vol. 10, no. 1, pp. 91–102, 1992.

[13] R. Zemek, M. Takashima, S. Hara et al., "An effect of anchor nodes placement on a target location estimation performance," in *Proceedings of the IEEE Region 10 Conference*, pp. 1–4, November 2006.

[14] K. K. Chintalapudi, A. Dhariwal, R. Govindan, and G. Sukhatme, "Ad-hoc localization using ranging and sectoring," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM ' 04)*, vol. 4, pp. 2662–2672, March 2004.

[15] H. M. Khan, S. Olariu, and M. Eltoweissy, "Efficient single-anchor localization in sensor networks," in *Proceedings of the 2nd IEEE Workshop on Dependability and Security in Sensor Networks and Systems (DSSNS '06)*, pp. 35–43, April 2006.

[16] X. Ji and H. Zha, "Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM ' 04)*, vol. 4, pp. 2652–2661, March 2004.

[17] A. A. Ahmed, Y. Shang, and H. Shi, "Variants of multidimensional scaling for node localization," in *Proceedings of the 11th International Conference on Parallel and Distributed Systems Workshops*, vol. 1, pp. 140–146, July 2005.

[18] G. Destino and G. T. Freitas de Abreu, "Localization from imperfect and incomplete ranging," in *Proceedings of the 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1–5, September 2006.

[19] S. T. Birchfield and A. Subramanya, "Microphone array position calibration by basis-point classical multidimensional

scaling," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 5, pp. 1025–1034, 2005.

[20] R. B. Fisher, "Self-organization of randomly placed sensors," in *Proceedings of the European Conference on Computer Vision*, pp. 146–160, Copenhagen, Denmark, May 2002.

[21] C. Taylor, A. Rahimi, J. Bachrach, H. Shrobe, and A. Grue, "Simultaneous localization, calibration, and tracking in an ad hoc sensor network," in *Proceedings of the 5th International Conference on Information Processing in Sensor Networks (IPSN '06)*, pp. 27–33, April 2006.

[22] A. Chiuso, P. Favaro, H. Jin, and S. Soatto, "Structure from motion causally integrated over time," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 523–535, 2002.

[23] E. D. Dickmanns and V. Graefe, "Dynamic monocular machine vision," *Machine Vision and Applications*, vol. 1, no. 4, pp. 223–240, 1988.

[24] D. B. Gennery, "Visual tracking of known three-dimensional objects," *International Journal of Computer Vision*, vol. 7, no. 3, pp. 243–270, 1992.

[25] J. E. Guivant and E. M. Nebot, "Optimization of the simultaneous localization and map-building algorithm for real-time implementation," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 242–257, 2001.

[26] S. J. Davey, "Simultaneous localization and map building using the probabilistic multi-hypothesis tracker," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 271–280, 2007.

[27] A. Rahimi, B. Dunagan, and T. Darrell, "Simultaneous calibration and tracking with a network of non-overlapping sensors," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. I187–I194, Washington, DC, USA, July 2004.

[28] O. Javed, Z. Rasheed, O. Alatas, and M. Shah, "KNIGHT: a real time surveillance system for multiple overlapping and non-overlapping cameras," in *Proceedings of the International Conference on Multimedia and Expo*, pp. 649–652, Baltimore, Md, USA, July 2003.

[29] I. N. Junejo, X. Cao, and H. Foroosh, "Autoconfiguration of a dynamic nonoverlapping camera network," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 37, no. 4, pp. 803–816, 2007.

[30] K.-Y. Cheng, V. Tam, and K.-S. Lui, "Improving APS with anchor selection in anisotropic sensor networks," in *Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services*, pp. 49–54, October 2005.

[31] N. Anjum, M. Taj, and A. Cavallaro, "Relative position estimation of non-overlapping cameras," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '07)*, pp. II281–II284, Honolulu, Hawaii, USA, April 2007.

[32] Y. H. Kim and A. Ortega, "Maximum a posteriori (MAP)-based algorithm for distributed source localization using quantized acoustic sensor readings," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '06)*, Toulouse, France, May 2006.

[33] N. Anjum and A. Cavallaro, "Automated localization of a camera network," *IEEE Intelligent Systems*. In press.

[34] N. Anjum and A. Cavallaro, "Single camera calibration for trajectory-based behavior analysis," in *Proceedings of the IEEE International Conference on Advanced Signal and Video based Surveillance*, London, UK, September 2007.

[35] M. S. Grewal and A. P. Andrews, *Kalman filtering: Theory and Practice*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.

[36] C. K. Chui and G. Chen, *Kalman Filtering with Real-Time Applications*, Springer, New York, NY, USA, 1987.

[37] J. C. Hung and B. S. Chen, "Two-stage signal reconstruction under unknown parameters and missing data," in *Proceedings of the IEEE International Conference on Accoustics, Speech, and Signal Processing (ICASSP '03)*, pp. 573–576, April 2003.

[38] V. Digalakis, J. R. Rohlicek, and M. Ostendorf, "ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 1, no. 4, pp. 431–442, 1993.

[39] S. T. Hagen and B. Krose, "Trajectory reconstruction for selflocalization and map building," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '02)*, vol. 2, pp. 1796–1801, May 2002.

[40] H. Zhou, M. Taj, and A. Cavallaro, "Audiovisual tracking using STAC sensors," in *Proceedings of the ACM / IEEE International Conference on Distributed Smart Cameras (ICDSC '07)*, Vienna, Austria, September 2007.