Automatic Camera Network Localization using Object Image Tracks

Marci Meingast
marci@eecs.berkeley.edu
Dept. of Electrical Engineering
and Computer Science
University of California, Berkeley

Songhwai Oh
songhwai.oh@ucmerced.edu
School of Engineering
University of California, Merced

Shankar Sastry
sastry@eecs.berkeley.edu
Dept. of Electrical Engineering
and Computer Science
University of California, Berkeley

Abstract

Camera networks are being used in more applications as different types of sensor networks are used to instrument large spaces. Here we show a method for localizing the cameras in a camera network to recover the orientation and position up to scale of each camera, even when cameras are wide-baseline or have different photometric properties. Using moving objects in the scene, we use an intra-camera step and an inter-camera step in order to localize. The intra-camera step compares frames from a single camera to build the tracks of the objects in the image plane of the camera. The inter-camera step uses these object image tracks from each camera as features for correspondence between cameras. We demonstrate this idea on both simulated and real data.

1. Introduction

Camera networks are becoming a more common type of high-bandwidth sensor network. These systems consists of a number of cameras which communicate through a network to perform some specified tasks. These tasks can be as simple as just sending image data or can be more sophisticated, such as automatic tracking of objects in the scene. With the increase in computing power and advances in camera technologies, camera networks are being used in more and more applications like surveillance [17, 18], intelligent environments [29, 30], and traffic monitoring [10, 5, 16].

An important step in any camera network deployment is localization of the cameras themselves. The position and orientation of the cameras must be determined in three-dimensional (3D) space. By localizing the cameras in the network, image data becomes more useful as we know where the images are captured from and the relations of images of one camera to images of another. This aids in tasks such as tracking and 3D position estimation of an object in the scene.

Many current automatic low-bandwidth sensor network are localized using acoustic information and radio frequency intensities and exploiting received signal strength indicators, time of arrival, time difference of arrival, or angle of arrival [12]. However, these methods will not provide all the localization parameters necessary for cameras, as there is no information on field of view orientation. Manually measuring the pose (location and orientation) of all cameras in the network is a very tedious and time consuming task and sometimes requires special environmental conditions that may not be present. For example, in [27], the authors use a point light sources to calibrate cameras and the EasyCal Calibration Toolbox [1] uses a point light source to calibrate a network of cameras. This method requires conditions in which the light source can actually be seen by the cameras as well as requiring time consuming manual intervention.

There has been some literature examining how to automatically localize multiple cameras in a network. In [15], the authors localize a network of cameras by assuming there is a common set of features points seen by each set of three cameras that can be used for calibration. However, this is often not the case in many camera networks, as they can be wide baseline and even if they see the same objects in the scene the image features of that object look different in each camera and do not lend well to correspondence. Points on the head and feet of a silhouette of a human are used as feature points for correspondence in calibration on a frame by frame basis in [25]. In order to do this a good representation of a human silhouette is needed as well as where the points on the head and feet are located on the silhouette.

In [8], the authors use a statistical method to find the localization parameters of the cameras. Objects are tracked over time and the image position at each time is recorded and used as a means to solve for the localization parameters. However, it is assumed that the height, the Z coordinate of the position vector, of the cameras as well as two of their orientation parameters, are already known. Thus, only the X-Y portion of the position vector and the orientation

around the principal axis of the camera are solved for.

In [23, 6], the authors assume a common global ground plane. Within a single camera, tracked objects are fit to a local ground plane and then using homography constraints, these local ground planes are matched to a global ground plane. In these works, how the camera is oriented to a local ground plane is already known and then tracked objects in time are used to provide the constraints for the homographies and determine which camera field of view relates to which others. In [2, 13], this ground plane concept is extended a bit, where the relation of the local ground plane to the camera is not known, but solved for based on tracked objects and then these local ground planes are aligned to a global ground plane based on homographies. While these methods work well in certain settings, it is not always the case that a global ground plane exists and trying to fit one to the data from the cameras can lead in wrong localization parameters.

In this paper, we present an automatic method for solving the localization task by using raw video from the cameras in the network. By observing moving objects in the scene, we are able to build trajectories of those objects in the image plane, which we call object image tracks, in each camera using a multi-target tracking algorithm. There is no assumption of a global ground plane or any assumptions on the external parameters of the cameras, unlike other work. Also, our method works on wide baseline cameras where common features points may not be able to be found. As demonstrated in Figure 1, it is unclear how these two wide baseline cameras are related until a moving object enters the scene.

The object image tracks from each camera are then used as spatio-temporal features to correlate against other object image tracks from other cameras in order to determine the orientation and position, up to a scale, of the cameras relative to one another. A single camera's coordinate frame can be chosen as the world coordinate frame such that each camera's position and orientation can be related to this world frame.

In Section 2, we discuss our problem formulation for the network. Section 3 discusses our method using motion of objects to build object image tracks using a multi-target tracking algorithm. In Section 4, we examine how object image tracks can be used as features and used for correspondence between cameras for the epipolar constraint. In Section 5, we present results, both simulated and real, on different camera network setups. We conclude in Section 6 and discuss future extensions of this work.

2. Overview of the Method

For our method of localization, we assume the following inputs and outputs:

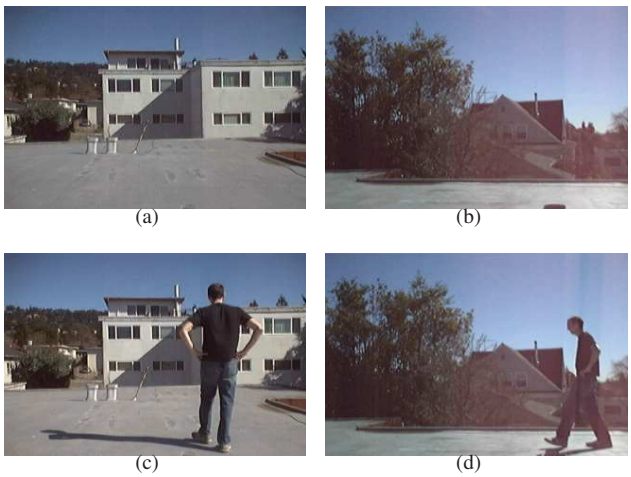


Figure 1. Wide baseline matching: The top row shows images from two wide baseline cameras without any moving object, and it is difficult to tell how the cameras' fields of view are related. The bottom row shows images from the same two cameras with a moving object in their fields of view and it is more clear how their coordinate frames relate.

- **Input:** synchronized video sequences from the N fixed cameras in the network which are at unknown positions and orientations and known internal calibration parameters for all cameras
- **Output:** The orientation, $R \in SO^{3\times3}$ and the position, up to scale factor, $T \in \mathbb{R}^3$ for all N cameras.

We make no prior assumptions on where the N cameras are placed except that there must be some field of view overlap between pairs of cameras if the orientation, up to scale, and position of those cameras are to be recovered. Further, we do not make any assumptions on the scene structure. For example, the cameras may be wide baseline and no prior knowledge of how each camera's coordinate frame is related to the ground of the scene is known. No prior correspondences of features between cameras is known, nor do we make any assumptions that the same static scene features appear in multiple cameras. The problem is finding correspondences between cameras as geometry and photometric characteristics can vary considerably between images from different cameras. Thus, one cannot necessarily use brightness or proximity constraints and traditional methods of features correspondence, such as SIFT features [14] or Salient Region [28] will not necessarily work for localization.

By observing moving objects in the scene, we use this information in order to localize the cameras. Our procedure consists of three main steps in order to use individual raw video streams to localize the cameras : an intra-camera step, an inter-camera step, and then a global recovery step. The intra-camera step, which we call track formation, involves exploiting similarities of objects between frames for each

camera separately. The inter-camera step, which we call track matching, involves using the object image tracks from each camera as features to compare against object image tracks from other cameras. The steps are as follows:

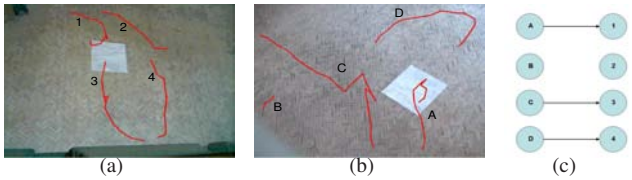
1. **Track Formation:** Find moving objects in each camera's field of view and based on correspondences between frames, build tracks of those objects within the image plane as shown in Figure 2.

2. **Track Matching:** Using the image tracks from each camera as features to compare against image tracks from another cameras in order to determine the relative orientation and position of each camera. This comparison is done in a pair wise manner and is based off of correspondences and the properties of the essential matrix. This is illustrated in Figure 3.

3. **Global Recovery:** The transformation of all relative coordinates of the cameras into a global coordinate frame

(a) Camera 1 (b) Camera 2

Figure 2. Track formation: formation of tracks based on object motion in two separate cameras.



(a) (b) (c)

Figure 3. Track Matching: (a) shows the tracks in a single camera (b) shows the tracks from another camera over the same time period and (c) shows the correct track matchings between the two cameras that is used in the epipolar constraint.

2.1. Track Formation

For this step, we assume there are P moving objects in the scene and each camera, C_i where $i \in 1, 2, ..., N$, observes some subset of objects, $p_i^t \subseteq P$ at each time t. No assumptions are made on what objects are seen by which cameras nor what type of objects can be seen. For example, humans, cars, and dogs could all be types of moving objects seen by the cameras.

We assume the frame rate of the cameras are fast enough to pick up the motion of objects moving in the scene. Within a single camera, C_i, moving objects are found and tracks of these objects within the image plane are built based on multi-target tracking. This is discussed in more detail in section 3.

2.2. Track Matching

Multiple view geometry is well studied. For two views there exist geometric constraints that relate the corresponding points in the 3D camera geometry, which come about in the epipolar constraint. Since we know the internal parameters of all cameras, here we focus on the essential matrix, based off the epipolar constraint.

Once object image tracks in each camera have been recovered, we use these tracks as spatio-temporal features. We compare these spatio-temporal features between each pair of cameras to get a relative orientation and position, up to scale, based on the essential matrix. We discuss this in more detail in section 4

2.3. Global Recovery

Once all relative 3D orientations and positions are found between each pair of cameras, a single camera can be chosen as the world coordinate frame. Without loss of generality we assume the world frame to be the camera coordinate frame of C_1. Then all C_i where $i \neq 1$, can be aligned to the world frame.

3. Building Tracks

Building tracks of objects within the image plane of each camera is the first thing that must be done using the raw video data. Here we discuss the necessary steps in order to build these tracks.

3.1. Filtering for Moving Objects

Filtering the video data in order to find the moving objects in each camera's field of view is the necessary first step. An adaptive background subtraction technique is applied in order to segment moving objects. We use the method proposed by [32] in order to do this segmentation. Items that are determined to be foreground are what we consider for moving objects.

After the background segmentation is run, the remaining foreground objects are further filtered. Bounding boxes around each foreground object are determined. If a bounding box is too close to the boundary of the image, based on some threshold q, then this foreground object is not used in that frame as a moving object. Thus, only the foreground objects that lie completely within the image are treated as the true moving objects. Thus, for each camera C_i we get some number of moving objects p_i^t at each time instance t.

For each moving object in the video frame of a camera, we compute the centroid of that object and end up with a list of centroids, M_i^t, for each camera C_i. We use the centroids of the objects as our measurements for the multi-target tracking in order to build object image tracks using data association.

3.2. Data Association and Multi-Target Tracking

Recently, *multi-target tracking* has received a considerable amount of attention in the computer vision community because the task of tracking multiple objects in video sequences is an important step towards understanding dynamic scenes. The essence of the multi-target tracking problem is to find a track of each object from the noisy measurements. If the sequence of measurements associated with each object is known, multi-target tracking reduces to a set of state estimation problems, for which many efficient algorithms are available. Unfortunately, the association between measurements and objects is unknown. The *data association* problem is to work out which measurements were generated by which objects; more precisely, we require a partition of measurements such that each element of a partition is a collection of measurements generated by a single object or clutter [26]. Due to this data association problem, the complexity of the posterior distribution of the states of objects grows exponentially as time progresses. It is well-known that the data association problem is NP-hard [3, 22], so we do not expect to find efficient, exact algorithms for solving this problem. The problem gets more challenging with video sequences due to the nonlinear camera projection, occlusions, and varying appearances, to name a few.

Since cameras are not calibrated, we cannot use the 3D model-based tracking approaches such as [9, 7]. However, we can still track moving objects on a 2D image plane. In addition, the computational complexity of the model-based approach, *e.g.*, [7], is not desirable for our rapid autonomous calibration task.

In order to handle highly nonlinear and non-Gaussian dynamics and observations, a number of methods based on particle filters has been recently developed to track multiple objects in video [9, 21, 11]. Although particle filters are highly effective in single-target tracking, it is reported that they provide poor performance in multi-target tracking [11]. It is because a fixed number of particles is insufficient to represent the posterior distribution with the exponentially increasing complexity (due to the data association problem). As shown in [11, 31], an efficient alternative is to use Markov chain Monte Carlo (MCMC) to handle the data association problem in multi-target tracking.

For our problem, there is an additional complexity. We do not assume the number of objects is known. A *single-scan* approach, which updates the posterior based only on the current scan of measurements, can be used to track an unknown number of targets with the help of trans-dimensional MCMC [31, 11] or a detection algorithm [21]. But a single-scan approach cannot maintain tracks over long periods because it cannot revisit previous, possibly incorrect, association decisions in the light of new evidence. This issue can be addressed by using a *multi-scan* approach, which updates the posterior based on both current and past scans of measurements. The well-known *multiple hypothesis tracking* (MHT) [4, 24] is a multi-scan tracker, however, it is not widely used due to its high computational complexity.

A newly developed algorithm, called Markov chain Monte Carlo data association (MCMCDA), provides a computationally desirable alternative to MHT [19]. The simulation study in [19] showed that MCMCDA was computationally efficient compared to MHT with heuristics (*i.e.*, pruning, gating, clustering, N-scan-back logic and k-best hypotheses). In this paper, we use the online version of MCMCDA to track multiple objects in a 2-D image plane. Due to the page limitation, we omit the description of the algorithm in this paper and refer interested readers to [20] or [19].

4. Track Matching

In this section we describe how the position and orientation of a pair of cameras can be found relative to one another using the object image tracks. Once we have determined the object image tracks, we use these as spatio-temporal features and look at the correspondence between these features in each pair of cameras.

Once the object image tracks have been formed we use the them as features to do feature correspondence between cameras. While it is possible just to use the centroids of objects from all the frames alone as features and do point correspondences, using the track information from the multi-target tracking algorithm is much more beneficial. By doing data association on the centroids, a track is formed which best fits the centroid data and smooths over the noisy centroid measurements. The new estimated tracks are then used for correspondence instead of the original noisy centroids. By using tracks, we can cut down the correspondence space as multiple points in one track can only correspond to multiple points in another single track, not separate points from multiple tracks. We can further constrain the correspondence based on timing data on the tracks. Using the object image tracks from the intra-camera data association greatly reduces computation time and further constrains the position and orientation of the cameras, leading us to a more accurate solution.

We define the problem as follows. For a given time period $[t_0, t_0 + 1, \ldots, t_n]$, let

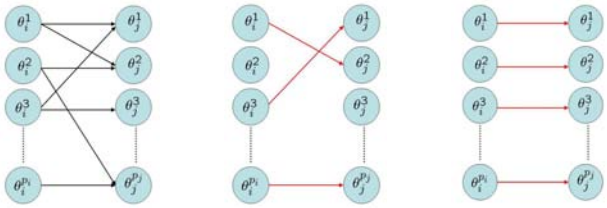
(C_i, C_j)	Pair of cameras where $i \neq j$
Θ_i	Set of tracks detected by C_i during $[t_0, t_n]$
$t_s(\theta_i)$	Starting time of track $\theta_i \in \Theta_i$
$t_e(\theta_i)$	Ending time of track $\theta_i \in \Theta_i$
$t_d(\theta_i)$	Duration of track $\theta_i \in \Theta_i$ (i.e., $t_d(\theta_i) = [t_s(\theta_i), t_s(\theta_i)+1, \ldots, t_e(\theta_i)]$)

where $t_0 \leq t_s(\theta_i) < t_e(\theta_i) \leq t_n, \forall \theta_i \in \Theta_i$. For a pair of cameras (C_i, C_j), we form a bipartite graph $G_{ij} = (\Theta_i, \Theta_j, E_{ij})$ where Θ_i and Θ_j are vertex sets and the edge set E includes all the pairs of tracks with overlapping times. More formally,

$$E_{ij} = \{(\theta_i, \theta_j) : \theta_i \in \Theta_i, \theta_j \in \Theta_j, t_d(\theta_i) \cap t_d(\theta_j) \neq \emptyset\}$$

Now, let Γ^0_{ij} be a set of all matchings in G_{ij}. A matching in G_{ij} is a subset $M \subset E_{ij}$ such that no two edges in M share a vertex.

It must be checked to see if there are enough tracks in a matching for computing the essential matrix. For the most general case, where all object image tracks are linear, we require at least 4 matching tracks between cameras and 3 or more overlapping times in each track pair. This provides enough constraints for the essential matrix to be solved for, excluding a few singular cases. Let $\Gamma_{ij} = \{\gamma \in \Gamma^0_{ij} : |\gamma| \geq k_1, |t_d(\theta_i) \cap t_d(\theta_j)| \geq k_2 \text{ for all } (\theta_i, \theta_j) \in \gamma\}$, where $|A|$ denotes the cardinality of set A. Γ_{ij} is a set of matchings in G_{ij} of size larger than or equal to k_1, consisting of pairs of tracks with larger than k_2 overlapping times (for instance, $k_1 = 4$ and $k_2 = 3$ for the linear case). We call $\gamma \in \Gamma_{ij}$ a *candidate track correspondence*. An example of a candidate track correspondence can be seen in Figure 4(c).



(a) (b) (c)

Figure 4. (a) An example of a bipartite graph G_{ij} for (C_i, C_j) $(k_1 = 4, p_i = |\Theta_i|)$. (b) An example of a matching in G_{ij} which does not qualify as a candidate matching as only 3 tracks match up; and (c) An example of a candidate track correspondence in G_{ij}. In this example, we have assumed that each pair of tracks has more than k_2 overlapping times.

Given a candidate track correspondence $\gamma \in \Gamma_{ij}$, let Φ_γ be a set of all 3-matchings in γ. For each $\phi \in \Phi_\gamma$, we solve for the essential matrix E_ϕ. Then, we search for a matching $\gamma^* \in \Gamma_{ij}$ which minimizes the following cost function:

$$d(\gamma) = \frac{\sum_{\phi_1, \phi_2 \in \Phi_\gamma, \phi_1 \neq \phi_2} \|E_{\phi_1} - E_{\phi_2}\|_F}{|\{\phi_1, \phi_2 \in \Phi_\gamma : \phi_1 \neq \phi_2\}|}, \qquad (1)$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The cost function $d(\gamma)$ can be considered as an average deviation of all the essential matrices formed by the matching γ. If γ is a correct matching, we can expect that all the essential matrices generated from γ are close to each other. Hence, $d(\gamma)$ should be small as well. Again, $\gamma^* = \arg\min d(\gamma)$. If $d(\gamma^*)$ is less than some threshold, E_{ϕ^*} for some $\phi^* \in \Phi_{\gamma^*}$ is chosen to be the essential matrix between C_i and C_j. Otherwise, we say the camera pair does not share a field of view.

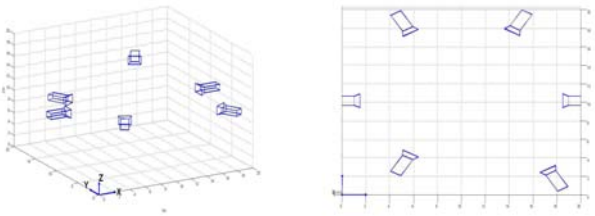
A natural question to ask is whether tracks based on the centroid of an object will actually correspond to the same points in space when dealing with the epipolar constraint. If the segmentation is perfect and the objects are spheres, then the centroids of the segmented objects do in fact correspond to the same points in space. Yet most objects do not have this nice property nor can be segmented out perfectly in the image such as people,cars, dogs, and other types of moving objects often in scenes. However, as an upper bound on the error, we know that the true centroid of the object will be inside the convex hull of the silhouette segmented out. Our results show that the estimated results are within 8 degrees, in both position and orientation, of the ground truth results.

5. Results

The method discussed in this paper was tested in a simulated camera network setup and a real camera network setup.

5.1. Simulation

To test our method, we simulated objects moving through a scene observed by a camera network. The simulation was done in matlab using 7 different camera views and different size cubes as the moving objects. Perspective projection and triangular fields of view were used for the simulated cameras. All measurements had gaussian noise of $N(0, 1)$ added to them. The camera setup can be seen in Figure 5(a) and 5(b).



(a) (b)

Figure 5. Camera network simulation setup

Using the ground truth to compare measurement for the corners of the cubes, the error in the estimated measurements was on average off by 3%, based on image size.

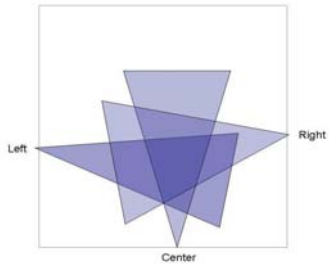


Figure 6. Overhead view of the position of the 3 real cameras

5.2. Images from Camera Network

We also tested our algorithm on a system of cameras. Here we show the results on data from a camera network of three cameras. Images from the network can be seen in Figure 7.

In this setup three cameras were placed in a wide baseline configuration on a second story balcony of a building looking down at the first floor. Figure 6 illustrates the layout of the cameras from an overhead view. Tracks of people walking on the first floor were used to localize the cameras. Three different sequences of 60 seconds of video footage were used. As we had control over the environment, ground truth for the position and orientation of the cameras was obtained using a point light source and methods in the [1].

The resulting error in the estimated position, up to scale, can be seen in Table 1 and the estimated orientation error can be seen in Table 2. The center camera's coordinate frame was chosen as the world coordinate frame and the right and left cameras aligned to the center camera's coordinate frame in the global recovery step. We compared the ground truth unit direction vectors for orientation and position of the left and right cameras in the world coordinate frame to the estimated direction vectors from our method. It can be seen seen that the error in the estimation of the localization parameters is small using the centroids of the objects in the scene to build the image tracks and then using the estimated track points to do the correspondence.

camera	sequence 1	sequence 2	sequence 3
Right	5.04	6.22	5.81
Left	4.11	7.68	5.91
Center	0	0	0

Table 1. Position Error: The error in the estimated position, up to scale, from the tracks is given in degrees here. The coordinate frame of the center camera is chosen as the world coordinate frame and all other coordinate frames are aligned to this.

6. Conclusion

In this paper we have demonstrated a novel technique using spatio-temporal features to provide an efficient method

camera	sequence 1	sequence 2	sequence 3
Right	1.42	3.67	2.84
Left	2.13	2.56	3.21
Center	0	0	0

Table 2. Orientation Error: The error in the estimated orientation from the tracks is given in degrees here. The coordinate frame of the center camera is chosen as the world coordinate frame and all other coordinate frames are aligned to this.

to automatically localize the cameras in a network. Intra-camera information about moving objects is exploited to find the tracks of these objects within a camera's field of view. These object image tracks are then used as features to do inter-camera correspondence and determine if the two cameras share a field of view and how they are related based on the essential matrix. We have demonstrated how this technique work on both simulated and real data. Future work on localizing using this technique involves running experiments on a real camera network setup with more cameras involved and adding in object descriptors to provide additional information for the intra-camera step. We also plan to demonstrate this method on heterogeneous camera pairs, such as a catadioptric and perspective camera pair.

References

[1] Easycal. http://www.cis.upenn.edu/ kostas/ tele-immersion/research/downloads/EasyCal/. 1, 6

[2] J. Black, T. Ellis, and P. Rosin. Multi view image surveillance and tracking. *Proceedings of Workshop on Motion and Video Computing*, 2002. 2

[3] J. Collins and J. Uhlmann. Efficient gating in data association with multivariate distributed states. *IEEE Trans. Aerospace and Electronic Systems*, 28(3):909–916, July 1992. 4

[4] I. Cox and S. Hingorani. An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. In *International Conf. on Pattern Recognition*, pages 437–443, 1994. 4

[5] R. Cucchiara, M. Piccardi, and P. Mello. Image analysis and rule-based reasoning for a traffic monitoring system. *IEEE Trans. Intelligent Transportation Systems*, 1(2):119–130, June 2000. 1

[6] R. Cucchiara, A. Prati, and R. Vezzani. A system for automatic face obscuration for privacy purposes. *in Pattern Recognition Letters*, 2006. 2

[7] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2000. 4

[8] S. Funiak, C. Guestrin, M. Paskin, and R. Sukthankar. Distributed localization of networked cameras. *The Fifth International Conference on Information Processing in Sensor Networks*, April 2006. 1

[9] M. Isard and J. MacCormick. BraMBLe: A Bayesian multiple-blob tracker. In *International Conference on Computer Vision*, pages 34–41, 2001. 4

[10] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi. Traffic monitoring and accident detection at intersections. *IEEE Trans. Intelligent Transportation Systems*, 1(2):108–118, June 2000. 1

[11] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1918, Nov. 2005. 4

[12] K. Langendoen and N. Reijers. Distributed localization in wireless sensor networks: a quantitative comparison. *Comput. Networks*, 43(4):499–518, 2003. 1

[13] L. Lee, R. Romano, and G. Stein. Monitoring activities from multiple video streams: Establishing a common coordinate frame. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000. 2

[14] D. Lowe. Distinctive image features from scale invariant keypoints. *In IJCV*, 60(2):91–110, 2004. 2

[15] W. Mantzel, C. Hyeokho, and R. Baraniuk. Distributed camera network localization. *Asilomar Conference on Signals, Systems and Computers*, Nov. 2004. 1

[16] O. Masoud, N. P. Papanikolopoulos, and E. Kwon. The use of computer vision in monitoring weaving sections. *IEEE Trans. Intelligent Transportation Systems*, 2(1):18–25, Marci 2001. 1

[17] M. McCahill and C. Norris. From cameras to control rooms: the mediation of the image by cctv operatives. *CCTV and Social Control:The politics and practice of video surviellance-European and global perspectives*, 2004. 1

[18] M. T. Moore. Cities opening more video surveillance eyes. *USA Today*, July 18 2005. 1

[19] S. Oh, S. Russell, and S. Sastry. Markov chain Monte Carlo data association for general multiple-target tracking problems. In *Proc. of the 43rd IEEE Conference on Decision and Control*, Paradise Island, Bahamas, Dec. 2004. 4

[20] S. Oh, S. Russell, and S. Sastry. Markov chain Monte Carlo data association for multi-target tracking. *IEEE Trans. Automatic Control (submitted)*, 2007. 4

[21] K. Okuma, A. Taleghani, N. de Freitas, J. Little, and D. Lowe. A boosted particle filter: Multitarget detection and tracking. In *European Conference on Computer Vision*, 2004. 4

[22] A. Poore. Multidimensional assignment and multitarget tracking. In I. J. Cox, P. Hansen, and B. Julesz, editors, *Partitioning Data Sets*, pages 169–196. American Mathematical Society, 1995. 4

[23] A. Rahimi, B. Dunagan, and T. Darrell. Simultaneous calibration and tracking with a network of non-overlapping sensors. *Computer Vision and Pattern Recognition*, 1, July 2004. 2

[24] D. Reid. An algorithm for tracking multiple targets. *IEEE Trans. Automatic Control*, 24(6):843–854, December 1979. 4

[25] S. Sinha, M. Pollefeys, and L. McMillan. Camera network calibration from dynamic silhouettes. *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, 2004. 1

[26] R. Sittler. An optimal data association problem on surveillance theory. *IEEE Trans. Military Electronics*, MIL-8:125–139, April 1964. 4

[27] D. Stevenson and M. Fleck. Robot aerobics: four easy steps to a more flexible calibration. In *International Conference on Computer Vision*, Paradise Island, Bahamas, June 1995. 1

[28] A. Z. T. Kadir and M. Brady. An affine invariant salient region detector. *In ECCV*, 2004. 2

[29] R. Wolff, D. J. Roberts, A. Steed, and O. Otto. A review of telecollaboration technologies with respect to closely coupled collaboration. *International Journal of Computer Applications in Technology*, 29(1), 2007. 1

[30] Z. Yang, W. Wu, K. Nahrstedt, G. Kurillo, and R. Bajcsy. Viewcast: View dissemination and management for multiparty 3d tele-immersive environments. *ACM Multimedia*, 2007. 1

[31] T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 4

[32] Z.Zivkovic. Improved adaptive gausian mixture model for background subtraction. *International Conference Pattern Recognition*, 2:28–31, 2004. 3

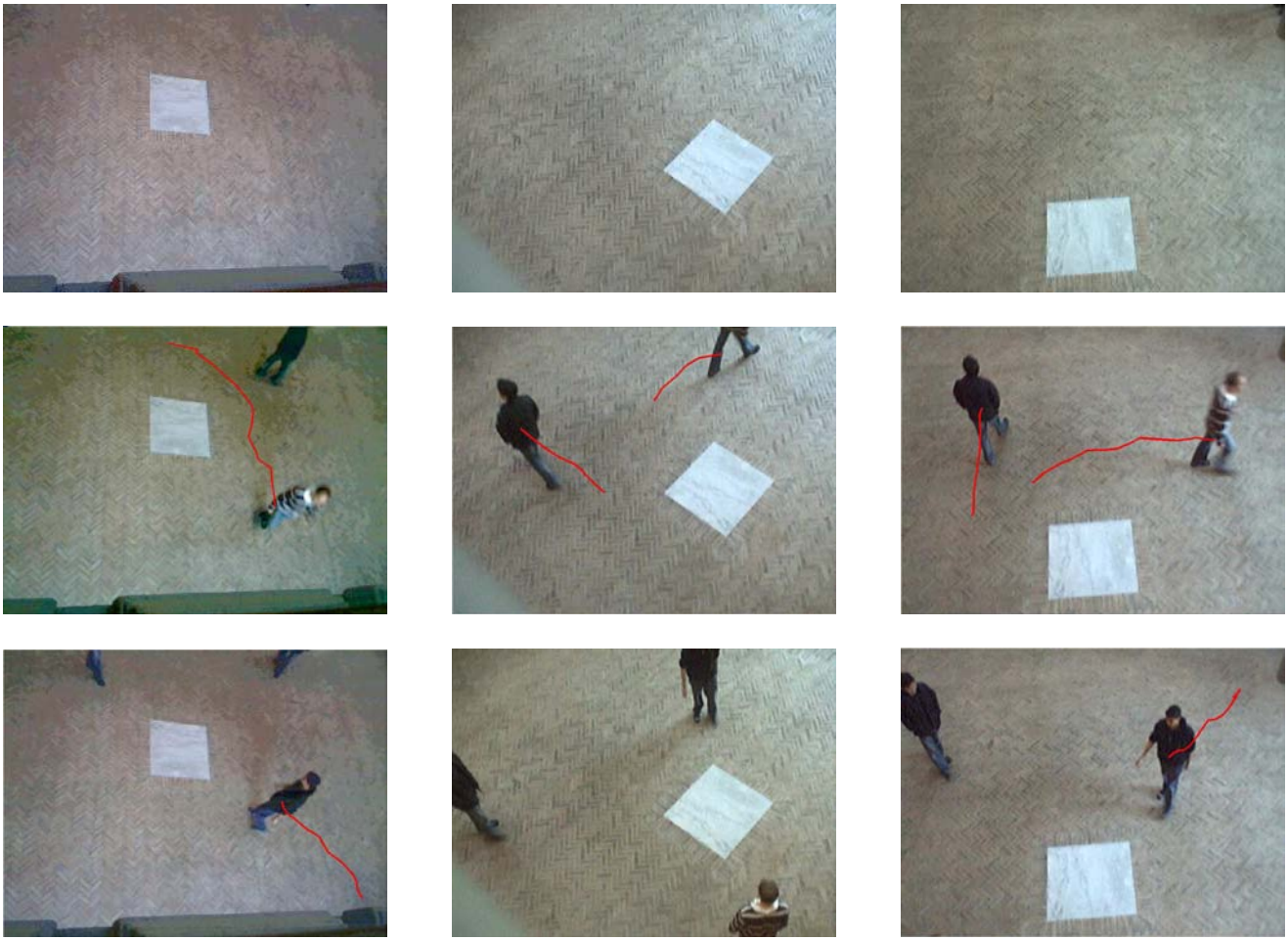


Figure 7. Real Camera Network: Top Row: Images from the cameras with no moving objects Middle and Bottom Rows: Images from cameras with tracks of moving objects shown over time.