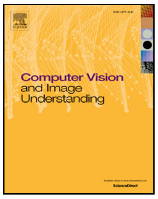
Computer Vision and Image Understanding 139 (2015) 40–58

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu



Sparsity-driven bandwidth-efficient decentralized tracking in visual sensor networks

Serhan Coşar*, Müjdat Çetin

Faculty of Engineering and Natural Sciences, Sabancı University, Orta Mahalle, Üniversite Caddesi No: 27 34956 Tuzla-İstanbul, Turkey

ARTICLE INFO

Article history:
Received 22 January 2014
Accepted 9 April 2015
Available online 11 June 2015

Keywords:
Camera networks
Visual sensor networks
Human tracking
Sparse representation
Designing overcomplete dictionaries

ABSTRACT

Recent developments in low-cost CMOS cameras have created the opportunity of bringing imaging capabilities to sensor networks and a new field called visual sensor networks (VSNs) has emerged. VSNs consist of image sensors, embedded processors, and wireless transceivers which are powered by batteries. Since energy and bandwidth resources are limited, setting up a tracking system in VSNs is a challenging problem. In this paper, we present a framework for human tracking in VSN environments. The traditional approach of sending compressed images to a central node has certain disadvantages such as decreasing the performance of further processing (i.e., tracking) because of low quality images. Instead, in our decentralized tracking framework, each camera node performs feature extraction and obtains likelihood functions. We propose a sparsity-driven method that can obtain bandwidth-efficient representation of likelihoods extracted by the camera nodes. Our approach involves the design of special overcomplete dictionaries that match the structure of the likelihoods and the transmission of likelihood information in the network through sparse representation in such dictionaries. We have applied our method for indoor and outdoor people tracking scenarios and have shown that it can provide major savings in communication bandwidth without significant degradation in tracking performance. We have compared the tracking results and communication loads with a block-based likelihood compression scheme, a decentralized tracking method and a distributed tracking method. Experimental results show that our sparse representation framework is an effective approach that can be used together with any probabilistic tracker in VSNs.

© 2015 Elsevier Inc. All rights reserved.

1. Introduction

Over the past decade, large-scale camera networks have enjoyed increased use in a wide range of applications, especially in security and surveillance. With the developments in wireless sensor networks and the availability of inexpensive image sensors, a new field has emerged: Visual sensor networks (VSNs), i.e., networks of wirelessly interconnected battery-operated devices that acquire video data.

Using a camera in a wireless network poses unique and challenging problems that do not exist either in traditional multi-camera video analysis systems or in sensor networks. In most of the multi-camera video analysis systems, a centralized approach, in which the raw data acquired by cameras are collected in a central unit and analyzed to perform the task of interest, is followed. However, performing complex tasks, such as tracking, recognition, etc., in a communication-constrained VSN environment is extremely challenging. For such constraints, with a data compression perspective, the common approach is to compress images in the process of transmission to the central unit. This strategy essentially focuses on low-level data compression without regard to the final inference goal. Such a strategy may not be appropriate for use under scenarios with severe bandwidth limitations and might cause significant degradation in tracking performance with large compression ratios. We provide a more in-depth review of existing work in Section 2.

In this paper, we propose a different strategy that is better matched to the final inference goal, which, in the context of this paper, is tracking. We propose a sparsity-driven tracking method that is suitable for energy and bandwidth constraints in VSNs. Our method is a decentralized tracking approach in which each camera node in the network performs feature extraction by itself and obtains image features (likelihood functions). In scenarios with overlapping cameras, tracking is performed by fusing the likelihoods obtained from each view. Instead of directly sending likelihood functions to the fusion node, we compute and transmit sparse representations of the likelihoods. By sending such sparse representations to the fusion node, multi-view tracking can be performed without overloading the network. We design special overcomplete dictionaries for the sparse

* Corresponding author.
E-mail addresses: serhancosar@sabanciuniv.edu (S. Coşar), mcetin@sabanciuniv.edu (M. Çetin).

http://dx.doi.org/10.1016/j.cviu.2015.04.010
1077-3142/© 2015 Elsevier Inc. All rights reserved.

representation of likelihood functions. The main contribution of this work is building a sparse representation framework and designing overcomplete dictionaries that are matched to the structure of likelihoods. In particular our dictionaries are designed in an adaptive manner by exploiting the specific known geometry of the measurement scenario and by focusing on the problem of human tracking. Each element in the dictionary for each camera corresponds to the likelihood that would result from a single human at a particular location in the scene. Hence actual likelihoods extracted from real observations from scenes containing multiple individuals can be very sparsely represented in our approach. By using these dictionaries, we can represent likelihoods with a very small number of coefficients, and thereby decrease the communication between camera and fusion nodes. To the best of our knowledge, such sparse representation based compression of likelihood functions computed in the context of tracking in a VSN has not been proposed in previous work.

We have used our method within the context of two multi-camera human tracking algorithms [1,2]. We have modified these methods in order to obtain decentralized tracking algorithms. Both by qualitative and quantitative results, we have shown that our method is better than using the block-based compression scheme in [3], the decentralized tracking method in [4], the distributed tracking methods in [5,6] and a traditional centralized approach that compresses raw images acquired by each camera. The sparse likelihood representation framework we present can be used within any probabilistic tracking method under VSN constraints without significantly degrading the tracking performance.

In Section 2, existing pieces of work on tracking in VSNs are reviewed. Section 3 presents our decentralized approach for multi-camera tracking in detail. In Section 4, our sparse representation framework and the details of our specially designed overcomplete dictionaries are described. Experimental results are presented in Section 5. Finally in Section 6, we provide a summary and conclusions.

2. Related work

There exists some previous work on tracking in VSNs. In several pieces of work, basic features or techniques are used to adapt centralized tracking methods to VSNs. For instance, visual hulls are used in [7,8] to detect the presence and number of humans. However, since a visual hull presents the largest volume in which a human can reside, the exact number of humans cannot be determined when humans are positioned close to one another. To minimize the amount of data to be communicated between cameras, in some methods simple features are used for communication. For instance, 2D trajectories are used in [9]. In [10], 3D trajectories together with color histograms are used. Hue histograms along with 2D position are used in [11].

Moreover, there are decentralized approaches in which cameras are grouped into clusters and tracking is performed by local cluster fusion nodes. This kind of approach has been applied to the multi-camera target tracking problem in various ways [4,12,13]. For a nonoverlapping camera setup, tracking is performed by maximizing the similarity between the observed features from each camera and minimizing the long-term variation in appearance using graph matching at the fusion node [12]. For an overlapping camera setup, a cluster-based Kalman filter in a network of wireless cameras has been proposed in [4,13]. In this work, local measurements of the target acquired by members of the cluster are sent to the fusion node. Then, the fusion node estimates the target position via an extended Kalman filter, relating the measurements acquired by the cameras to the actual position of the target by nonlinear transformations.

To further increase scalability and to reduce communication costs, distributed estimation operates without local fusion centers. The estimates generated in a camera are transmitted to its immediate neighbors only. The received estimates are used to refine the estimates at these immediate neighbors, and these refined estimates are then transmitted to the next group of neighbors [5,6,14]. This process ends after a predefined number of steps after all cameras viewing the target are visited or when the uncertainty has decreased below a desired value. In [5], the Kalman–Consensus algorithm [15] is adapted to take into account the directional nature of video sensors and the network topology. Each camera estimates the locations of the people in the scene based on its own sensed data which is then shared locally with the neighboring cameras in an iterative fashion, and a final estimate is arrived at in the network using the Kalman–Consensus algorithm. As an extension of this approach, in [6] authors presented the Information Weighted Consensus filter that weights the estimates coming from neighboring cameras by their information. Thus a camera node which has less information about a person's state is given less weight in the overall estimation process. A wireless embedded smart camera system for cooperative human tracking has been proposed in [14]. At each camera lightweight foreground detection and color histogram based tracking algorithms are implemented and run. Important portions of video and trajectories are determined by detecting events of interest that are pre-defined by users. Communication in the network is minimized by sending messages only when an event of interest occurs.

There are certain limitations of previous work which motivate further research. The methods in [7–11] that use simple features may be capable of decreasing the communication, but they are not capable of maintaining robustness of tracking performance in the case of reduced communication. For the sake of bandwidth efficiency, these methods choose to change the features from complex and robust to simpler, but not so effective ones. Distributed tracking methods [5,6,14] fit well to the needs of VSNs, but suffer from several disadvantages. In the literature of multi-camera tracking, there are many algorithms that can perform robust tracking. In order to use such algorithms in VSN environments, we need to implement existing centralized trackers in a distributed way. In order to do that, usually, one needs to modify pretty much each step from feature extraction to final inference, which is not a straight-forward task and which can affect the performance of the tracker. Performing local processing and collecting features to the fusion node, as in [4,12,13], may not satisfy the bandwidth requirements in a communication-constrained VSN environment. In particular, depending on the size of image features and the number of cameras in the network, even collecting features to the fusion node may become expensive for the network. In such cases, further approximations on features are necessary.

Over the last decade, an alternative sampling/sensing theory, known as "compressed sensing" has emerged. Compressed sensing enables the recovery of signals, images, and other data from what appear to be undersampled observations. Compressed sensing is a technique for acquiring and reconstructing a signal from small amount of measurements utilizing the prior knowledge that the signal has a sparse representation in a proper space. As a consequence, compressed sensing and sparse representation (SR) have become important signal recovery techniques because of their success for acquiring, representing, and compressing high-dimensional signals in various application areas [16–19]. In the past few years, variations and extensions of l_1 minimization have been applied to many vision tasks, including face recognition [20], denoising and inpainting [17], background modeling [21], and image classification [22]. Compressed sensing has also been combined with distributed estimation to perform distributed video coding in VSNs [23]. In almost all of these applications, using sparsity as a prior leads to state-of-the-art results [24].

Following the observations about SR and considering the problems of existing methods, we propose a decentralized approach that fits well to the needs of VSNs and exploits desirable features of a successful centralized tracking algorithm. By transmitting sparse representations of image features, our method can reduce the bandwidth requirements without significantly decreasing their quality.

3. Tracking in visual sensor networks

3.1. Decentralized tracking

In a traditional setup of camera networks, which we call centralized tracking, each camera acquires an image and sends this raw data to a central unit. In the central unit, multi-view data are collected, relevant features are extracted and combined, finally, using these features, the positions of the humans are estimated. Hence, integration of multi-view information is done at the raw-data level by pooling all images into a central unit. The presence of a single global fusion center leads to high data-transfer rates and the need for a computationally powerful machine. Such an approach cannot satisfy the scalability, bandwidth-efficiency, and energy-efficiency requirements of a VSN. Compressing raw image data may decrease the communication in the network, however high compression ratios imposed by severe bandwidth limitations could lead to degraded tracking performance. For this reason, centralized trackers are not very appropriate for use in VSN environments. In decentralized tracking, there is no central unit that collects all raw data from the cameras. Cameras are grouped into clusters and nodes communicate with their local cluster fusion nodes only [25]. Communication overhead is reduced by limiting the cooperation within each cluster and among fusion nodes.

After acquiring the images, each camera extracts useful features from the images it has observed and sends these features to the local fusion node. The processing capability of camera nodes in emerging VSNs enable feature extraction at the camera nodes without the need to send the images to the central unit [26–29]. Using the multi-view image features, tracking is performed in the local fusion node. Hence, in decentralized tracking, multi-view information is integrated in feature-level by combining the features in small clusters. This both reduces the communication in the network and removes the need of powerful processors in the fusion node.

Decentralized approaches are appropriate for VSNs in many aspects. The processing capability of each camera is utilized by performing feature extraction at the camera-level. Since cameras are grouped into clusters, the communication overhead is reduced by limiting the cooperation within each cluster and among fusion nodes. In other words, by a decentralized approach, feature extraction and communication are distributed among cameras in clusters, therefore, efficient estimation can be performed.

Modeling the dynamics of humans in a probabilistic framework is a common perspective of many multi-camera human tracking methods [1,30–33]. In tracking methods based on a probabilistic framework, data and/or extracted features are represented by likelihood functions, $p(y|x)$ where $y \in \mathbb{R}^d$ and $x \in \mathbb{R}^m$ are the observation and state vectors, respectively. In other words, for each camera, a likelihood function is defined in terms of the observations obtained from its field of view. In centralized tracking, of course, the likelihood functions are computed after collecting the image data of each camera at the central unit. For a decentralized approach, since each camera node extracts local features from its field of view, these likelihood functions can be evaluated at the camera nodes and they can be sent to the fusion node. Then, in the fusion node the likelihoods can be combined and tracking can be performed in the probabilistic framework. A flow diagram of a generic decentralized approach is illustrated in Fig. 1. Following this line of thought, we have converted the tracking approaches in [1,2] to decentralized trackers as explained in the next section.

3.2. Multi-camera tracking algorithms

We have applied our proposed framework within the context of two different tracking methods. In this section, we describe these tracking methods and explain how we have converted these trackers to decentralized trackers.

Fusion node selection and sensor resource management (sensor tasking) is out of scope of this paper. We have assumed that one of the camera nodes, a relatively more powerful one, has been selected as the fusion node. In a practical implementation, resource management can be performed using existing work in [34–37].

3.2.1. Algorithm 1

In this section we describe the tracking method of [1]. In [1], the visible part of the ground plane is discretized into a finite number G of regularly spaced 2D locations. Let $\mathbf{L}_t = (L_t^1, \ldots, L_t^{N^*})$ be the locations of individuals at time t, where N^* stands for the maximum allowable number of individuals. Given T temporal frames from C cameras, $\mathbf{I}_t = (I_t^1, \ldots, I_t^C)$, $t = \{1 \ldots T\}$, the goal is to estimate the trajectory of person n, $\mathbf{L}^n = (L_1^n, \ldots, L_T^n)$, by seeking the maximum of the probability of both the observations and the trajectory ending up at location k at time t:

$$\Phi_t^n(k) = \max_{l_1^n, \ldots, l_{t-1}^n} P\big(\mathbf{I}_1, L_1^n = l_1^n, \ldots, \mathbf{I}_t, L_t^n = k\big) \quad (1)$$

Under a hidden Markov model, it turns into the classical recursive expression:

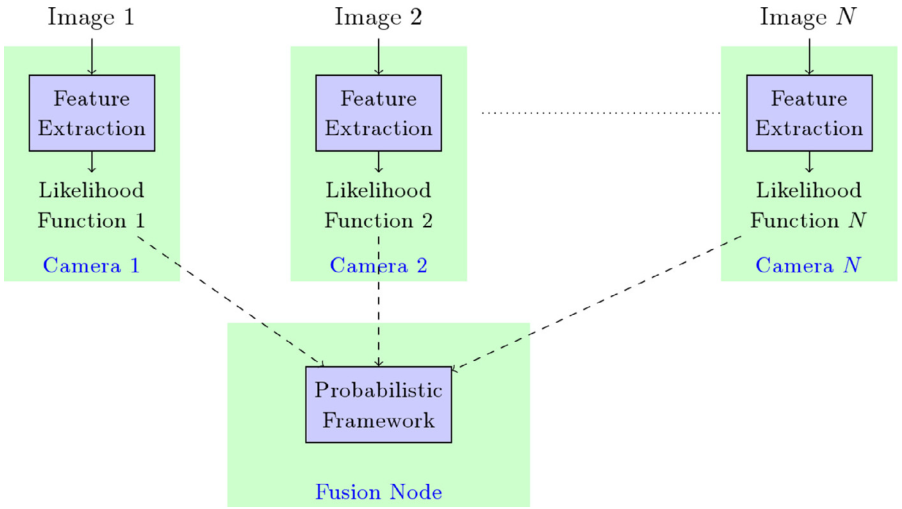


Fig. 1. The flow diagram of a decentralized tracker using a probabilistic framework.

$$\Phi_t^n(k) = \underbrace{P\left(\mathbf{I}_t \middle| L_t^n = k\right)}_{\text{Appearance model}} \max_{\tau} \underbrace{P\left(L_t^n = k \middle| L_{t-1}^n = \tau\right)}_{\text{Motion model}} \Phi_{t-1}(\tau) \tag{2}$$

The motion model $P(L_t^n = k|L_{t-1}^n = \tau)$ is a circular uniform distribution with a limited radius and center τ, which corresponds to a loose bound on the maximum speed of a walking human.

From the input images $\mathbf{I}_t$, by using background subtraction, foreground binary masks, $\mathbf{B}_t$, are obtained. Let the colors of the pixels inside the blobs be denoted by $\mathbf{T}_t$ and let X_k^t be a Boolean random variable denoting the presence of an individual at location k of the grid at time t. It is shown in [1] that the appearance model in Eq. (2) can be decomposed as:

$$\overbrace{P\left(\mathbf{I}_t \middle| L_t^n = k\right)}^{\text{Appearance model}} \propto \underbrace{P\left(L_t^n = k \middle| X_k^t = 1, \mathbf{T}_t\right)}_{\text{Color model}} \underbrace{P\left(X_k^t = 1 \middle| \mathbf{B}_t\right)}_{\text{Ground plane occupancy}} \tag{3}$$

In [1], humans are represented as simple rectangles used to create synthetic ideal images that would be observed if people were at given locations. Within this model, the ground plane occupancy is approximated by measuring the similarity between ideal images and foreground binary masks.

Let $T_t^c(k)$ denote the color of the pixels taken at the intersection of the foreground binary mask, B_t^c, from camera c at time t and the rectangle A_k^c corresponding to location k in that same field of view. Say we have the color distributions of the N^* individuals present in the scene, $\mu_1^c, \ldots, \mu_{N*}^c$. The color model of person n in Eq. (3) can be expressed as:

$$\begin{aligned} P\left(L_t^n = k \middle| X_k^t = 1, \mathbf{T}_t\right) \propto P\left(\mathbf{T}_t \middle| L_t^n = k\right) &= P\left(T_t^1(k), \ldots, T_t^C(k) \middle| L_t^n = k\right) \\ &= \prod_{c=1}^{C} P\left(T_t^c(k) \middle| L_t^n = k\right) \end{aligned} \tag{4}$$

Different from [1], we represent $P(T_t^c(k)|L_t^n = k)$ by comparing the estimated color distribution (histogram) of the pixels in $T_t^c(k)$ and the color distribution μ_n^c with the Bhattacharya coefficient between two distributions [3]. By performing a global search with dynamic programming using Eq. (2), the trajectory of each person can be estimated.

3.2.2. Decentralized version of Algorithm 1

From the above formulation, we can see that there are two different likelihood functions defined in the method. One is the ground plane occupancy map (GOM), $P(X_k^t = 1|\mathbf{B}_t)$, approximated using the foreground binary masks. The other is the ground plane color map (GCM), $P(L_t^n = k|X_k^t = 1, \mathbf{T}_t)$, which is a multi-view color likelihood function defined for each person individually. This map is obtained by combining the individual color maps, $P(T_t^c(k)|L_t^n = k)$, evaluated using the images each camera acquires. Since foreground binary masks are simple binary images that can be easily compressed by a lossless compression method, they can be directly sent to the fusion node without overloading the network. Therefore, we keep these binary images as in the original method and GOM is evaluated at the fusion node. In our framework, we evaluate GCM in a decentralized way (as presented in Fig. 1): At each camera node ($c = 1, \ldots, C$), the local color likelihood function for the person of interest ($P(T_t^c(k)|L_t^n = k)$) is evaluated by using the image acquired from that camera. Then, these likelihood functions are sent to the fusion node. At the fusion node, these likelihood functions are integrated to obtain the multi-view color likelihood function (GCM) (Eq. (4)). By combining GCM and GOM with the motion model, the trajectory of the person of interest is estimated at the fusion node using dynamic programming (Eq. (2)). The whole process is run for each person in the scene.

Since each camera keeps a reference color histogram individually for each person in the scene, data association between different people is performed at the camera level. After the likelihoods are transmitted to the fusion node, the association of these likelihoods obtained by each camera should be performed in the fusion node. Since the main focus of this paper is to decrease communication in the network using sparse representation, we have followed a simple approach for data association in the fusion node. By assuming there is only one person in the scene in the beginning of the tracking process, we assign an ID number for each likelihood function coming from cameras to the fusion node. Likelihoods with the same ID number from different cameras are associated with one another at the fusion node, which avoids high level of computations in the fusion node. For scenarios that do not satisfy our assumption, a color-based data association algorithm can be used before running our approach. We have presented such a case in Section 5.2 and shown that the likelihoods of four people can be correctly matched using a color-based data association algorithm. More advanced data association algorithms that can reliably work under severe conditions such as significantly varying lighting conditions across cameras or a very crowded scene, are beyond the scope of this paper.

3.2.3. Algorithm 2

This section describes the second tracking algorithm [2] used together with our proposed approach. In [2], a planar homographic occupancy constraint that fuses foreground likelihood information from multiple views to resolve occlusions and localize people on a reference scene plane is developed. For better performance, this process is extended to multiple planes parallel to the reference plane in the framework of plane to plane homography. The formulation of likelihood function in this approach ($p(y|x)$ where y and x are the image observation and position of the person, respectively) is compactly described by:

$$p(y|x) = \prod_{h=1}^{H} p(y_h|x) = \prod_{h=1}^{H} \prod_{c=1}^{C} p(y_{c,h}|x) \tag{5}$$

Here, $p(y_{c,\,h}|x)$ represents the foreground likelihood information extracted from camera c and projected onto plane h, $p(y_h|x)$ represents the fused foreground likelihood information from multiple views on plane h, and finally H and C represent the number of parallel planes and cameras, respectively.

Unlike the method in [2], we have used the human detection algorithm in [38] for extracting foreground likelihood information ($p(y_{c,\,h}|x)$). The human detector outputs a probability map that represents the probable locations of people in the image plane. We project this probability map onto the ground plane (Z=0) and onto planes at different heights (Z=200, 400, …, 1600) that are parallel to the ground. Then, we combine these multi-layered projected probability maps and obtain a likelihood function for a camera view. Similar to the first tracker described in Section 3.2.1, after the fusion of likelihoods from multiple views for multiple planes, a posterior probability is obtained by combining the likelihood with a motion model and the position of people are estimated by running dynamic programming on the posterior probability. The association of observations to people is achieved in two levels: at the camera level based upon appearance (color) and at the fusion node based on motion information.

3.2.4. Decentralized version of Algorithm 2

In our framework, we evaluate the multi-layer projected probability maps ($p(y_h|x)$) in a decentralized way (Fig. 1). In [2], fusion is first performed on camera views and then on parallel homography planes. Here, we switch this order by first fusing the likelihood information on parallel planes. At each camera node, the likelihood functions obtained by the human detection algorithm [38] ($p(y_{c,\,h}|x)$) are projected on multiple parallel planes and combined to obtain the single-view likelihood function:

$$p(y_c|x) = \prod_{h=1}^{H} p(y_{c,h}|x) \tag{6}$$

Then, this likelihood is sent to the fusion node. In the fusion node, the likelihoods are fused on camera views to obtain the multi-view likelihood function:

$$p(y|x) = \prod_{c=1}^{C} p(y_c|x) \quad (7)$$

Using the multi-view likelihood function and the motion model, the position of people in the scene are estimated at the fusion node using dynamic programming.

4. Sparse representation of likelihood functions

4.1. Overview

The bandwidth required for sending local likelihood functions depends on the size of likelihoods (i.e., the number of "pixels" in a 2D likelihood function) and the number of cameras in the network. To make the communication in the network feasible, in [3] a block-based transform domain compression scheme is followed. In this block-based compression scheme, after each camera node performs feature extraction and obtains likelihood functions, likelihood functions are split into blocks and each block is transformed to an appropriate wavelet domain. Then, by taking only the significant coefficients, the likelihood functions are compressed and this new representation is sent to the fusion node. Here, following the great success of compressed sensing in different application areas, we propose a sparse representation framework. At each camera node we propose to represent the likelihood functions sparsely in a proper dictionary and then send this representation instead of sending the function itself. If one can find a dictionary through which the likelihood functions can be represented accurately by a small number of coefficients, then this approach would have the potential to contribute to accurate tracking with minimal use of communication resources. The main contribution of this paper is designing overcomplete dictionaries that are matched to the structure of likelihoods and building a sparse representation framework for bandwidth-efficient decentralized tracking in VSNs. Thanks to the developments in processor technology and fast solver algorithms for l_1-minimization problems, we believe sparse representation based methods, such as our approach, also have a high potential for real-world scenarios.

Mathematically, we have the following linear system:

$$y_c = A_c \cdot b_c \quad (8)$$

where y_c and b_c represents the likelihood function of the camera c (e.g., $P(T_t^c(k)|L_t^n = k)$ in Eq. (4) or $p(y_c|x)$ in Eq. (6)) and its sparse coefficients, respectively, and A_c is the overcomplete[1] dictionary matrix for camera c that represents the domain in which y_c has a sparse representation. To obtain the sparse representation of the likelihood function, at each camera we solve the optimization problem in Eq. (9).

$$\min_{b_c} ||y_c - A_c \cdot b_c||_2 + \lambda ||b_c||_1 \quad (9)$$

Notice that in our sparse representation framework, we do not require the use of specific image features or likelihood functions. The only requirement is that the tracking method should be based on a probabilistic framework, which is a common approach for modeling the dynamics of humans. Hence, our framework is a generic framework that can be used with many probabilistic tracking algorithms in a VSN environment. In fact, we have applied our framework using two different tracking algorithms (Section 3.2) and shown the tracking results in Section 5.

At the fusion node, likelihood functions of each camera can be reconstructed simply by multiplying the new representation with the matrix A_c. In general, this may require an initialization step to decide the sparsifying space (A_c) at each camera that is matched with the task of interest and to send all dictionary matrices to the fusion node. In the next subsection, we go through the question of how one can design the representation dictionary A_c in Eq. (8).

4.2. Designing overcomplete dictionaries

In [3], it has been shown that the block-based compression approach can be used to reduce bandwidth. However, this approach does not offer a natural representation for likelihood functions, as it does not exploit the specific spatial structure exhibited by these functions. The block-based processing breaks the global structure of the likelihoods. For these reasons, we need a special dictionary that is matched to the structure of the likelihood functions. A well-known approach to obtain such a dictionary is to learn from training data [39,40]. However, such a learning-based approach is mostly used in signal/image denoising or restoration problems and the training data consist of signal/image patches [40]. In the problem of tracking, Aharon et al. [39] would treat likelihoods as arbitrary images and would not properly exploit the knowledge on the structure of the likelihood functions. Hence, it might not be perfectly matched for parsimonious representation of likelihoods, for the objective of reducing the bandwidth. For this reason, rather than a generic learning-based method, we follow a different approach by building a dictionary exploiting the geometry of the measurement scenario. In particular, for an extremely parsimonious representation, we assert that natural atoms of the dictionary for each camera view would be likelihood functions generated by targets at each possible location in the scene.

The likelihood functions we obtain from the color model in [1] have a special structure. As it has been explained in Section 3.2.1, the color model likelihood functions for a person of interest are obtained by comparing the color histogram of rectangular patches in the foreground image and the color distribution of the person of interest. Hence, a similarity score is obtained for each grid cell on the ground plane. In Fig. 2, two sample foreground images and the likelihood function obtained from these foreground images are shown. Fig. 2(a) and (c) show foreground images captured from two different camera views when there is only one person in the scene and when the scene is crowded, respectively. The likelihood functions obtained from these image are shown in Fig. 2(b) and (d). Here, x and y coordinates represent the grid cell coordinates on the ground plane. We can clearly see that likelihood functions consist of quadrilateral-shaped components. A person in the scene creates a quadrilateral-shaped component in the likelihood function. One of the important properties of these components is that their shape do not depend on the value of the foreground pixels. The values inside the quadrilateral change according to the color pixel intensities in the foreground image. But the shape of the quadrilateral only depends on the camera view and the position of the foreground pixels. For this reason, we can say that these quadrilateral-shaped components are building blocks of likelihood functions. By creating a dictionary from these building blocks, we can naturally and properly exploit the structure of the likelihood functions.

As we have mentioned above, the scale and orientation of the quadrilateral depends on the camera view and the position of the foreground pixels. In order to find all the building blocks of likelihood functions, we need to create likelihood functions from all the possible foreground images. Similar to a 2D Dirac delta function, we create a foreground image that is all-black except a single white pixel and obtain a likelihood function from this image (Fig. 3). By changing the position of the white pixel and obtaining the likelihood function from that foreground image, we can create a pool composed of building blocks. For each camera, we create the dictionary matrix (A_c in Eq. (8)) by arranging the building blocks of likelihoods as the columns of

[1] The number of columns is bigger than the number of rows.

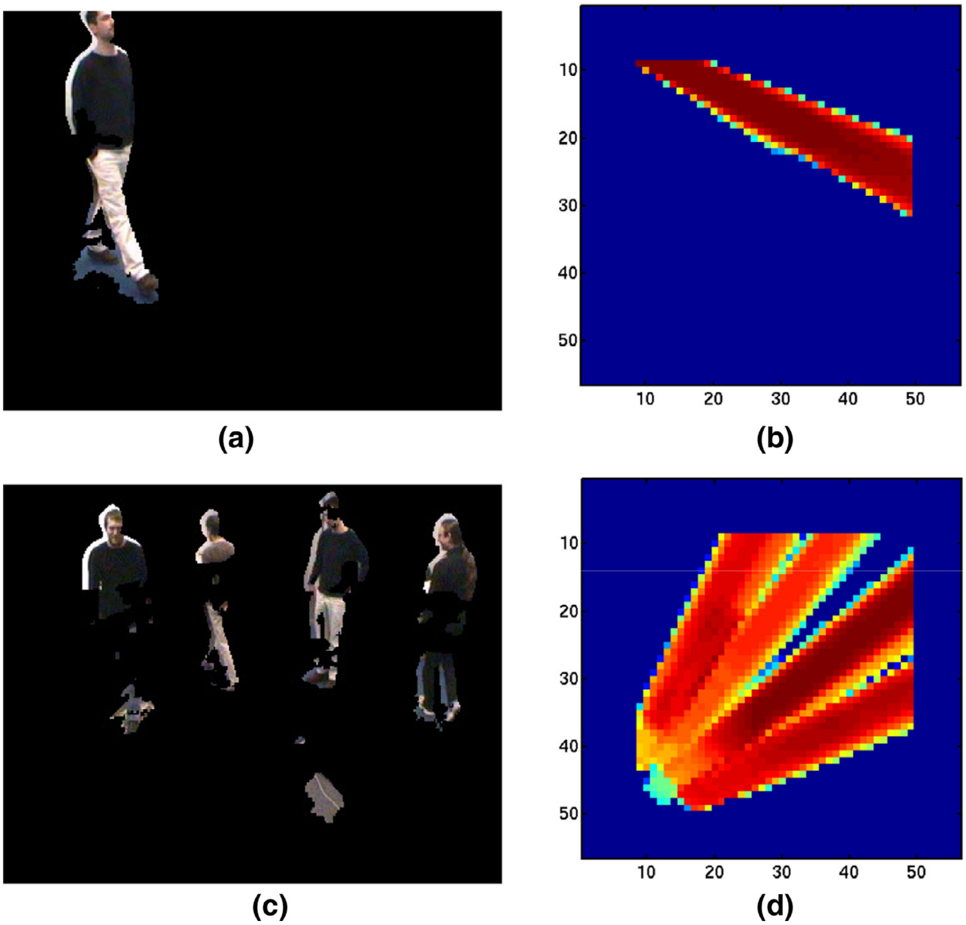


Fig. 2. Foreground images captured from two different camera views (a) when there is only one person in the scene, (c) when the scene is crowded and (b) and (d) color model likelihood functions obtained from the images based on the approach in [1].

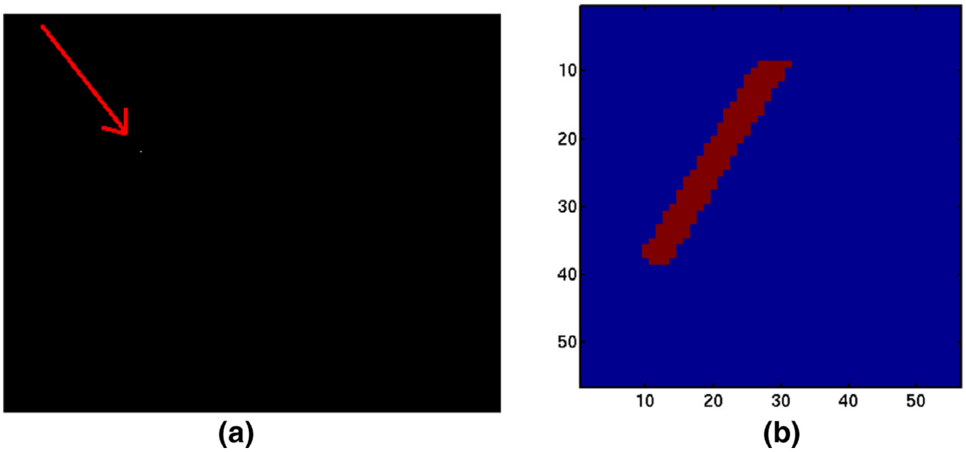


Fig. 3. (a) A sample foreground image that is all-black except a white pixel (pointed with a red arrow) and (b) the likelihood function obtained from this foreground. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the matrix. Note that dictionaries constructed in this manner depend on the geometry of the observation scenario. Hence, our dictionaries naturally adapt to and exploit the geometry of the sensing scenario.

In the tracking method [2], the likelihood functions of each camera view are obtained by fusing the projection of foreground maps obtained by the human detection step on parallel planes. A sample foreground image and a foreground map obtained from this foreground image are shown in Fig. 4. Again, we can observe that a person in the scene creates a quadrilateral-shaped component in the projected likelihood function. In order to find the building blocks of these likelihood functions, we imitate a person in the scene by setting a 100 × 30 rectangular patch in an all-zero likelihood function and projecting this likelihood onto the ground plane (Fig. 5). Similar to the procedure above, as we shift this rectangular patch, we can create a pool of building blocks and consequently build the dictionary.

4.3. Comparison of solvers for l_1-minimization

Solving optimization problems with l_1 constraints has become a well-established research area. Many solvers have been proposed for

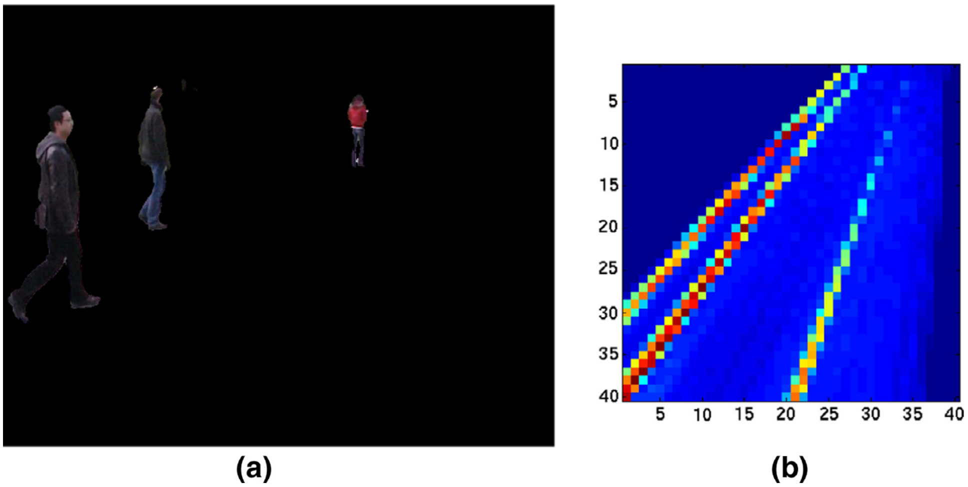


Fig. 4. (a) A foreground image and (b) likelihood function obtained from the image based on the approach in [2].

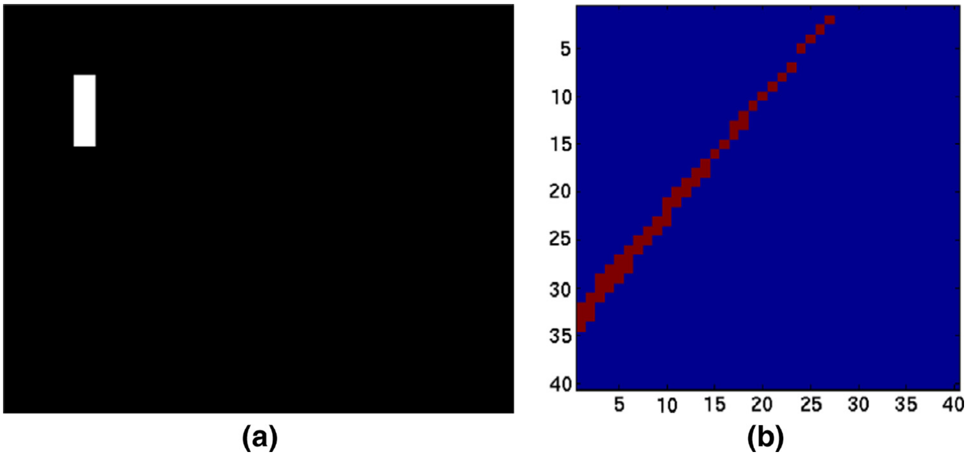


Fig. 5. (a) A sample foreground map that is all-black except a white 100 × 30 rectangular patch and (b) the likelihood function obtained from this foreground.

l_1-minimization. In [41], popular solvers have been compared in order to find the fastest solver to be used in sparsity-driven face recognition. Following this work, we have compared these solvers in order to choose a solver that is fast and accurate for our sparse representation framework.

In the comparison, we have used 20 different likelihood functions obtained from the tracker in [1] as a test set. Our specially designed dictionaries, that have been created using the approach described in Section 4.2, have been used as the A_c matrix in Eq. (9). Homotopy [42], L1LS [43], SpaRSA [44], FISTA [45], and ALM [41] algorithms have been used to solve the optimization problem in Eq. (9) and find the sparse representation of likelihood functions in the test set. The regularization parameter in Eq. (9), λ, is set to several values: 0.1, 0.5, 1, 10, 100, 200. The average run-times of the solvers in seconds for different λ values are shown in Table 1. To avoid trivial solutions, we have also checked the number of iterations of each algorithm . The average iteration counts of solvers for different λ values are given in Table 2.

We can see that ALM algorithm for λ values between 0.1–10 and SpaRSA algorithm for $\lambda = 200$ are the fastest solvers. But, average iteration counts of ALM and SpaRSA show that they find the trivial solution. When we look at the Homotopy algorithm, we can see that, independent of the λ parameter, Homotopy algorithm works fast and does not give the trivial solution. We have observed that selecting $\lambda = 0.1$ enforces sufficient level of sparsity to achieve reasonable results. Therefore, we select the Homotopy algorithm and set $\lambda = 0.1$ in our tracking experiments.

5. Experimental results

This section contains results of a set of experiments testing the performance of our approach in various tracking scenarios and comparing it with several existing techniques proposed for tracking applications in VSNs. In Section 5.1, we have demonstrated the use of our approach within the framework of the tracking algorithm in [1] and presented comparisons of our method with a block-based compression framework in [3], a decentralized method [4] and a centralized approach. In Section 5.2, we have demonstrated the use of our approach within the framework of the tracking algorithms in [1,2] and we have presented the comparisons of our approach with distributed methods in [5,6]. In all experiments, the first camera node is assumed to be the most powerful node and selected as the fusion node.

5.1. Comparison with a block-based compression framework

In this subsection, we present experimental results based on the tracking framework in [1]. We have compared our method with a block-based compression framework [3], a traditional centralized approach of compressing raw images, and a decentralized method in which, similar to [4], a Kalman filter is used in the fusion node to estimate the position of a person in the scene using the observations coming from cameras. In the centralized approach, after the raw images are acquired by the cameras, similar to JPEG compression, each color channel in each image is compressed and sent to the central node. In the central node, features are extracted from the

Table 1
Average run-times of solvers in seconds for different regularization parameters.

	$\lambda = 0.1$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 10$	$\lambda = 100$	$\lambda = 200$
Homotopy [42]	0.60133	0.64638	0.62925	0.62143	0.49892	0.32343
L1_LS [43]	816.1898	134.1769	81.6986	30.0675	13.7835	12.8253
SpaRSA [44]	24.695	22.9193	22.1939	21.7195	22.0352	0.063627
FISTA [45]	636.0806	381.3541	343.2617	206.3525	130.385	131.4214
ALM [41]	0.085406	0.078551	0.079087	0.078963	6.8105	127.8755

Table 2
Average iteration counts of solvers for different regularization parameters.

	$\lambda = 0.1$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 10$	$\lambda = 100$	$\lambda = 200$
Homotopy [42]	15.4792	15.4792	15.4792	15.1042	8.6354	1.3125
L1_LS [43]	20.8333	17.9479	18.8542	21.1979	20.5104	21.5833
SpaRSA [44]	833.5938	848.9271	833.5729	813.4583	826.0208	0.5
FISTA [45]	269.9688	144.4583	121.9375	47.7917	7.8542	4.75
ALM [41]	2	2	2	2	206.0313	3498.6563

reconstructed images and tracking is performed using the method in [1]. In this decentralized method, after likelihood functions are computed, each camera sends the peak point of the distribution to the fusion node as observation.[2] In the fusion node, the observations of each camera are spatially averaged and using the average position as its observation, a Kalman filter is applied to estimate the position of the person on the ground plane. The positions of all people in the scene are estimated by running an individual Kalman filter for each person.

5.1.1. *Setup*

In the experiments, we have simulated the VSN environment by using the indoor and outdoor multi-camera dataset in [1]. The indoor dataset consists of a video sequence of four people sequentially entering a room and walking around. The sequence was shot by four synchronized cameras that were located at each corner of the room. In this sequence, the area of interest was discretized into $G = 56 \times 56 = 3136$ locations. The outdoor dataset was shot in a university campus and it includes up to four individuals appearing simultaneously. This sequence was shot by three synchronized cameras. The area of interest for this sequence was discretized into $G = 40 \times 40 = 1600$ locations. For the correspondence between camera views and the top view, the homography matrices provided with the dataset are used. The size of the images are 360 × 288 pixels and the frame rate for all of the cameras is 25 fps.

Using the procedure described in Section 4.2, we have created the dictionaries for each view. For the indoor dataset, we end-up with dictionaries with 36,073, 46,986, 28,155 and 30,195 atoms for the first, second, third and fourth view, respectively. Some elements of these dictionaries are presented in Fig. 6. For the outdoor dataset, we end-up with dictionaries with 12,777, 11,984, and 19,846 atoms for the first, second, and third view, respectively.

Following our observations in Section 4.3, we have solved the optimization problem using the Homotopy algorithm [42] with λ set to 0.1 for all dictionaries.

5.1.2. *Indoor tracking results*

In this subsection, we present the performance of our method used for indoor multi-person tracking and compare it with the block-based compression approach of [3], our implementation of the decentralized approach in [4] and a traditional centralized approach of compressing raw images. For the block-based compression framework and the centralized approach, we use DCT for compression with a block size of 8 × 8 with several levels of compression, taking 1, 2, 3, 4, 5, 10, and 25 most significant coefficient(s) per block. Consequently, with 56 × 56 likelihoods, at each camera in total we end up with at most 49, 98, 147, 196, 245, 490 and 1225 coefficients per person. Since there are four individuals in the scene at most, each camera sends at most 196, 392, 588, 784, 980, 1960 and 4900 coefficients. In the centralized approach we compress three color channels of each image. Since the images used in this experiment are composed of 360 × 288 pixels, at each camera we end up with 4860, 9720, 14,580, 19,440, 24,300, 48,600 and 121,500 coefficients. In our method, after sparse representation of color model likelihood of a person of interest is found, we considered transmission of 10, 15, 20, 25, 50 and 100 most significant coefficients. Since there are four individuals in the scene at most, each camera ends up sending at most 40, 60, 80, 100, 200 and 400 coefficients to the fusion node. In the decentralized method that uses a Kalman filter, for each person, each camera sends only two points, namely the 2D position of the peak point, to the fusion node. In total, we end up with eight points in maximum for four individuals.

A ground truth for this sequence is obtained by manually marking the people in the ground plane, in intervals of 25 frames. Tracking errors are evaluated via Euclidean distance between the tracking and manual marking results (in intervals of 25 frames). We have also used two evaluation metrics [46]: Returning-Fragments (R-Frag) and Returning-ID Switches (R-IDS). R-Frag corresponds to the total number of times that there is a link between two trajectories which represents a person leaving and returning to the scene in groundtruth, but there is no link in the tracking result. R-IDS is the total number of times that there is no link between two trajectories which means they represent different people in groundtruth, but there is a link in the tracking results. Table 3 shows the average of tracking errors over all people, and the R-Frag and R-IDS metrics obtained for each approach. Since people do not exit the scene in this sequence, R-Frag metric is zero for all methods. Note that the actual number of significant coefficients sent by a camera at each time point depends on the number of people in the scene at that moment. The number of significant coefficients shown in Table 3 is computed based on the worst case assumption of the presence of four people in the scene all the time. So this is actually an upper bound on the number of coefficients that will be sent by each camera at each time point. Note this is handled in exactly the same way for all methods, so our comparison is fair. Table 3 also presents the bandwidth requirement of each method in Kbps calculated using a 32-bit precision for each coefficient value. Fig. 7 presents the average tracking errors versus the total number of significant coefficients used in communication for all methods.

[2] Previously, the word “observation” was used to refer to the data acquired by cameras. Here, we use it as the information, that is obtained by feature extraction at the camera nodes, shared by cameras to be used as "data" by the tracker.

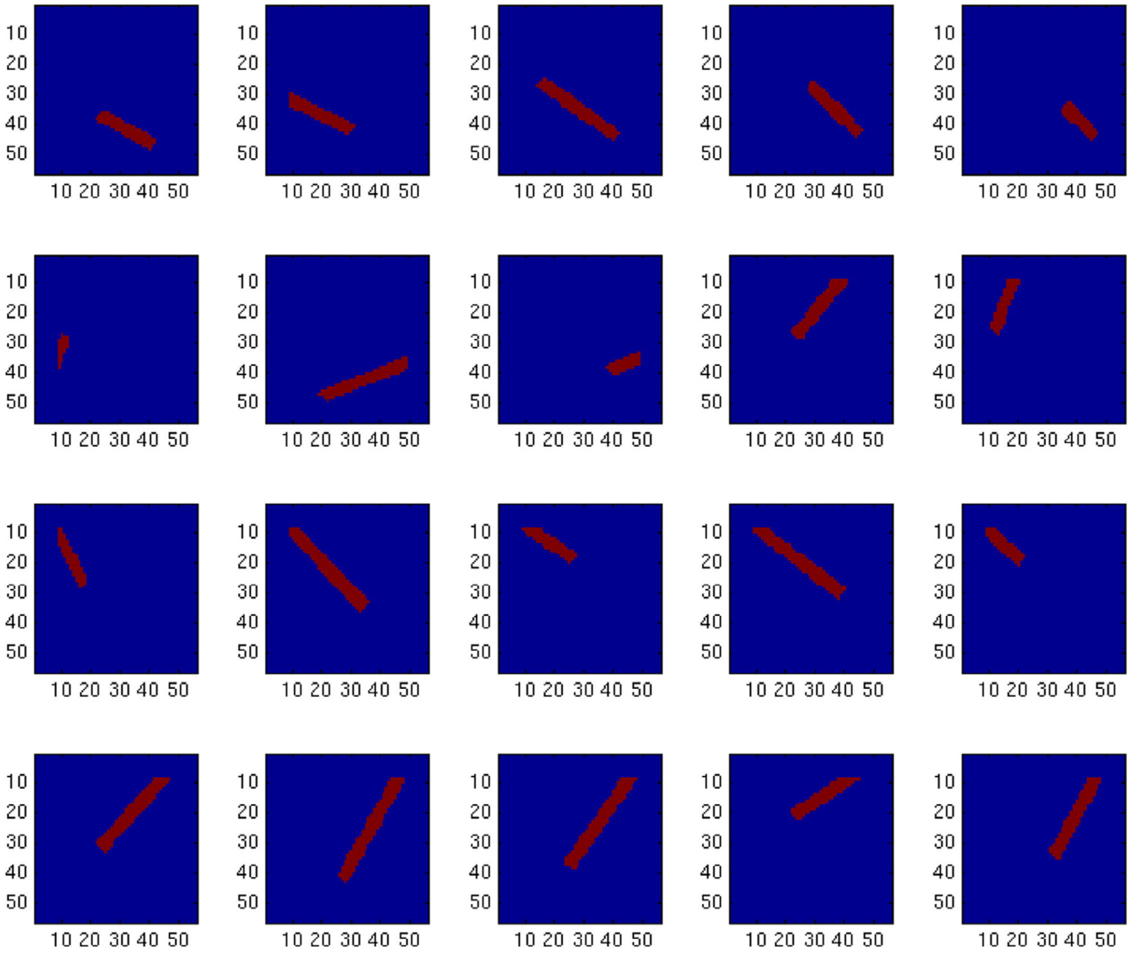


Fig. 6. Each row presents some elements of the dictionary of each view created for the indoor dataset by the procedure described in Section 4.2.

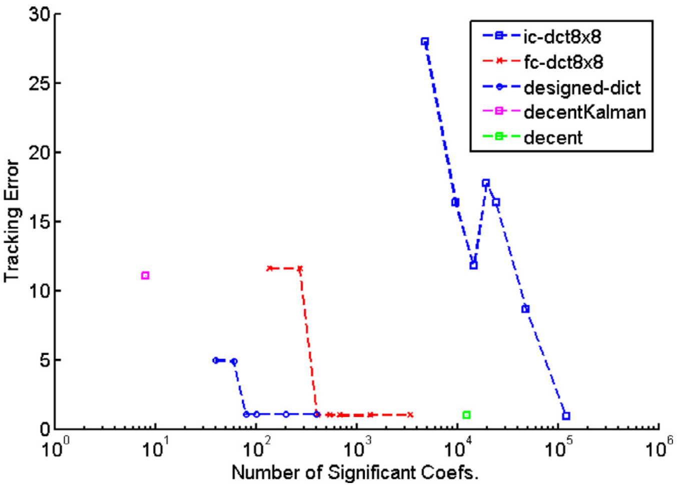


Fig. 7. Indoor sequence: The average tracking errors vs. the number of coefficients for the centralized approach (blue square), the block-based compression framework in [3] (red), our sparse representation framework (blue circles), the decentralized Kalman approach (purple) and a decentralized method (green) that directly sends the likelihood functions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

It can be clearly seen that by using custom-designed dictionaries, our sparse representation framework achieves much more bandwidth reduction than the block-based compression framework and the centralized method. To achieve an error of 1 pixel in the grid on average and zero ID switch (R-IDS), our sparse representation framework using custom-designed dictionaries needs at least 20 coefficients per person, whereas the block-based compression framework needs at least 147 coefficients per person. The centralized approach is not capable of decreasing the communication without affecting the tracking performance. It needs at least 121,500 significant coefficients in total. By using the decentralized Kalman approach, we can obtain a huge reduction in communication, but we cannot perform robust tracking. R-IDS metric shows that in 144 frames the people are misassociated. Our framework is also advantageous over an ordinary decentralized approach that directly sends likelihood functions to the fusion node. In such an approach, we send each data point in the likelihood function, resulting in the transmission of 3136 values per person. It can be seen that our approach can significantly reduce the amount of communication in the network as compared to this approach while achieving the same level of tracking accuracy.

The tracking results of the block-based compression framework using 49 coefficients per person, the decentralized Kalman approach, and our sparse representation framework using 20 coefficients per person are given in Figs. 8, 9, and 10, respectively. It can be seen that, although the block-based compression approach can track the first and the second individuals very well, there is an identity association problem for the third and fourth individuals. The decentralized Kalman approach fails to track the people in the scene. Nearly for all people, there occurs identity association problems. In some frames, it loses the track of the person and starts tracking a virtual person in the scene (frame no. 1173 in Fig. 9(b)). These failures occur because the amount of information coming from cameras is not enough to

Table 3
Indoor sequence: Tracking results of (a) a centralized approach that sends compressed images to central node ("IC"), (b) the block-based compression framework in [3] ("FC"), (c) the decentralized Kalman approach ("DecentKalman") and a decentralized method that directly sends the likelihood functions ("Decent"), and (d) our sparse representation framework for several levels of compression.

	Number of coefficients	BW (Kbps)	Tracking error	R-Frag	R-IDS
		a			
IC	4860	3888	27.9838	0	162
IC	9720	7776	16.3918	0	164
IC	14,580	11,664	11.7918	0	139
IC	19,440	15,552	17.7786	0	183
IC	24,300	19,440	16.3503	0	183
IC	48,600	38,880	8.6414	0	82
IC	**121,500**	**97,200**	**0.9508**	**0**	**0**
		b			
FC	136	108.8	11.5734	0	76
FC	272	217.6	11.5734	0	76
FC	**408**	**326.4**	**1.0092**	**0**	**0**
FC	544	435.2	1.0092	0	0
FC	680	544	1.0092	0	0
FC	1360	1088	1.0207	0	0
FC	3400	2720	1.0207	0	0
		c			
Decent	**12,544**	**10,035.2**	**1.0207**	**0**	**0**
DecentKalman	8	6.4	11.0319	0	144
		d			
Designed-Dict	40	32	4.984	0	23
Designed-Dict	60	48	4.9246	0	23
Designed-Dict	**80**	**64**	**1.0367**	**0**	**0**
Designed-Dict	100	80	1.0367	0	0
Designed-Dict	200	160	1.0367	0	0
Designed-Dict	400	320	1.0367	0	0

Bold lines represent the best tracking and bandwidth reduction performance.

perform robust tracking. In [4], the decentralized Kalman filtering approach has only been tested on a simple scenario that involves tracking people using cameras mounted on the ceiling, and the approach does not perform very well on the more challenging tracking scenario we consider in this paper. In Fig. 10, we observe that all people in the scene can be tracked very well by our sparse representation framework. All methods suffer from an error for the third person around the 700th frame, because tracking starts a few frames after the third person enters the room. When the number of coefficients taken per person is fewer than 20, since the quality of likelihoods decreases, we also observe identity problems. We can also observe this in Fig. 11 which shows average PSNR values between the original and reconstructed likelihoods versus the number of coefficients. However, by selecting the number of coefficients per person greater than or equal to 20, we can track all the people in the scene accurately. The block-based compression framework, in total, requires at least five times more coefficients to achieve this level of accuracy.

In the light of the results we obtained, for the same tracking performance, our sparse representation based method saves 80.39% of the bandwidth used by the block-based compression approach and uses only 0.06% of the bandwidth required by the centralized approach. As compared to the ordinary decentralized approach transmitting full likelihood functions, our approach saves 99.37% of the bandwidth, while achieving the same level of tracking accuracy.

5.1.3. Outdoor tracking results

The performance of our sparse representation based method for outdoor multi-person tracking is presented in this subsection. Again, we have compared our sparse representation framework with the block-based compression framework in [3] and the centralized approach using DCT domain with a block size of 8 × 8 and the decentralized Kalman approach in [4]. For the block-based compression approach, we again considered several levels of compression, taking 5, 10, 15, 20, 30, and 50 most significant coefficient(s) per block. Consequently, with 40 × 40 likelihoods, at each camera in total we end up with at most 125, 250, 375, 500, 750 and 1250 coefficients per person. Since there are four individuals in the scene at most, each camera sends at most 500, 1000, 1500, 2000, 3000, and 5000 coefficients. For the centralized approach, we considered taking only 1, 2, 3, 4, 5, 10, and 25 most significant coefficient(s) per block. Hence, at each camera we end up with 4860, 9720, 14,580, 19,440, 24,300, 48,600 and 121,500 coefficients. In our method, after sparse representation of color model likelihood of a person of interest is found, we considered transmitting 5, 10, 15, 20, 25, 50 and 100 the most significant coefficients. Since there are four individuals in the scene at most, each camera ends up sending at most 20, 40, 60, 80, 100, 200 and 400 coefficients to the fusion node. As mentioned in the previous section, in the decentralized Kalman approach, we end up sending at most eight points for four individuals.

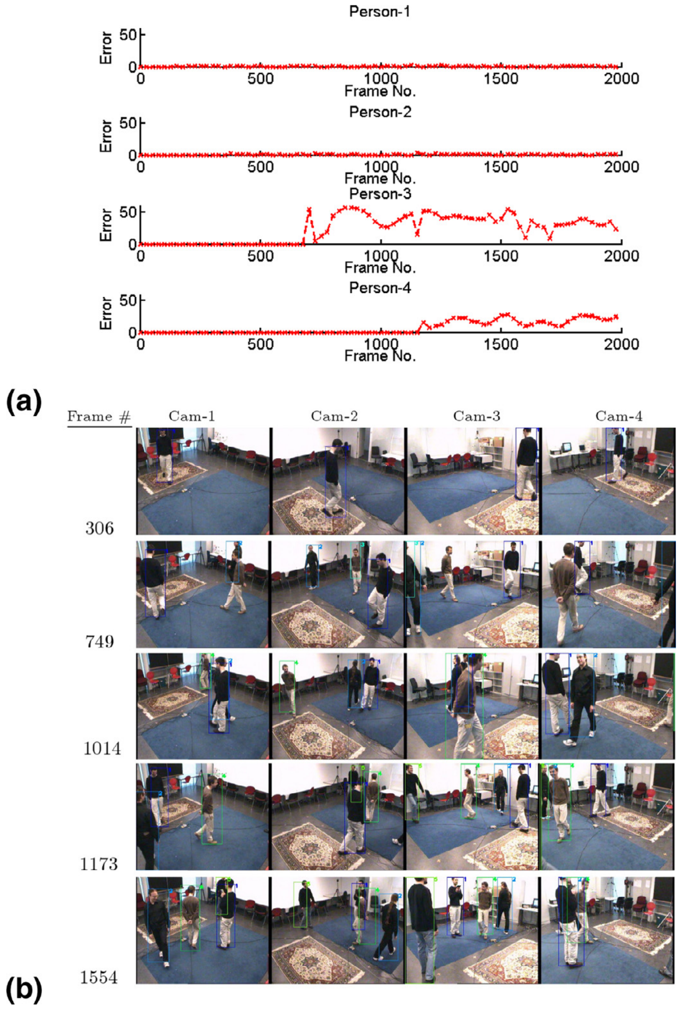


Fig. 8. (a) The tracking errors for each person and (b) tracking results for the indoor dataset obtained by the block-based compression framework in [3] using 49 coefficients per person used in communication.

As in the indoor sequence, tracking errors are evaluated via the Euclidean distance between the tracking and manual marking results. We have also computed the R-Frag and R-IDS metrics. Table 4 shows the average of tracking errors over all people, and the R-Frag and R-IDS metrics obtained for our sparse representation framework. Fig. 12 presents the average of tracking errors over all people versus the total number of significant coefficients used in communication for our approach. The performance of the block-based compression framework, the decentralized Kalman approach and the centralized approach

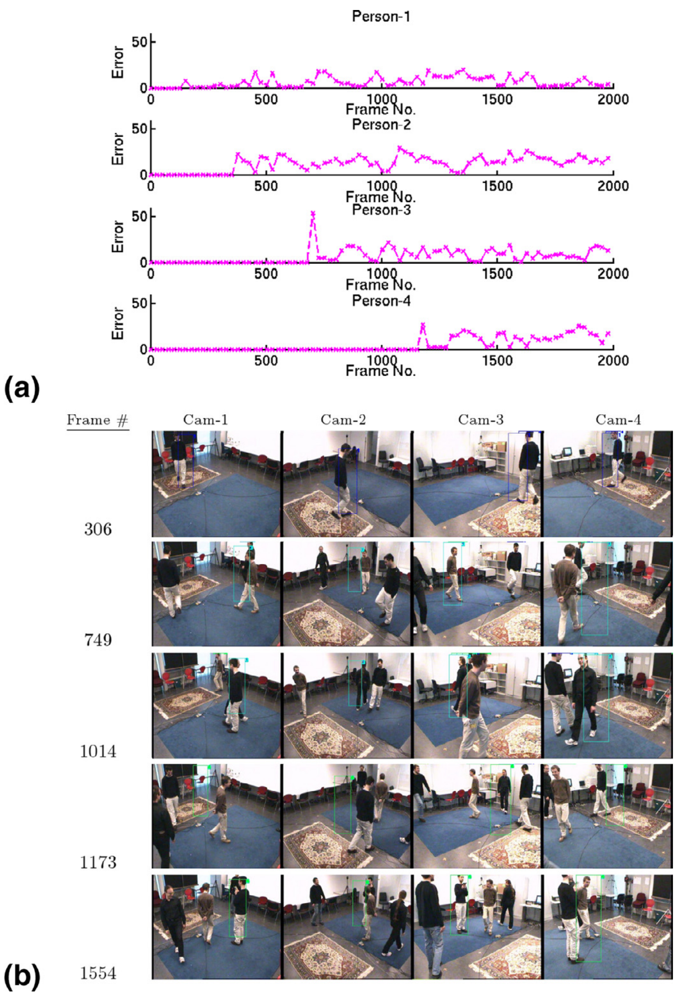


Fig. 9. (a) The tracking errors for each person and (b) tracking results for the indoor dataset obtained by the decentralized Kalman approach.

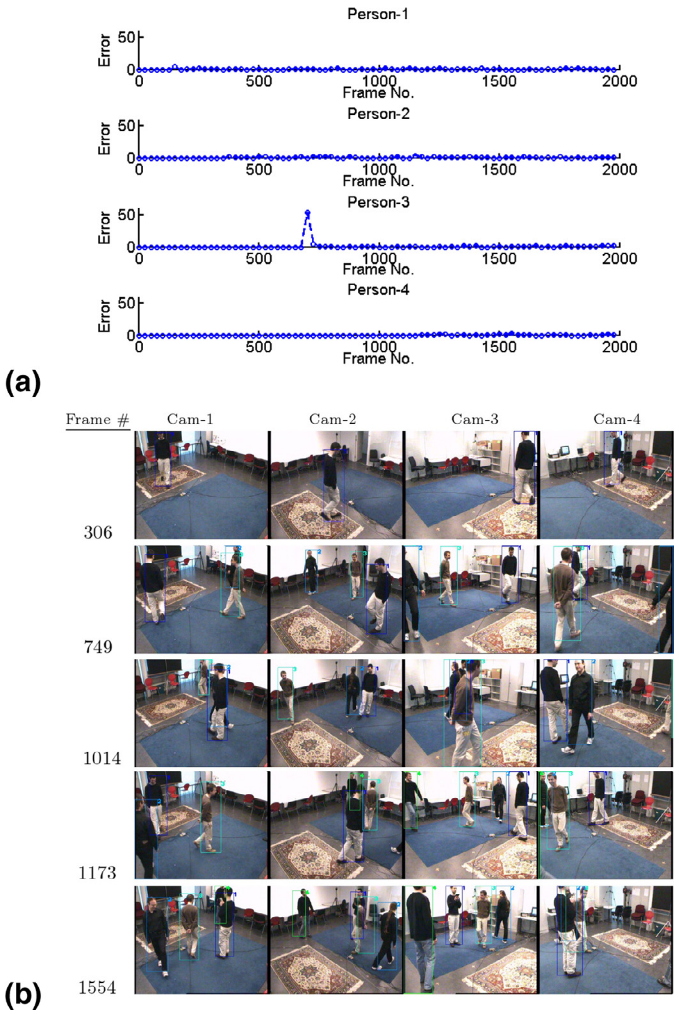


Fig. 10. (a) The tracking errors for each person and (b) tracking results for the indoor dataset obtained by our sparse representation framework using 20 coefficients per person used in communication.

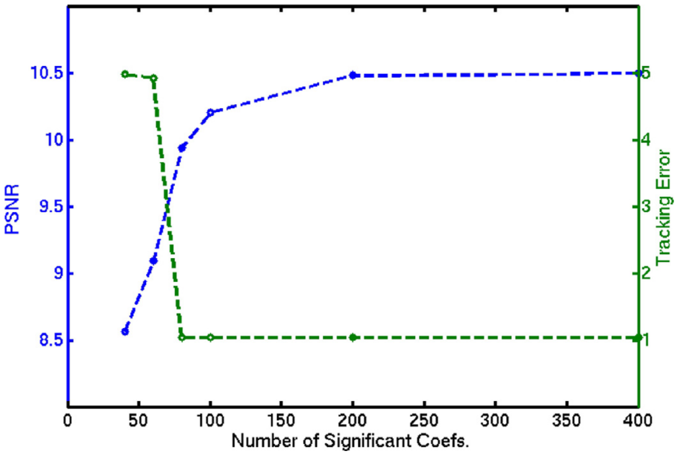


Fig. 11. Indoor sequence: The average PSNR vs. the number of coefficients for our sparse representation framework together with average tracking errors.

are also presented in Table 4 and Fig. 12. It can be clearly seen that our sparse representation framework works better in decreasing the communication than the block-based compression framework. The block-based compression framework requires at least 375 coefficients per person to achieve an error of 2 pixels in the grid on average and minimum number of ID switches. The R-IDS metric results show that, using fewer coefficients with this approach causes identity association problems. On the other hand, by using our sparse representation framework, we achieve a similar tracking error and even less number of ID switches with 10 coefficients per person. The decentralized Kalman approach enables a huge reduction in the communication, but cannot perform robust tracking. The R-IDS metric results show that people are misassociated in 124 frames. For the centralized approach, at least 121,500 coefficients are required to achieve robust tracking. Our framework is also advantageous over an ordinary decentralized approach that directly sends each data point in the likelihood functions to the fusion node. Such an approach requires sending 1600 values per person. The performance of this approach is also shown in Table 4 and Fig. 12. It can be seen that we can achieve the same level of tracking accuracy as the ordinary decentralized method while significantly decreasing the communication in the network.

The tracking results of the block-based compression framework using 250 coefficients per person, the decentralized Kalman approach, and our sparse representation based approach with custom-designed dictionaries using 10 coefficients per person are given in Figs. 13, 14, and 15, respectively. It can be seen that, block-based compression fails to preserve identities with this level of compression. In particular, when a person leaves the scene and comes back, the person cannot be recognized and he or she is considered as a new person in the scene. For the decentralized Kalman approach, nearly for all people, there occurs identity association problems. Usually, it loses the track of the person and starts tracking a virtual person in the scene (Fig. 14(b)). As in the indoor tracking results presented

Table 4
Outdoor sequence: Tracking results of (a) a centralized approach that sends compressed images to central node ("IC"), (b) the block-based compression framework in [3] ("FC"), (c) the decentralized Kalman approach ("DecentKalman") and a decentralized method that directly sends the likelihood functions ("Decent"), and (d) our sparse representation framework for several levels of compression.

	Number of coefficients	BW (Kbps)	Tracking error	R-Frag	R-IDS
		a			
IC	4860	3888	18.0632	2	79
IC	9720	7776	12.7814	2	79
IC	14,580	11,664	8.5199	1	47
IC	19,440	15,552	7.8093	2	45
IC	24,300	19,440	7.7203	2	43
IC	48,600	38,880	7.8957	2	47
IC	**121,500**	**97,200**	**3.0881**	**1**	**8**
		b			
FC	575	460	8.58	2	35
FC	1150	920	8.6179	2	34
FC	**1725**	**1380**	**2.1823**	**1**	**12**
FC	2300	1840	2.1823	1	12
FC	3450	2760	2.1823	1	12
FC	5750	4600	1.9346	1	4
		c			
Decent	**6400**	**5120**	**1.9346**	**1**	**4**
DecentKalman	8	6.4	23.3423	2	124
		d			
Designed-Dict	20	16	3.1662	1	9
Designed-Dict	**40**	**32**	**1.9432**	**1**	**3**
Designed-Dict	60	48	2.1819	1	12
Designed-Dict	80	64	1.9341	1	4
Designed-Dict	100	80	1.9406	1	4
Designed-Dict	200	160	1.9406	1	4
Designed-Dict	400	320	1.9406	1	4

Bold lines represent the best tracking and bandwidth reduction performance.

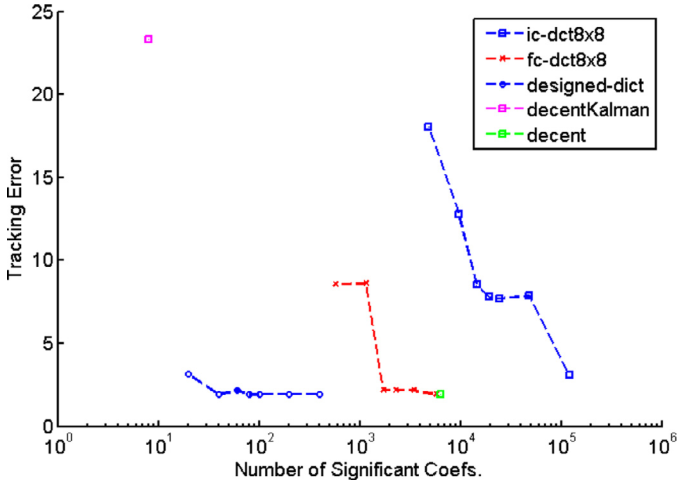


Fig. 12. Outdoor sequence: The average tracking errors vs. the number of coefficients for the centralized approach (blue square), the block-based compression framework in [3] (red), our sparse representation framework (blue), the decentralized Kalman approach (purple) and a decentralized method (green) that directly sends likelihood functions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

in the previous section, these failures occur because the amount of information coming from cameras is not enough to perform robust tracking. In Fig. 15, we observe that likelihoods can be reconstructed properly and all people in the scene can be tracked very well by our approach with custom-designed dictionaries using five times fewer coefficients. This can also be observed in the plot of PSNR values versus the number of coefficients in Fig. 16.

Based on these results, we can say that, by using the custom-designed dictionaries, our sparse representation framework successfully decreases communication load in the network without significantly degrading tracking performance. For the same tracking performance, our approach saves 97.62% of the bandwidth used by the block-based compression approach and uses only 0.03% of the bandwidth required by the centralized approach. As compared to the ordinary decentralized approach, our approach uses only 0.63% of the bandwidth needed by the decentralized approach. Hence what we propose is a bandwidth-efficient approach.

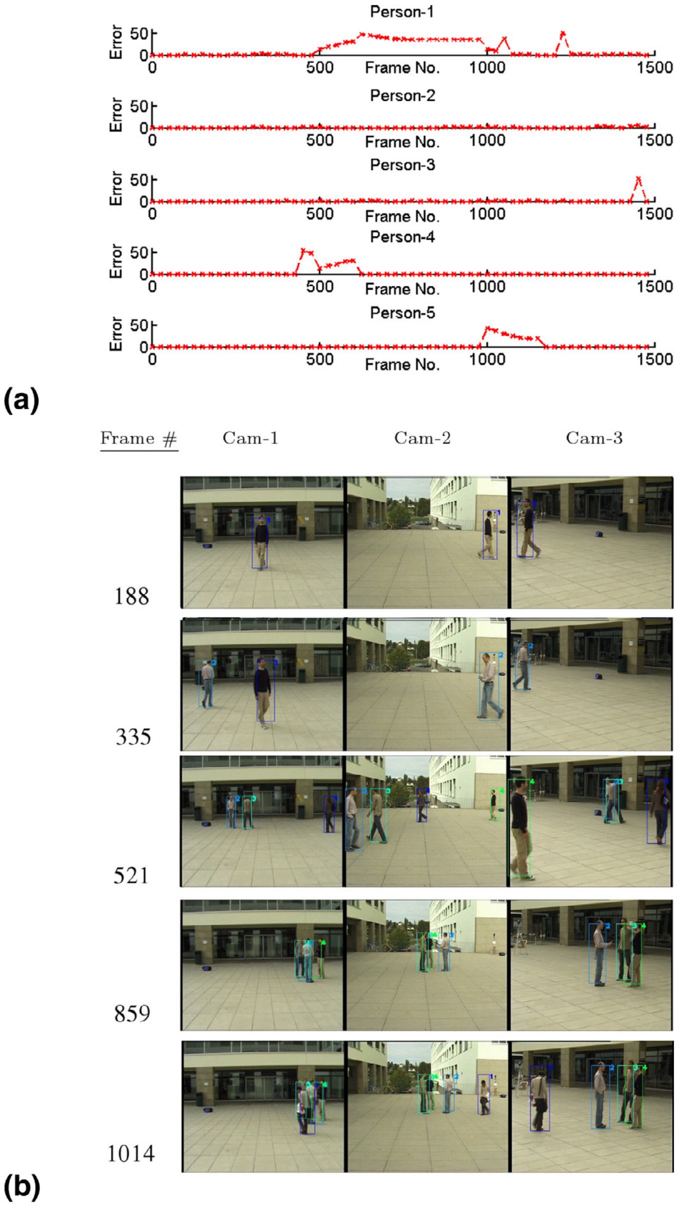


Fig. 13. (a) The tracking errors for each person and (b) tracking results for the outdoor dataset obtained by the block-based compression framework in [3] using 250 coefficients per person in communication.

5.2. Comparison with a distributed approach

In this subsection, we compare our method with a distributed approach in which each camera node fuses its observations with tracking results received from its neighbors and sends the updated estimates to the next neighbor. For this comparison, we have used the distributed tracking methods in [5,6] within the tracking frameworks in [1,2].

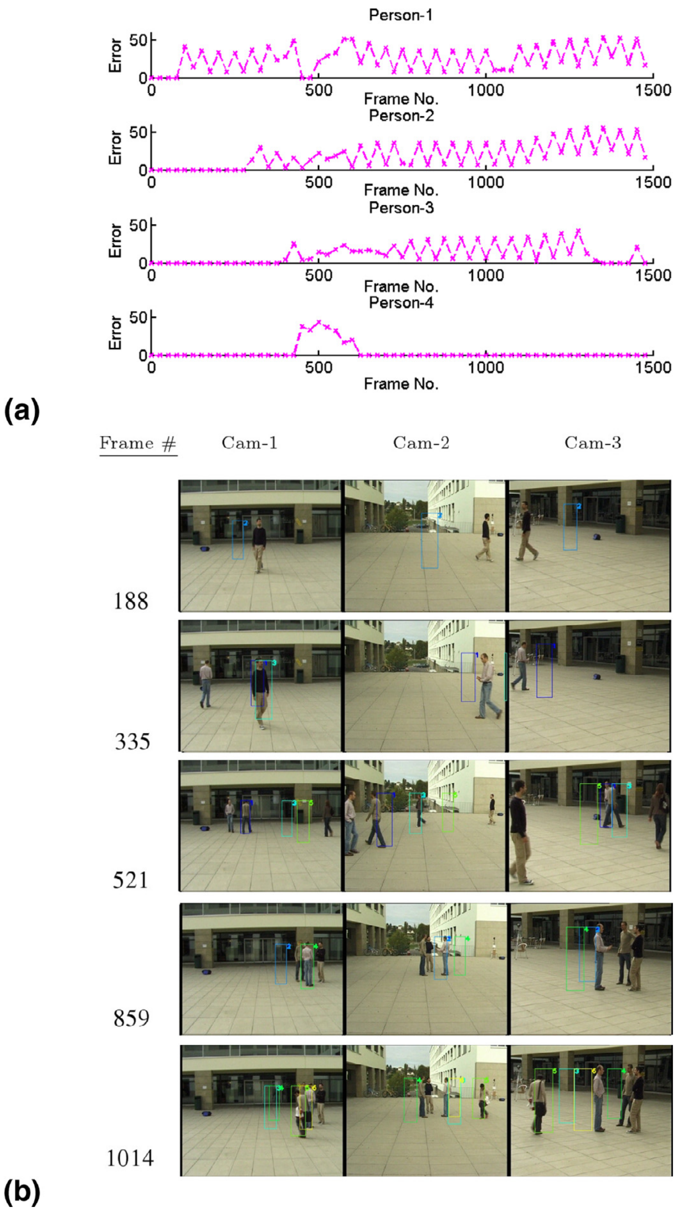


Fig. 14. (a) The tracking errors for each person and (b) tracking results for the outdoor dataset obtained by the decentralized Kalman approach.

In [5], at each camera node, the position of each person on the ground plane is estimated by applying individual Kalman–Consensus filters (KCF) [15] on its own observations together with observations and estimates coming from neighboring cameras. In [6], rather than a Kalman–Consensus filter, an information weighted consensus filter (ICF) that weights the estimates coming from neighboring cameras by their information matrix is used. The state of each person in the filters is a four-element vector representing the position and velocity in the horizontal and vertical directions on the ground plane. The observation vector of each camera is obtained by finding the local maximum points of its likelihood function ($P(T_t^c(k)|L_t^n = k)$ and $P(X_k^t = 1|\mathbf{B}_t)$ in Section 3.2.1, and $p(y_c|x)$ in Section 3.2.3). In the tracking framework of [1], the local maximum points obtained from $P(T_t^c(k)|L_t^n = k)$ and $P(X_k^t = 1|\mathbf{B}_t)$ are spatially averaged and the average position is used as the observation. At each time step, camera nodes share their observation vectors and observation covariances together with the predicted states of each person.

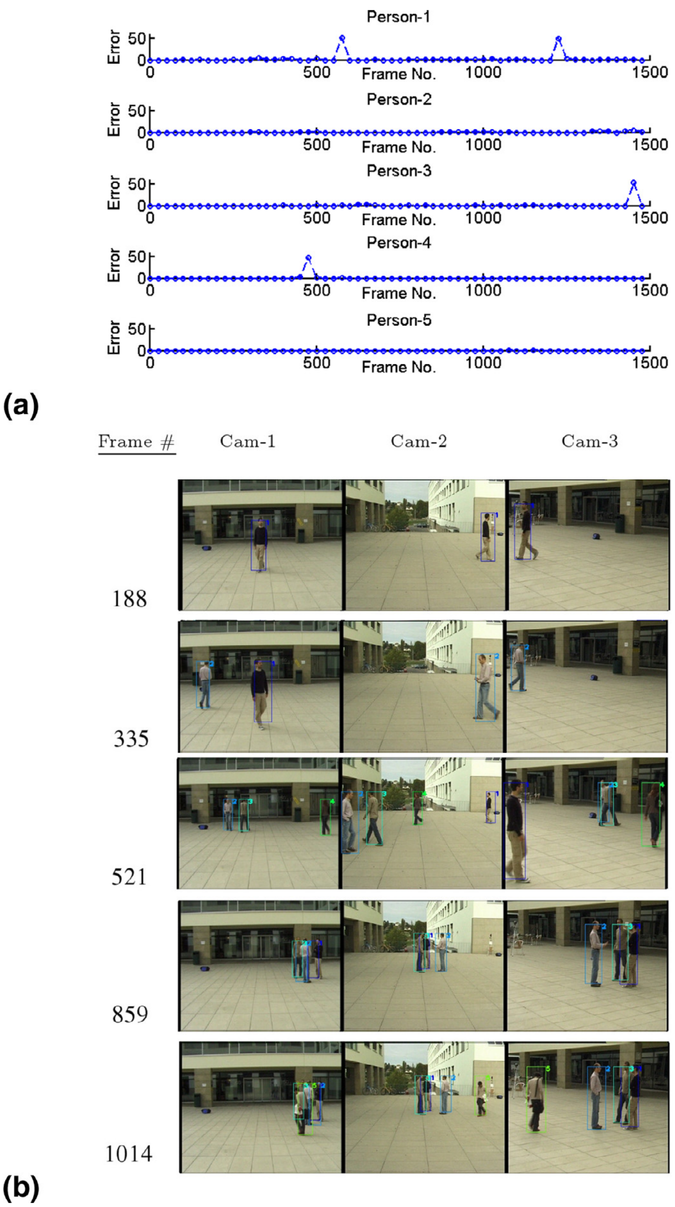


Fig. 15. (a) The tracking errors for each person and (b) tracking results for the outdoor dataset obtained by our sparse representation framework using 10 coefficients per person in communication.

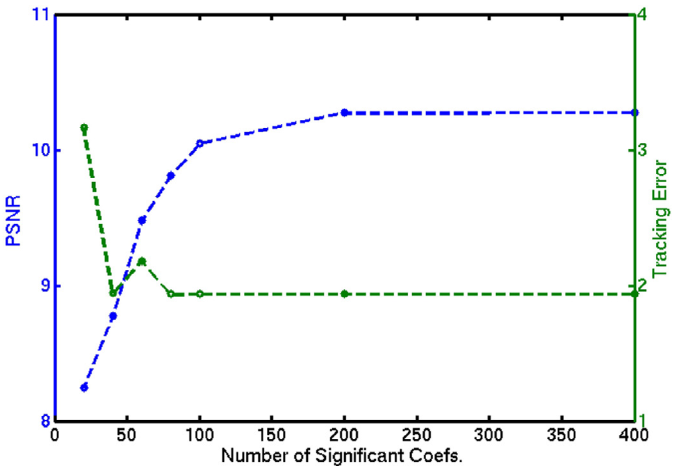


Fig. 16. Outdoor sequence: The average PSNR vs. the number of coefficients for our sparse representation framework together with average tracking errors.

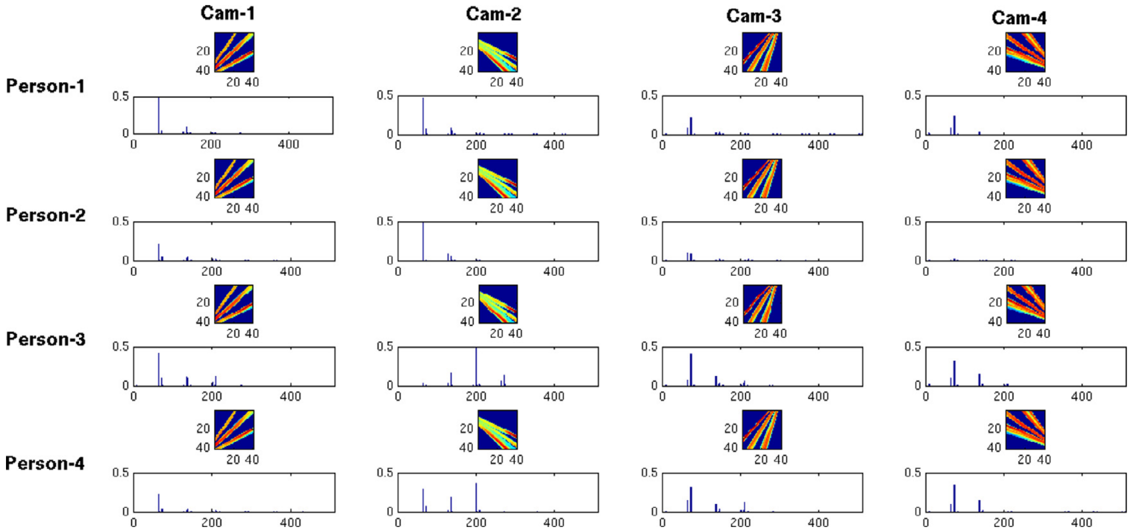


Fig. 17. Likelihoods extracted by each camera node (top) and corresponding 512-bin RGB color-histograms (bottom) for four people in the beginning of the PETS 2009 sub-sequence. Color-histograms are used for data association in the beginning of the sequence.

5.2.1. *Setup*

In the experiments, we have simulated the VSN environment by using the PETS 2009 benchmark dataset [47]. The data were collected in a university campus and it includes many people appearing simultaneously. We have used the four cameras that cover a rectangular region on the ground and a sub-sequence in which at most five people appear in the scene. The area of interest in this dataset is of size 6 m × 6 m and discretized into $G = 40 \times 40 = 1600$ locations, corresponding to a regular grid with a resolution of 15 cm. The calibration parameters for each camera are provided within the dataset. The size of the images are 720 × 576 pixels and the frame rate for all of the cameras is 7 fps.

As we have described in Section 4.2, we design a dictionary for each camera by using the building blocks of the likelihood functions. For the tracker in [2], we obtain dictionaries with 6932, 7870, 7768 and 6844 atoms for the first, second, third, and fourth view, respectively. For the tracker in [1], we obtain dictionaries with 28,380, 32,259, 28,399 and 20,505 atoms for the first, second, third, and fourth view, respectively. Again following our observations in Section 4.3, we have solved the optimization problem using the Homotopy algorithm [42] with λ set to 0.1 for all dictionaries.

In experiments within the tracker in [1], we have used a subsequence that includes four people in the beginning. In the fusion node, a color-based data association algorithm is used to match likelihoods extracted by each camera node. For the first frame of the sequence, likelihoods are sent to the fusion node together with a corresponding 512-bin RGB color histogram. In the fusion node, the histograms are matched using the intersection metric [48] and a voting procedure. Each histogram, corresponding to a likelihood function extracted for a person in each camera node, is compared with the color histograms from other nodes using the intersection metric. The ID of the closest histogram is taken as a vote for the corresponding node. Then, the most voted ID is selected as the ID of the likelihood. In order to obtain a globally consistent solution among cameras, we run the procedure for each likelihood and find the globally maximum voted ID. Fig. 17 presents the likelihood functions and the corresponding color histograms for each person at each camera node. Using this color-based data association algorithm we can correctly match the likelihoods and perform tracking in the fusion node.

5.2.2. *Tracking results*

The results of the comparison between our method and the distributed methods in [5,6] are presented in this subsection. In both distributed approaches, at each iteration of the filter, each camera shares the observation[3] vector (two element vector) and observation covariance (2 × 2 matrix) together with the predicted states (four element vector) with neighboring cameras. In our implementation, we have selected a common observation covariance at each camera, hence we save on communications by not sending this information. Consequently, to estimate the position of a person using any tracker, in total six elements are shared among cameras. Since there are four individuals in the scene at most, each camera sends at most 24 elements. In our method, after sparse representation of the likelihood of a person of interest is found, within the tracker in [2] we consider the transmission of 30, 40, 45, 50, 75, and 100 most significant coefficients and within the tracker in [1] we consider the transmission of 10, 20, 30, 40, 50, and 100 most significant coefficients. In addition, we have also compared our approach with the block-based compression framework in [3] and the centralized approach using DCT domain with a block size of 8 × 8 and the decentralized Kalman approach in [4]. For the block-based compression approach, we again considered several levels of compression, taking 20, 30, 40, 50, and 60 most significant coefficient(s) per block within the tracker in [2] and taking 1, 2, 3, 4, 5, 10, and 20 most significant coefficient(s) per block within the tracker in [1]. Within the tracker in [2], we have also tried transmitting all the coefficients which corresponds to 64 values per block. Consequently, with 40 × 40 likelihoods, at each camera in total we end up with at most 500, 750, 1000, 1250, 1500 and 1600 coefficients within the tracker in [2] and at most 25, 50, 75, 100, 125, 250, and 500 coefficients per person within the tracker in [1]. Since there are four individuals in the scene at most, each camera sends at most 100, 200, 300, 400, 500, 1000, and 2000 coefficients within the tracker in [1]. For the centralized approach within any tracker, we considered taking only 1, 2, 3, 4, 5, 10, and 25 most significant coefficient(s) per block. Hence, at each camera we end up with 19,440, 38,880, 58,320, 77,760, 97,200, 194,400 and 486,000 coefficients. For the decentralized Kalman approach, as we have mentioned previously, each camera sends eight points in maximum for four individuals.

A ground truth for this sequence is obtained by manually marking the people in the ground plane. Tracking errors are evaluated via Euclidean distance between the tracking and manual marking results. We have used the R-Frag and R-IDS metrics. Tables 5 and 6 present

[3] We refer to the information shared by cameras.

Table 5
PETS 2009 sequence using the tracker in [2]: Tracking results of (a) a centralized approach that sends compressed images to central node ("IC"), (b) the block-based compression framework in [3] ("FC"), (c) the decentralized Kalman approach ("DecentKalman"), a decentralized method that directly sends the likelihood functions ("Decent"), the distributed approach in [5] ("Distributed-KCF") and the distributed approach in [6] ("Distributed-ICF"), and (d) our sparse representation framework for several levels of compression.

	Number of coefficients	BW (Kbps)	Tracking error	R-Frag	R-IDS
		a			
IC	19,440	4354.56	11.1148	1	185
IC	38,880	8709.12	9.0774	1	197
IC	58,320	13,063.68	9.0828	1	155
IC	77,760	17,418.24	4.1325	1	116
IC	97,200	21,772.8	4.1091	1	101
IC	194,400	43,545.6	8.1468	1	169
IC	**486,000**	**108,864**	**2.9917**	**1**	**53**
		b			
FC	480	107.52	8.7572	1	163
FC	720	161.28	8.4399	1	111
FC	960	215.04	4.9828	1	80
FC	1200	268.8	8.3583	1	175
FC	1440	322.56	8.5446	1	127
FC	**1536**	**344.064**	**2.1815**	**1**	**30**
		c			
Decent	**1600**	**358.4**	**0.6056**	**1**	**30**
DecentKalman	8	1.792	23.4272	1	249
Distributed-KCF	24	16.128	16.965	1	249
Distributed-ICF	24	16.128	19.4994	0	279
		d			
Designed-Dict	15	13.44	16.6413	0	215
Designed-Dict	30	26.88	13.5002	0	186
Designed-Dict	40	35.84	10.9498	0	131
Designed-Dict	45	40.32	11.0045	1	177
Designed-Dict	**50**	**44.8**	**2.6001**	**1**	**32**
Designed-Dict	75	67.2	2.6005	1	32
Designed-Dict	100	89.6	2.6005	1	32

Bold lines represent the best tracking and bandwidth reduction performance.

Table 6
PETS 2009 sequence using the tracker in [1]: Tracking results of (a) a centralized approach that sends compressed images to central node ("IC"), (b) the block-based compression framework in [3] ("FC"), (c) the decentralized Kalman approach ("DecentKalman"), a decentralized method that directly sends the likelihood functions ("Decent"), the distributed approach in [5] ("Distributed-KCF") and the distributed approach in [6] ("Distributed-ICF"), and (d) our sparse representation framework for several levels of compression.

	Number of coefficients	BW (Kbps)	Tracking error	R-Frag	R-IDS
		a			
IC	19,440	4354.56	14.3745	2	230
IC	38,880	8709.12	12.6730	2	133
IC	58,320	13063.68	9.4096	2	166
IC	77,760	17418.24	5.3548	1	103
IC	97,200	21772.8	4.8907	1	169
IC	194,400	43545.6	8.5290	1	105
IC	**486,000**	**108864**	**1.2755**	**1**	**16**
		b			
FC	105	23.52	13.9950	0	249
FC	210	47.04	15.7702	0	280
FC	315	70.56	17.7792	0	254
FC	420	94.08	18.1580	0	344
FC	525	117.6	16.3548	0	270
FC	**1050**	**235.2**	**0.9529**	**0**	**12**
FC	2100	470.4	0.9201	0	13
		c			
Decent	**6400**	**1433.6**	**0.8099**	**0**	**13**
DecentKalman	8	1.792	22.9449	1	454
Distributed-KCF	24	5.376	24.6673	0	467
Distributed-ICF	24	5.376	24.9856	1	453
		d			
Designed-Dict	40	8.96	7.4445	0	102
Designed-Dict	**80**	**17.92**	**0.9583**	**0**	**12**
Designed-Dict	120	26.88	0.9583	0	12
Designed-Dict	160	35.84	0.9583	0	12
Designed-Dict	200	44.8	0.9583	0	12
Designed-Dict	400	89.6	0.9583	0	12

Bold lines represent the best tracking and bandwidth reduction performance.

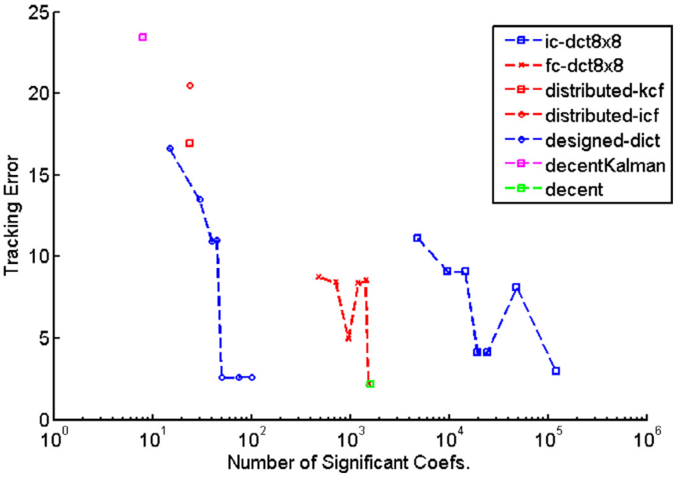


Fig. 18. PETS 2009 sequence using the tracker in [2]: The average tracking errors vs. the number of coefficients for the centralized approach (blue square), the block-based compression framework in [3] (red), our sparse representation framework (blue), the distributed approach in [5] (red square), the distributed approach in [6] (red circle), the decentralized Kalman approach (purple) and a decentralized method (green) that directly sends likelihood functions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

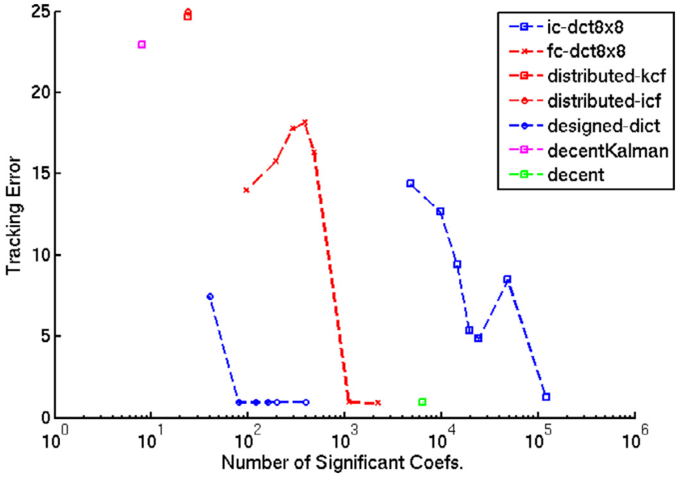


Fig. 19. PETS 2009 sequence using the tracker in [1]: The average tracking errors vs. the number of coefficients for the centralized approach (blue square), the block-based compression framework in [3] (red), our sparse representation framework (blue), the distributed approach in [5] (red square), the distributed approach in [6] (red circle), the decentralized Kalman approach (purple) and a decentralized method (green) that directly sends likelihood functions. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the average of tracking errors over all people, as well as the R-Frag, and R-IDS metrics for all methods within the trackers in [2] and [1], respectively. Figs. 18 and 19 show the average tracking error versus the total number of significant coefficients for all methods within the trackers in [2] and [1], respectively. Since bandwidth usage is not adjustable in distributed, ordinary decentralized, and decentralized Kalman approaches, these methods are represented by single operating points in Figs. 18 and 19. Tables 5 and 6 also present bandwidth requirement of each method in Kbps calculated using a 32-bit precision for each coefficient value. It can be clearly seen that the distributed approaches within both trackers can provide a huge reduction in communication in the network, but they cannot perform

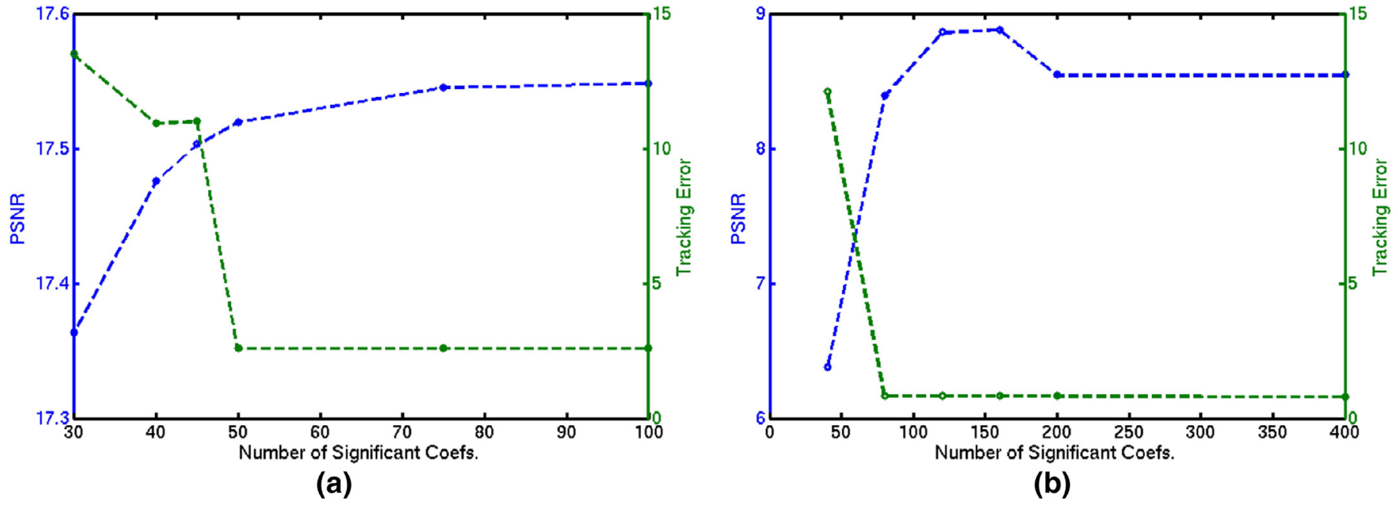


Fig. 20. PETS 2009 sequence: The average PSNR vs. the number of coefficients for our sparse representation framework together with average tracking errors within the tracker (a) in [2] and (b) in [1].

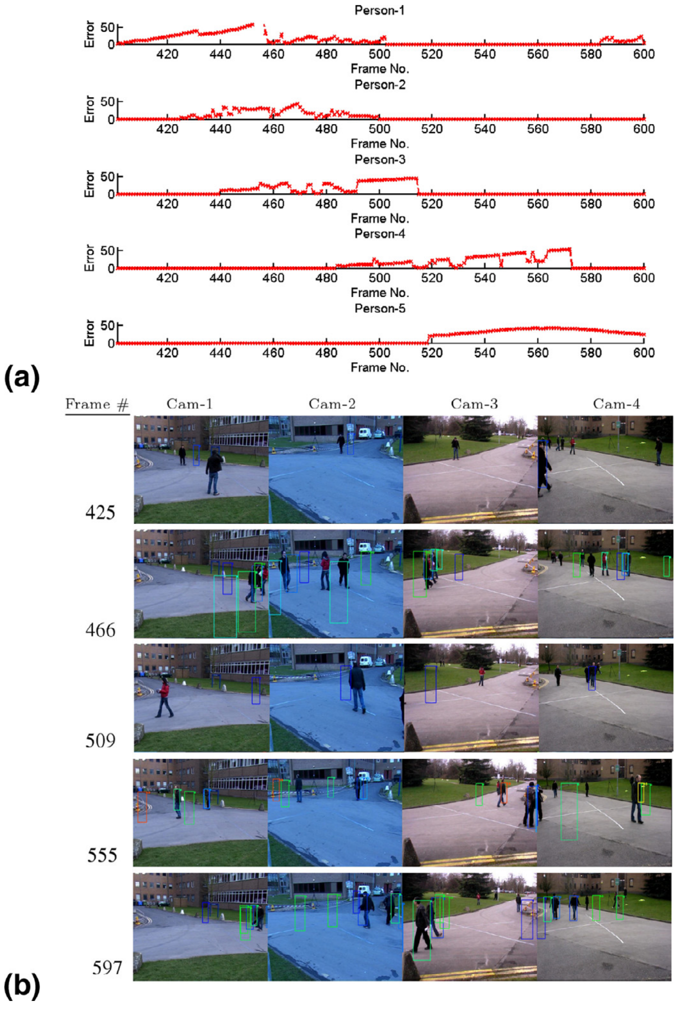


Fig. 21. (a) The tracking errors for each person and (b) tracking results for the PETS 2009 dataset obtained by the distributed approach in [5] within the tracker in [2].

robust tracking. Both KCF and ICF based distributed approaches we consider here appear to depend on each camera to provide good tracking performance on its own, and may not perform well in a challenging tracking scenario when that is not satisfied as we observe in our experiments. Our sparse representation framework achieves a smaller bandwidth reduction than the distributed approaches, but, by using the custom-designed dictionaries, our framework has the ability to decrease the communication without affecting the tracking performance significantly. Within the tracker in [2], by using at least 50 coefficients, our sparse representation framework reconstructs the likelihoods with minimum error. Thereby it achieves a tracking error of 2.5 pixels in the grid on average and minimum number of ID switches. Within the tracker in [1], by using at least 80 coefficients, our approach properly reconstructs the likelihoods and achieves an error of 0.95 pixels in the grid on average and only 12 ID switches. This is also observed in PSNR values given in Fig. 20. On the other hand, while using 24 coefficients, the distributed approaches, within the tracker in [2], has a tracking error of more than 16 pixels in the grid on average and in more than 240 times the people in the scene are misassociated and, within the tracker in [1], has a tracking error of more than 24 pixels in the grid on average and in more than 450 times the people in the scene are misassociated. Within both trackers, our method is also advantageous over the block-based compression approach, the centralized approach and the decentralized Kalman approach. The block-based compression approach fails to perform compression on likelihoods without affecting the tracking accuracy. To achieve robust tracking within the tracker in [2], it needs all coefficients to be transmitted to the fusion node. Within the tracker in [1], it needs at least 1050 coefficients to be transmitted to the fusion node for robust tracking. Within both trackers, the centralized approach requires 486,000 coefficients to achieve an error of 1.3–3 pixels in the grid on average and minimum number of ID switches. Although, the decentralized Kalman approach achieves a huge reduction in bandwidth, it cannot perform robust tracking. The R-IDS metric shows that people are misassociated in 249 and 454 frames within the tracker in [2] and [1], respectively. Our approach also achieves a better performance than an ordinary decentralized approach that directly sends likelihood functions to the fusion node. In such an approach, we send each data point in the likelihood function, resulting in the transmission of 1600 values within the tracker in [2] and 6400 values within the tracker in [1]. The performance of this approach is also presented in Tables 5 and 6, and Figs. 18 and 19. Within the tracker in [2], by using all the information in likelihoods, the ordinary decentralized approach achieves marginally better tracking performance than our method. However, our framework provides a significant reduction in bandwidth use while achieving a tracking performance very close to the performance of the ordinary decentralized method.

The tracking results of the distributed approach in [5] and our sparse representation framework using 50 coefficients, within the

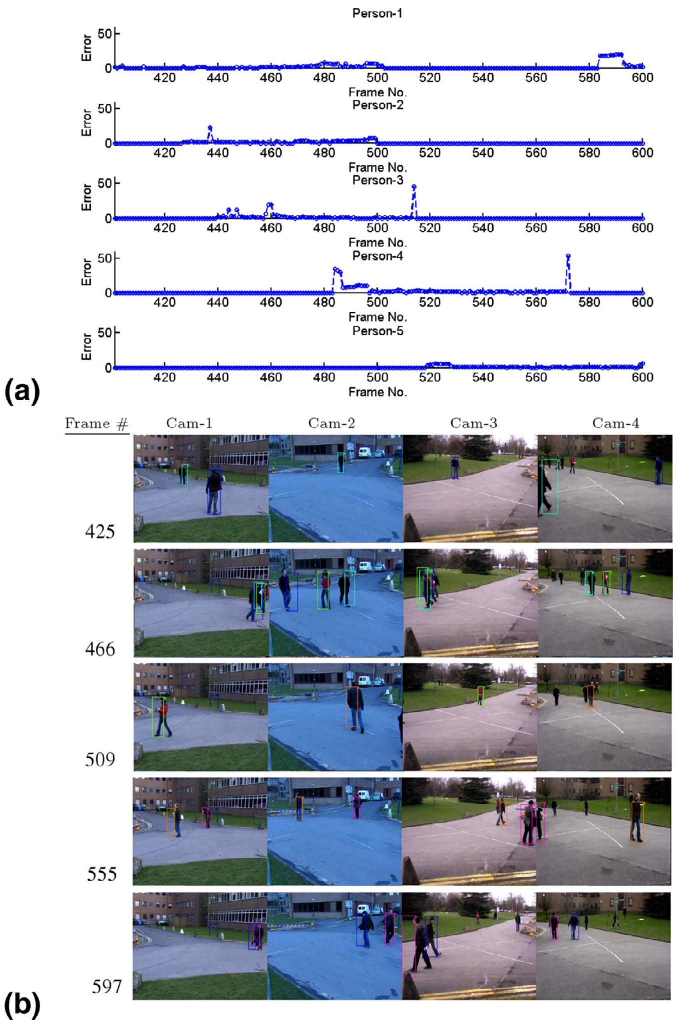


Fig. 22. (a) The tracking errors for each person and (b) tracking results for the PETS 2009 dataset obtained by our sparse representation framework, within the tracker in [2], using 50 coefficients per person used in communication.

tracker in [2], are shown in Figs. 21 and 22, respectively. It can be seen that, the distributed approach fails to preserve identities. For all people in the scene, there occurs identity switches. One of the main reasons of this problem is that the observations extracted from single view likelihood functions are not sufficient for representing the whole likelihood function. The distributed approaches appear to equally incorporate all observations coming from cameras. Since the observations are not extracted after fusing the multi-view likelihoods, noisy single view likelihoods create inaccurate observations for tracking. An example for such a case is given in Fig. 23. We can see that, since the observations extracted from likelihoods of each view are inaccurate, the estimated position of the person is not accurate. Hence, the distributed approach fails to track the people. On the other hand, in our sparse representation framework, we first fuse the single view likelihoods and then use the multi-view likelihood function in tracking. By using the custom-designed dictionaries, we represent the likelihood functions with small number of coefficients without significantly reducing the amount of information they contain. Thus, the multi-view likelihoods, obtained by fusing the reconstructed single view likelihoods, are accurate enough to perform robust tracking under severe bandwidth limitations. In Fig. 22, it can be seen that our framework can successfully preserve identities and track all people in the scene robustly. Even if a person leaves the scene and comes back (the first person in Fig. 22(b)), he or she is recognized and a true label is assigned to the person. Occasionally in this sequence, a person enters the scene while another person leaves. For this reason, sometimes our method starts tracking a few frames after the person enters the scene or ends tracking before the person leaves the scene. Thereby, it suffer from some errors. When the number of coefficients taken is fewer than 50, we also observe identity problems. But by selecting the number of coefficients greater than or equal to 50, we can track all the people in the scene accurately.

In the light of the results we obtained, we can say that our sparse representation based method outperforms the both KCF and ICF based distributed approaches in [5,6]. By using the custom-designed dictionaries, we can both decrease the communication in the network and perform robust tracking. Our method requires only 3.12% of the bandwidth needed by the ordinary decentralized method in order to achieve a tracking performance very close to that method. In the distributed approaches in [5,6], since the observations are modeled with single Gaussian distributions, we only share mean and covariance information with the fusion center. However such a simple model is often insufficient for robust tracking [49]. There are particle filtering based distributed algorithms in which the particles sampled from likelihood functions are approximated using models (e.g., mixture of Gaussians) or quantized in order to reduce communications [49,50]. Since our approach involves representing likelihood functions using custom-designed dictionaries, we expect to obtain more parsimonious representations, and hence more efficient communication than such methods.

6. Conclusions

Using a camera in a wireless network leads to unique and challenging problems. This paper presents a novel method that can be used in VSNs for multi-camera person tracking applications. In our method, tracking is performed in a decentralized way: each camera extracts useful features from the images it has observed and sends them to a fusion node which performs tracking. Most probabilistic tracking systems involve computation of a likelihood function. Instead of sending the likelihood functions themselves to the fusion node, we compress the likelihoods via sparse representation. Special overcomplete dictionaries that are matched to the structure of

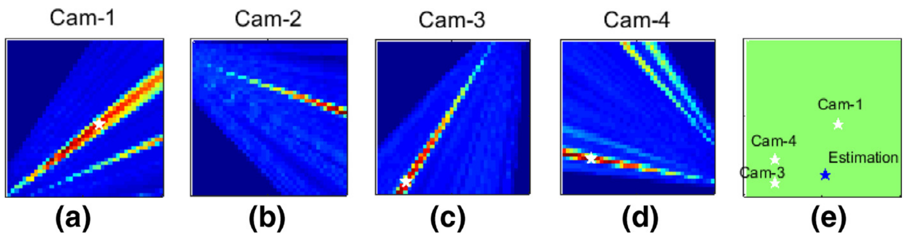


Fig. 23. (a)–(d) Inaccurate observations extracted from the likelihoods of camera 1,3 and 4 (white stars) lead to (e) inaccurate estimation (blue star) in the distributed approach in [5]. Since the person of interest is not visible in camera 2, we do not have an observation coming from this view. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the likelihood functions are designed in an adaptive fashion exploiting information about the geometry of the sensing scenario, and used for sparse representation of likelihoods. This enables us to decrease the communication between cameras and fusion nodes. To the best of our knowledge, this is the first method that uses sparse representation and specially designed dictionaries to compress likelihood functions and applies this idea for VSNs. This framework fits well within the needs of the VSN environment. By extracting image features at the camera-level, the processing capabilities of cameras are utilized. Transmitting only the most significant coefficients, obtained from the sparse representation of likelihoods, saves energy and bandwidth resources. In this manner, we have achieved a goal-directed compression scheme for the tracking problem in VSNs by performing local processing at the nodes and compressing the resulting likelihood functions which are related to the tracking goal, rather than compressing raw images.

Another advantage of this framework is that it does not require the use of a specific tracking method. In our experiments, we have used two different tracking algorithms and achieved bandwidth reduction in the network without degrading the tracking performance significantly. We believe our sparse representation framework is an effective approach that can be used together with any probabilistic tracker in VSNs. Thereby, existing centralized methods can be used within our framework in VSN environments without making significant changes (e.g., using simpler features, etc.) which may degrade their performance. By using overcomplete dictionaries that are matched to the structure of the likelihoods, for the same tracking performance, we achieve more bandwidth savings compared to existing methods.

In our experiments, we have also observed that there is a negative correlation between PSNR levels and the tracking error. This can be used to automatically choose an operating point (i.e., number of coefficients), e.g., based on the slope of the PSNR curve. While the absolute value of the tracking error is scenario-dependent, this automatic choice would enable one to avoid the operating points with significantly high tracking errors and achieve a low tracking error without using unnecessarily large number of coefficients. While we do not claim this is the best approach for automatic selection of the number of coefficients in our approach, we believe it certainly provides a reasonable approach demonstrating the feasibility of automatic selection. We believe that trying our sparse representation framework in nonoverlapping VSN setups, by compressing feature descriptors used for re-identification (such as in [46,51]), can be another possible direction for future work.

Acknowledgments

This work was partially supported by a Turkish Academy of Sciences Distinguished Young Scientist Award and by a graduate scholarship from the Scientific and Technological Research Council of Turkey.

References

[1] F. Fleuret, J. Berclaz, R. Lengagne, P. Fua, Multicamera people tracking with a probabilistic occupancy map, PAMI 30 (2) (2008) 267–282, doi:10.1109/TPAMI.2007.1174.

[2] S.M. Khan, M. Shah, Tracking multiple occluding people by localizing on multiple scene planes, IEEE Trans. Pattern Anal. Mach. Intell. 31 (3) (2009) 505–519, doi:10.1109/TPAMI.2008.102.

[3] S. Coşar, M. Çetin, Feature compression: A framework for multi-view multi-person tracking in visual sensor networks, J. Visual Commun. Image Represent. 25 (5) (2014) 864–873.

[4] H. Medeiros, J. Park, A. Kak, Distributed object tracking using a cluster-based kalman filter in wireless camera networks, IEEE J. Select. Top. Signal Process. 2 (4) (2008) 448–463, doi:10.1109/JSTSP.2008.2001310.

[5] B. Song, A.T. Kamal, C. Soto, C. Ding, J.A. Farrell, A.K. Roy-Chowdhury, Tracking and activity recognition through consensus in distributed camera networks, IEEE Trans. Image Process. 19 (10) (2010) 2564–2579, doi:10.1109/TIP.2010.2052823.

[6] A. Kamal, J. Farrell, A. Roy-Chowdhury, Information consensus for distributed multi-target tracking, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2013, pp. 2403–2410, doi:10.1109/CVPR.2013.311.

[7] D. Yang, H. Gonzalez-Banos, L. Guibas, Counting people in crowds with a real-time network of simple image sensors, in: Proceedings of Ninth IEEE International Conference on Computer Vision, 1, IEEE, 2003, pp. 122–129, doi:10.1109/ICCV.2003.1238325.

[8] D. Yang, J. Shin, A.O. Ercan, L. Guibas, Sensor tasking for occupancy reasoning in a camera network, in: Proceedings of IEEE/ICST 1st Workshop on Broadband Advanced Sensor Networks (BASENETS'04), 2004. http://geometry.stanford.edu//paper.php?id=yseg-basn-04.

[9] P.V. Pahalawatta, A.K. Katsaggelos, Optimal sensor selection for video-based target tracking in a wireless sensor network, in: Proceedings of the International Conference on Image Processing (ICIP'04), IEEE, 2004, pp. 3073–3076.

[10] S. Fleck, F. Busch, W. Straß er, Adaptive probabilistic tracking embedded in smart cameras for distributed surveillance in a 3d model, EURASIP J. Embed. Syst. 2007 (1) (2007) 24. http://dx.doi.org/10.1155/2007/29858.

[11] E. Oto, F. Lau, H. Aghajan, Color-based multiple agent tracking for wireless image sensor networks, in: ACIVS06, Springer-Verlag, 2006, pp. 299–310.

[12] B. Song, A. Roy-Chowdhury, Robust tracking in a camera network: A multi-objective optimization framework, IEEE J. Select. Top. Signal Process. 2 (4) (2008) 582–596, doi:10.1109/JSTSP.2008.925992.

[13] J. Yoder, H. Medeiros, J. Park, A. Kak, Cluster-based distributed face tracking in camera networks, IEEE Trans. Image Process. 19 (10) (2010) 2551–2563, doi:10.1109/TIP.2010.2049179.

[14] Y. Wang, S. Velipasalar, M. Casares, Cooperative object tracking and composite event detection with wireless embedded smart cameras, IEEE Trans. Image Process. 19 (10) (2010) 2614–2633, doi:10.1109/TIP.2010.2052278.

[15] R. Olfati-Saber, Distributed kalman filtering for sensor networks, in: 46th IEEE Conference on Decision and Control, IEEE, 2007, pp. 5492–5498, doi:10.1109/CDC.2007.4434303.

[16] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, R. Baraniuk, Single-pixel imaging via compressive sampling, IEEE Signal Process. Mag. 25 (2) (2008) 83–91, doi:10.1109/MSP.2007.914730.

[17] J. Mairal, M. Elad, G. Sapiro, Sparse representation for color image restoration, IEEE Trans. Image Process. 17 (1) (2008) 53–69, doi:10.1109/TIP.2007.911828.

[18] M. Plumbley, T. Blumensath, L. Daudet, R. Gribonval, M. Davies, Sparse representations in audio and music: From coding to source separation, Proc. IEEE 98 (6) (2010) 995–1005, doi:10.1109/JPROC.2009.2030345.

[19] L. Potter, E. Ertin, J. Parker, M. Cetin, Sparsity and compressed sensing in radar imaging, Proc. IEEE 98 (6) (2010) 1006–1020, doi:10.1109/JPROC.2009.2037526.

[20] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2) (2009) 210–227, doi:10.1109/JPROC.2010.2040797.

[21] V. Cevher, A.C. Sankaranarayanan, M.F. Duarte, D. Reddy, R.G. Baraniuk, R. Chellappa, Compressive sensing for background subtraction, in: ECCV, Springer, Marseille, France, 2008, pp. 155–168.

[22] J. Mairal, F. Bach, J. Ponce, G. Sapiro, A. Zisserman, Discriminative learned dictionaries for local image analysis, in: IEEE Conference on Computer Vision and Pattern Recognition, 2008 (CVPR'08), IEEE, 2008, pp. 1–8, doi:10.1109/CVPR.2008.4587652.

[23] Y. Liu, X. Zhu, L. Zhang, S.H. Cho, Distributed compressed video sensing in camera sensor networks, Int. J. Distrib. Sens. Netw. 2012 (2012) 1–10.

[24] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. Huang, S. Yan, Sparse representation for computer vision and pattern recognition, Proc. IEEE 98 (6) (2010) 1031–1044, doi:10.1109/JPROC.2010.2044470.

[25] M. Taj, A. Cavallaro, Distributed and decentralized multicamera tracking, IEEE Signal Process. Mag. 28 (3) (2011) 46–58, doi:10.1109/MSP.2011.940281.

[26] S. Hengstler, D. Prashanth, S. Fong, H. Aghajan, Mesheye: a hybrid-resolution smart camera mote for applications in distributed intelligent surveillance, in: Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN'07), ACM, New York, NY, USA, 2007, pp. 360–369, doi:10.1145/1236360.1236406.

[27] W. Wolf, B. Ozer, T. Lv, Smart cameras as embedded systems, Computer 35 (9) (2002) 48–53, doi:10.1109/MC.2002.1033027.

[28] B. Tavli, K. Bicakci, R. Zilan, J. Barcelo-Ordinas, A survey of visual sensor network platforms, Multim. Tool Appl. 60 (3) (2012) 689–726, doi:10.1007/s11042-011-0840-z.

[29] I. Akyildiz, T. Melodia, K. Chowdury, Wireless multimedia sensor networks: A survey, IEEE Wireless Commun. 14 (6) (2007) 32–39, doi:10.1109/MWC.2007.4407225.

[30] J. Yao, J.-M. Odobez, Multi-camera multi-person 3d space tracking with mcmc in surveillance scenarios, in: ECCV Workshop on Multi Camera and Multi-modal Sensor Fusion Algorithms and Applications, Springer, Berlin, Heidelberg, 2008.

[31] A. Gupta, A. Mittal, L. Davis, Constraint integration for efficient multiview pose estimation with self-occlusions, PAMI 30 (3) (2008) 493–506, doi:10.1109/TPAMI.2007.1173.

[32] M. Hofmann, D. Gavrila, Multi-view 3d human pose estimation combining single-frame recovery, temporal integration and model adaptation, in: CVPR, IEEE, 2009, pp. 2214–2221, doi:10.1109/CVPR.2009.5206508.

[33] M. Ayazoglu, B. Li, C. Dicle, M. Sznaier, O. Camps, Dynamic subspace-based coordinated multicamera tracking, in: 2011 IEEE International Conference on Computer Vision (ICCV), IEEE, 2011, pp. 2462–2469, doi:10.1109/ICCV.2011.6126531.

[34] C. Yu, G. Sharma, Camera scheduling and energy allocation for lifetime maximization in user-centric visual sensor networks, IEEE Trans. Image Process. 19 (8) (2010) 2042–2055, doi:10.1109/TIP.2010.2046794.

[35] D. Karuppiah, R. Grupen, Z. Zhu, A. Hanson, Automatic resource allocation in a distributed camera network, Mach. Vis. Appl. 21 (4) (2010) 517–528, doi:10.1007/s00138-008-0182-7.

[36] E. Monari, K. Kroschel, Task-oriented object tracking in large distributed camera networks, in: 2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), IEEE, 2010, pp. 40–47, doi:10.1109/AVSS.2010.66.

[37] B. Dieber, C. Micheloni, B. Rinner, Resource-aware coverage and task assignment in visual sensor networks, IEEE Trans. Circ. Syst. Video Technol. 21 (10) (2011) 1424–1437, doi:10.1109/TCSVT.2011.2162770.

[38] J. Yao, J.-M. Odobez, Fast human detection from videos using covariance features, in: European Conference on Computer Vision, Workshop on Visual Surveillance (ECCV-VS), Springer, Berlin, Heidelberg, 2008.

[39] M. Aharon, M. Elad, A. Bruckstein, k -svd: An algorithm for designing overcomplete dictionaries for sparse representation, IEEE Trans. Signal Process. 54 (11) (2006) 4311–4322, doi:10.1109/TSP.2006.881199.

[40] R. Rubinstein, A. Bruckstein, M. Elad, Dictionaries for sparse representation modeling, Proc. IEEE 98 (6) (2010) 1045–1057, doi:10.1109/JPROC.2010.2040551.

[41] A. Yang, S. Sastry, A. Ganesh, Y. Ma, Fast l_1-minimization algorithms and an application in robust face recognition: A review, in: 2010 17th IEEE International Conference on Image Processing (ICIP), IEEE, 2010, pp. 1849–1852, doi:10.1109/ICIP.2010.5651522.

[42] M.S. Asif, J. Romberg, Fast and accurate algorithms for re-weighted l1 norm minimization, IEEE Trans. Signal Process. 61 (23) (2012) 5905–5916.

[43] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, D. Gorinevsky, An interior-point method for large-scale l_1-regularized least squares, IEEE J. Select. Top. Signal Process. 1 (4) (2007) 606–617, doi:10.1109/JSTSP.2007.910971.

[44] S. Wright, R. Nowak, M.A.T. Figueiredo, Sparse reconstruction by separable approximation, IEEE Trans. Signal Process. 57 (7) (2009) 2479–2493, doi:10.1109/TSP.2009.2016892.

[45] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM J. Img. Sci. 2 (1) (2009) 183–202, doi:10.1137/080716542.

[46] C.-H. Kuo, C. Huang, R. Nevatia, Inter-camera association of multi-target tracks by on-line learned appearance affinity models, in: K. Daniilidis, P. Maragos, N. Paragios (Eds.), Computer Vision (ECCV'10), *Lecture Notes in Computer Science*, 6311, Springer, Berlin, Heidelberg, 2010, pp. 383–396.

[47] Eleventh IEEE international workshop on performance evaluation of tracking and surveillance (pets), 2009. http://www.cvg.reading.ac.uk/PETS2009/a.html.

[48] M.J. Swain, D.H. Ballard, Color indexing, Int. J. Comput. Vis. 7 (1) (1991) 11–32, doi:10.1007/BF00130487. http://dx.doi.org/10.1007/BF00130487

[49] Z. Ni, S. Sunderrajan, A. Rahimi, B. Manjunath, Distributed particle filter tracking with online multiple instance learning in a camera sensor network, in: Image Processing (ICIP), 2010 17th IEEE International Conference on, 2010, pp. 37–40.

[50] M. Coates, Distributed particle filters for sensor networks, in: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN'04), ACM, New York, NY, USA, 2004, pp. 99–107, doi:10.1145/984622.984637. http://doi.acm.org/10.1145/984622.984637

[51] A. Gilbert, R. Bowden, Tracking objects across cameras by incrementally learning inter-camera colour calibration and patterns of activity, in: Proceedings of the 9th European Conference on Computer Vision—Volume Part II (ECCV'06), Springer-Verlag, Berlin, Heidelberg, 2006, pp. 125–136.