Feature Compression: A Framework for Multi-View Multi-Person Tracking in Visual Sensor Networks

Serhan Coşar*, Müjdat Çetin

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

---

Abstract

Visual sensor networks (VSNs) consist of image sensors, embedded processors and wireless transceivers which are powered by batteries. Since the 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 VSNs. 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 method, each camera performs feature extraction and obtains likelihood functions. By transforming to an appropriate domain and taking only the significant coefficients, these likelihood functions are compressed and this new representation is sent to the fusion node. An appropriate domain is selected by performing a comparison between well-known transforms. We have applied our method for indoor people tracking and demonstrated the superiority of our system over the traditional approach.

Keywords: Visual sensor networks, Camera networks, Human tracking, Communication constraints, Compressing likelihood functions

---

*Corresponding author. Tel: +90 216 4830000-2117, Fax: +90 216 483-9550.

Email addresses: serhancosar@sabanciuniv.edu (Serhan Coşar), mcetin@sabanciuniv.edu (Müjdat Çetin)

1. Introduction

With the birth of wireless sensor networks, new applications are enabled by large-scale networks of small devices capable of (i) measuring information from the physical environment, such as temperature, pressure, etc., (ii) performing simple processing on the extracted data, and (iii) transmitting the processed data to remote locations by also considering the limited resources such as energy and bandwidth. More recently, the availability of inexpensive hardware such as CMOS cameras that are able to capture visual data from the environment has supported the development of Visual Sensor Networks (VSNs), i.e., networks of wirelessly interconnected devices that acquire video data.

Using a camera in a wireless network leads to unique and challenging problems that are more complex than the traditional wireless sensor networks might have. For instance, most sensors provide measurements of temporal signals that represent physical quantities such as temperature. On the other hand, at each time instant image sensors provide a 2D set of data points, which we see as an image. This richer information content increases the complexity of data processing and analysis. Performing complex tasks, such as tracking, recognition, etc., in a communication-constrained VSN environment is extremely challenging. With a data compression perspective, the common approach is to compress images and collect them in a central unit to perform the tasks of interest. In this strategy, the main goal is to focus on low-level communication. The communication load is decreased by compressing the raw data without regard to the final inference goal based on the information content of the data. Since such a strategy will affect the quality of the transmitted data, it may decrease the performance of further inference tasks. In this paper, we propose a different strategy for decreasing the communication that is better matched to problems with a defined final inference goal, which, in the context of this paper, is tracking.

There has been some work proposed for solving the problems mentioned above.

To minimize the amount of data to be communicated, in some methods simple features are used for communication. For instance, 2D trajectories are used in [1]. In [2], 3D trajectories together with color histograms are used. Hue histograms along with 2D position are used in [3]. Moreover, there are decentralized approaches in which cameras are grouped into clusters and tracking is performed by local cluster fusion nodes. This kind of approaches have been applied to the multi-camera target tracking problem in various ways [4, 5, 6]. 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 [4]. For an overlapping camera setup, a cluster-based Kalman filter in a network of wireless cameras is proposed in [5, 6]. 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.

Previous works proposed for VSNs have some handicaps. The methods in [1, 2, 3] that use simpler features may be capable of decreasing the communication, but they are not capable of maintaining robustness. For the sake of bandwidth constraints, these methods choose to change the features from complex and robust to simpler but not so effective ones. As in the methods proposed in [4, 5, 6], performing local processing and collecting features to the fusion node 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. An efficient approach that reduces the bandwidth requirements without significantly decreasing the quality of image features is needed.

In this paper, we propose a framework that is suitable for energy and bandwidth constraints in VSNs. It is capable of performing multi-person tracking without significant performance loss. 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). Instead of directly sending likelihood functions to the fusion node, a block-based compression is performed on likelihoods by transforming each block to an appropriate domain. Then, in this new representation we only take the significant coefficients and send them to the fusion node. Hence, multi-view tracking can be performed without overloading the network. The main contribution of this work is the idea of performing goal-directed compression in a VSN. In the tracking context, this is achieved 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. To the best of our knowledge, 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 a well-known multi-camera human tracking algorithm [7]. We have modified the method in [7] to obtain a decentralized tracking algorithm. In order to choose an appropriate domain for likelihood functions, we have performed a comparison between well-known transforms. A traditional approach in camera networks is transmitting compressed images. Both by qualitative and quantitative results, we have shown that our method is better than the traditional approach of sending compressed images and can work under VSN constraints without degrading the tracking performance.

In Section 2, how we integrate multi-view information in our decentralized approach is described. Section 3 presents our feature compression framework in detail and contains a comparison of various domains for likelihood representation. Experimental setup and results are given in Section 4. Finally in Section

5, we conclude and suggest a number of directions for potential future work.

2. Multi-Camera Integration

2.1. 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 in raw-data level by pooling all images in 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, thereby, to a lack of scalability and energy efficiency. Compressing raw image data may decrease the communication in the network, but since the quality of images drops, it might also decrease the 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 [8]. 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. Using the multi-view image features, tracking is performed in the local fusion node. Hence, we can say that in decentralized tracking, multi-view information is integrated in feature-level by combining the features in small clusters. The decentralized approaches fits very well to VSNs in many aspects. The processing capability of each camera is utilized by performing feature extraction at 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, effi-

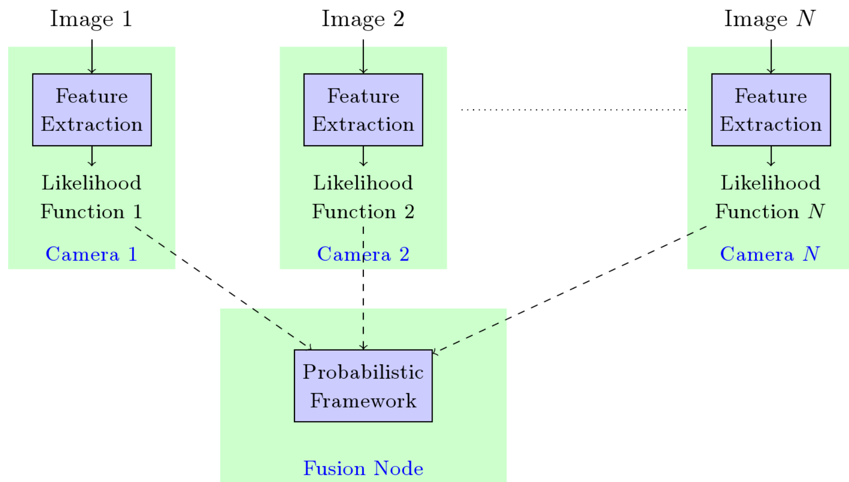


Figure 1: The flow diagram of a decentralized tracker using a probabilistic framework.

cient estimation can be performed.

Modeling the dynamics of humans in a probabilistic framework is a common perspective of many multi-camera human tracking methods [7, 9, 10, 11]. In tracking methods based on a probabilistic framework, data and/or extracted features are represented by likelihood functions, $p(y|x)$ where $y \in R^d$ and $x \in R^m$ 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 the decentralized approach is illustrated in Figure 1. Following this line of thought, we have converted the tracking approach described in Section 2.2 to a decentralized tracker as explained in Section 2.3.

2.2. Multi-Camera Tracking Algorithm

In this section we describe the tracking method of [7], as we apply our proposed approach within in the context of this method in this paper. In [7], the visible part of the ground plane is discretized into a finite number G of regularly spaced 2D locations. Let $\mathbf{L}_t = (L_t^1, ..., L_t^{N^*})$ 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, $\mathbf{I} = (\mathbf{I}_1, ..., \mathbf{I}_T)$ where $\mathbf{I}_t = (I_t^1, ..., I_t^C)$, the goal is to maximize the posterior conditional probability:

$$P(\mathbf{L}^1 = \mathbf{l}^1, ..., \mathbf{L}^{N^*} = \mathbf{l}^{N^*} | \mathbf{I}) = P(\mathbf{L}^1 = \mathbf{l}^1 | \mathbf{I}) \prod_{n=2}^{N^*} P(\mathbf{L}^n = \mathbf{l}^n | \mathbf{I}, \mathbf{L}^1 = \mathbf{l}^1, ..., \mathbf{L}^{n-1} = \mathbf{l}^{n-1}) \tag{1}$$

where $\mathbf{L}^n = (L_1^n, ..., L_T^n)$ is the trajectory of person n. Simultaneous optimization of all the L^is would be intractable. Instead, one trajectory after the other is optimized. $\mathbf{L}^n$ is estimated by seeking the maximum of the probability of both the observations and the trajectory ending up at location k at time t:

$$\Phi_t(k) = \max_{l_1^n, ..., l_{t-1}^n} P(\mathbf{I}_1, L_1^n = l_1^n, ..., \mathbf{I}_t, L_t^n = k) \tag{2}$$

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

$$\Phi_t(k) = \underbrace{P(\mathbf{I}_t | L_t^n = k)}_{Appearance\ model} \max_{\tau} \underbrace{P(L_t^n = k | L_{t-1}^n = \tau)}_{Motion\ model} \Phi_{t-1}(\tau) \tag{3}$$

The motion model $P(L_t^n = k | L_{t-1}^n = \tau)$ is a distribution into a disc of 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 are denoted as $\mathbf{T}_t$ and 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 [7] that the appearance model in Eq. 3 can be decomposed as:

$$\overbrace{P(\mathbf{I}_t|L_t^n = k)}^{Appearance\ model} \propto \underbrace{P(L_t^n = k|X_k^t = 1, \mathbf{T}_t)}_{Color\ model} \underbrace{P(X_k^t = 1|\mathbf{B}_t)}_{Ground\ plane\ occupancy} \quad (4)$$

In [7], humans are represented as simple rectangles and these rectangles are 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 reference color distributions (histograms) of the N^* individuals present in the scene, $\mu_1^c, ..., \mu_{N^*}^c$. The color model of person n in Eq. 4 can be expressed as:

$$\begin{aligned}\overbrace{P(L_t^n = k|X_k^t = 1, \mathbf{T}_t)}^{Color\ model} \propto P(\mathbf{T}_t|L_t^n = k) = & \quad P(T_t^1(k), ..., T_t^C(k)|L_t^n = k) \\ = & \quad \textstyle\prod_{c=1}^{C} P(T_t^c(k)|L_t^n = k) \quad (5)\end{aligned}$$

In [7], by assuming the pixels whose colors are represented by $T_t^c(k)$ are independent, $P(T_t^c(k)|L_t^n = k)$ is evaluated by a product of the marginal color distribution μ_n^c at each pixel,– $P(T_t^c(k)|L_t^n = k) = \prod_{r \in T_t^c(k)} \mu_n^c(r)$. In this approach, a patch with constant color intensity corresponding to the the mode of the color distribution would be most likely. Hence, this approach may fail to capture the statistical color variability represented by the full probability density function estimated from a spatial patch. Instead, we represent $P(T_t^c(k)|L_t^n = k)$ by comparing the observed and reference color distributions, which is a well known approach used in many computer vision methods [12, 13, 14]. In particular, we compare the estimated color distribution (histogram) of the pixels in $T_t^c(k)$ and the color distribution μ_n^c with a distance metric – $P(T_t^c(k)|L_t^n = k) = exp(-S(H_t^{c,k}, \mu_n^c))$ where $H_t^{c,k}$ denotes the histogram of the pixels in $T_t^c(k)$ and $S(.)$ is a distance metric. As a distance metric, we use the Bhattacharya coefficient between two distributions. In this way, we can evaluate the degree of match between the intensity distribution of an observed patch and the reference color distribution.

By performing a global search with dynamic programming using Eq. 3, the trajectory of each person can be estimated.

2.3. Decentralized Version of the Tracking Algorithm

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 acquired. 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 Figure 1): At each camera node ($c = 1, \cdots, C$), 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. 5). 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. 3). The whole process is run for each person in the scene.

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, relatively more powerful one, has been selected as the fusion node.

3. Feature Compression Framework

3.1. Compressing Likelihood Functions

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, we propose a feature compression framework. In our framework, similar to image compression, we compress the likelihood functions by transforming them to a proper domain and keeping only the significant coefficients, assuming significant parts of the likelihood functions are sufficient for performing tracking. At each camera node, we first split the likelihood function into blocks. Then, we transform each block to a proper domain and take only the significant coefficients in the new representation. Instead of sending the function itself, we send this new representation of each block. In this way, we reduce the communication in the network.

Mathematically, we have the following linear system:

$$y_c^b = A \cdot x_c^b \tag{6}$$

where y_c^b and x_c^b represent the bth block of the likelihood function of camera c (for a person of interest in a particular time instant, $P(T_t^c(k)|L_t^n = k)$ in Eq. 5) and its representation, respectively, and A is the domain we transform y_c^b to. In most of the compression methods, the matrix A is chosen to be a unitary matrix. Hence, we can obtain x_c^b by multiplying y_c^b with the Hermitian transpose of A:

$$x_c^b = A^* \cdot y_c^b \tag{7}$$

Figure 2 illustrates our likelihood compression scheme.

Notice that in our feature compression framework, we do not require the use

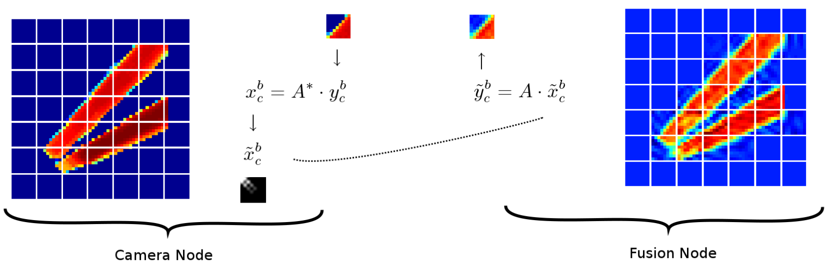


Figure 2: Our Likelihood compression scheme. On the left, there is a local likelihood function ($P(T_t^c(k)|L_t^n = k)$ in Eq. 5). First, we split the likelihood into blocks, then we transform each block to the domain represented by matrix A and obtain the representation x_c^b. We only take significant coefficients in this representation and obtain a new representation $\tilde{x}_c^b$. For each block, we send this new representation to fusion node. Finally, by reconstructing each block we obtain the whole likelihood function on the right.

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 all camera nodes and fusion nodes, the matrix A is common, therefore, at the fusion node, likelihood functions of each camera can be reconstructed simply by multiplying the new representation with the matrix A. In general, this may require an offline coordination step to decide the domain that is matched with the task of interest. In the next subsection, we go through the question of which domain should be selected in Eq. (6).

3.2. A Proper Domain for Compression

By sending the compressed likelihoods to the fusion node, our goal is to decrease the communication in the network without affecting the tracking performance significantly. On one hand, we want to send less coefficients, on the other hand, we do not want to decrease the quality of the likelihoods, i.e., we want to have small reconstruction error. For this reason, we need to select a domain that is well-matched to the likelihood functions, providing the opportunity to accurately reconstruct the likelihoods back using a small number of coefficients.

Image compression using transforms is a mature research area. Numerous transforms such as the discrete cosine transform (DCT), the Haar transform, symmlets, coiflets have been proposed and proven to be successful [15, 16, 17]. DCT is a well-known transform that has the ability to analyze non-periodic signals. Haar wavelet is the first known wavelet basis that consists of orthonormal functions. In wavelet theory, *number of vanishing moments* and *size of support* are two important properties that affect the ability of wavelet bases to approximate a particular class of functions with few non-zero wavelet coefficients [18]. In order to reconstruct likelihoods accurately using from a small number of coefficients, we wish wavelet functions to have large number of vanishing moments and small size of support. Coiflets [19] are a wavelet basis with large number of vanishing moments and Symmlets [20] are a wavelet basis that have minimum size of support. The performance of these domains has been analyzed in the context of our experiments and a proper domain has been selected accordingly as described in Section 4.2.

4. Experimental Results

4.1. Setup

In the experiments, we have simulated the VSN environment by using the indoor multi-camera dataset in [7]. This dataset includes four people sequentially entering a room and walking around. The sequence was shot by four synchronized cameras in a 50 m^2 room. The cameras were located at each corner of the room. In this sequence, the area of interest was of size 5.5 $m \times$ 5.5 $m \simeq 30$ m^2 and discretized into $G = 56 \times 56 = 3136$ locations, corresponding to a regular grid with a $10cm$ resolution. For the correspondence between camera views and the top view, the homography matrices provided with the dataset are used. The

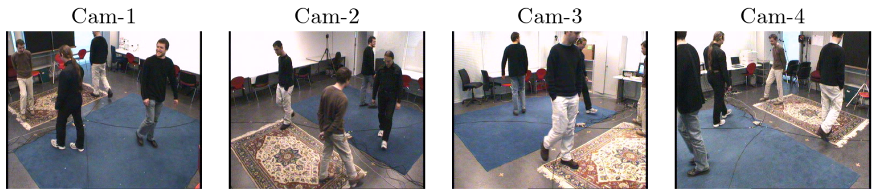


Figure 3: A sample set of images from the indoor multi-camera dataset [7].

size of the images are 360×288 pixels and the frame rate for all of the cameras is 25 fps. The sequence is approximately 2.5 minutes ($\simeq 3,800$ frames) long.

Starting from the frames around the 2,000th, we have observed failures in the original method [7] on preserving identities. For this reason, we have used the sequence consisting of the first 2,000 frames for testing. A sample set of images is shown in Figure 3.

4.2. Comparison of Domains

As discussed in Section 3.2, it is very important to select a domain (matrix A in Eq. (6)) that can compress the likelihood functions effectively. To select a proper domain, we have performed a comparison between DCT, Haar, Symmlet, and Coiflet domains and examined the errors in reconstructing the likelihoods using various number of coefficients. For the Symmlet domain, the size of support is set to 8 and for the Coiflet domain, the number of vanishing moments is set to 10. In the comparison, we have used 20 different likelihood functions obtained from the tracker in [7]. We have also analyzed the effect of block size by choosing two different block sizes: $8{\times}8$ and $4{\times}4$. After we transform each block to a domain, we have reconstructed the blocks by using only 1, 2, 3, 4, 5, and 10 most significant coefficient(s). In total, for a block size of $8{\times}8$, taking the most significant 2 coefficients results in 98 coefficients overall. According to the structure of the likelihood functions, the elements in a block may all be zero. For such a block all the coefficients will be zero, thereby we do not need to take coefficients. Thus, we may end up with even smaller number of coefficients.

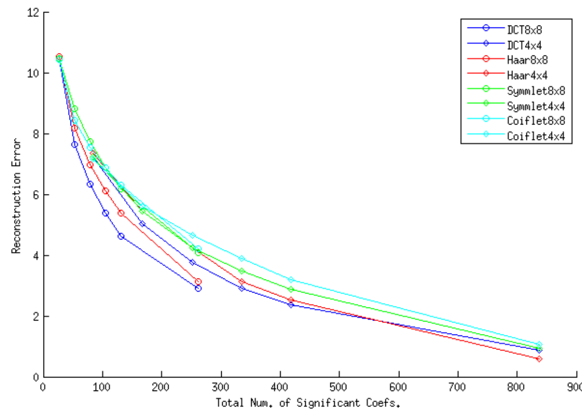


Figure 4: The average reconstruction errors of DCT, Haar, Symmlet, and Coiflet domain for block sizes of 8×8 and 4×4 using 1, 2, 3, 4, 5 and 10 most significant coefficient(s) per block.

Figure 4 shows the average of reconstruction errors of each domain for different block sizes. As explained above, the total number of significant coefficients used for reconstruction may change depending on the structure of likelihoods. For this reason, the x-axis in Figure 4 are the average of number of coefficients obtained by taking the 1, 2, 3, 4, 5 and 10 most significant coefficient(s) per block. We can see that using DCT with a block size of 8×8 outperforms other domains. Following this observation, in our tracking experiments, this setting has been used.

4.3. Tracking Results

In this subsection, we present the performance of our method used for multi-view multi-person tracking. In the experiments, we have compared our method with the traditional centralized approach of compressing raw images. In this centralized approach, after the raw images are acquired by the cameras, similar to JPEG compression, each color channel in the images are compressed and sent to the central node. In the central node, features are extracted from the reconstructed images and tracking is performed using the method in [7]. For both our method and the centralized approach we have used DCT domain with a block size of 8×8 and took only the 1, 2, 3, 4, 5, 10, and 25 most significant coefficient(s). Consequently, in our method with the likelihoods of 56×56 size, 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 maximum, each camera sends at most 196, 392, 588, 784, 980, 1960 and 4900 coefficients. As mentioned in the previous section, these are the maximum number of coefficients, since there may be some all-zero blocks. To make a fair comparison, in the centralized approach we compress the images with 360×288 size and 3 color channels. Hence, at each camera we end up with 4860, 9720, 14580, 19440, 24300, 48600 and 121500 coefficients.

A groundtruth for this sequence is obtained by manually marking the people on 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). Figure 5 presents the average of tracking errors over all people versus the total number of significant coefficients used in communication for the centralized approach and for our method. Since the total number of significant coefficients sent by a camera in our method may change depending on the structure of likelihood functions and the number of people at that moment, the maximum is shown in Figure 5. It can be clearly seen that the centralized approach is not capable of decreasing the communication without affecting the tracking performance. It needs at least 121500 significant coefficients in total to achieve an error of around 1 pixel in the grid on average. On the other hand, our method, down to using 3 significant coefficients per block, achieves an error of around 1 pixel in the grid on average. In our experiments, this led to sending at most 408 coefficients for four people. Taking less than 3 coefficients per block affects the performance of the tracker and produces an error of 11.5 pixels in the grid on average. But in overall, our method significantly outperforms the centralized approach.

The tracking errors for each person and the tracking results, obtained by the centralized approach using 48600 coefficients in total, are given in Figure 6-a and Figure 6-b, respectively. It can be seen that although the centralized approach can track the first and the second individuals very well, there is an identity association problem for the third and fourth individuals. In Figure 7-a and Figure 7-b, we present the tracking errors for each person and the tracking results obtained with our method using 3 coefficients per block, respectively. Clearly, we can see that all people in the scene can be tracked very well by our method. The reason of the peak error value in the third person is because the tracking starts a few frames after the third person enters the room. For this reason, there is a big error at the time third person enters the room. When the number of coefficients taken per block are less then 3, we also observe identity problems. But by selecting the number of coefficients per block greater than or equal to 3, we can track all the people in the scene accurately. The centralized approach, in total, requires at least more than two orders of magnitude coefficients to achieve this level of accuracy.

In the light of the results we obtained, for the same tracking performance, our framework saves 99.6% of the bandwidth compared to the centralized approach. 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 a need of sending 12544 values for tracking four people. The performance of this approach is also given in Figure 5. For the same level of tracking accuracy, our framework achieves saving 96.75% compared to the decentralized approach.

5. Conclusion

Visual sensor networks constitute a new paradigm that merges two well-known topics: computer vision and sensor networks. Consequently, it poses unique and challenging problems that do not exist either in computer vision or

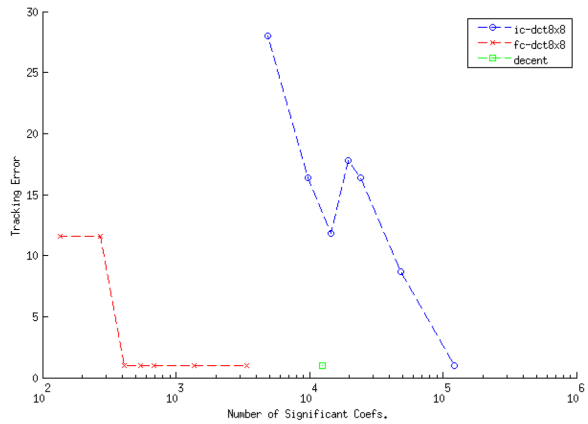
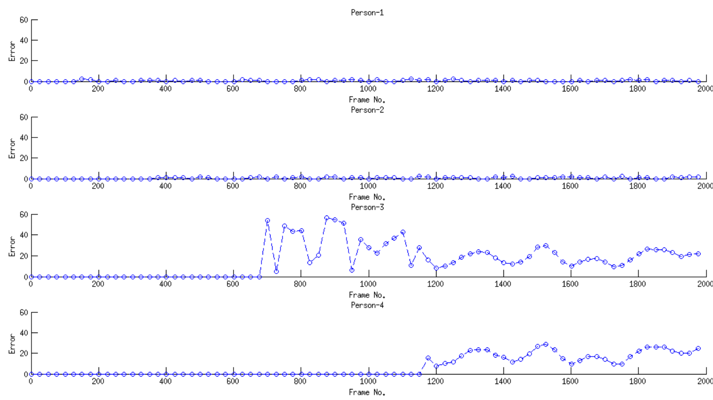


Figure 5: The average tracking errors of the centralized approach (“ic-dct8x8“), our framework (“fc-dct8x8“) both using DCT with 8×8 blocks and a decentralized method (“decent“) that directly sends likelihood functions versus the total number of significant coefficients used in reconstruction.

in sensor networks. This paper presents a novel method that can be used in VSNs for multi-camera person tracking applications. In our framework, 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 collects the multi-view image features and performs tracking. In tracking, extracting features usually results a likelihood function. Instead of sending the likelihood functions itself to the fusion node, we compress the likelihoods by first splitting them into blocks, and then transforming each block to a proper domain and taking only the most significant coefficients in this representation. By sending the most significant coefficients to the fusion node, we decrease the communication in the network. At the fusion node, the likelihood functions are reconstructed back and tracking is performed. The idea of performing goal-directed compression in a VSN is the main contribution of this work. Rather than focusing on low-level communication without regard to the final inference goal, we propose a different compressing scheme that is better matched to the final inference goal, which, in the context of this paper, is tracking.



(a)

Frame # | Cam-1 | Cam-2 | Cam-3 | Cam-4

306

749

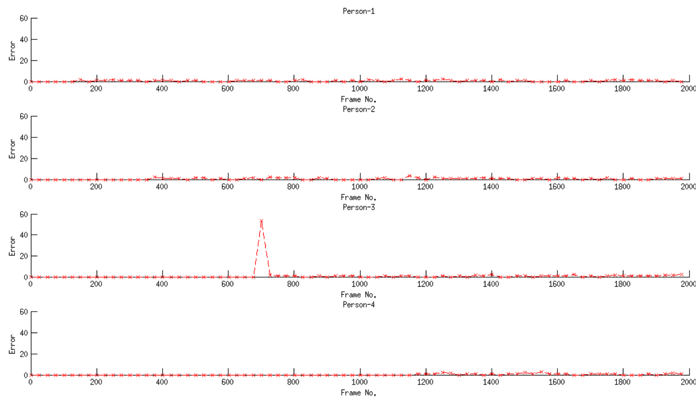
1014

1173

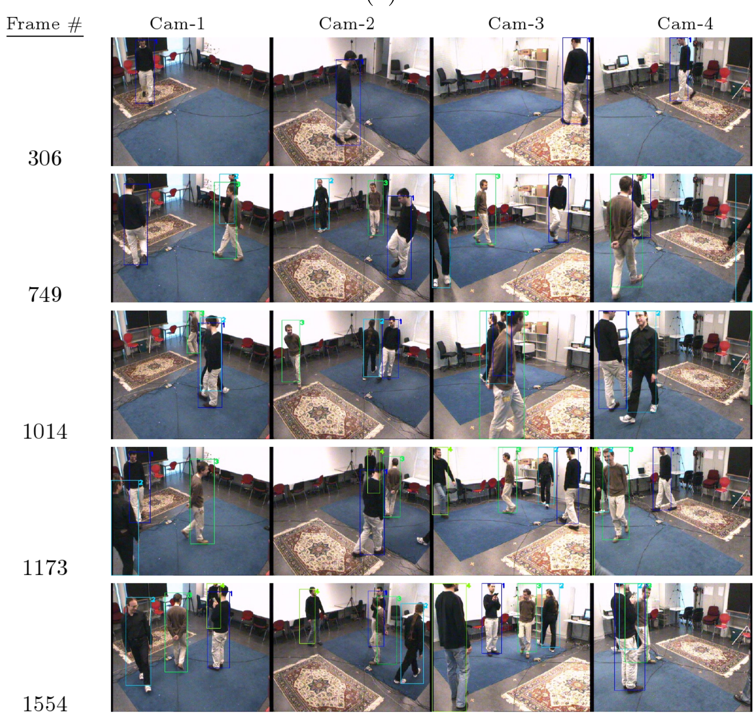
1554

(b)

Figure 6: (a) The tracking errors for each person and (b) tracking results obtained by the centralized approach using 48600 coefficients in total used in communication.



(a)



(b)

Figure 7: (a) The tracking errors for each person and (b) tracking results obtained by our framework using 3 coefficients per block used in communication.

This framework fits well to the needs of the VSN environment in two aspects: i) the processing capabilities of cameras in the network are utilized by extracting image features at the camera-level, ii) using only the most significant coefficients in network communication saves energy and bandwidth resources. 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. To the best of our knowledge, this method is the first method that compresses likelihood functions and applies this idea for VSNs. Another advantage of this framework is that it does not require the use of a specific tracking method. Without making significant changes on existing tracking methods (e.g., using simpler features, etc.), which may degrade the performance, such methods can be used within our framework in VSN environments. In the light of the experimental results, we can say that our feature compression approach can be used together with any robust probabilistic tracker in the VSN context.

We believe that trying different dictionaries that are better matched to the structure of likelihood functions, thereby, leading to further reductions in the communication load, can be a possible direction for future work. In addition, an interesting future work direction can be the implementation of our method in a real VSN setup.

Acknowledgements

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] P. V. Pahalawatta, A. K. Katsaggelos, Optimal sensor selection for video-based target tracking in a wireless sensor network, in: in Proc. International Conference on Image Processing (ICIP ?04, 2004, pp. 3073–3076.

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

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

[4] B. Song, A. Roy-Chowdhury, Robust tracking in a camera network: A multi-objective optimization framework, Selected Topics in Signal Processing, IEEE Journal of 2 (4) (2008) 582 –596. doi:10.1109/JSTSP.2008.925992.

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

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

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

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

[9] 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, 2008.

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

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

[12] D. Comaniciu, V. Ramesh, P. Meer, Kernel-based object tracking, Pattern Analysis and Machine Intelligence, IEEE Transactions on 25 (5) (2003) 564 – 577. doi:10.1109/TPAMI.2003.1195991.

[13] T.-L. Liu, H.-T. Chen, Real-time tracking using trust-region methods, Pattern Analysis and Machine Intelligence, IEEE Transactions on 26 (3) (2004) 397 –402. doi:10.1109/TPAMI.2004.1262335.

[14] P. Perez, C. Hue, J. Vermaak, M. Gangnet, Color-based probabilistic tracking, in: Heyden, A and Sparr, G and Nielsen, M and Johansen, P (Ed.), COMPUTER VISON - ECCV 2002, PT 1, Vol. 2350 of LECTURE NOTES IN COMPUTER SCIENCE, IT Univ Copenhagen; Univ Copenhagen; Lund Univ, 2002, pp. 661–675, 7th European Conference on Computer Vision (ECCV 2002), COPENHAGEN, DENMARK, MAY 28-31, 2002.

[15] G. Wallace, The jpeg still picture compression standard, Consumer Electronics, IEEE Transactions on 38 (1) (1992) xviii –xxxiv. doi:10.1109/30.125072.

[16] M. Antonini, M. Barlaud, P. Mathieu, I. Daubechies, Image coding using wavelet transform, Image Processing, IEEE Transactions on 1 (2) (1992) 205 –220. doi:10.1109/83.136597.

[17] L. Winger, A. Venetsanopoulos, Biorthogonal modified coiflet filters for image compression, in: Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on, Vol. 5, 1998, pp. 2681 –2684 vol.5. doi:10.1109/ICASSP.1998.678075.

[18] S. Mallat, A Wavelet Tour of Signal Processing, Second Edition (Wavelet Analysis & Its Applications), 2nd Edition, Academic Press, 1999.

[19] I. Daubechies, Orthonormal bases of compactly supported wavelets, Communications on Pure and Applied Mathematics.

[20] I. Daubechies, Ten lectures on wavelets, 1st Edition, Society for Industrial and Applied Mathematics, 1992.