Algorithms for Distributed Feature Extraction in Multi-camera Visual Sensor Networks

Emil Eriksson, György Dán, Viktoria Fodor
School of Electrical Engineering, KTH Royal Institute of Technology, Stockholm, Sweden
{emieri,gyuri,vfodor}@kth.se

***Abstract*—Real-time visual analysis tasks, like tracking and recognition, require swift execution of computationally intensive algorithms. Enabling visual sensor networks to perform such tasks can be achieved by augmenting the sensor network with processing nodes and distributing the computational burden among several nodes, in a way that the cameras contend for the processing nodes while trying to minimize their completion times. In this paper, we formulate the problem of minimizing the completion time of all camera sensors as an optimization problem. We propose algorithms for fully distributed optimization, analyze the existence of equilibrium allocations, and evaluate their performance. Simulation results show that distributed optimization can provide good performance despite limited information availability at low computational complexity, but the predictable and stable performance is often not provided by the algorithm that provides lowest average completion time.**

***Index Terms*—Visual feature extraction, Sensor networks, Divisible load theory, Distributed optimization**

I. Introduction

Many real-time computer vision applications, like surveillance, tracking, traffic monitoring and augmented reality, require the timely processing of visual information from several cameras [1], [2], [3], [4], [5], [6]. Timely processing allows improved application precision, and allows events in 3D to be reconstructed, but has significant computational requirements. The emergence of cheap cameras and network devices, visual sensor networks (VSNs), could in principle enable the wide-spread deployment of these popular applications, but in practice visual processing in VSNs faces two challenges. On the one hand, the high computational complexity of the image processing tasks prevents the processing to be performed locally at the cameras. On the other hand, the large amount of pixel data in the images makes it infeasible to transmit all data through the sensor network to a central processing node, considering the delay limit and the energy resources of the network nodes.

A promising solution to overcome these challenges is to augment the sensor network with processing nodes that have suitable memory and computational capacity, and to perform the image processing at the processing nodes. The processing nodes do not need to be calibrated and can be installed or replaced with ease, possibly extending the lifetime of the camera equipped nodes. Reduced maintenance and extended lifetime are particularly important for VSNs deployed in remote or hazardous areas, or in protected animal habitats. As in this case multiple sensors need to share the processing nodes as well as the wireless channel, the optimization of the distribution of the processing tasks is non-trivial.

In this paper we consider the case of local visual feature-based visual analysis [7], [8], [4], [5], where the visual analysis application utilizes the extracted features from the images captured by multiple sensors and the role of the processing nodes is to detect and to extract the descriptors within the time limit determined by the application. The sensors can leverage the processing capabilities of the processing nodes, and aim at optimizing the size of the image sub-areas sent to the processing nodes and the schedule of the transmissions of the pixel information through the shared wireless channel, so that the time when the extraction of all features from all the images is completed is minimized. The optimization is, however, based on the information that the sensors have about the system, from measurements or obtained through signaling, and thus it is important to understand how the amount of information available and the coordination between the sensors would affect the completion time.

In this paper we address this question. We provide an analytic model of the system and formulate completion time minimization as an optimization problem for the case when all system parameters are known, and the optimization can be performed centrally. We then propose fully distributed optimization algorithms that are executed by the sensors based on locally available information. The sensors can obtain the information via measurements and through signaling between the processing nodes and the sensors. We consider three different levels of available information, provide sufficient conditions for the existence of equilibrium allocations, and analyze the convergence properties of the algorithms under synchronous and asynchronous revisions. We use simulations to give insight into the convergence properties and the performance of the algorithms under various VSN topologies, and we use a multi-camera surveillance trace to assess the algorithms' performance in a realistic scenario. Our results show that predictable and low completion times may be contradicting requirements, and the choice of the revision rule plays a crucial role in determining an algorithm's performance.

The rest of the paper is organized as follows. In Section II we review related work. In Section III we describe the considered system and in Section IV we formulate the problem of completion time minimization. In Section V we

The project GreenEyes acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number: 296676.

ISBN 978-3-901882-68-5

present and analyze fully distributed algorithms for solving the completion time minimization problem. In Section VI we present numerical results and we conclude the paper in Section VII.

II. Related work

Visual analysis applications utilizing many camera nodes are discussed among others in [1], [2] for free viewpoint television, in [3], [4], [5] for localization and tracking and in [6] for high accuracy object recognition. The challenge of visual analysis at nodes with limited processing power is addressed in [7], [8], defining feature extraction schemes with low computational complexity. To decrease the transmission bandwidth requirements of pixel information, [9], [10] propose lossy image coding schemes optimized for descriptor extraction, while [11], [12], [13] give solutions to decrease the number and the size of the descriptors to be transmitted. Considering video sequences with temporal correlation, [14] limits the image areas of interest, while [15], [16] proposes intra- and inter-frame coding for the descriptors. However, results in [17] show that even under optimized extraction and coding, the processing at the camera sensor or at the sink node of the VSN leads to significant delay, which motivates the introduction of in-network processing in VSNs [18], [19].

Optimal load scheduling for distributed systems is addressed in [20], in the framework of Divisible Load Theory (DLT), with the general result that minimum completion time is achieved, if all processors finish the processing at the same time. Usually three decisions need to be made: the subset of the processors used, the order they receive their share of workload, and the division of the workload. Unfortunately, the results are specific to a given system setup. Tree networks with heterogeneous link capacities and processor speeds are addressed in [21], concluding that scheduling should be in decreasing order of the transmission capacities, while the processing speed does not affect the scheduling decision. However, [22] shows that the optimal scheduling order may be different if the processing has constant overhead, and under equal link capacities the scheduling should happen in decreasing order of the processing speeds. In [18] we show that the application of DLT for distributed visual processing is non-trivial even in the case of a single sensor node, due to the transmission overhead introduced by distributed feature extraction, and due to the dynamism of the image content in the video. Contrary to previous work, in this paper we investigate the optimal distribution of the processing load for the case of several camera sensors sharing the processing resources, in the framework of distributed optimization.

Related to our work is the problem of learning in games [23], [24]. Studies of learning in games usually consider models of perfect information for the analysis of convergence [23], [24], [25], [26]. Recent works on experimentation dynamics and regret testing models consider that players can only observe their own payoffs [27], [28], [29], but these learning models provide asymptotic convergence guarantees, and thus convergence is prohibitively slow. In this paper we consider three models of imperfect information, and provide equilibrium existence and convergence results. Our results also highlight the potential trade-off between predictable and good performance in learning.

III. System model

We consider a visual sensor network (VSN) that consists of a set of sensor nodes $\mathcal{S}$, $|\mathcal{S}| = S$, and a set of processing nodes $\mathcal{N}$, $|\mathcal{N}| = N$. Sensor node $s \in \mathcal{S}$ captures a sequence $\mathcal{I}_s = \{1, ...\}$ of images of width w pixels. For the delegation of the computation, sensor node s divides image i into $V_s^i \leq N$ vertical slices. This scheme was referred to as area-split in [30], [31]. We define slice v using its normalized leftmost and rightmost horizontal coordinates, $x_{s,v-1}^i$ and $x_{s,v}^i$, i.e., $x_{s,0}^i = 0$ and $x_{s,V_s^i}^i = 1$, and we define the cutpoint location vector for image i as $x_s^i = \{x_{s,0}^i, \ldots, x_{s,V_s^i}^i\}$. For convenience, we use $y_{s,v}^i = x_{s,v}^i - x_{s,v-1}^i$ to denote the normalized width of slice v, and we define $y_{s,v}^i = 0$ for $v \leq 0$ and for $v > V_s^i$. Thus, by definition, $\sum_{v=1}^{V_s^i} y_{s,v}^i = 1$. Sensor s transmits slice $1 \leq v \leq V_s^i$ to processing node $d_s^i(v) \in \mathcal{N}$ for processing. We define d_s^i as a sequence with V_s^i distinct elements, and with slight abuse of notation we use $n \in d_s^i$ if $d_s^i(v) = n$ for some $1 \leq v \leq V_s^i$, i.e., node n is used by sensor s. We use the notation $\mathcal{p}_s^i(n)$ for the slice that sensor s assigns to node n. That is, we refer to d_s^i as the assignment by sensor s, and to $\mathcal{p}_s^i$ as its inverse.

A. Visual feature extraction

Each processing node n computes local visual features from the image slices assigned it. The computation of local features starts with interest point detection in an image, by applying a blob detector or an edge detector at every pixel of an image [32], [33], [8]. For each pixel, the detector computes a response score based on a square area centered around it. We denote the side length of the square normalized by the width of the image by $2o$. The side length $2ow$ of the square (in pixels) depends on the applied detector.

A pixel is identified as an interest point if the response score exceeds the detection threshold $\vartheta \in \mathbf{\Theta} \subseteq \mathbb{R}^+$. The time it takes to detect interest points can be modeled as a linear function of image size in pixels and of the number $\xi_{s,v}^i = f_i(x_{s,v}^i, x_{s,v-1}^i)$ of interest points detected; this model was validated recently on a BeagleBone Black single board computer in [18] and on an Intel Imote2 platform in [17]. We can thus model the detection time for slice v from sensor s at processing node n as a function of the image slice width $y_{s,v}^i$ and of the number $\xi_{s,v}^i$ of interest points detected in the image slice as an affine function $P_n(y_{s,v}^i + \alpha_f \xi_{s,v}^i)$, where P_n is the per unit processing time of node n. Note that $\xi_{s,v}^i$ is unknown before processing image slice v, but efficient low-complexity predictors exist, such as the last value predictor [18].

After detection, a feature descriptor is extracted for each interest point by comparing pixel intensities. The time it takes to extract the descriptors can be modeled as a linear function of the number $\xi_{s,v}^i$ of interest points detected, as shown in [18]. We can thus model the detection and extraction time as $P_n(y_{s,v}^i + (\alpha_f + \alpha_e)\xi_{s,v}^i) = P_n(y_{s,v}^i + \alpha_d \xi_{s,v}^i)$. We consider that α_f, α_e and thus α_d are the same for all processing nodes, which is reasonable if the nodes have a similar computer architecture (e.g., instruction set).

In applications where features extracted from images captured by multiple cameras are needed, e.g., in the case of

multi-camera tracking for updating a hidden-Markov model or a particle filter, the extraction of the features from all images should finish at the same time. We thus consider that if a processing node n has to process image slices from different sensors simultaneously, then it allocates its processing power in a way that ensures that the processing of all slices is completed at the same time.

B. Communication Model

The nodes communicate using a wireless communication protocol, such as IEEE 802.15.4 or IEEE 802.11, in which transmissions suffer from packet losses due to wireless channel impairments. As measurement studies show [34], [35], the loss burst lengths at the receiver have low mean and variance in the order of a couple of frames [36], [37]. Therefore, a widely used model of the loss process is a low-order Markov-chain, with fast decaying correlation and short mixing time. In the system we consider, the amount of data to be transmitted to the processing nodes is relatively large, and therefore it is reasonable to model the average transmission time from sensor s to processing node n as a linear function of the amount of transmitted data. We denote the transmission time coefficient by $C_{s,n}$, which can be interpreted as the average per image transmission time, including potential retransmissions. As the throughput is close to stationary over short timescales, $C_{s,n}$ can be estimated [38]. When there are several sensors transmitting data, the MAC protocol provides airtime fairness for the transmitters [39], thus the actual transmission time coefficient is proportional to the number of sensors transmitting. For example, when there are S sensors transmitting, the actual transmission time coefficient is $SC_{s,n}$.

Recall that interest point detection involves applying a square filter of size $2o$ at each pixel. Thus, for correct operation, each slice v has to be appended by an overlap area of width o on one or both sides. The resulting regions of overlap in adjacent slices could in principle be transmitted in multicast to the appropriate processing nodes, but experimental results show that multicast transmission suffers from low throughput in practice due to lack of link layer retransmissions and missing channel quality information [40], we thus consider that all data transmissions are done using unicast.

The processing nodes can receive data and perform processing simultaneously. Thus, a processing node can start processing slice v from sensor s after it has received o (for $v = 1$) and $2o$ (for $1 < v \leq V_s^i$) worth of data. Motivated by the rapid increase of 802.11 wireless transmission capacities, we consider that processing is slower than transmission, i.e.,

$$min_n P_n(1 + \alpha_d) \geq S \max_{s,n} C_{s,n}, \tag{1}$$

and thus processing an image slice takes at least as much time as receiving it.

IV. Completion Time and Problem Formulation

Using the model of transmission and processing above, let us consider the completion time of the processing of image $i \in \mathcal{I}_s$ captured by sensor s. Figure 1 illustrates the transmission and processing of slices to $N = 3$ processing nodes. Let us denote by $t^b_{s,v}$ the time instant when processing node $d^i_s(v)$ receives the first bit of slice v from sensor s, by $t^p_{s,v}$ the time instant when processing of slice v from sensor s starts at processing node $d^i_s(v)$ (i.e., after receiving the overlap), by $t^r_{s,v}$ the time instant when processing node $d^i_s(v)$ receives the last bit of slice v from sensor s, and by $t^c_{s,v}$ the time instant when processing of slice v from sensor s is completed at processing node $d^i_s(v)$.

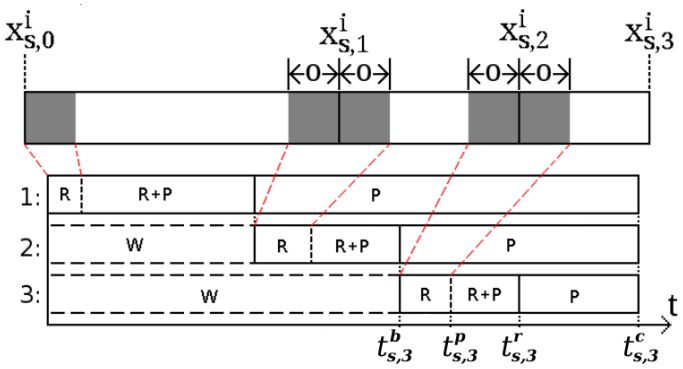

Figure 1: Transmission and processing schedule of slices from a single sensor to three processing nodes. 'R' stands for receiving the image slice, 'P' for processing, and 'W' for waiting.

Observe that the time $t^r_{s,v} - t^b_{s,v}$ it takes node s to transmit slice v to processing node n depends on the number of sensor nodes that are transmitting simultaneously, which depends on the cutpoint location vectors $x^i_{s'}$ and on the assignment functions $d^i_{s'}$ of the other sensors. To capture the dependence of the transmission time on the sensors' cutpoint location vectors $(x^i_s)_{s\in\mathcal{S}}$ and assignment functions $(d^i_s)_{s\in\mathcal{S}}$, we define the experienced transmission time coefficient

$$\tilde{C}_{s,n}(\mathbf{x^i}, \mathbf{d^i}) = \begin{cases} (t^r_{s,v} - t^b_{s,v})/(y^i_{s,v} + o) & \text{for } v = 1, V^i_s \\ (t^r_{s,v} - t^b_{s,v})/(y^i_{s,v} + 2o) & \text{for } 1 < v < V^i_s \end{cases}. \tag{2}$$

Similarly, the time it takes processing node n to complete the processing of slice v sent by sensor s depends on whether or not the processing node has to process slices from other sensors simultaneously. We define the experienced processing time coefficient of sensor s at processing node n as $\tilde{P}_{s,n}(\mathbf{x^i}, \mathbf{d^i}) = (t^c_{s,v} - t^p_{s,v})/(y^i_{s,v} + \alpha_d \xi^i_{s,v})$.

We can express the mean completion time of slice v delegated by sensor s to processing node $n = d^i_s(v)$ as a function of the experienced transmission time coefficients and of the experienced processing time coefficients. For the first slice, i.e., $n = d^i_s(1)$, we have

$$T^i_{s,n}(\mathbf{x^i}, \mathbf{d^i}) = \tilde{C}_{s,n}(\mathbf{x^i}, \mathbf{d^i})o + \tilde{P}_{s,n}(\mathbf{x^i}, \mathbf{d^i})(y^i_{s,1} + \alpha_d \xi^i_{s,1}). \tag{3}$$

For the remaining slices, i.e., $n = d^i_s(v)$, $v > 1$, the completion time depends also on the transmission times of previous slices

$$\begin{aligned} T^i_{s,n}(\mathbf{x^i}, \mathbf{d^i}) &= \tilde{C}_{s,d^i_s(1)}(\mathbf{x^i}, \mathbf{d^i})[y^i_{s,1} + o] + \tilde{C}_{s,n}(\mathbf{x^i}, \mathbf{d^i})2o \\ &+ \sum_{\nu=2}^{v-1} \tilde{C}_{s,d^i_s(\nu)}(\mathbf{x^i}, \mathbf{d^i})[y^i_{s,\nu} + 2o] \\ &+ \tilde{P}_{s,n}(\mathbf{x^i}, \mathbf{d^i})(y^i_{s,v} + \alpha_d \xi^i_{s,v}). \end{aligned} \tag{4}$$

Finally, we define the completion time of image i for sensor s as the completion time of the processing node that finishes

last

$$T_s^i(\mathbf{x^i}, \mathbf{d^i}) = \max_{n \in d_s^i}(T_{s,n}^i(\mathbf{x^i}, \mathbf{d^i})). \quad (5)$$

Observe that the maximum is taken only over the processing nodes used by sensor s.

A. Completion Time Minimization

Given the set of sensor nodes $\mathcal{S}$, the set of processing nodes $\mathcal{N}$, the transmission and processing time coefficients $C_{s,n}$ and P_n, we can formulate the problem of minimizing the completion time for a single image i as a combinatorial optimization problem

$$\min_{(\mathbf{x^i}, \mathbf{d^i})} t \quad (6)$$

s.t.

$$\begin{aligned} T_s^i(\mathbf{x^i}, \mathbf{d^i}) &\leq t \ \forall s \in \mathcal{S} && (7)\\ x_{s,v-1}^i - x_{s,v}^i &\leq -o \ 1 \leq v \leq V_s^i && (8)\\ x_{s,v}^i w &\in \{1, \ldots, w\} \ 1 \leq v \leq V_s^i && (9) \end{aligned}$$

where w is the width of the image in pixels. Solving this optimization problem in the considered VSN scenario faces three challenges. First, the solution has to consider all profiles of partial permutations $(d_s^i)_{s \in \mathcal{S}}$, and for each profile find the optimal allocation vector $\mathbf{x^i}$. Thus, given the computational constraints in the sensors it may be infeasible to solve even moderate instances of the problem. Second, even if every sensor could solve the optimization problem, there may be multiple solutions, and deciding which solution to use would require communication between the sensors, which introduces delay. Third, the sensors may not have sufficient information to formulate the optimization problem, e.g., because the interest point distribution is unknown before processing an image. We are thus interested in fully distributed solutions that require no communication between the sensors.

V. Distributed Algorithms

We consider distributed solutions in which the sensors use information they can observe (e.g., measure) or they may receive from the processing nodes. We denote the information available to sensor s before processing image i by Υ_s^i.

To make the information Υ_s^i explicit, we introduce the mean expected transmission time coefficient $\bar{C}_{s,n}(x_s^i, d_s^i|\Upsilon_s^i)$, and the mean expected processing time coefficient $\bar{P}_{s,n}(x_s^i, d_s^i|\Upsilon_s^i)$. Thus, based on the information Υ_s^i, the expected mean completion time of sensor s can be expressed for $n = d_s^i(1)$ as

$$\begin{aligned} \bar{T}_{s,n}^i(x_s^i, d_s^i|\Upsilon_s^i) &= \bar{C}_{s,n}(x_s^i, d_s^i|\Upsilon_s^i)o \quad (10)\\ &+ \bar{P}_{s,n}(x_s^i, d_s^i|\Upsilon_s^i)(y_{s,1}^i + \alpha_d \xi_{s,1}^i), \end{aligned}$$

and for the remaining slices, i.e., $n = d_s^i(v)$, $v > 1$, we have

$$\begin{aligned} \bar{T}_{s,n}^i(x_s^i, d_s^i|\Upsilon_s^i) &= \bar{C}_{s,d_s^i(1)}(x_s^i, d_s^i|\Upsilon_s^i)[y_{s,1}^i + o] \quad (11)\\ &+ \sum_{\nu=2}^{v-1} \bar{C}_{s,d_s^i(\nu)}(x_s^i, d_s^i|\Upsilon_s^i)[y_{s,\nu}^i + 2o]\\ &+ \bar{C}_{s,n}(x_s^i, d_s^i|\Upsilon_s^i)2o\\ &+ \bar{P}_{s,n}(x_s^i, d_s^i|\Upsilon_s^i)(y_{s,v}^i + \alpha_d \xi_{s,v}^i). \end{aligned}$$

Finally, sensor s aims to minimize its expected completion time

$$\bar{T}_s^i(x_s^i, d_s^i|\Upsilon_s^i) = \max_{n \in d_s^i} \bar{T}_{s,n}^i(x_s^i, d_s^i|\Upsilon_s^i). \quad (12)$$

The times when the sensors can revise their allocations is determined by the *revision opportunity*, which can be either synchronous or asynchronous.

Definition 1. Revision opportunity: *Asynchronous revision allows one sensor $s \in \mathcal{S}$ to update its allocation upon each image i. Synchronous revision allows every sensor to update its allocation upon every image i.*

While in a VSN synchronous revision is easy to implement, asynchronous revision could, e.g., be implemented by configuring a static revision order through modulo division of the image sequence number.

A basic requirement in networked system design is to have predictable, constant performance. For a distributed algorithm, constant system performance can be guaranteed if the algorithm reaches an allocation at which the sensors would settle, and thus the completion time would remain constant, assuming that the image contents do not change.

Definition 2. Equilibrium: *An equilibrium is an assignment profile $(d_s^i)_{s \in \mathcal{S}}$ and $(x_s^i)_{s \in \mathcal{S}}$ compared to which no sensor s can decrease its expected completion time by deviating unilaterally, given the information Υ_s^i.*

In the case of perfect information (i.e., Υ_s^i contains all transmission time coefficients, processing time coefficients and interest point distributions), the notion of an equilibrium corresponds to the notion of a Nash equilibrium in game theory [23], [25].

In the following we consider and analyze three different models of information Υ_s^i that the sensors base their revisions upon. We define an *algorithm* to be the *combination* of the *revision opportunity* and of the revision made by the sensors based on the *information available* to them. For the analysis we assume that the interest points are evenly spaced along the horizontal axis in every image, we can thus let $\alpha_d = 0$. For notational convenience we omit the index i whenever the expected transmission time and processing time coefficients are used.

A. Measurement Only (MO) Information

We start with considering a system with no signaling between the processing nodes and the sensor nodes, thus, all parameters need to be estimated by the sensors. We call this the measurement only (*MO*) scenario. Sensor s can measure the experienced transmission time coefficient $\tilde{C}_{s,n}$ to processing node $n \in d_s^i$, and it can measure the experienced processing time coefficient $\tilde{P}_n$ of processing node $n \in d_s^i$. In lack of more information, sensor s would estimate the sensitivity of the completion time at node n to the slice width as $\frac{dt_{s,n}^c}{dy_{s,d_s^i(v)}^i} = \tilde{P}_n$, and would use it as follows.

Let us consider sensor s and let us derive the optimal offloading for a particular assignment function d_s, given $\bar{C}_{s,n} = \tilde{C}_{s,n}$ and $\bar{P}_n = \tilde{P}_n$. In order to compute the optimal

assignment d_s and the optimal allocation $\mathbf{x}_s$, we recall a fundamental result from divisible load theory [20].

Lemma 1. *The completion time T_s^i for sensor s is minimized if all processing nodes $n \in d_s$ complete processing at the same time. Furthermore, if all processing time coefficients are equal then the optimal assignment is in increasing order of the transmission time coefficients $C_{s,n}$ (i.e., use the node with fastest link first).*

This result is illustrated in Figure 1 for $N = 3$ processing nodes. The fact that at optimality all used processing nodes $n \in d_s$ complete processing at the same time allows us to establish a relationship between the optimal slice widths for a particular assignment d_s as

$$\bar{P}_{d_s(1)} y_{s,1} = \bar{C}_{s,d_s(1)} y_{s,1} + \bar{C}_{s,d_s(2)} 2o + \bar{P}_{d_s(2)} y_{s,2} \quad (13)$$

$$\bar{P}_{d_s(2)} y_{s,2} = \bar{C}_{s,d_s(2)} y_{s,2} + \bar{C}_{s,d_s(3)} 2o + \bar{P}_{d_s(3)} y_{s,3} \quad (14)$$

$$\dots$$

$$\bar{P}_{d_s(V-1)} y_{s,V-1} = \bar{C}_{s,d_s(V-1)} y_{s,V} + \bar{C}_{s,d_s(V)} 2o + \bar{P}_{d_s(V)} y_{s,V}. \quad (15)$$

We can use the above equations to formulate a recursive expression for the normalized width of slices $1 \leq v < V$ as

$$y_{s,v} = \frac{2o\bar{C}_{s,d_s(v+1)}}{\bar{P}_{d_s(v)} - \bar{C}_{s,d_s(v)}} + \frac{\bar{P}_{d_s(v+1)}}{\bar{P}_{d_s(v)} - \bar{C}_{s,d_s(v)}} y_{s,v+1}, \quad (16)$$

and together with the normalization constraint $\sum_{v=1}^{V} y_{s,v} = 1$ we can compute the optimal allocation vector. Based on this recursive expression we can formulate the following result.

Lemma 2. *Given an assignment function d_s, the optimal slice widths $y_{s,v}^*$ are insensitive to the scaling of the expected transmission time coefficients $\bar{C}_{s,n}$ and of the expected processing time coefficients $\bar{P}_n$ by the same factor $\sigma > 0$.*

Proof: Observe that based on (16) the ratio $y_{s,v}/y_{s,v+1}$ does not change as long as the ratios $\frac{\bar{C}_{s,d_s(v+1)}}{\bar{P}_{d_s(v)} - \bar{C}_{s,d_s(v)}}$ and $\frac{\bar{P}_{d_s(v+1)}}{\bar{P}_{d_s(v)} - \bar{C}_{s,d_s(v)}}$ are unchanged. Since the optimal slice widths are obtained by using the fact that $\sum_{v=1}^{V} y_{s,v} = 1$, the optimal slice widths $y_{s,v}^*$ are only a function of $y_{s,v}/y_{s,V}$, and thus the result follows. ∎

Lemma 3. *Let d_s^* be the assignment function that together with cutpoint location vector x_s^* minimizes the completion time for sensor s. Then d_s^* and x_s^* are optimal after scaling all expected transmission time coefficients $\bar{C}_{s,n}$ and all expected processing time coefficients $\bar{P}_n$ by the same factor $\sigma > 0$.*

Proof: Observe that by Lemma 2, the cutpoint location vector x_s^* remains optimal for d_s^* after scaling. Furthermore, the completion time is a linear function of the transmission and of the processing time coefficients, and thus all completion times $T_s(x_s, d_s)$ are scaled by σ. Thus, $\bar{T}_s(x_s^*, d_s^*)$ remains minimal after scaling. ∎

We are now ready to prove a sufficient condition for an equilibrium allocation to exist for the *MO* scenario.

Theorem 1. *Consider a VSN with symmetric transmission time coefficients $C_{s,n} = C_{s',n}, \forall s, s' \in \mathcal{S}$, and denote by d_s^* and by x_s^* the assignment function and the corresponding cutpoint location vector that are optimal for $\tilde{C}_{s,n} = C_{s,n}$ and $\tilde{P}_n = P_n$, i.e., when s is the only sensor in the system. If the sensors use all processing nodes, i.e., $d_s^* = \mathcal{N}$, then an equilibrium allocation exists under* MO*, and $(d_s^*)_{s\in S}$ and $(x_s^*)_{s\in S}$ is an equilibrium allocation profile.*

Proof: Observe that for every sensor s the experienced transmission time coefficients $\tilde{C}_{s,n} = S \times C_{s,n}$ and the experienced processing time coefficients $\tilde{P}_n = S \times P_n$. By Lemma 3 the optimal assignment function d_s^* and the optimal cutpoint location vector x_s^* are insensitive to scaling and thus they remain optimal for all sensors. Consequently, $(d_s^*, x_s^*)_{s\in\mathcal{S}}$ is an equilibrium. ∎

Thus, an equilibrium may exist, and *synchronous* revision starting from an appropriate initial allocation (d_s^* and x_s^*) would reach an equilibrium after one revision. It is thus important to understand whether, in general, the completion time would be minimal in an equilibrium. The following result shows that this is not the case.

Proposition 1. *An equilibrium allocation under the* MO *scenario may not be optimal.*

Proof: We prove the proposition through an example. Let $S = 2$, $N = 2$, $C_{s,n} = 1$ and $P_n = 5$, $o = 0.1$. Then, in isolation $d_s^* = (1, 2)$, $x_s^* = (0, \frac{5.2}{9}, 1)$, which is an equilibrium according to Theorem 1, with completion time $T_s^* = 5.98$. By the assignment function of sensor 2 to $d_s^* = (2, 1)$, with the same cutpoint location vectors $x_s^* = (0, \frac{5.2}{9}, 1)$ the completion time is $T_s = 5.2 < T_s^*$, thus $d_s^* = (1, 2)$, x_s^* cannot be optimal. ∎

Thus, while the *MO* scenario requires no signaling, even if sensors would converge to an equilibrium using synchronous or asynchronous revisions, the performance may not be optimal.

B. Transmission Time (TT) Information

In the second scenario each sensor can measure its transmission and processing time coefficients as in the *MO* scenario. Besides, upon completion, every processing node n broadcasts to each sensor s its processing time coefficient P_n, and the times $(t^b_{s',\mathrm{p}^i_{s'}(n)}, t^r_{s',\mathrm{p}^i_{s'}(n)})$ and the corresponding slice widths $y^i_{s',\mathrm{p}^i_{s'}(n)}$ for all sensors that used node n, i.e., $s' \in \{s' | \exists v \text{ s.t. } d^i_{s'}(v) = n\}$. We refer to this as the transmission time (*TT*) scenario.

Observe that $(t^b_{s',\mathrm{p}^i_{s'}(n)} - t^r_{s',\mathrm{p}^i_{s'}(n)})$ is a known linear function of $y^i_{s',\mathrm{p}^i_{s'}(n)}$ and of the transmission time coefficient $C_{s',n}$, and thus every sensor s can compute $C_{s',n}$ for $n \in d^i_{s'}$. The sensors can also compute the time $t^p_{s',\mathrm{p}^i_{s'}(n)}$ for every processing node n.

In order to get analytic insight into the problem, let us make the simplifying assumption that the experienced transmission times do not change as an effect of the sensors' assignments. Under this simplifying assumption we can show that an equilibrium allocation exists for the *TT* scenario.

Theorem 2. *There is an equilibrium allocation* $(\mathbf{x}^*, \mathbf{d}^*)$ *such that no sensor can decrease its completion time by unilaterally changing its allocation.*

Proof: For an allocation $(\mathbf{x}, \mathbf{d})$ let us define the vector $\tau(\mathbf{x}, \mathbf{d}) = (T_{s,1}, \ldots, T_{s,N})$ of completion times sorted in decreasing order, i.e., $T_{s,1}(\mathbf{x}, \mathbf{d}) \geq T_{s,2}(\mathbf{x}, \mathbf{d})$, etc.

Let us now consider that every sensor s chooses a cutpoint location vector x_s^1 and an assignment vector d_s^1 that minimizes its completion time assuming there are no other sensors. We refer to this initial assignment as $(\mathbf{x^1}, \mathbf{d^1})$.

Let us consider now that given assignment $(\mathbf{x^i}, \mathbf{d^i})$, $i \geq 1$, a single sensor s revises its assignment and/or allocation to (x'_s, d'_s) and thereby it minimizes its completion time given the assignments $\mathbf{d^i_{-s}}$ and allocations $\mathbf{x^i_{-s}}$ of the other sensors, i.e.,

$$T_s((x'_s, \mathbf{x^i_{-s}}), (d'_s, \mathbf{d^i_{-s}})) < T_s(\mathbf{x^i}, \mathbf{d^i}). \tag{17}$$

Let us denote by $(\mathbf{x^{i+1}}, \mathbf{d^{i+1}}) = ((x'_s, \mathbf{x^i_{-s}}), (d'_s, \mathbf{d^i_{-s}}))$ the resulting assignment profile. Observe that (17) implies that

$$max_{n \in d'_s} T_{s,n}((x'_s, \mathbf{x^i_{-s}}), (d'_s, \mathbf{d^i_{-s}})) < max_{n \in d_s} T_s^n(\mathbf{x^i}, \mathbf{d^i}). \tag{18}$$

At the same time, for $n \notin d'_s$ we have $T_s^n((x'_s, \mathbf{x^i_{-s}}), (d'_s, \mathbf{d^i_{-s}})) \leq T_s(\mathbf{x^i}, \mathbf{d^i})$. Thus,

$$\tau(\mathbf{x^{i+1}}, \mathbf{d^{i+1}}) <_L \tau(\mathbf{x^i}, \mathbf{d^i}), \tag{19}$$

where $<_L$ stands for *lexicographically smaller*. Since among all vectors τ of ordered completion times there is a vector that is lexicographically minimal, the vector that corresponds to all sensors completing at the same time. Thus, there is an allocation $(\mathbf{x}, \mathbf{d})$ compared to which no sensor can decrease its completion time. ∎

Observe that the proof is based on an *asynchronous* revision opportunity. A consequence of the proof is that using *asynchronous* revision the sensors can reach an equilibrium in the *TT* scenario.

Corollary 1. *Assume that sensors follow the asynchronous revision opportunity. Then the sensors' allocations converge to an equilibrium under the* TT *scenario.*

Thus, using asynchronous revision under the *TT* scenario guarantees convergence to equilibrium. Unfortunately, this result cannot be extended to the case of *synchronous* revision, as shown by the following example.

Example 1. *Let* $\mathcal{S} = \{1, 2\}$, $\mathcal{N} = \{1, 2\}$, $C_{s,1} = 1$, $C_{s,2} = 2$, $P_n = 5$, *and* $o = 0.1$. *Let the initial assignment be* $d_1^1 = (1, 2)$, $d_2^1 = (2, 1)$ *and the allocations* $x_s^1 = (0, 0.6, 1)$. *Then* $T^1 = (5.2, 5.4)$ *and the senors will update their allocations to* $x_1^2 = (0, 0.62, 1)$ *and* $x_2^2 = (0, 0.58, 1)$, *which results in* $T^2 = (5.4, 5.2)$. *The next allocation is* $x_s^3 = (0, 0.6, 1) = x_s^1$, *thus the sensors will cycle between these two allocations.*

Nonetheless, it is easy to verify that the cycle in the above example can be avoided if upon processing image i sensor s uses $x_s^i = \frac{1}{S} x'_s + \frac{S-1}{S} x_s^{i-1}$, as in this case the sensors would reach the equilibrium $x_1^2 = (0, 0.61, 1)$ and $x_2^2 = (0, 0.59, 1)$ in one step. We call this revision rule the *synchronous/S* revision.

C. Processing Time (PT) Information

The third scenario is between the *MO* and the *TT* scenarios in terms of available information. Each sensor can measure its transmission and processing time coefficients as in the previous scenario. Besides, upon completion, every processing node broadcasts to each sensor its processing time coefficient P_n and the time $t_n^p = \min_{s'}(t^p_{s', \mathrm{p}^i_{s'}(n)})$ when it started the processing. We refer to this as the processing time (*PT*) scenario.

The time $t^c_{s,v}$ at which the processing of slice v completes satisfies

$$t^c_{s,v} = \min_{s'}(t^p_{s', \mathrm{p}^i_{s'}(n)}) + P_n(y^i_{s,v} + \alpha_d \xi^i_{s,v}) \tag{20}$$

$$+ P_n \sum_{s' \in \mathcal{S} \setminus \{s\}} (y^i_{s', \mathrm{p}^i_{s'}(n)} + \alpha_d \xi^i_{s', \mathrm{p}^i_{s'}(n)}), \tag{21}$$

and hence sensor s is able to compute the total workload that n received from the other sensors (i.e., (21)), as it knows the time when n started to process the first slice (the first term in (20)). Furthermore, it can compute the remaining workload at $t^p_{s,v}$ as $(t^c_{s,v} - t^p_{s,v})/P_n - (y^i_{s,v} + \alpha_d \xi^i_{s,v})$.

Based on these data, sensor s can estimate the sensitivity of the experienced processing time coefficient at node n to the slice width as $\frac{d\tilde{P}_{s,n}}{dy^i_{s,d_s(v)}} = P_n$ as long as $t^p_{s,v} + P_n(y^i_{s,v} + \alpha_d \xi^i_{s,v}) < t^p_n$, and s can influence $t^p_{s,v}$ at node n through the assignment function d_s^i and the cutpoint location vector x_s^i, and can use these to compute the expected completion time. In what follows we investigate the convergence properties and the resulting completion times using simulations.

VI. Numerical Results

We consider three VSN topologies for the numerical evaluation. In the first topology the system consists of $S = 2$ sensors and $N = 4$ processing nodes. The transmission time coefficients are $C_{s,n} = 0.3\,\mathrm{s/image}$, $\forall s \in \mathcal{S}, n \in \mathcal{N}$. In this topology all processing nodes are equivalent from the sensors' points of view, it is only the interaction between the sensors that affects the behaviour of the algorithms. We refer to this topology as *Homogeneous*. Observe that this topology satisfies the conditions of Theorem 1 and of Theorem 2, and thus an equilibrium allocation exists for the *MO* and the *TT* scenarios.

In the second topology we consider a system with $S = 4$ sensor nodes and $N = 4$ processing nodes. The transmission time coefficients are

$$C = \begin{pmatrix} 0.25 & 0.25 & 0.05 & 0.05 \\ 0.05 & 0.25 & 0.25 & 0.05 \\ 0.05 & 0.05 & 0.25 & 0.25 \\ 0.25 & 0.05 & 0.05 & 0.25 \end{pmatrix} \mathrm{s/image},$$

which corresponds to a system where each sensor is located at a distinct corner of a square, e.g., directed towards the center of the square, and a processing node is placed at the midpoint of each edge. We refer to this topology as *Square*.

In the third topology there are $S = 2$ sensors, and $N = 2$ to $N = 5$ processing nodes. The two sensors are at two adjacent corners of a square, and the two first processing nodes at the remaining two corners. The remaining processing nodes

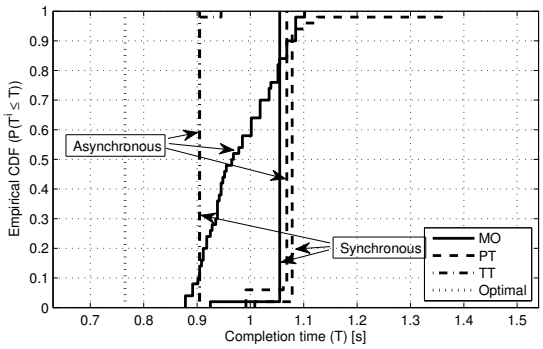

Figure 2: Cumulative distribution function of the completion time for the *Homogeneous* topology.

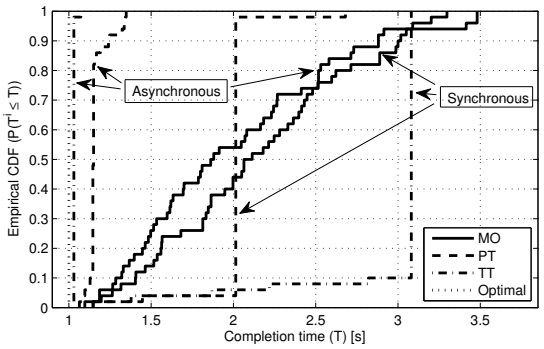

Figure 3: Cumulative distribution function of the completion time for the *Square* topology.

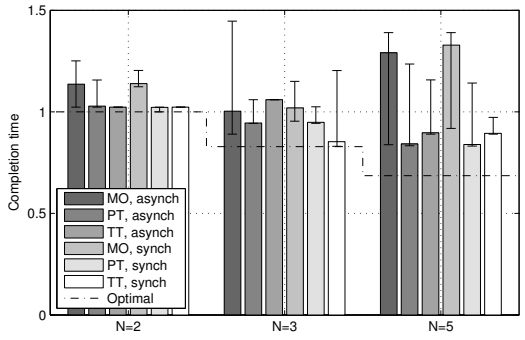

Figure 4: Mean, minimum and maximum completion times vs. number of nodes for the *Line* topology.

are placed on the edge connecting the first two nodes at even intervals. The transmission time coefficients for $N = 5$ are

$$C = \begin{pmatrix} 0.050 & 0.063 & 0.100 & 0.163 & 0.250 \\ 0.250 & 0.163 & 0.100 & 0.063 & 0.050 \end{pmatrix} {}^{\mathrm{s}}/_{\mathrm{image}}.$$

We refer to this topology as the *Line* topology.

For each topology we evaluate the completion time of the system under the *MO*, *PT*, and *TT* scenarios with both asynchronous and synchronous revisions, i.e., 6 algorithms, and compare the results to the optimal completion time. For obtaining an initial estimate of the transmission and processing time coefficients, the sensors use a bootstrap cutpoint location vector x_s^i in which $y_{s,n}^i = max(o, 1/N)$.

A. Convergence vs. Performance

We first evaluate the algorithms on an image sequence in which every image has a uniform interest point distribution, which is the case on average [30], [31], and we can thus use $\alpha_d = 0$. For the overlap we use $o = 0.06$, which corresponds to a filter width of 84 pixels in an image of width $w = 720\,\mathrm{pixels}$. The processing time coefficient is $P_n = 1\,{}^{\mathrm{s}}/_{\mathrm{img}}$ for all processing nodes, and serves as a reference: a single processing node would complete processing an image in 1 second of time. This configuration allows us to observe the convergence properties of the algorithms and how convergence affects the completion time.

Figure 2 shows the cumulative distribution function (CDF) of the completion times for the *Homogeneous* topology for the three scenarios, using the asynchronous and the synchronous/S revision. To interpret the figure, observe that an algorithm will produce a constant completion time after it reaches an equilibrium, hence a step function-like CDF. The *TT* scenario results in approximately the same completion time at equilibrium for synchronous and for asynchronous revision, hence the two curves overlap.

Based on the figure, we can see that all algorithms reach an equilibrium after a few images, except for the asynchronous *MO* algorithm. The reason for the non-convergence of the asynchronous *MO* algorithm is twofold. First, when sensor s updates its slice sizes, its estimates $\tilde{P}_{s,n}$ of the processing time coefficients are correct only if the proportion of the slice size from s to the total size of the slices from all sensors remains constant. Second, as slices evolve based on the experienced processing time coefficients, eventually an image i is reached where the sensors will change the assignment function d_s^i, and changing the assignment function has a large impact on the experienced processing time coefficients of all sensors in the system, preventing convergence.

The non-convergence of the asynchronous *MO* algorithm is remarkable with respect to the game theoretical literature of learning. Observe that the *MO* scenario corresponds to a best reply dynamics based on imperfect information. For game theoretical models of learning under perfect information, e.g., using best- and better reply dynamics and fictitious play, convergence under asynchronous revision is typically necessary but not sufficient for convergence under synchronous revision [23], [24]. In contrast, for the *MO* scenario convergence happens under synchronous revision, but not under asynchronous revision, which emphasizes the importance of information availability for the convergence of learning rules (c.f., Theorem 1).

It is interesting to note that even though all but the asynchronous *MO* algorithm converge after a few images, only the average system completion time achieved in equilibrium for the *TT* scenario is lower than that of the non-convergent asynchronous *MO* algorithm, which is $0.98\,\mathrm{s}$. This observation raises an important system design question, i.e., whether it is preferable to have a predictable constant but higher completion time or a lower average completion time that changes over time.

Figure 3 shows the CDF of the completion times for the *Square* topology for the three scenarios combined with the

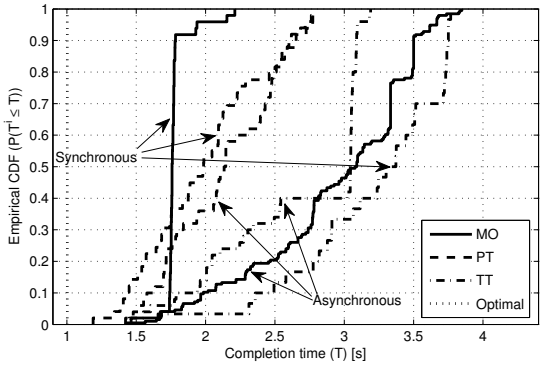

Figure 5: Cumulative distribution function of the completion time for the *Square* topology with the *Parking lot* data set.

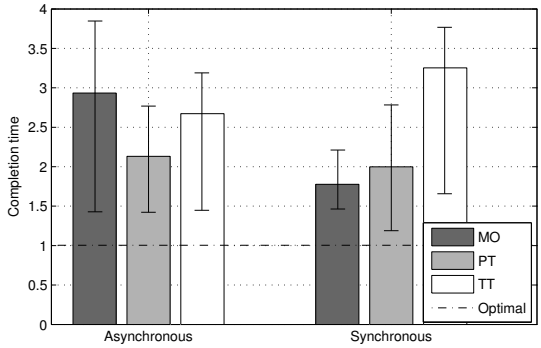

Figure 6: Mean, minimum and maximum completion time for the *Square* topology and the *Parking lot* data set.

two revision rules. We see that under this topology neither the synchronous nor the asynchronous *MO* algorithms converge, while all other algorithms converge within a few revisions. Nonetheless, non-convergence does not mean that the synchronous and the asynchronous *MO* algorithms exhibit worst performance (the average completion times are $2.14\,\mathrm{s}$ and $2.02\,\mathrm{s}$, respectively). Among the convergent algorithms it is interesting to note the difference between the synchronous and the asynchronous *TT* algorithms, which highlights the importance of the revision opportunity used.

Figure 4 shows the mean completion times obtained with each of the six algorithms maximum completion times, i.e., the variation around the mean. The figure shows that the mean completion times decrease with the number of processing nodes, but at the same time the variation tends to increase. Observe that under the *PT* and the *TT* scenarios the algorithms converge, hence the minimum is close to the average completion time.

We can conclude that the topology has a major impact on which algorithms converge to equilibrium, but reaching an equilibrium does not guarantee low completion times.

B. Trace-based Evaluation

We now turn to the evaluation of the algorithms using a measured multi-camera surveillance video trace. The trace we use is the *Parking lot* surveillance video data set used for the evaluation of algorithms for tracking humans [41]. The data set consists of video traces captured in a parking lot by 4 surveillance cameras at a resolution of 720×480 pixels and frame rates of $30\,\mathrm{fps}$, showing 9 people moving around. The cameras are approximately located at the corners of a square and are facing the center of the square, and thus we use the *Square* topology for the evaluation. Given the topology, the optimal solution is for each sensor to use one of the closer processing nodes.

We use BRISK [8] for detecting local visual features with a filter width of up to 84 pixels (i.e., $o = 0.06$), and use the top 400 interest points to compute the interest point distribution of each frame. We do not perform background subtraction on the traces prior to interest point detection, and thus there are a number of interest points belonging to the background that do not change their locations. As the sensor nodes can not know the distribution of interest points in image i before image i has been processed, they assume that image $i+1$ has the same interest point distribution as image i. This corresponds to the last value predictor used in [18], which was shown to provide a good trade-off between prediction accuracy and computational complexity. We use a processing time coefficient of $P_n = \frac{1}{6}\,\mathrm{s}/\mathrm{img}$ and $\alpha_d = 5$; these match measured data reported in [18] for a target of 400 interest points using BRISK. Again, as a reference, with these parameters a single processing node would complete processing an image in 1 second of time.

Figure 5 shows the CDF of the completion times using the six algorithms based on 50 video frames. We see that the algorithms lead to increased completion times, and they exhibit very different behaviour compared to the results with uniform interest point distribution (c.f. Fig. 3). Furthermore, it is for the *TT* scenario that performance deteriorates most.

To facilitate the comparison of the completion times, Figure 6 shows the mean, minimum and maximum completion times for the different algorithms, and the optimal completion time. Based on the averages, the results for the *PT* scenario are most consistent and thus *processing time* information may be the most robust signaling scheme to be used when the interest point distributions change due to the image content.

VII. Conclusion and Future Work

We considered the problem of minimizing the completion time of distributed feature extraction in visual sensor networks consisting of several camera sensors and image processing nodes. As centralized optimization would require excessive control information exchange, we considered distributed solutions where each camera sensor decides locally about the size and the allocation of the image slices. We defined three schemes that differ in terms of the information available to the sensors, and evaluated synchronous and asynchronous revisions. Our results show that for images with uniform interest point distribution, transmission and processing time information from the processing nodes helps to achieve equilibrium and close to optimal completion time. Convergence however does not necessarily mean good performance, and in some scenarios the non-convergent scheme, based on local measurements only, outperforms convergent solutions. Evaluations with real traces showed that most of the algorithms do not converge, and lead to low performance, reflecting that under changing image content, tighter cooperation among the

sensor nodes may be necessary. Therefore, as future work, we plan to design algorithms with a central coordinator to improve system performance.

REFERENCES

[1] K. Muller, P. Merkle, and T. Wiegand, "3-d video representation using depth maps," *Proc. of the IEEE*, vol. 99, no. 4, pp. 643–656, 2011.

[2] P. Rana, J. Taghia, and M. Flierl, "Statistical methods for interviewdepth enhancement," in *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, 2014.

[3] M. Liem and D. M. Gavrila, "Multi-person localization and track assignment in overlapping camera views," in *Pattern Recognition*, ser. Lecture Notes in Computer Science, 2011, pp. 173–183.

[4] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using sift features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345–352, 2009.

[5] M. Ayazoglu, B. Li, C. Dicle, M. Sznaier, and O. I. Camps, "Dynamic subspace-based coordinated multicamera tracking," in *Proceedings of the 2011 International Conference on Computer Vision*, ser. ICCV '11, 2011, pp. 2462–2469.

[6] S. Helmer and D. Lowe, "Using stereo for object recognition," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2010.

[7] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, 2010.

[8] S. Leutenegger, M. Chli, and R. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, 2011.

[9] L.-Y. Duan, X. Liu, J. Chen, T. Huang, and W. Gao, "Optimizing JPEG quantization table for low bit rate mobile visual search," in *Proc. of IEEE Visual Communications and Image Processing Conference (VCIP)*, 2012.

[10] J. Chao, H. Chen, and E. Steinbach, "On the design of a novel JPEG quantization table for improved feature detection performance," in *Proc. of IEEE International Conference on Image Processing (ICIP)*, 2013.

[11] V. R. Chandrasekhar, S. S. Tsai, G. Takacs, D. M. Chen, N.-M. Cheung, Y. Reznik, R. Vedantham, R. Grzeszczuk, and B. Girod, "Low latency image retrieval with progressive transmission of CHoG descriptors," in *Proc. of the ACM Multimedia Workshop on Mobile Cloud Media Computing*, 2010.

[12] H. Jegou, M. Douze, and C. Schmid, "Product quantization for nearest neighbor search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 1, pp. 117–128, 2011.

[13] A. Redondi, L. Baroffio, J. Ascenso, M. Cesana, and M. Tagliasacchi, "Rate-accuracy optimization of binary descriptors," in *Proc. of IEEE International Conference on Image Processing (ICIP)*, 2013.

[14] D.-N. Ta, W.-C. Chen, N. Gelfand, and K. Pulli, "SURFTrac: Efficient tracking and continuous object recognition using local feature descriptors," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[15] G. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[16] L. Baroffio, M. Cesana, A. Redondi, S. Tubaro, and M. Tagliasacchi, "Coding video sequences of visual features," in *Proc. of IEEE International Conference on Image Processing (ICIP)*, 2013.

[17] A. Redondi, L. Baroffio, A. Canclini, M. Cesana, and M. Tagliasacchi, "A visual sensor network for object recognition: Testbed realization," in *Proc. of International Conference on Digital Signal Processing (DSP)*, 2013.

[18] E. Eriksson, G. Dán, and V. Fodor, "Real-time distributed visual feature extraction from video in sensor networks," in *Proc. of IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2014.

[19] L. Baroffio, A. Canclini, M. Cesana, A. Redondi, M. Tagliasacchi, G. Dán, E. Eriksson, V. Fodor, J. Ascenso, and P. Monteiro, "Enabling visual analaysis in wireless sensor networks," in *Proc. of IEEE Intl. Conf. on Image Procesing (ICIP), Show and Tell*, October 2014.

[20] V. Bharadwaj, D. Ghose, and T. Robertazzi, "Divisible load theory: A new paradigm for load scheduling in distributed systems," *Cluster Computing*, vol. 6, no. 1, pp. 7–17, 2003.

[21] V. Bharadwaj, D. Ghose, and V. Mani, "Optimal sequencing and arrangement in distributed single-level tree networks with communication delays," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, no. 9, pp. 968–976, 1994.

[22] B. Veeravalli, X. Li, and C.-C. Ko, "On the influence of start-up costs in scheduling divisible loads on bus networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 12, pp. 1288–1305, 2000.

[23] D. Monderer and L. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, pp. 124–143, 1996.

[24] ——, "Fictitious play property for games with identical interests," *Journal of Economic Theory*, vol. 68, pp. 258–265, 1996.

[25] V. Pacifici and G. Dán, "Convergence in player-specific graphical resource allocation games," *IEEE J. Sel. Areas Commun. (JSAC)*, vol. 30, no. 11, pp. 2190–2199, 2012.

[26] U. Berger, "Brown's original fictitious play," *Journal of Economic Theory*, vol. 135, no. 1, pp. 572–578, 2007.

[27] D. P. Foster and H. Young, "Regret testing: Learning to play nash equilibrium without knowing you have an opponent," *Theoretical Economics*, vol. 1, pp. 241–367, 2006.

[28] L. Cigler and B. Faltings, "Reaching correlated equilibria through multi-agent learning," in *Proc. of Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2011, pp. 509–516.

[29] B. Pradelsky and H. Young, "Learning efficient nash equilibria in distributed systems," *Games and Economic Behavior*, vol. 75, pp. 882–897, 2012.

[30] M. A. Khan, G. Dan, and V. Fodor, "Characterization of SURF interest point distribution for visual processing in sensor networks," in *Proc. of International Conference on Digital Signal Processing (DSP)*, 2013.

[31] ——, "Characterization of SURF and BRISK interest point distribution for distributed feature extraction in visual sensor networks," *IEEE Transactions on Multimedia*, vol. 17, no. 5, May 2015.

[32] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346 – 359, 2008.

[33] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *Proc. of European Conference on Computer Vision (ECCV)*, 2010.

[34] C. Tang and P. K. McKinley, "Modeling multicast packet losses in wireless lans," in *Proc. of ACM International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems*, 2003.

[35] J. Lacan and T. Perennou, "Evaluation of error control mechanisms for 802.11b multicast transmissions," in *Proc. of International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, 2006.

[36] J. Hartwell and A. Fapojuwo, "Modeling and characterization of frame loss process in IEEE 802.11 wireless local area networks," in *Proc. of IEEE Vehicular Technology Conference. (VTC-Fall)*, 2004.

[37] R. Guha and S. Sarkar, "Characterizing temporal SNR variation in 802.11 networks," *IEEE Transactions on Vehicular Technology,*, vol. 57, no. 4, pp. 2002–2013, 2008.

[38] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella, "Performance study of ieee 802.15.4 using measurements and simulations," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, 2006.

[39] T. Joshi, A. Mukherjee, Y. Younghwan, and D. Agrawal, "Airtime fairness for IEEE 802.11 multirate networks," *IEEE Trans. on Mob.Comp.*, Apr. 2008.

[40] A. Kostuch, K. Gierlowski, and J. Wozniak, "Performance analysis of multicast video streaming in ieee 802.11 b/g/n testbed environment," in *Wireless and Mobile Networking*, ser. IFIP Advances in Information and Communication Technology, J. Wozniak, J. Konorski, R. Katulski, and A. Pach, Eds. Springer, 2009, vol. 308, pp. 92–105.

[41] M. S. Khan and M. Shah, "Tracking multiple occluding people by localizing on multiple scene planes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 3, pp. 505–519, 2009.