This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

Predictive Distributed Visual Analysis for Video in Wireless 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—We consider the problem of performing distributed visual analysis for a video sequence in a visual sensor network that contains sensor nodes dedicated to processing. Visual analysis requires the detection and extraction of visual features from the images, and thus the time to complete the analysis depends on the number and on the spatial distribution of the features, both of which are unknown before performing the detection. In this paper we formulate the minimization of the time needed to complete the distributed visual analysis for a video sequence subject to a mean average precision requirement as a stochastic optimization problem. We propose a solution based on two composite predictors that reconstruct randomly missing data, on quantile-based linear approximation of feature distribution and on time series analysis methods. The composite predictors allow us to compute an approximate optimal solution through linear programming. We use two surveillance video traces to evaluate the proposed algorithms, and show that prediction is essential for minimizing the completion time, even if the wireless channel conditions vary and introduce significant randomness. The results show that the last value predictor together with regular quantile-based distribution approximation provide a low complexity solution with very good performance.

Keywords—Image analysis; wireless sensor networks; scheduling; distributed computation

1 Introduction

Low cost cameras and networking hardware make a new class of sensor networks viable, namely, visual sensor networks (VSNs), where visual information is captured at one or several cameras and processed and transmitted through several network nodes, until the useful information reaches a central unit. VSNs differ from traditional sensor networks, where the transmission of sensed information requires little bandwidth and the complexity of the information processing is rather low. VSNs may instead capture high bitrate video sequences, requiring in-network processing in order to reduce the amount of data which is delivered to the sink node. The information processing needed for visual analysis, such as for tracking and for object recognition is, however, computationally intensive even using state-of-the-art algorithms like FAST and BRISK [1], [2].

A promising solution to allow real-time processing of the visual information in a VSN is to delegate the computationally intensive tasks of interest point detection and descriptor extraction to sensor nodes without cameras. Doing so allows the use of the processing capacity of those nodes, and not only their transmission capacity, enabling more processing intense applications without the need for an infrastructure. Unlike sensors with cameras, which typically need to be calibrated, the processing nodes can be installed or replaced with ease once their batteries are depleted. A VSN with processing nodes may thus be particularly useful in applications where easy maintenance and extended lifetime are important, such as surveillance systems in remote or hazardous areas, or in protected animal habitats. Reducing the cost of VSNs means they can even be used as disposable surveillance systems, suitable for detecting and tracking the progress of forest fires or other large scale natural disasters.

In this paper we consider visual analysis of video sequences based on local feature descriptors [1], [2], which are widely used for object recognition and tracking. To allow parallel computation, the camera node distributes the workload of interest point detection and descriptor extraction by assigning image sub-areas to the processing nodes. The precision and the processing workload of the visual analysis task depend both on the image size and on the number of detected and extracted descriptors [3]. However, the time it takes for a particular processing node to complete the processing depends on the available communication and computational resources of the node and, importantly, on the image content, which is not known prior to performing the processing. Therefore, optimizing the distribution of the processing tasks among the network nodes such that the completion time of the VSN is minimized is a challenging problem.

In this paper we consider a network consisting of a single camera node, several processing nodes, and a sink node, and we formulate the processing task distribution problem as a multi-objective stochastic optimization problem, first assuming deterministic communication resources. To solve the optimization problem we leverage the temporal correlation among the consecutive images

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

in the video sequence. The temporal correlation allows us to develop a predictor of the detection threshold, such that the number of descriptors is close to the required number. To minimize the completion time, we find the optimal schedule of the transmissions to the processing nodes, and we predict the optimal division of the image into sub-areas using a percentile-based approximation, such that the time of completing the feature extraction is minimized. Numerical results show that prediction is essential to achieve our objectives, and that the proposed prediction algorithms combine low computational complexity with good prediction performance. Finally, simulations results considering realistic, stochastic wireless channels show that the proposed prediction based solution is essential, and leads to a performance gain that is almost independent from the channel quality.

The rest of the paper is organized as follows. In Section 2 we review related work. In Section 3 we describe the considered system and in Section 4 we provide the problem formulation. In Section 5 we provide a method for reconstructing randomly missing threshold data. In Section 6 we develop the proposed predictors and in Section 7 we identify the optimal scheduling order. In Section 8 we present smiulation results and we conclude the paper in Section 9.

2 Related work

The challenge of networked visual analysis is addressed in [1], [2], defining feature extraction schemes with low computational complexity. To decrease the transmission bandwidth requirements, [4], [5] propose lossy image coding schemes optimized for descriptor extraction, while [6], [7], [8] give solutions to decrease the number and the size of the descriptors to be transmitted. In [9] the number and the quantization level of the considered descriptors are jointly optimized to maximize the accuracy of the recognition, subject to energy and bandwidth constraints. This approach is motivated by the measurement results of [9], [10], [11], [12] demonstrating that the performance of the visual analysis task increases with the number of features considered, for threshold based feature selection [9], [10], [11] as well as for more complex selection methods [12]. [11] shows that the MAP score decreases monotonically as the BRISK threshold is increased, unless the threshold value is very low, and thus the number of detected interest points is very high, but that region is not relevant for wireless sensor networks.

To decrease the transmission requirements of feature extraction in the case of video sequences [13] selects candidate descriptor locations based on motion prediction, and transmits and processes these areas only. In [14], [15] intra- and inter-frame coding of descriptors is proposed to decrease the transmission requirements.

Our work is motivated by recent results on the expected transmission and processing load of visual analysis in sensor networks [9], [3], [16], [17]. Measurements in [3] demonstrate that processing at the camera or at the sink node of the VSN leads to significant delays, and thus distributed processing is necessary for real-time applications. The requirement of prediction based system optimization is motivated by the statistical analysis of a large public image database in [16], [17], showing that the number and the spatial distribution of the descriptors have high variability and depend significantly on the image content. Thus, the temporal correlation in the video sequence needs to be utilized to achieve the efficient control of the visual analysis parameters. Finally, experiments in [9] show that the processing delay and the energy consumption increase linearly with the image size and with the number of detected descriptors. Consequently, to limit the time needed for descriptor extraction, the number of descriptors need to be controlled, and the workload allocation has to consider both the size of the sub-areas and the distribution of the descriptors.

Optimal load scheduling for distributed systems is addressed in [18], in the framework of Divisible Load Theory, 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. Works closest to ours address tree networks with heterogeneous link capacities and processor speeds [19], concluding that scheduling should be in decreasing order of the transmission capacities, while the processing speed does not affect the scheduling decision. However, [20] 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. As we show in the paper, this result can not be used in general either, for example, in our scenario where unicast and multicast transmissions are combined, and the link transmission capacities differ.

We introduced the main components of prediction based distributed visual analysis in [21], [22]. In this paper we provide a discussion of application scenarios, a revised and more detailed description of the problem and a more thorough performance evaluation of the proposed algorithms. Moreover, we use simulations to evaluate the effect of the randomness of the wireless channel on the prediction accuracy.

3 Background and System model

We consider a VSN consisting of a camera node $\mathcal{C}$, a set of processing nodes $\mathcal{N}$, $|\mathcal{N}| = N$, and a sink node $\mathcal{S}$. The camera node captures a sequence of images. Each image is transmitted to and processed at nodes in $\mathcal{N}$, and finally the results are transmitted to $\mathcal{S}$ where the visual analysis task is completed.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

3.1 Communication model

The nodes communicate using a multicast/broadcast capable wireless communication protocol, such as IEEE 802.15.4 or IEEE 802.11. Transmissions suffer from packet losses due to wireless channel impairments. As measurement studies show [23], [24], the loss burst lengths at the receivers have low mean and variance in the order of a couple of frames [25], [26]. Therefore, widely used wireless channel models are time independent fading channels, or finite state Markov channels, 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 $\mathcal{C}$ to a node $n \in \mathcal{N}$ as a linear function of the amount of transmitted data. We can thus model the time it takes to transmit p pixels from the camera node $\mathcal{C}$ to processing node $n \in \mathcal{N}$ as a random variable K_n, with a conditional expected value of $E[K_n|p] = C_n \cdot p$, where C_n denotes the average per pixel transmission time, including potential retransmissions, and is referred to as the transmission time coefficient throughout the paper. As the throughput is close to stationary over short timescales, C_n can be estimated [27]. When using multicast or broadcast transmission, the throughput is determined by the receiver with lowest achievable throughput.

3.2 Feature detection and extraction

We consider a sequence $\{Z_i\}, i = 1, \ldots, I$, of images is captured at $\mathcal{C}$. Each image has a height of h and a width of w pixels. For each image, $\mathcal{C}$ sends the image data to the processing nodes, which perform interest point detection and feature descriptor extraction.

Interest point detection is performed by applying a blob detector or an edge detector at every pixel of the image area [28], [29], [2]. The detector computes a response score for each pixel based on a square area centered around the pixel, with side length $2ow$ pixels, where o 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 number of interest points detected in an image depends on the image and on the detection threshold ϑ, we thus describe the number of interest points detected in image i is by an integer valued, left continuous, non-negative, decreasing step function $f_i(\vartheta)$ of the detection threshold ϑ. f_i is not known before processing image i; we model it as a random function chosen from the family of integer valued, left continuous, non-negative, decreasing step functions. The inverse function $f_i^{-1} : \mathbb{N} \to \mathbf{\Theta}$ can be defined as $f_i^{-1}(m) = \max\{\vartheta | f_i(\vartheta) = m\}$. The maximum exists because f_i is a left continuous, decreasing step function. We denote the sequence of thresholds used in the images by $\boldsymbol{\vartheta} = (\vartheta_1, \ldots, \vartheta_I)$.

In order to distribute the workload among the processing nodes in $\mathcal{N}$, the camera node divides each image i into at most N sub-areas. Sub-area $Z_{i,n}$ is then assigned to processing node n. For simplicity, we consider that the sub-areas are slices of the image formed along the horizontal axis. This scheme was referred to as area-split in [16], [17]. We specify the sub-areas by the horizontal coordinates of the vertical lines separating them, normalized by the image width w, which we refer to as the cut-point location vector $\boldsymbol{x}_i = (x_{i,1}, \ldots, x_{i,N})$, $x_{i,1} < \ldots < x_{i,N} = 1$. For notational convenience, we define $x_{i,0} = 0$, the left edge of image i, and $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_I)$, the sequence of cut-point vectors used for the trace. Since interest point detection at a pixel requires a square area around the pixel to be available, all points within ow pixels of the horizontal coordinate $x_{i,n}$ need to be transmitted to both node n and $n+1$. We call o the overlap, and we express its value normalized by the image width w (hence the multiplication above). We consider that $\frac{1}{N} >> o$, which holds if the image size is reasonably large.

The number of interest points detected in sub-area $Z_{i,n}$ depends on the image, the detection threshold ϑ_i and on the cut-point location vector $\boldsymbol{x}_i$. We thus describe the number of interest points detected in sub-area $Z_{i,n}$ by the function $f_{i,n}(\vartheta_i, \boldsymbol{x}_i)$, and we define the vector function $\boldsymbol{f}_i(\vartheta_i, \boldsymbol{x}_i) = (f_{i,1}(\vartheta_i, \boldsymbol{x}_i), \ldots, f_{i,N}(\vartheta_i, \boldsymbol{x}_i))$. The function $\boldsymbol{f}_i(\vartheta_i, \boldsymbol{x}_i)$ can be modeled as a random function from the family of integer valued vector functions with $\sum_{n=1}^{N} f_{i,n}(\vartheta, \boldsymbol{x}_i) = f_i(\vartheta)$. We consider that the time it takes to detect the interest points is a linear function of the size of the sub-area (not including the overlap) with rate $P_n^{d,px}$, and of the number of interest points detected with rate $P_n^{d,ip}$.

As the next step, a feature descriptor is extracted for each interest point. The time it takes to extract the descriptors is a linear function of the number of interest points detected with rate $P_n^{e,ip}$.

To validate this model, we performed interest point detection and feature descriptor extraction on a BeagleBone Black single board computer for 3 different image sizes using OpenCV [30]. The results shown in Figure 1 confirm that the computation time can be well approximated by a linear function. Similar results were reported on an Intel Imote2 platform in [3].

When node n completes the extraction of descriptors within sub-area $Z_{i,n}$, it transmits them to $\mathcal{S}$, where various computer vision tasks can be performed. In order for $\mathcal{S}$ to be able to perform its computer vision tasks, it requires M^* interest points to be detected in each image. While the required number of interest points, M^*, may vary over time, depending on the observed scene, scene changes would happen at a time-scale much larger than the time between subsequent images, we thus consider it to be constant. In Section 5 we will briefly discuss how to handle varying M^*. To optimize the distributed processing, ϑ and $\boldsymbol{x}$ should be selected based on information available at $\mathcal{S}$. Since for each already transmitted image i the sink has access to the parameters $(\vartheta_i, \boldsymbol{x}_i)$, as well as all the interest point descriptors, it

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

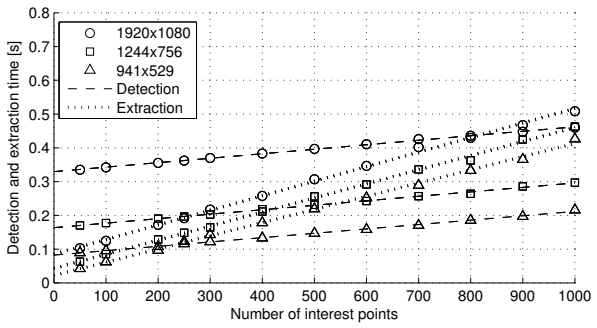

Figure 1: Average time for performing detection and extraction as a function of the number of interest points found for three image sizes. The linear regression shows a good fit.

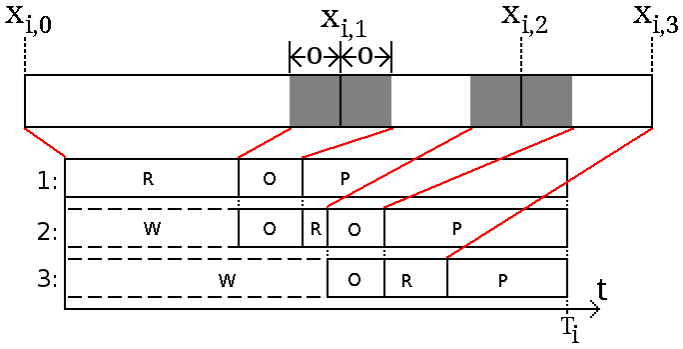

Figure 2: Example with $N = 3$ processing nodes. The image is cut along the horizontal axis, the sub-areas and the overlaps are received and processed by the processing nodes. A node is either (w)aiting to receive data, is (r)eceiving individual data, is receiving (o)verlapping data, or is (p)rocessing data.

knows the location and score of each detected interest point. It can therefore calculate $f_i(\vartheta)$ for any $\vartheta \geq \vartheta_i$, i.e., the total workload the system would have had with detection threshold ϑ. Nevertheless, if $f_i(\vartheta_i) < M^*$ then $f_i^{-1}(M^*)$ cannot be computed by $\mathcal{S}$. Similarly, $\mathcal{S}$ can compute $\boldsymbol{f}_i(\vartheta, \boldsymbol{x}_i)$ for any $\vartheta \geq \vartheta_i$ and any cut-point location vector $\boldsymbol{x}$. We use Υ_i to denote the information available to $\mathcal{S}$ about image i, and Υ_{i-} to denote the information available about all images previous to image i.

4 Problem Formulation

Based on the model of the wireless links and of the detection and extraction of features, we first express the reception and processing times of the N processing nodes as a function of the threshold ϑ_i and the cut-point location vector $\boldsymbol{x}_i$. We then define the performance metrics and formulate our objective.

4.1 Expected completion time

Assume the processing nodes are numbered by the order in which they receive their data from the camera node $\mathcal{C}$, and let us consider a node n. Initially, node n is waiting while all preceding nodes receive their data. It then starts receiving data once $\mathcal{C}$ starts to transmit the overlap shared between nodes n and $n-1$, followed by the data destined to node n only, and finally the overlap shared between nodes n and $n+1$. Once node n has received the data, feature detection and descriptor extraction are performed on the sub-area $Z_{i,n}$. Finally the descriptors are transmitted to $\mathcal{S}$. For high resolution grayscale images, the image data is in the order of tens of Mbits, while in the case of modern binary descriptors like BRISK [2], the size of the descriptors is in the order of a few hundred bits, which gives a few hundreds of kbits descriptor data for realistic M^*. Due to the two orders of magnitude difference, we do not consider the time it takes to transmit the descriptors to $\mathcal{S}$, and define the completion time as the sum of the reception, feature detection and descriptor extraction times. Figure 2 illustrates the phases for $N = 3$.

Lemma 1. *Minimizing the completion time based on the transmission time coefficients C_n minimizes the expected completion time.*

Proof: Observe that the completion time of processing node n is the sum of a number of random variables, in particular, the transmission times to the preceding processing nodes ($K_m, m < n$), its own transmission time (K_n) and its processing time. Furthermore, given the size $x_n - x_{n-1}$ of sub-area n, the processing time of node n is conditionally independent of its transmission time. Since the expected value of a sum of random variables equals the sum of their expectations, and the expected transmission times are determined by the transmission time coefficients C_n, the result follows. □

Motivated by Lemma 1, we can use the transmission time coefficients C_n and processing time coefficients $P_n^{d,px}$, $P_n^{d,ip}$ and $P_n^{e,ip}$ to provide matrix expressions for the expected reception, processing and completion time. The expressions given describe the case when $2o < x_{i,n-1} - x_{i,n}$, i.e., an overlap spans only two nodes; similar expressions can be obtained for $2o \geq x_{i,n-1} - x_{i,n}$. Let D_j and E_j be $N \times N$ matrices, and let G_j be an $N \times 1$ column vector. Also, let us use the notation $C_n^M \triangleq \max(C_n, C_{n+1})$ as a shorthand for the effective transmission time coefficient for multicast transmission to nodes n and $n+1$.

The average time node $n > 1$ spends waiting before it receives the first bit depends on the transmission time coefficients and the size of the sub-area of the preceding processing nodes,

$$T_{i,n}^w = hwC_1(x_1 - o) + hw \sum_{m=2}^{n-1} C_m(x_m - x_{m-1} - 2o) + C_{m-1}^M 2o.$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

This can be expressed in matrix notation as $T_i^w = D^w \boldsymbol{x}_i + G^w$, where

$$d_{m,n}^w = \begin{cases} hwC_n, & m = n+1 \\ hwC_n - hwC_{n+1}, & m > n+1 \\ 0, & otherwise, \end{cases}$$

and

$$g_m^w = \begin{cases} 0, & m = 1 \\ -hwoC_1 + \sum_{j=2}^{m-1} \left(2hwoC_{j-1}^M - 2hwoC_j\right), & m > 1. \end{cases}$$

Node n receives the overlaps with its neighbours in multicast transmission. As nodes 1 and N are assigned the first and last sub-areas of the image respectively, they will only receive a single overlap. The time to receive the overlap is determined by the multicast transmission time coefficient C_n^M and by the overlap o, $T_i^o = G^o$, where

$$g_n^o = \begin{cases} 2hwoC_1^M, & n = 1 \\ 2hwoC_{n-1}^M + 2hwoC_n^M, & 1 < n < N \\ 2hwoC_{N-1}^M, & n = N. \end{cases}$$

The average time it takes node n to receive the non-overlapping data depends on the size of the sub-area $Z_{i,n}$, and on the transmission time coefficient,

$$T_{i,n}^r = hwC_n(x_n - x_{n-1} - 2o) + C_{n-1}^M 2o, 1 < n < N.$$

This can be expressed in matrix notation as $T_i^r = D^r \boldsymbol{x}_i + G^r$, where

$$d_{m,n}^r = \begin{cases} hwC_n, & m = n \\ -hwC_{n+1}, & m = n+1 \\ 0, & otherwise \end{cases}$$

and

$$g_n^r = \begin{cases} -hwoC_n, & n \in \{1, N\} \\ -2hwoC_n, & otherwise. \end{cases}$$

The time it takes to perform interest point detection is a function of the size of sub-area $Z_{i,n}$ and of the number of detected interest points,

$$T_{i,n}^d = \frac{hw}{P_n^{d,px}}(x_n - x_{n-1}) + \frac{f_{i,n}(\vartheta, \mathbf{x}_i)}{P_n^{d,ip}},$$

which can be expressed in matrix notation as $T_i^d = D^d \boldsymbol{x}_i + E^d \boldsymbol{f}_i(\vartheta_i, \boldsymbol{x}_i)$, where

$$d_{m,n}^d = \begin{cases} \frac{hw}{P_n^{d,px}}, & m = n \\ -\frac{hw}{P_{n+1}^{d,px}}, & m = n+1 \\ 0, & otherwise \end{cases}$$

and

$$e_{m,n}^d = \begin{cases} \frac{1}{P_n^{d,ip}}, & m = n \\ 0, & otherwise. \end{cases}$$

Finally, the time needed for descriptor extraction is a function of the number of detected interest points

$$T_{i,n}^e = \frac{f_{i,n}(\vartheta, \mathbf{x}_i)}{P_n^e},$$

which can be expressed in matrix notation as $T_i^e = E^e \boldsymbol{f}_i(\vartheta_i, \boldsymbol{x}_i)$, where

$$e_{m,n}^e = \begin{cases} \frac{1}{P_n^e}, & m = n \\ 0, & otherwise. \end{cases}$$

Let us define $D \triangleq D^w + D^r + D^d$, $E \triangleq E^d + E^e$, and $G \triangleq G^w + G^o + G^r$. Using this notation we can express the expected completion time of each node n for image i, $T_i(\vartheta_i, \boldsymbol{x}_i) = (T_{i,1}(\vartheta_i, \boldsymbol{x}_i), \ldots, T_{i,N}(\vartheta_i, \boldsymbol{x}_i))$ as

$$T_i(\vartheta_i, \boldsymbol{x}_i) = D\boldsymbol{x}_i + E\boldsymbol{f}_i(\vartheta_i, \boldsymbol{x}_i) + G, \tag{1}$$

which is a non-linear vector function of ϑ_i and $\boldsymbol{x}_i$.

4.2 Performance optimization

We are interested in two key aspects of the VSN's performance. First, we want to ensure that the VSN can perform the visual analysis task at the required level of mean average precision. The mean average precision is a concave function the number of detected interest points [9], [10], [11], while the completion time is an affine function of the number of interest points, hence detecting too few or too many interest points can both be detrimental. Therefore we define our first performance metric to be the squared error in detected interest points in image i compared to the target value M^* required by the computer vision task

$$e_i^D(\vartheta_i) = \left(f_i(\vartheta_i) - M^*\right)^2, \tag{2}$$

and we define the corresponding mean square error (MSE) as $e^D(\boldsymbol{\vartheta}) = \frac{1}{I}\sum_{i=1}^I e_i^D(\vartheta_i)$. We define the optimal detection threshold for image i as $\vartheta_i^* = \max(\theta_i^*)$, where $\theta_i^* = \{\vartheta | e_i^D(\vartheta) = 0\}$.

Second, we are interested in minimizing the time it takes to complete the detection and the extraction of all descriptors. We therefore define our second performance metric based on the VSN's completion time, which we define as the largest completion time among all processing nodes. We define the squared completion time error of the VSN for image i as the squared difference compared to the smallest possible VSN completion time

$$e_i^C(\vartheta_i, \boldsymbol{x}_i) = \left(\max_n\left(T_i(\vartheta_i, \boldsymbol{x}_i)\right) - \max_n\left(T_i(\vartheta_i^*, \boldsymbol{x}_i^*)\right)\right)^2, \tag{3}$$

and the mean squared completion time error as $e^C(\boldsymbol{\vartheta}, \boldsymbol{x}) = \frac{1}{I}\sum_{i=1}^I e_i^C(\vartheta_i, \boldsymbol{x}_i)$, where the optimal cut-point location vector $\boldsymbol{x}_i^* \in \arg\min_{\boldsymbol{x}_i} \max_n\left(T_i(\vartheta_i^*, \boldsymbol{x}_i)\right)$.

Observe that both (2) and (3) depend on the functions f_i and $\boldsymbol{f}_i$, which are not known prior to processing image i. By modeling f_i and $\boldsymbol{f}_i$ as random functions, we can

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

formulate our problem as a stochastic multi-objective optimization problem

$$\text{lexmin}(\mathbb{E}[e^D(\boldsymbol{\vartheta})], \mathbb{E}[e^C(\boldsymbol{\vartheta}, \boldsymbol{x})]) \quad (4)$$

s.t.

$$\boldsymbol{\vartheta} \in \Theta^I, \boldsymbol{x} \in \mathcal{X}^I, \quad (5)$$

where $lexmin$ stands for lexicographical minimization, and we are looking for an expected value efficient solution [31]. Since the choice of ϑ_i and $\boldsymbol{x}_i$ for image i does not influence the error at images $j > i$, this problem is equivalent to solving

$$\text{lexmin}(\mathbb{E}[e_i^D(\vartheta_i)], \mathbb{E}[e_i^C(\vartheta_i, \boldsymbol{x}_i)]) \quad (6)$$

s.t.

$$\vartheta_i \in \Theta, \boldsymbol{x}_i \in \mathcal{X}, \quad (7)$$

for every image i based on the information Υ_{i-}. We therefore search for the solution in the form of a predictor $\tau^*(\Upsilon)$ that minimizes the expected square error

$$\tau^* \in \arg\min_{\tau} \mathbb{E}[e_i^D(\tau(\Upsilon_{i-}))], \quad (8)$$

and a predictor $\gamma^*(\Upsilon)$ that minimizes the expected squared completion time error

$$\gamma^* \in \arg\min_{\gamma} \mathbb{E}[e_i^C(\tau^*(\Upsilon_{i-}), \gamma(\Upsilon_{i-}))]. \quad (9)$$

In what follows we develop and analyze predictors with low complexity and little overhead suitable for sensor networks.

5 Regression-Based Threshold Reconstruction

We will first address prediction problem (8). Solving this problem with conventional methods is not straight forward for two reasons. First, since $f_i(\vartheta)$, and thus (2) are step functions in ϑ, the sets of minimizers $\theta_i^* = \{\vartheta | e_i^D(\vartheta) = 0\}$ may not be singleton. Second, if $f_i(\hat{\vartheta}_i) < M^*$ then θ_i^* is unknown and can not be used for prediction.

We propose two regression-based methods for estimating θ_i^*, which allows us to evaluate various autoregressive models that require ϑ_i^* for prediction.

Consider an image i for which the predicted detection threshold $\hat{\vartheta}_i$ results in $f_i(\hat{\vartheta}_i) < M^*$. The goal is to estimate some $\hat{\vartheta}_i^* \in \theta_i^*$ that can be used to predict $\hat{\vartheta}_{i+1} \in \theta_{i+1}^*$. The key tenet of the proposed approach is to use preceding images for which $f_j(\hat{\vartheta}_j) \geq M^*$ for estimating the slope of the function f_i^{-1} around M^*. Let $\mathcal{I}_{i-}$ be the set of indices of the images before image i for which the estimated detection threshold $\hat{\vartheta}_j$ resulted in more than M^* interest points, i.e., $f_j(\hat{\vartheta}_j) \geq M^* \; \forall j \in \mathcal{I}_{i-}$. We can use the images in $\mathcal{I}_{i-}$ to estimate the slope of the function f_i^{-1} around M^* in two ways: in the forward direction and in the backward direction.

Forward estimate: To obtain the forward estimate of the slope of the function we define the forward regression coefficient

$$\beta_{i-}^f = \frac{\frac{1}{|\mathcal{I}_{i-}|}\sum_{j\in\mathcal{I}_{i-}}(f_j(\hat{\vartheta}_j) - M^*)(\hat{\vartheta}_j - f_j^{-1}(M^*))}{\frac{1}{|\mathcal{I}_{i-}|}\sum_{j\in\mathcal{I}_{i-}}(f_j(\hat{\vartheta}_j) - M^*)^2}, \quad (10)$$

which is the estimated slope of the piecewise linear extension of f_i^{-1} in the forward direction (i.e., beyond M^*). We then use the forward regression coefficient to obtain the estimated threshold

$$\hat{\vartheta}_i^{f*} = \hat{\vartheta}_i - (f_i(\hat{\vartheta}_i) - M^*)\beta_{i-}^f \quad (11)$$

Backward estimate: To obtain the backward estimate of the function's slope we use the same linear regression but in the backward direction. In the backward direction (i.e., less than M^* interest points) we can compute the regression for arbitrary difference $d < M^*$ based on the available data $\mathcal{I}_{i-}$. For a particular difference d after simplification we obtain

$$\beta_{i-}^b(d) = \frac{1}{|\mathcal{I}_{i-}|}\sum_{j\in\mathcal{I}_{i-}}\frac{f_j^{-1}(M^*) - f_j^{-1}(M^* - d)}{d}, \quad (12)$$

which is the average backward difference quotient of f^{-1} at M^* over the images in $\mathcal{I}_{i-}$. Using the backward regression coefficient we obtain the estimated threshold

$$\hat{\vartheta}_i^{b*} = \hat{\vartheta}_i - (f_i(\hat{\vartheta}_i) - M^*)\beta_{i-}^b. \quad (13)$$

Proposition 1. *Assume that for every d the backward difference quotient $\frac{f_i^{-1}(M^*) - f_i^{-1}(M^*-d)}{d}$ of f_i^{-1} at M^* can be modeled by an i.i.d. random variable, and is independent of $f_i^{-1}(M^*)$. Then the estimated threshold $\hat{\vartheta}_i^{b*} = \arg\min_{\vartheta} E[e_i^D(\vartheta)]$.*

Proof: Since the backward difference quotient is independent of $f_i^{-1}(M^*)$, the backward difference quotient of images $j \in \mathcal{I}_{i-}$ is an unbiased sample of that of all images $j < i$. Since $\beta_{i-}^b(d)$ is the sample mean of the backward difference quotient, it is the minimum variance unbiased estimator, and thus it minimizes the expected square error. □

If the target value M^* varies over time, then one can maintain the difference ratio quotient estimates for different values of M^*. Given $\hat{\vartheta}_i^{b*}$ or $\hat{\vartheta}_i^{f*}$, and a policy to choose an element of the set θ_i^*, we can use a time series model such as AR, MA or ARMA, to predict $\hat{\vartheta}_{i+1}$.

6 Predictive Completion Time Minimization

We now proceed to address the minimization of the completion time, i.e., the problem of finding a predictor that solves (9). As with prediction problem (8), there are two issues that make this problem non-straightforward to solve.. First, since $T_i(\vartheta_i, \boldsymbol{x}_i)$, and thus (3) are step functions in ϑ and $\boldsymbol{x}$, the sets of minimizers $\Xi_i^* =$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

$\left\{\boldsymbol{x}|e_i^C\left(\vartheta^*,\boldsymbol{x}\right)=0,\,\vartheta^*\in\theta_i^*\right\}$ may not be singleton. Second, a predictor for solving (9) also needs to predict the location of all interest points in each captured image. First, we consider a given ordering of the processing nodes and provide an algorithm to approximate the cut-point location vector $\boldsymbol{x}_i$ that minimizes the completion time for the ordering. Second, in Section 7 we show how to find the ordering that allows the smallest completion time.

6.1 Distribution-based Cut-point Location Vector Selection

Without loss of generality we consider that sub-area $Z_{i,n}$ has to be processed by node n, and for image i we need to find the cut-point vector $\boldsymbol{x}_i$ that minimizes $e_i^C(\vartheta_i,\boldsymbol{x}_i)$.

Let us assume that the distribution of the interest points' horizontal coordinates $F_i(\vartheta_i,x)$ is known, thus $\boldsymbol{f}_i\left(\vartheta_i,\boldsymbol{x}\right)$ can be computed for an arbitrary cut-point location vector $\boldsymbol{x}$. We can then compute the cut-point location vector $\boldsymbol{x}_i^*$ for image i that minimizes $e_i^C(\vartheta_i,\boldsymbol{x}_i)$ by solving the integer programming (IP) problem

$$\min t \tag{14}$$

s.t.

$$\begin{aligned} D\mathbf{x}_i+E\mathbf{f}_i\left(\vartheta_i,\mathbf{x}_i\right)+G &\leq t\mathbf{1} && (15)\\ x_{i,n-1}w-x_{i,n}w &\leq -2o\ \ \forall n && (16)\\ x_{i,n}w &\in \{1,\ldots,w\}\ \ \forall n && (17)\end{aligned}$$

where (15) is componentwise, (16) enforces that the cut-point coordinates are increasing, (17) ensures they are aligned with pixels, and $\mathbf{1}$ is a $N\times 1$ column vector of ones.

Using the IP (14)-(17) for the considered VSN faces two challenges. First, solving IP problems is computationally intensive, in general. Second, the distribution $F_i(x)$ is not available until the processing of image i is completed, at which point solving the IP problem is no longer necessary. Thus, the biggest challenge in solving (14)-(17) is that it needs a prediction of the distribution $F_i(\vartheta_i,x_i)$ of interest points in image i. This prediction would require predicting the locations and appearance/disappearance of all interest points for every image, which is computationally infeasible. We address this challenge in the following.

6.2 Percentile-based Cut-point Location Vector Selection

We propose to solve the above problem by approximating the distribution $F_{i-1}(\vartheta_i,x)$ of interest points through its percentiles, and by predicting the approximation of the distribution $F_i(\vartheta_{i+1},x)$ for the optimization through *predicting the percentiles*. Here we focus on the approximation and the optimization, and will compare various predictors in Section 8.

We approximate the distribution $F_i(\vartheta_i,x)$ with the distribution $\tilde{F}_i(\vartheta_i,x)$, obtained as the linear interpolation of $F_i(\vartheta_i,x)$ between its values at Q percentiles, denoted by $\boldsymbol{\xi}=\xi_1,\ldots,\xi_Q$,

$$\tilde{F}_i(\vartheta_i,x)=\frac{x-\xi_{q-1}}{\xi_q-\xi_{q-1}}\pi_q+\Pi_{q-1}, \tag{18}$$

where $\xi_0=0$, $\xi_{q-1}<x\leq\xi_q$, $\pi_q=F_i(\vartheta_i,\xi_q)-F_i(\vartheta_i,\xi_{q-1})$ is the portion of interest points in the interval $\xi_{q-1}<x\leq\xi_q$, and $\Pi_{q-1}=F_i(\vartheta_i,\xi_{q-1})$ is the portion of interest points left of ξ_{q-1}. $\tilde{F}_i(\vartheta_i,x)$ is a non-decreasing, non-negative, continuous, piecewise linear function, which we can use to compute the approximate number of interest points assigned to node n for cut-point location vector $\boldsymbol{x}_i$ as

$$\tilde{f}_{i,n}\left(\vartheta_i,\boldsymbol{x}_i\right)=M^*\left(\tilde{F}_i(\vartheta_i,x_{i,n})-\tilde{F}_{i,x}(\vartheta_i,x_{i,n-1})\right). \tag{19}$$

We can use (19) to express the approximate time needed for interest point detection

$$\begin{aligned}\tilde{T}_i^d &= \tilde{D}^d\boldsymbol{x}_i+\tilde{G}^d, \text{where}\\ \tilde{d}_{m,n}^d &= \begin{cases}\frac{hw}{P_n^{d,px}}+\frac{M^*}{P_n^{d,ip}}\frac{\pi_q}{\xi_q-\xi_{q-1}}, & m=n\\ \frac{-hw}{P_{n+1}^{d,px}}-\frac{M^*}{P_{n+1}^{d,ip}}\frac{\pi_r}{\xi_r-\xi_{r-1}}, & m=n+1\\ 0, & \text{otherwise}\end{cases}\\ \tilde{g}_n^d &= \frac{M^*}{P_n^{d,ip}}\left(\frac{\xi_{r-1}\pi_r}{\xi_r-\xi_{r-1}}-\frac{\xi_{q-1}\pi_q}{\xi_q-\xi_{q-1}}+\Pi_{q-1}-\Pi_{r-1}\right),\forall n\end{aligned}$$

and the approximate time needed for descriptor extraction

$$\begin{aligned}\tilde{T}_i^e &= \tilde{D}^e\boldsymbol{x}_i+\tilde{G}^e, \text{where}\\ \tilde{d}_{m,n}^e &= \begin{cases}\frac{M^*}{P_n^e}\frac{\pi_q}{\xi_q-\xi_{q-1}}, & m=n\\ -\frac{M^*}{P_{n+1}^e}\frac{\pi_r}{\xi_r-\xi_{r-1}}, & m=n+1\\ 0, & \text{otherwise}\end{cases}\\ g_n^e &= \frac{M^*}{P_n^e}\left(\frac{\xi_{r-1}\pi_r}{\xi_r-\xi_{r-1}}-\frac{\xi_{q-1}\pi_q}{\xi_q-\xi_{q-1}}+\Pi_{q-1}-\Pi_{r-1}\right),\forall n\end{aligned}$$

By forming the matrices $\tilde{D}\triangleq D^w+D^r+\tilde{D}^d+\tilde{D}^e$ and $\tilde{G}\triangleq G^w+G^o+G^r+\tilde{G}^d+\tilde{G}^e$, we obtain for the approximate completion times of the nodes the set of linear equations

$$\tilde{T}_i=\tilde{D}\boldsymbol{x}_i+\tilde{G}. \tag{20}$$

The cut-point location vector $\tilde{\boldsymbol{x}}_i^*$ that minimizes (20) can be obtained by solving the integer-linear programming problem

$$\min t \tag{21}$$

s.t.

$$\begin{aligned}\tilde{D}\boldsymbol{x}_i+\tilde{G} &\leq t\mathbf{1} && (22)\\ x_{i,n-1}w-x_{i,n}w &\leq -2o\ \ \forall n && (23)\\ x_{i,n}w &\in \{1,\ldots,w\}\ \ \forall n && (24)\end{aligned}$$

Since (22) is piecewise linear, a linear relaxation of the problem can be solved efficiently, and the rounding error is negligible if the distribution $\tilde{F}_x(\vartheta,x)$ is reasonably smooth. Observe that by using $Q=f_i(\vartheta_i)$ percentiles, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

approximate distribution $\tilde{F}_x(\vartheta, x)$ is a linear interpolation of $F_x(\vartheta, x)$.

An important question is how close to optimal would be the completion time of the processing with this approximate solution. To answer this question we introduce $T_i^N(\vartheta_i, \boldsymbol{x}_i^*) = \max_n(T_i(\vartheta_i, \boldsymbol{x}_i^*))$, the optimal processing completion time in the VSN based on (14)-(17), $\tilde{T}_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) = \max_n(T_i(\vartheta_i, \tilde{\boldsymbol{x}}_i^*))$ the optimal completion time with the linear interpolation according to (21)-(24), and finally $T_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*)$ the experienced completion time if $\tilde{\boldsymbol{x}}_i^*$ is applied for the real distribution $F_x(\vartheta, x)$. In the following we give a bound on the maximum difference between $T_i^N(\vartheta_i, \boldsymbol{x}_i^*)$ and $\tilde{T}_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*)$

Proposition 2. *For any $\epsilon > 0$ there exists Q such that $T_i^N(\vartheta_i, \boldsymbol{x}_i^*) \leq \tilde{T}_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) + \epsilon$.*

Proof: As $T_i^N(\vartheta_i, \boldsymbol{x}_i^*) \leq T_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*)$, we prove $T_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) \leq \tilde{T}_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) + \epsilon$. Despite the linear interpolation, for each node n the components of $\tilde{T}_{i,n}(\vartheta_i, \tilde{\boldsymbol{x}}_i^*)$ and $T_{i,n}(\vartheta_i, \tilde{\boldsymbol{x}}_i^*)$ are identical: the transmission time, the sub-area size dependent part of the detection time, and the detection and extraction times that depend on the interest points in the percentiles following $\tilde{x}_{i,n-1}^*$ and preceding $\tilde{x}_{i,n}^*$. If we define $\Delta_i = max_x|F_i(\vartheta_i, x) - \tilde{F}_i(\vartheta_i, x)|$, we can obtain the worst case bound $\tilde{\epsilon}_n = T_{i,n}(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) - \tilde{T}_{i,n}(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) \leq 2\Delta_i \left(\frac{1}{P_n^{d,ip}} + \frac{1}{P_n^e}\right)$, and consequently $\tilde{\epsilon} = T_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) - \tilde{T}_i^N(\vartheta_i, \tilde{\boldsymbol{x}}_i^*) \leq 2\Delta_i \max_n \left(\frac{1}{P_n^{d,ip}} + \frac{1}{P_n^e}\right)$.

Let us now consider Q quantiles. The number of interest points between neighboring quantile points is $\frac{f_i(\vartheta_i)}{Q} - 1 \geq \Delta_i$, and consequently we have

$$\tilde{\epsilon} \leq 2\left(\frac{f_i(\vartheta_i)}{Q} - 1\right) \max_n \left(\frac{1}{P_n^{d,ip}} + \frac{1}{P_n^e}\right). \tag{25}$$

Thus, $\tilde{\epsilon} \leq \epsilon$ for any $Q \geq \frac{2f_i(\vartheta_i)T_p}{\epsilon + 2T_p}$, with $T_p = \frac{1}{P_n^{d,ip}} + \frac{1}{P_n^e}$. □

6.3 On-line Cut-point Location Vector Optimization

So far we considered minimizing the expected completion time, assuming that data are transmitted with the expected transmission time coefficients C_n. The actual transmission times are however random, and would differ from the expected values. In the following we address whether one should recompute the cut-point location vector after the data transmission to node m completes, to further minimize the completion time of the distributed processing.

Let us consider image i and denote the expected time of completing the transmission to node m by $\tau_{i,m}(\vartheta_i, \boldsymbol{x}_i^*)$, and the expected remaining time until completing the processing of the image by $\tau_{i,m+}(\vartheta_i, \boldsymbol{x}_i^*)$, such that $\tau_{i,m}(\vartheta_i, \boldsymbol{x}_i^*) + \tau_{i,m+}(\vartheta_i, \boldsymbol{x}_i^*) = T_i^N(\vartheta_i, \boldsymbol{x}_i^*)$ according to (14)-(17). Let us furthermore denote by $\tau_{i,m}^m(\vartheta_i, \boldsymbol{x}_i^*)$ the experienced time of completing the transmission to node m, using the optimal cut-point vector $\boldsymbol{x}_i^*$. Also we denote by $\boldsymbol{x}^m$ and $\boldsymbol{x}^{m+}$ the first m and the remaining $N-m$ elements of vector $\boldsymbol{x}$.

Proposition 3. *For all $m = 1 \ldots N-1$, $\mathbf{x}_i^{m+*} = \{x_{i,m+1}^*, x_{i,m+2}^*, \ldots x_{i,N}^*\}$, that is, the cut-point location vector calculated according to (14)-(17) minimizes the expected completion time $T_i^{N,m}\left(\vartheta_i, [\mathbf{x}_i^{m*}, \mathbf{x}_i^{m+}]\right)$ for any given $\tau_{i,m}^m(\vartheta_i, \mathbf{x}_i^*)$.*

Proof: We prove the theorem by contradiction. $T_i^{N,m}\left(\vartheta_i, [\mathbf{x}_i^{m*}, \mathbf{x}_i^{m+}]\right)$ can be minimized by minimizing the remaining expected completion time $\tau_{i,m+}\left(\vartheta_i, [\mathbf{x}_i^{m*}, \mathbf{x}_i^{m+}]\right)$, where $\mathbf{x}_i^{m+}$ is arbitrary.

Assume now that there exists $\boldsymbol{x}_i^{m+} \neq \boldsymbol{x}_i^{m+*}$, such that $\tau_{i,m+}^m(\vartheta_i, [\boldsymbol{x}_i^{m*}, \boldsymbol{x}_i^{m+}]) < \tau_{i,m+}^m(\vartheta_i, \boldsymbol{x}_i^*)$. Then, exchanging $\boldsymbol{x}_i^{m+*}$ with $\boldsymbol{x}_i^{m+}$, the completion time expected before the start of the transmission of the first sub-area of image i would be $\tau_{i,m-}\left(\vartheta_i, [\boldsymbol{x}_i^{m*}, \boldsymbol{x}_i^{m+}]\right) + \tau_{i,m+}\left(\vartheta_i, [\boldsymbol{x}_i^{m*}, \boldsymbol{x}_i^{m+}]\right) < T_i\left(\vartheta_i, \boldsymbol{x}_i^*\right)$, which is a contradiction. □

Thus, the optimal cut-point location vector does not need to be recomputed after the transmission starts.

7 Scheduling Order

From [18] it is known, that for given scheduling order, that is, given order of transmission to the processing nodes, the task completion time is minimized, if all the processing nodes completes the processing at the same time, while the achievable minimum depends on the scheduling order. Below we show that the existence of data transmission overlap affects the optimal scheduling method. We show that to minimize the completion time decisions need to be made: i) on the order of the transmission to the utilized processors, ii) on the number of processors to be utilized, and iii) whether the overlap should be transmitted multicast or by separate, unicast transmission to the two involved processors.

Let us consider the simplified case, when the processing time is proportional to the amount of received data, that is, $P_n = P_n^{d,px}$ and $P_n^{d,ip} = P_n^e = 0$, and there are only two processing nodes $\mathcal{N} = \{A, B\}$.

Proposition 4. *If overlap is not required, i.e. $o = 0$, the completion time is minimized by scheduling the nodes in increasing order of per bit transmission time.*

Proof: Here we recall the proof for $N = 2$. The extended version can be found in [19]. Consider two processing nodes, A and B, with $C_A \leq C_B$ and arbitrary processing capacities P_A and P_B. When node A is scheduled before node B, the completion times for image i are

$$T_{i,AB} = hw \begin{bmatrix} x_{i,1}C_A + \frac{x_{i,1}}{P_A} \\ x_{i,1}C_A + (1 - x_{i,1})C_B + \frac{1-x_{i,1}}{P_B} \end{bmatrix}. \tag{26}$$

This gives optimal cut-point location, under which the processing at node A and B completes at the same time

$$x_{i,1,AB}^* = \frac{P_A(1 + C_B P_B)}{P_A + P_B + C_B P_A P_B}, \tag{27}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

and the resulting minimum completion time is

$$T^*_{i,AB} = \frac{(1+C_AP_A)(1+C_BP_B)}{P_A+P_B+C_BP_AP_B}. \quad (28)$$

Scheduling the nodes in the reverse order gives

$$T_{i,BA} = hw \left[\begin{array}{c} x_{i,1}C_B + (1-x_{i,1})C_A + \frac{1-x_{i,1}}{P_A} \\ x_{i,1}C_B + \frac{x_{i,1}}{P_B} \end{array} \right], \quad (29)$$

The optimal cut-point location in this case is

$$x^*_{i,1,BA} = \frac{P_B(1+C_AP_A)}{P_A+P_B+C_AP_AP_B}, \quad (30)$$

and minimum completion time becomes

$$T^*_{i,BA} = \frac{(1+C_AP_A)(1+C_BP_B)}{P_A+P_B+C_AP_AP_B}. \quad (31)$$

Assume, $T^*_{i,BA} < T^*_{i,AB}$. As (28) and (31) differ only in one term in the denominator, $T^*_{i,BA} < T^*_{i,AB} \rightarrow C_A > C_B$, which contradicts the initial assumption $C_A \leq C_B$. □

Now we introduce transmission overlap $o > 0$.

Proposition 5. *Consider overlap $o > 0$. There exists some configuration of per bit transmission times and processing rates for which the scheduling order in increasing per bit transmission times is not optimal.*

Proof: Consider, as before $\mathcal{N} = \{A, B\}$, with $C_A \leq C_B$ and arbitrary P_A and P_B. The overlap is transmitted via multicast transmission with C_B.

Transmitting first to node A the completion times are

$$T_{i,AB} = hw \left[\begin{array}{c} (x_{i,1}-o)C_A + 2oC_B + \frac{x_{i,1}}{P_A} \\ (x_{i,1}-o)C_A + (1-x_{i,1}+o)C_B + \frac{1-x_{i,1}}{P_B} \end{array} \right], \quad (32)$$

and in the reverse order they become

$$T_{i,BA} = hw \left[\begin{array}{c} (x_{i,1}+o)C_B + (1-x_{i,1}-o)C_A + \frac{1-x_{i,1}}{P_A} \\ (x_{i,1}+o)C_B + \frac{x_{i,1}}{P_B} \end{array} \right]. \quad (33)$$

The optimal cut-point location and the related minimum completion time can be calculated as in Proposition 4. The expressions are rather cumbersome in this case. However, as $\frac{C_B}{C_A} \rightarrow \infty$, they give $T^*_{i,BA} < T^*_{i,AB}$, if

$$\frac{P_A}{P_B} > \frac{1-o}{o} \quad (34)$$

There is thus a ratio of processing rates for which reversed scheduling order, with increasing per bit transmission times, is optimal. □

Similar derivations provide the (C_A, P_A, C_B, P_B) parameter combinations where the unicast transmission of the overlap area is preferable, and when only one of the processors should be utilized. Leaving the exact expressions aside, in Figure 3 we show representative results, with parameters $C_A = 1/P_A$, $C_A \leq C_B$ and $o = 0.2$ Under given C_AP_A product and o value, the optimal transmission scheduling is a function of the ratios C_B/C_A and P_A/P_B. Only a single processor, processor A should be used in the parameter region marked with (A,-,U), that is, when the relative transmission time to A is low, and its relative processing speed is high. Moreover, the border of this region does not depend on the value of the overlap. If there is a significant difference in the transmission times, but the processing speeds are similar, then both of the processors should be used, and the overlap areas should be transmitted separately with unicast transmission. In the case of unicast transmission, it always holds that the fastest link should be scheduled first. This region is marked as (A,B,U) on the figure. Finally, according to Proposition 4, when the multicast transmission of the overlap area is optimal, the scheduling order depends on the ratio of the processing speeds, leading to parameter regions (A,B,M) and (B,A,M).

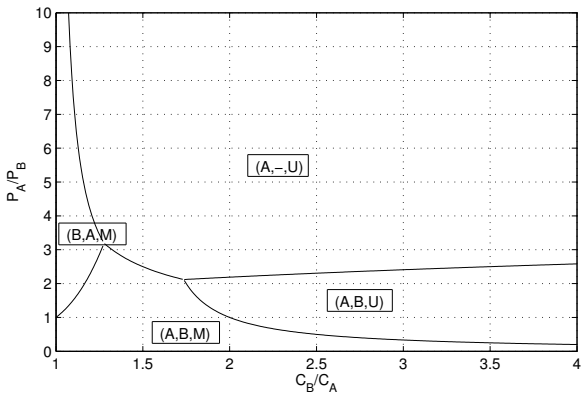

Figure 3: Optimal transmission scheduling schemes as a function of C_B/C_A and P_A/P_B, for $C_A = 1/P_A$, $C_A \leq C_B$ and $o = 0.2$.

8 Numerical Results

We performed simulations to evaluate the proposed algorithms on two surveillance video traces, both with $8\,\mathrm{bit}$ grayscale colorspace, 1920×1080 resolution, and frame rates of 25 frames per second, resulting in a raw bitrate of $415\,\mathrm{Mbps}$. One trace, referred to as the "Pedestrian" trace, consists of 375 frames and shows a pedestrian intersection with people moving horizontally across the field of view, covering and uncovering interest points in the background. The other trace, referred to as the "Rush hour" trace, consists of 473 frames and shows a road with vehicles moving slowly along the camera's line of sight, leading to mostly minor changes in the horizontal distribution of interest points.

We use BRISK [2] for interest point detection and feature description extraction, with $M^* = 400$ as the target number of interest points. As the size of the BRISK descriptor is $512\,\mathrm{bits}$, the average bitrate required for transmitting the descriptors to the sink node is $5\,\mathrm{Mbps}$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

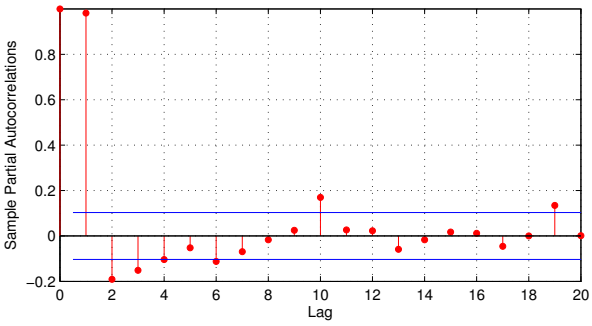

Figure 4: The partial autocorrelation function of the optimal detection threshold ϑ_i^* time series of the Pedestrian video sequence. The optimal detection threshold ϑ_i^* has a mean of 55.2 and standard deviation of 6.6.

When not noted otherwise, video is generated by one camera node, and is processed at N=6 processing nodes, all with equal processing rates similar to those of an Intel iMote2 ($P^{d,px} = 9 \times 10^4$ px/s, $P^{d,ip} = 94$ ip/s, $P^e = 25$ ip/s). The transmission time coefficients are $C = 6.7 \times 10^{-8}$ s/bit, and we use $Q = 10$ quantiles for the approximation of the interest point distribution $F_i(\vartheta, x)$, i.e., $\tilde{F}_i(\vartheta, \xi_q) = \frac{q}{Q}, q = 1, 2, \ldots, Q$.

8.1 Detection threshold reconstruction

We first evaluate our proposed threshold reconstruction schemes by comparing the threshold MSE, $\frac{1}{I}\sum_{i=1}^{I}(\vartheta_i^* - \hat{\vartheta}_i)^2$, of the different schemes.

As a basis of comparison for the proposed regression based threshold reconstruction we use two methods. The first method, referred to as the *Scaling* method, scales the predicted threshold $\hat{\vartheta}_i$ by a constant factor α, $0 < \alpha < 1$ whenever $f_i(\hat{\vartheta}_i) < M^*$, i.e., the reconstructed threshold for image i is $\hat{\vartheta}_i^{S*} = \alpha \cdot \hat{\vartheta}_i$. Off-line evaluation showed that the scaling method produces best results for $\alpha = 0.98$. The second method, referred to as the *Clairvoyant* method, has knowledge of $\hat{\vartheta}_i^* = \vartheta_i^* = f_i^{-1}(M^*)$ and $\hat{x}_i^* = x_i^* = \frac{\max(\Xi_i^*)+\min(\Xi_i^*)}{2}$, and can use it for predicting $\hat{\vartheta}_{i+1}$ and $\hat{x}_{i+1}$.

Figure 4 shows the partial autocorrelation function of the optimal threshold values ϑ_i^* for the Pedestrian trace, similar results were found for the Rush hour trace. The figure suggests that autoregressive (AR) models up to order 10 should be considered for predicting $\hat{\vartheta}_i$. Based on this observation we chose to use AR models of order 1, 2 and 10 for our numerical evaluations. The predictors are initially trained using the first 100 frames of the trace, and then retrained after each frame. Alongside the AR predictors we use the *Last value* predictor (denoted by $Y(i-1)$), which assumes that the content of image i is identical to that of image $i-1$, i.e. $\hat{\vartheta}_i = \hat{\vartheta}_{i-1}^*$.

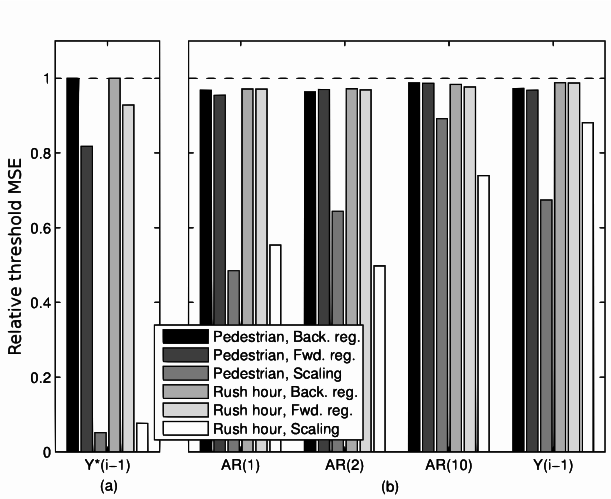

Figure 5: Relative MSE of (a) the reconstructed thresholds (predictors trained with *Clairvoyant* method) and (b) the predicted thresholds (predictors trained with reconstructed thresholds).

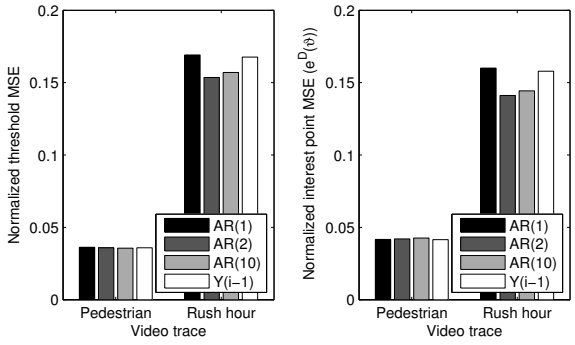

Figure 6: Mean square error of four threshold predictors in terms of threshold ϑ_i and in terms of detected interest points ($e^D(\boldsymbol{\vartheta})$).

In Figure 5 we show the threshold MSE for three reconstruction methods. Subfigure 5(a) was obtained by using the *Last value* predictor for predicting $\hat{\vartheta}_i$; whenever $f_i(\hat{\vartheta}_i) < M^*$ we use one of three reconstruction methods to reconstruct $\hat{\vartheta}_i^*$ and compute the resulting squared error $(\hat{\vartheta}_i^* - \vartheta_i^*)^2$. We then use the *Clairvoyant* method in order to avoid the propagation of the reconstruction error, when predicting $\hat{\vartheta}_{i+1}$. The values plotted are the MSE of the backward regression method, i.e., $(\hat{\vartheta}_i^{b*} - \vartheta_i^*)^2$, divided by the MSE of the three methods. The results show that backward regression performs best, in accordance with Proposition 1.

Subfigure 5(b) shows the threshold MSE of the *Last value* predictor combined with the *Clairvoyant* method, divided by the MSE of the predicted thresholds for four

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

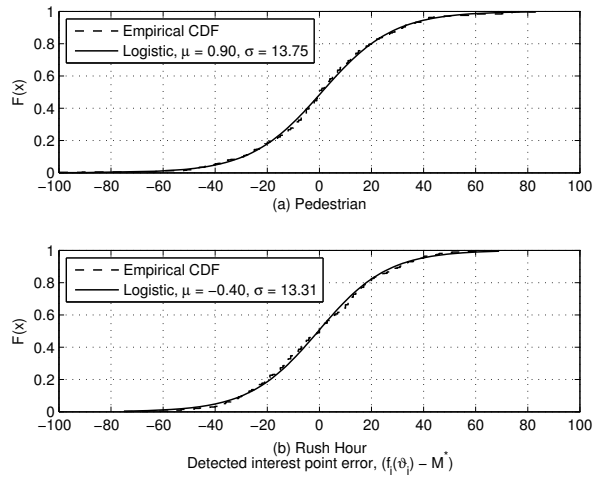

Figure 7: Distributions of $f(\hat{\vartheta}_i) - M^*$ using the *Last value* threshold predictor on the (a) Pedestrian and (b) Rush hour video traces.

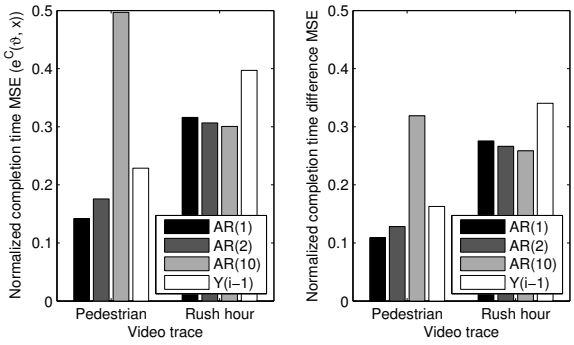

Figure 8: Mean square error of the completion time (e^C) and of the completion time difference of four percentile predictors.

predictors combined with three reconstruction methods. The results were obtained by using the reconstructed thresholds $\hat{\vartheta}_i^*$ for predicting the subsequent thresholds $\hat{\vartheta}_j, j > i$. This creates a feedback loop where the choice of predictor influences the frames for which reconstruction is needed, and reconstruction influences the prediction performance. From the figure we see that the backward regression method has a slight advantage over the forward regression method in almost all scenarios, and they both greatly outperform the *scaling* method. Therefore from this point on we only consider the backward regression method for threshold reconstruction.

8.2 Detection threshold prediction

We next evaluate the performance of different predictors in terms of threshold and interest point MSE. We normalize the performance results to the performance of a non-adaptive offline scheme, which we call the *Fixed* scheme. The *Fixed* scheme has complete knowledge of all parameters in each frame; it uses a fixed detection threshold $\vartheta^s = \arg\min_\vartheta e^D(\vartheta)$ and a fixed cut-point location vector that minimizes the completion time assuming the interest point distribution is $F(\vartheta, x) = \frac{1}{I}\sum_{i=1}^{I} F_i(\vartheta_i^*, x)$.

Figure 6 shows the performance in terms of MSE of three AR models and of the *Last value* predictor $Y(i-1)$. The AR models are initially trained using the first 100 frames of the trace and are then retrained after each frame. The left plot shows the MSE of the threshold prediction, i.e., $\frac{1}{I}\sum_{i=1}^{I}(\vartheta_i^* - \hat{\vartheta}_i)^2$, the right plot shows the MSE in terms of detected interest points, i.e., $e^D(\boldsymbol{\vartheta})$. The MSE results are normalized by the corresponding MSE of the *Fixed* scheme. The figure shows that threshold prediction decreases the MSE compared to the *Fixed* scheme by a factor of 5 to 20 depending on the trace. At the same time the gain of using a higher order predictor is small when compared to the *Last value* or the AR(1) predictor, especially for the Pedestrian trace.

Figure 7 shows the CDF of the deviation for the number of detected interest points from M^* for the Pedestrian and Rush hour sequences. The distributions fit well to logistic distributions with parameters $\mu = 0.90$, $\lambda = 13.75$ and $\mu = -0.40$, $\lambda = 13.31$, respectively. The heavier tail of the logistic distribution compared to the normal well describes the occasional large error in the number of detected interest points caused by sudden changes in the contents of a trace. It is also interesting to note that, despite the differences in the contents of the traces, the two distributions show great similarities.

8.3 Completion time minimization

Figure 8 shows results for the completion time MSE using the proposed percentile based prediction, i.e., each of the Q percentile points is predicted by an AR model or by the *Last value* predictor. Prediction decreases the MSE by up to a factor of 10 compared to the *Fixed* scheme. The two traces show different results in the performance of the predictors. For the Rush hour trace there is some advantage of choosing a higher order predictor, although the marginal performance gain decreases as the order increases. In this case the choice of the predictor should be based on the trade-off between the achieved performance and the computational complexity of training the predictor. For the Pedestrian trace, however, we see very different results, as the performance deteriorates with higher prediction order; the AR(10) performs significantly worse than the *Last value* predictor. In the following we discuss the reasons for this counter-intuitive result.

Figure 9 shows the completion time MSE achieved when using AR and vector AR (VAR) predictors for percentile prediction as a function of the predictor order p, again normalized by the MSE of the *Fixed* scheme.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

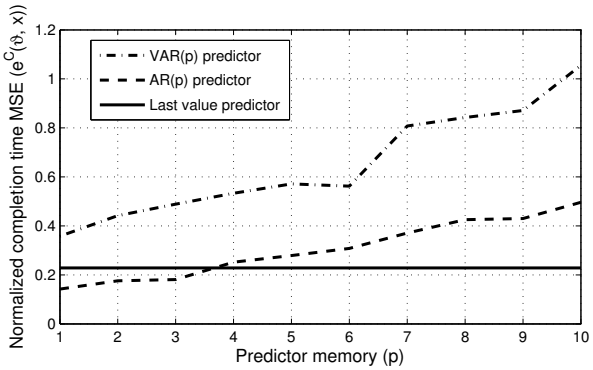

Figure 9: Mean square error of completion times for different predictors and for varying p, under percentile prediction.

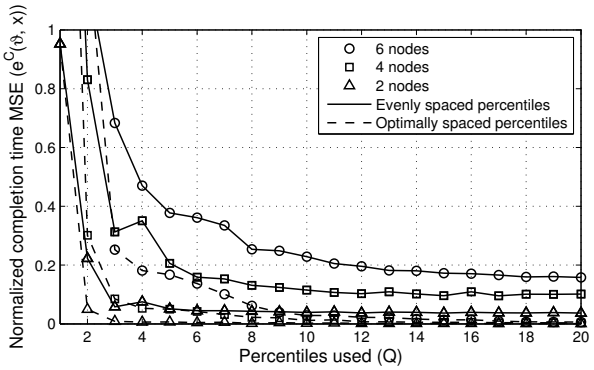

Figure 11: Normalized completion time MSE as a function of the number of percentiles used for distribution approximation.

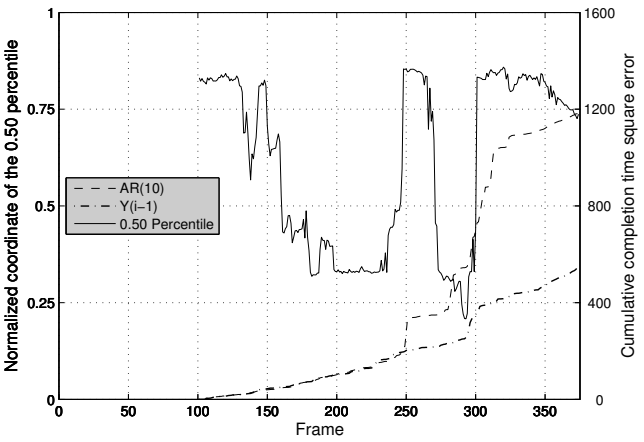

Figure 10: Cumulative square errors of two different predictors, together with the coordinate of the 0.50 interest point distribution percentile.

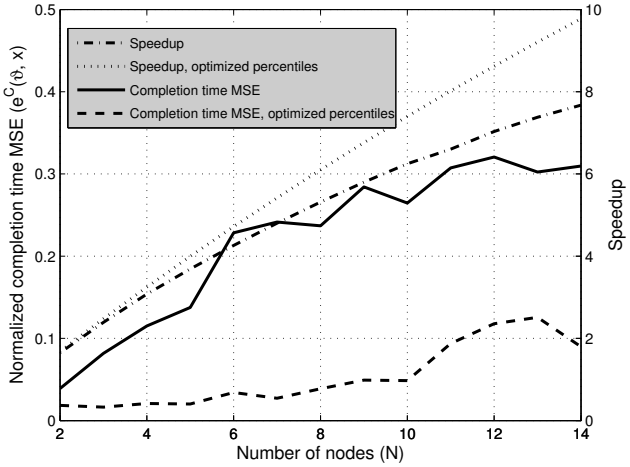

Figure 12: Normalized completion time MSE and mean completion time speedup as a function of N. As a baseline for the speedup values, the completion time when using a single processing node is approximately $44.4\,\mathrm{s}$.

We see that the performance of the AR predictor decreases with increased prediction order. Interestingly, the VAR predictor, which could capture the correlation between the different percentile coordinates, performs consistently worse than the independent AR predictors.

To explain the reason for the poor performance of high order predictors, Figure 10 shows the evolution of the cumulative square error (i.e., not normalized by the number of images I) for the sequence of images for the AR(10) and the *Last value* predictor. The results confirm that due to the longer memory of the AR(10) predictor it needs longer time to adjust to large and sudden changes in the image contents. We see, for instance, that a large portion of the total square error for AR(10) emerges during frames 250–320. These frames correspond to a 3 second part of the trace where a tight cluster of interest points in the right side of the scene is first revealed, concealed, and then revealed again very suddenly. The reason why the *Last value* predictor can outperform the AR predictors is that the error criterion that has to be used to train the predictors is not the deviation from the minimal completion time but the error in predicting the percentile coordinates. As the interest points tend to appear in clusters, a small error in percentile prediction and cut-point selection can produce a large discrepancy between the actual number of interest points in the slices.

8.4 Approximation of the interest point distribution

So far we used quantiles as the percentiles for approximating the interest point distributions. Figure 11 compares the normalized MSE of the completion time

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

for the quantile based approximation to an approximation that chooses the percentiles so as to minimize the square error of the approximation. The predictor used is the *Last value* predictor, and the results are normalized by the completion time MSE of the *Fixed* scheme. The figure shows that optimizing the percentiles improves the prediction performance significantly and reduces the number of percentiles needed for the same performance, especially when the number of processing nodes is high ($N = 6$). However, achieving this performance improvement comes at the price of optimizing the percentile locations, which is again computationally intensive.

In Figure 12 we show the MSE (left axis) of the *Last value* predictor as a function of the number of nodes N for $Q = 10$ percentiles and normalized by the completion time MSE of the *Fixed* scheme. We see that as N increases so does the difference in completion time MSE. On the right axis we show the speed-up of the mean completion times achieved using the two approximations relative to the completion time when using a single processing node. The speed-up is close to linear for both percentile fitting methods, but using optimized percentiles provides consistently higher speedup. It is worth noting that the difference in terms of speedup (and hence completion time) between the two percentile fitting methods is rather small, which indicates that the large difference in terms of completion time MSE is due to occasional large errors caused by the quantile-based approximation, which are penalized by the quadratic error function.

Consequently, if large completion times can be tolerated occasionally then the quantile based approximation with the *Last value* predictor constitute a computationally simple algorithm with good performance.

8.5 Impact of the channel randomness

So far we considered that the time it takes to transmit data to the processing nodes is deterministic, given by the transmission time coefficient C_n. In practice, however, the time needed to transmit data varies due to wireless channel impairments. In the following we show simulation results from a system with $N = 6$ processing nodes to assess the impact of the variation of the transmission time on the completion time MSE.

For the simulations we use the OMNET++/INET framework, implementing standard low-power IEEE 802.11b physical and link layer protocols, suitable for use in VSNs. We consider fixed transmission power, fixed modulation and coding at the physical layer, resulting in $199\,\mathrm{m}$ free-space transmission range. The wireless channels are subject to independent block Rician fading, on a per frame basis. The parameter K of the Rician fading channel determines the mean and the variance of the receivers' SNR, a higher K corresponding to higher mean and lower variance. We use UDP at the transport layer, and an application layer protocol providing reliable transmission. The application layer protocol transmits image subareas in multicast or in unicast, fragmented into 2276 bytes long UDP segments. After the camera node transmits all fragments, each receiver sends either a positive acknowledgement, if all fragments were correctly received, or a negative acknowledgement requesting the retransmission of lost fragments, otherwise. If the acknowledgement is lost, a time-out will be triggered after which the camera will ping the processing node to request a retransmission of the lost acknowledgement. Once the camera has received a positive acknowledgement from all processing nodes, the transmission has been completed successfully.

First, we consider the case when K, and thus the transmission time coefficient C_n, is known. We set the distance between the camera and the processing nodes to $187\,\mathrm{m}$. Figure 13 shows the completion time MSE, e^C for five different scenarios as a function of K. The corresponding frame loss rates are between $55\,\%$ and $45\,\%$. In the three scenarios denoted by *Rician* the transmission times are determined by the simulated Rician channels with parameter K, and *Fixed*, *Last Value* or *Oracle* is used for prediction. In the two scenarios denoted by *Constant* the transmission times are constant as given by C_n and *Fixed* or *Last Value* is used for prediction. Note, that the *Constant* and *Oracle* combination gives the optimal completion time value with $e^C = 0$.

Comparing the curve *Constant, Fixed* with the curve *Rician, Oracle* we can compare the MSE introduced by the randomness of the channel and of the video content, respectively. We see that for all values of K, the channel randomness has a small effect on the MSE, and as K increases, the MSE due to the channel randomness decreases. Comparing the three *Rician* curves to the corresponding *Constant* curves (and the $e^C = 0$ line in the case of *Rician, oracle*), we see that the completion time MSE increase due to channel randomness is slightly higher when the MSE due to video content randomness is already high.

Figure 14 shows the completion time MSE, as a function of the distance between the camera and the processing nodes, for $K = 3$. The corresponding frame loss rate is between $10\,\%$ and $50\,\%$. We see that for short distances and consequently relatively low loss rates, the randomness of the Rician channel has a negligible effect on the completion time MSE, but as the distance increases, the error contribution from the channel randomness shows an exponential increase. Still, the gap between the *Rician, Fixed* and *Rician, Last Value* MSE remains. These results show that even if the wireless channel introduces significant randomness in the time it takes to transmit sub-areas of the image, the randomness of the image content is the main source of completion time error, thus prediction-based completion time minimization is essential.

Next we consider the more practical case that K is unknown, and thus C_n needs to be estimated based on observed transmissions. We consider two estimators for C_n. The first estimator is the arithmetic mean: before transmission of image i the transmission time coefficient of a processing node n is estimated as the sum of all

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

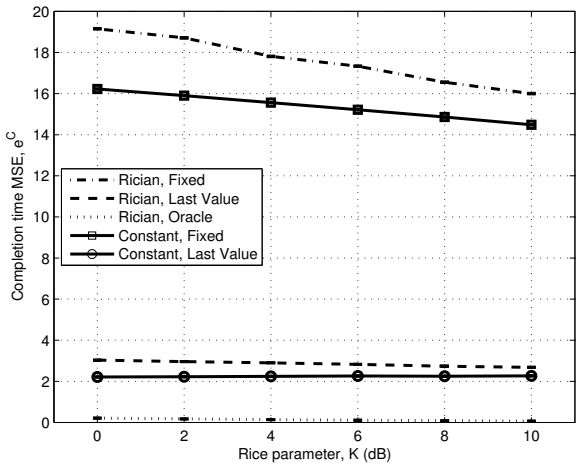

Figure 13: Completion time MSE (e^C) for different predictors and channel models. The 95 % confidence intervals are small and therefore not visible.

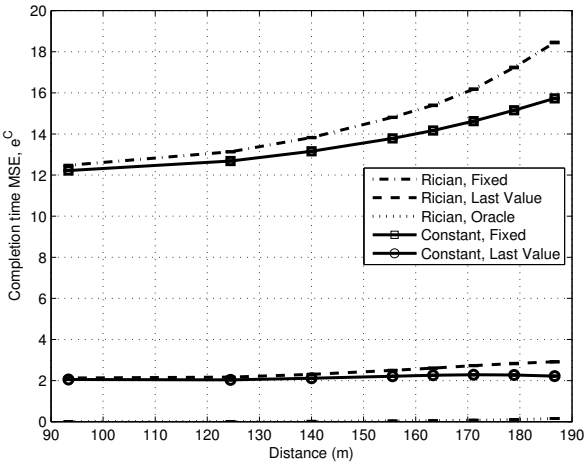

Figure 14: Completion time MSE (e^C) for different predictors and channel models, with Rice parameter $K = 3$, varying transmission distance. The 95 % confidence intervals are small and therefore not visible.

past transmission times divided by the total number of pixels transmitted to the processing node, i.e., $\widehat{C}_{i,n} = \frac{\sum_{j=1}^{i-1} T_{transmit,j,n}+T_{overlap,j,n}/2}{hw\sum_{j=1}^{i-1} x_{j,n}-x_{j,n-1}}$. If the channel is stationary and fading is independent then the arithmetic mean is the minimum variance unbiased estimator of C_n. The second estimator is exponential smoothing with smoothing factor α, $\widehat{C}_{i,n} = \alpha\frac{T_{transmit,i-1,n}+T_{overlap,i-1,n}/2}{hw(x_{i-1,n}-x_{i-1,n-1})} + (1-\alpha)\widehat{C}_{i-1,n}$, which is often used in practice when channels exhibit temporal correlation or are non-stationary.

Figure 15 shows the normalized completion time MSE for different K values, with estimated transmission time coefficients. The MSE is normalized by the completion time MSE of the case with fading and known C_n. We see that the estimator that gives the highest completion time MSE is exponential smoothing with $\alpha = 1.0$, which is simply a last value predictor. The MSE decreases with α, because the estimator becomes less sensitive to fluctuations in the transmission times, and at $\alpha = 0.1$ exponential smoothing almost matches the results of the arithmetic mean. Thus, a relatively high value of $\alpha = 0.1$ could represent a good trade-off between adaptation capability to non-stationary channels and low MSE on stationary channels.

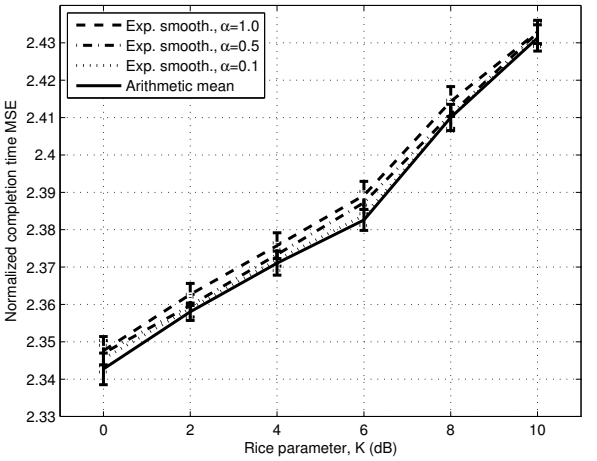

Figure 15: Normalized completion time MSE (e^C) for different estimators of the average C_n using the *Last value* predictor and Rician channel model. MSE is normalized by the completion time MSE for the case when C_n is known. Bars show 95 % confidence intervals.

9 Conclusion and Future Work

We considered the problem of minimizing the completion time of distributed interest point detection and feature extraction in a visual sensor network. We formulated the problem as a stochastic multi-objective optimization problem. We proposed a regression scheme to support the prediction of the detection threshold so as to maintain a target number of interest points, and a prediction

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

scheme based on a percentile-based approximation of the interest point distribution for minimizing the completion time. Our numerical results show that prediction is essential for achieving good system performance. The gain of high order predictors is moderate in general, and depending on the characteristics of the video trace it may even be detrimental to system performance to use higher order prediction models. Our results show that the simple AR(1) and the last value predictors together with a quantile-based approximation of the interest point distribution offer good performance at low computational complexity, making them good candidates for use in visual sensor networks. We considered the effect of the randomness of the wireless channel and demonstrated that prediction-based completion time minimization is crucial, even when the transmission times can vary significantly.

Our model could be extended to fast fading and correlated wireless channels and to dynamically evolving network topologies, in which case node unreachability needs to be handled. Another interesting direction for future work could be to maximize the network lifetime under completion time constraints which may require pipelined processing.

ACKNOWLEDGMENTS

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 No.: 296676.

REFERENCES

[1] 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.

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

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] A. Redondi, M. Cesana, and M. Tagliasacchi, "Rate-accuracy optimization in visual wireless sensor networks," in *Proc. of IEEE International Conference on Image Processing (ICIP)*, 2012.

[10] P. Monteiro and J. Ascenso, "Clustering based binary descriptor coding for efficient transmission in visual sensor networks," in *Picture Coding Symposium*, Dec 2013.

[11] L. Baroffio, M. Cesana, A. Redondi, and M. Tagliasacchi, "Performance evaluation of object recognition tasks in visual sensor networks," in *26th International Teletraffic Congress (ITC)*, Sept 2014.

[12] P. Monteiro, J. Ascenso, and F. Pereira, "Local feature selection for efficient binary descriptor coding," in *IEEE International Conference on Image Processing (ICIP)*, Oct 2014.

[13] 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.

[14] 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.

[15] 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.

[16] M. A. Khan, G. Dán, 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.

[17] ——, "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.

[18] 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.

[19] 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.

[20] 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.

[21] E. Eriksson, G. Dán, and V. Fodor, "Prediction-based load control and balancing for feature extraction in visual sensor networks," in *Proc. of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[22] ——, "Real-time distributed visual feature extraction from video in sensor networks," in *Proc. of International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2014.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2015.2465390, IEEE Transactions on Mobile Computing

[27] 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.

[28] 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.

[29] 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.

[30] "OpenCV." [Online]. Available: http://opencv.org/

[31] F. B. Abdelaziz, "Solution approaches for the multiobjective stochastic programming," *European Journal of Operations Research*, vol. 216, pp. 1–16, 2012.

Emil Eriksson (S'13) received the M.Sc. in engineering physics from Uppsala University, Uppsala, Sweden, in 2014. He is currently pursuing his Ph.D. at KTH Royal Institute of Technology, Stockholm, Sweden. His research interests include wireless sensor networks, resource constrained systems and distributed computing.

György Dán (M'07) is an associate professor at KTH Royal Institute of Technology, Stockholm, Sweden. He received the M.Sc. in computer engineering from the Budapest University of Technology and Economics, Hungary in 1999, the M.Sc. in business administration from the Corvinus University of Budapest, Hungary in 2003, and the Ph.D. in Telecommunications from KTH in 2006. He worked as a consultant in the field of access networks, streaming media and videoconferencing 1999-2001. He was a visiting researcher at the Swedish Institute of Computer Science in 2008, a Fulbright research scholar at University of Illinois at Urbana-Champaign in 2012-2013, and an invited professor at EPFL in 2014-2015. His research interests include the design and analysis of content management and computing systems, game theoretical models of networked systems, and cyber-physical system security in power systems.

Viktoria Fodor (M'03) received the M.Sc. and Ph.D. degrees from the Budapest University of Technology and Economics, Budapest, Hungary, in 1992 and 1999, respectively, both in computer engineering. In 1994 and 1995, she was a Visiting Researcher with Polytechnic University of Turin, Turin, Italy, and with Boston University, Boston, MA. In 1998, she was a Senior Researcher with the Hungarian Telecommunication Company. Since 1999, she has been with the KTH Royal Institute of Technology, Stockholm, Sweden, where she now acts as an Associate Professor with the Laboratory for Communication Networks. Her current research interests include network performance evaluation, cognitive and cooperative communication, protocol design for sensor and multimedia networking.