A Mathematical Programming Approach to Task Offloading in Visual Sensor Networks

Alessandro Enrico Redondi, Matteo Cesana*, Luca Baroffio, Marco Tagliasacchi
Dip. Elettronica, Informazione e Bioingegneria
Politecnico di Milano
Email: {name.surname}polimi.it

***Abstract*—This work studies how visual analysis tasks based on feature extraction can be speeded up in the context of Visual Sensor Networks. The main catch is for the camera node to leverage the presence of neighboring sensor nodes and offload the task, thus parallelizing its execution. We propose two mathematical programming formulations for the optimal visual task offloading problem: the first one targets the minimization of the overall task completion time while enforcing energy consumption constraints onto the nodes; the second maximizes the overall sensor network lifetime subject to a temporal constraint on the task completion time. The aforementioned formulations are used to characterize the achievable speed-up and consequent energy consumption in representative visual sensor network topologies.**

***Index Terms*—Mathematical Programming, Visual Sensor Networks, Cooperative Processing**

I. Introduction

The Internet of Things vision is pushing the role of powerful smart cameras down to battery-operated sensing nodes empowered with sight and capable of visual analysis tasks (e.g., object recognition, event detection, localization, tracking), giving rise to Visual Sensor Networks (VSNs). As an example, in the context of smart cities, the availability of battery-operated visual nodes enables a capillary coverage of the urban landscape, reaching a wider area and limiting the costs of the required infrastructure. On the other side, enabling visual analysis in energy-constrained VSNs calls for the design of novel interdisciplinary solutions in the fields of image/signal processing and networking, as data acquisition, processing and transmission have to be implemented with limited resources.

Many analysis tasks are implemented by relying on a features-based representation of the visual content according to the following pipeline: (i) the camera nodes acquire an image; (ii) local visual features are extracted from the image: this allows to efficiently capture the image's most salient points; (iii) the camera nodes send a compressed version of the extracted features to a central controller, which typically performs the analysis by relying on some sort of processing on the received features. As an example, for the task of object recognition, the received local features are matched against a local database to identify the most similar object.

Applying the aforementioned pipeline to VSNs is indeed challenging, as all the steps above need to be adapted to the resource constrained scenario of sensor networks. In particular, extracting features from visual data is often a computationally-intensive task [1]. For the case of local features, this process entails detecting image *keypoints* and computing the corresponding *descriptors*, a process whose computational complexity grows linearly with the image size and with the number of scales of the image which are processed[1] [2]. As an example, even when using feature extraction algorithms tailored to low-power architectures (e.g. BRISK [3]), the complete process can take as much as 3.5s for a single VGA image, as illustrated in Figure 1. Clearly, this is a severe limit for those tasks which require responsiveness, such as event detection or tracking.

For this reason, in this work we aim at reducing the time taken by the feature extraction algorithm. The main catch is for the camera node to leverage the presence of neighboring sensor nodes to offload the visual processing task and parallelizing its execution. Namely, we address the optimal sharing of the processing burden of features extraction among cooperating sensor nodes, by thoroughly analyzing the trade offs between the processing time, which is reduced through offloading, and the communication (signaling) overhead which is needed to support the offloading process.

The problem of offloading jobs/tasks in computer networks is largely debated in the literature and dates back to first works on orchestrating tasks across grids of processors. In this context, Divisible Load Theory (DLT) provides elegant and compact results on the optimal offloading pattern from a central processor to individual cooperating processors under different topologies [4] [5] [6]. More recently, DLT has been applied to wireless sensor networks [7] [8] to orchestrate the processing of large data. However, the main focus of these works is to minimize the task completion time without explicitly targeting the energy consumed in the process. Energy consumption is often analyzed after the computation of the optimal offloading schedule (i.e., number, identity, and sequence of sensor nodes to be used), as done in [8].

Here we take a different approach, and we propose Mathematical programming formulations to derive the offloading schedule, instead of relying upon DLT. Two different scenarios are addressed: first, we propose a mathematical programming formulation to find the offloading schedule which minimizes

* Matteo Cesana is also with MobiMesh srl (Spin-off Politecnico di Milano)

[1] Downsampled versions of the image which are used in the feature extraction process to guarantee invariance to scale transformations.

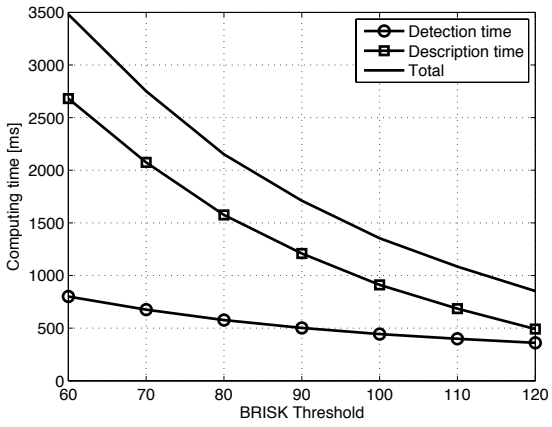

Fig. 1. BRISK detection/description completion time when running on a Linux-based Beaglebone hardware with VGA images (640x480).

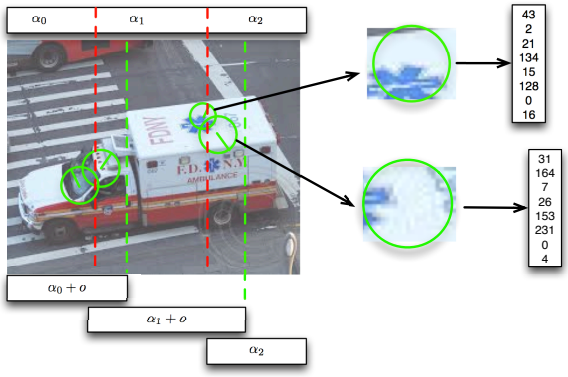

Fig. 2. Visual features extraction: several keypoints (green circles) are detected on the input image. The patch around each keypoint is then encoded in a numerical vector by the descriptor algorithm.

the overall visual task completion time subject to energy budget constraints for the sensor nodes involved in the offloading process; second, we introduce a formulation which finds the offloading schedule such that network lifetime is maximized, imposing constraints on the overall execution time for the features extraction task. The proposed formulations are then leveraged to assess the optimal offloading pattern in different network configurations.

The manuscript is organized as follows: Section II gives an overview on local visual feature extraction, whereas Section III introduces the reference scenario and describes the addressed problem; in Section IV we introduce the mathematical programming formulations to optimize the offloading of visual feature extraction; Section V reports on numerical results obtained by applying the proposed formulation to realistic VSNs; Section VI concludes the manuscript.

II. Offloading the Task of Features Extraction

Visual features are routinely adopted in several applications, including image/video retrieval, object recognition, tracking, etc. First, a *detector* analyzes the image to extract salient keypoints that represent its most informative parts; the number of extracted keypoints (and in turn the overall execution time) depends on (i) the image size, (ii) the visual content and (iii) a *threshold* parameter which can be tuned to set the detector sensitiveness. Then, image patches around each keypoint are further processed and compactly represented by means of fixed-dimensional vectors, known as descriptors, that capture their photometric properties. Referring to Figure 2, four keypoints are detected (centers of the four circles) and the patches around each keypoint are encoded in the corresponding descriptors. Keypoints and descriptors are then transmitted to a central controller, which uses them to perform high-level analysis tasks. Even if the present work is general and applicable to any detector/descriptor combination, here we focus on the BRISK [3] features extractor, which is tailored to low-power visual sensor node platforms, such as the BeagleBone Linux computer [9]. Figure 1 shows BRISK running times on such platform, for a VGA input image.

When it comes to offloading the task of features extraction in a visual sensor network, the camera node may distribute parts of the load to neighboring nodes by dividing the acquired image in vertical overlapping slices of fixed height and variable width (see Figure 2). Each slice is then transmitted and assigned for processing (keypoint detection and descriptor extraction) to a cooperating node. Recalling that each descriptor is computed starting from a patch of pixels around a keypoint, the overlap between slices is required to correctly compute descriptors near slices borders. We denote by o (in percentage with respect to the total image size I) the overlap between two adjacent slices. For convenience we will consider that the overlap between two slices is attached to the slice with lower index. Note that this implies that the last slice of the image does not need any overlap.

III. Reference Scenario and Problem Statement

We consider a scenario where a single camera node acquires an image whose size is I [bit] at a given frequency of $1/t$ [images/s]. The camera node is characterized by a processing rate of v_0 [s/bit]. The camera node is surrounded by N sensor nodes in direct communication range. Each sensor node is characterized by a processing rate of v_i [s/bit], and the link between the camera node and the i-th sensor node is characterized by an equivalent bit rate of c_i [s/bit]. A node's equivalent bit rate captures several factors, such as the the nominal bit rate, the effect of retransmissions needed to deal with packet losses, and the time-variant nature of the wireless signal quality. The camera node and the cooperators are characterized by an initial energy budget, $\bar{E}_i$, and by the values of the processing and transmission/reception power consumption P_i^{cpu} and $P_i^{\text{tx/rx}}$, with $i = 0, 1, \ldots, N$.

The camera node may decide to offload the features extraction to the N sensor nodes. An offloading schedule is defined as $\mathcal{S} = \{\alpha_0, \alpha_1, \alpha_2, \cdots, \alpha_N\}$, being $\alpha \in \mathbb{R}$, $\alpha_i \geq 0$ and $\sum_{i=0}^{N} \alpha_i = 1$ the fraction of the original image offloaded for

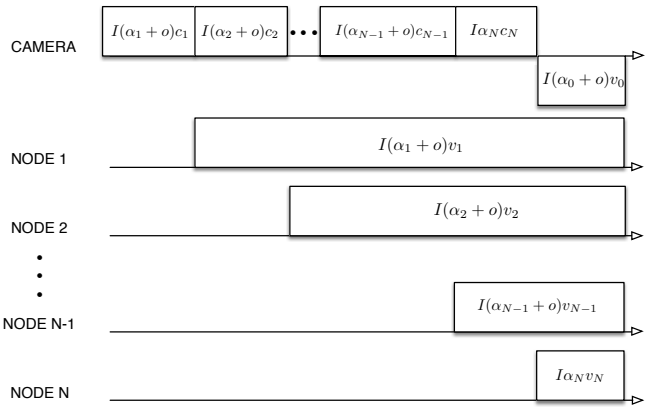

Fig. 3. Image splitting and unicast offloading

processing to the i-th cooperator in the offloading schedule (see Figure 2). In this notation, α_0 is the fraction of the original image which the camera decides to process locally.

Once the offloading sequence has been determined, the camera node distributes the loads to the cooperating nodes through sequential unicast transmissions as illustrated in Figure 3; in details, the camera first sends the loads to each cooperator, which requires a transmission time $I(\alpha_i + o)c_i$; each cooperator starts processing its load upon receiving it from the camera, and the processing time is assumed to be linear with respect to the slice size, that is, $I(\alpha_i + o)v_i$. This linear model is consistent with recent findings about the feature extraction time growing linearly with the number of keypoints in the image, and the keypoint distribution being uniform along the horizontal and vertical directions in many reference image data sets [10]. In case the assumption on uniformity of keypoint distribution is not valid, load balancing techniques can be applied in order to achieve a given number of detected keypoints in a given image slice [11], [12]. In this reference scenario, the Divisible Load Theory addresses the problem of determining the optimal offloading sequence by relying on the following fundamental result [13]:

Proposition III.1. *In the unicast offloading scenario of Figure 3, once the nodes that are used for cooperation are decided, the offloading schedule which minimizes the overall task completion time requires all the cooperators to complete processing at the same time.*

Such proposition applies only when number and identity of cooperators are given. Moreover, the DLT in general does not consider the energy consumption of offloading process. Differently, we aim at finding both the number/identity of cooperators to be used and the relative load shares further accounting for the energy consumed in the offloading process. Namely, the following problems are addressed:

- Find the offloading schedule $\mathcal{S}_{\mathcal{T}}^*$ which minimizes the total completion time of the reference visual task subject to per-node energy consumption constraints.
- Find the schedule $\mathcal{S}_{\mathcal{L}}^*$ which maximizes the network lifetime subject to constraints on the required frame rate $1/t$.

IV. Mathematical Programs

A. Time-Optimal Offloading Schedule

The first problem we address is the minimization of the overall image processing time, that is, we aim to find the offloading schedule $\mathcal{S}_{\mathcal{T}}{}^* = \{\alpha_0, \alpha_2, \alpha_3..\alpha_N\}$ such that the feature extraction completion time t is minimized; energy constraints are further added to enforce that under the offloading schedule, the network lifetime is at least R rounds, that is, the system can process up to R consecutive images before the first node in the network depletes its energy.

Without loss of generality, we assume that the available cooperators are characterized by an ordering rule, that is, each cooperator is assigned an index $j \in \mathbb{N}$ with $j \leq N$. To express which cooperators are used at a specific position of the offloading schedule, we define the binary decision variables y_{ij} which are 1 if processor j is used for offloading at position i of the offloading schedule. Moreover, it is useful to keep track of the number of cooperators which are actually used for offloading out of the original N. To this end, the following variables are defined:

$$n_k = \sum_{j=1}^{N} y_{kj} \qquad k = 1, \dots N \tag{1}$$

$$l_k = n_k - n_{k+1} \qquad k = 1, \dots N-1 \tag{2}$$

$$l_N = n_N \tag{3}$$

Variables n_k specify if the k-th position in the offloading schedule is used, that is, if at least k cooperators are used for offloading; variables l_k are 1 only if position k is the last-used position in the offloading schedule, that is, exactly k cooperators are used for offloading.

The time-optimal offloading schedule problem with energy constraints can then be formalized as follows:

$$\text{minimize } t \tag{4}$$

s.t.

(1) − (3)

$$I\left[(\alpha_0+o)v_0 + \sum_{k=1}^{N}\sum_{j=1}^{N} y_{kj}(\alpha_j+o)c_j - \sum_{k=1}^{N}\sum_{j=1}^{N} l_k y_{kj} o c_j\right] \leq t \tag{5}$$

$$I\left[\sum_{k=1}^{i}\sum_{j=1}^{N} y_{kj}(\alpha_j+o)c_j + \sum_{j=1}^{N} y_{ij}(\alpha_j+o)v_j - l_i o \sum_{j=1}^{N} y_{ij}(c_j+v_j)\right] \leq t \qquad i = 1 \dots N \tag{6}$$

$$\sum_{i=1}^{N} y_{ij} \leq 1 \qquad j = 1 \dots N \tag{7}$$

$$\sum_{j=1}^{N} y_{ij} \leq 1 \qquad i = 1 \dots N \tag{8}$$

$$\alpha_0 + \sum_{k=1}^{N}\sum_{j=1}^{N} y_{kj}\alpha_j = 1 \tag{9}$$

$$I\left[P_0^{\text{cpu}}(\alpha_0+o)v_0 + P_0^{\text{rx}}\sum_{k=1}^{N}\sum_{j=1}^{N} y_{kj}c_j(\alpha_j+o-l_k o)\right] \leq \frac{\bar{E}_0}{R} \tag{10}$$

$$I\left[\sum_{k=1}^{N} y_{kj}(\alpha_j+o)(v_j P_j^{\text{cpu}} + P_j^{\text{rx}} c_j) - \sum_{k=1}^{N} y_{kj} l_k o (P_j^{\text{rx}} c_j + P_j^{\text{cpu}} v_j)\right] \leq \frac{\bar{E}_j}{R} \qquad j = 1 \dots N \tag{11}$$

$$\alpha_j \geq 0 \qquad j = 0 \dots N \tag{12}$$

The objective function (4) together with constraints (5) and (6) enforce the minimization of the task completion time. Namely, the left-hand side of constraint (5) is the overall time required by the camera node to process its own share of the original image (first term) and send the remaining image shares to the cooperating nodes (second and third terms); similarly, the constraints (6) impose that the time required by each chosen cooperator to process the assigned image share is upper-bounded by t. Constraints sets (7) and (8) are used to enforce that one processor is scheduled in at most one position of the offloading schedule. Constraints (9) require that the entire image is processed. Constraints (10) and (11) enforce that the lifetime of the generic processor and of the camera is such that at least R images can be processed. The left-hand side of these constraints represent the energy consumption of a node which is calculated as the product of the required processing, transmission and reception time with the relative power consumption of the reference hardware. Parameters $\bar{E}_j$ are the energy budget of the generic processor j. Constraints (12) define decision variables α. Note that, differently from the DLT, such formulation does not ensure that all the nodes involved in the offloading process finish their processing at the same time.

B. Energy-Optimal Offloading Schedule

Specific application scenarios may require a minimum image processing rate. As an example, face recognition tasks of people entering a room may require to acquire and process images at a rate $\frac{1}{t} = 10$ [images/second]. It is thus reasonable to design the offloading schedule in order to maximize the network lifetime while guaranteeing that the image processing rate is above the target rate. The previous formulation can be modified as follows in order to maximize the network lifetime under strict constraints on the minimum required image processing rate:

$$\text{maximize } \epsilon \tag{13}$$

s.t.

$$(1) - (3)$$

$$I\left[(\alpha_0+o)v_0 + \sum_{k=1}^{N}\sum_{j=1}^{N} y_{kj}(\alpha_j+o)c_j - \sum_{k=1}^{N}\sum_{j=1}^{N} l_k y_{kj} o c_j\right] \leq t \tag{14}$$

$$I\left[\sum_{k=1}^{i}\sum_{j=1}^{N} y_{kj}(\alpha_j+o)c_j + \sum_{j=1}^{N} y_{ij}(\alpha_j+o)v_j - l_i o \sum_{j=1}^{N} y_{ij}(c_j+v_j)\right] \leq t \quad i = 1 \ldots N, \tag{15}$$

$$\sum_{i=1}^{N} y_{ij} \leq 1 \qquad j = 1 \ldots N \tag{16}$$

$$\sum_{j=1}^{N} y_{ij} \leq 1 \qquad i = 1 \ldots N \tag{17}$$

$$\alpha_0 + \sum_{k=1}^{N}\sum_{j=1}^{N} y_{kj}\alpha_j = 1 \tag{18}$$

$$E_0 - I\left[P_0^{\text{cpu}}(\alpha_0+o)v_0 + P_0^{\text{tx}}\sum_{k=1}^{N}\sum_{j=1}^{N} y_{kj}c_j(\alpha_j+o-l_k o)\right] \geq \epsilon \tag{19}$$

$$E_j - I\left[\sum_{k=1}^{N} y_{kj}(\alpha_j+o)(v_j P_j^{\text{cpu}} + P_j^{\text{rx}}c_j) - \sum_{k=1}^{N} y_{kj} l_k o (P_j^{\text{rx}}c_j + P_j^{\text{cpu}}v_j)\right] \geq \epsilon \quad j = 1 \ldots N \tag{20}$$

$$\alpha_j \geq 0 \qquad j = 0 \ldots N \tag{21}$$

The objective function (13) together with constraints (20) and (19) aim at maximizing the minimum residual energy after the completion of the offloading task. Constraints (14) and (15) have similar function as constraints (5) and (6) in the previous formulation with the difference that the temporal constraints for the completion of the offloading task is now given by parameter t. Constraints (16), (17), (18) and (21) are equivalent to the corresponding constraints in the previous formulation.

V. Numerical Analysis

To evaluate the benefits and the performance of the proposed formulations, we consider a VSN consisting of BeagleBone Linux computers integrated with a 802.11g compliant wireless interface. The VSN consists of one camera node and several cooperator nodes. The camera node is equipped with a camera sensor able to acquire VGA (640x480) grayscale images, with a color depth of 8 bits per pixel. The effective transmission rate c_i of each link is uniformly chosen from the set of available data rates in 802.11g (i.e., 6, 9, 12, 18, 23, 36, 38, 54 Mbps), while the CPUs of all sensor nodes in the VSN are equal (i.e., $v_i = v_0, \forall\, i$). In all our experiments we select for the BRISK algorithm a threshold equal to 90 and an image overlap o equal to 15%. For this choice of the parameters, the camera node alone can process an entire VGA image in $t_0 = 1.6572s$. The complete set of parameters used in our analysis is reported in Table I.

We consider several VSNs instances, characterized by an increasing number of cooperating nodes from 6 to 10. For each instance, we formalize the proposed formulation through AMPL and we solve them using the BONMIN solver on a standard laptop with a 2.6 GHz Intel processor and 4 GB of RAM under Windows. For the settings under consideration, each instance is solved in no more than 5 minutes.

In the presentation of the results, we use the concepts of *speedup* and *lifetime*; the former is obtained by dividing the completion time t^* after offloading by the completion time in case the processing is entirely carried out by the camera node, i.e., t_0; the latter measure the number of consecutive images which can be processed until the first node in the network depletes its original energy budget. As a performance benchmark, we compare the completion time obtained by solving our formulation against the optimal completion time predicted by the DLT, which constitutes an upper bound to the achievable speedup.

Figure 4 shows the results obtained by solving the time-optimal formulation for different values of the parameter

TABLE I
PARAMETERS USED FOR THE ANALYSIS

Name	Symbol	Value
Image Load	I	2.45×10^6 bits
Image overlap	o	15%
CPU speed	v_i	1.48 Mbps
Energy budget	$\bar{E}$	32.4×10^3 J
CPU power	P_i^{cpu}	2.1 W
RADIO power	$P_i^{\text{rx/tx}}$	1.5 W

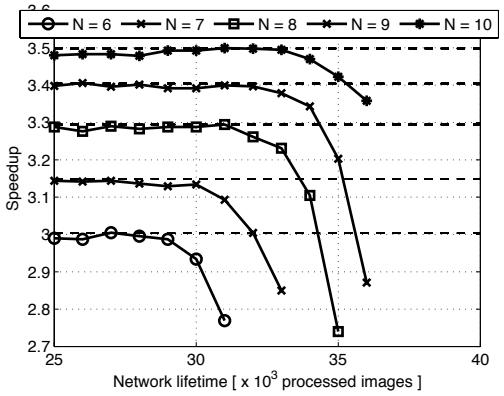

Fig. 4. Tradeoff between lifetime and speedup for the time-optimal formulation

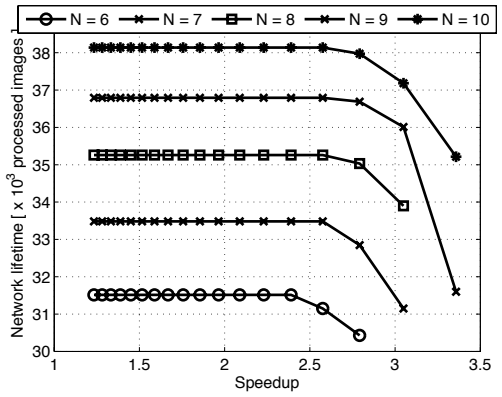

Fig. 5. Tradeoff between speedup and lifetime for the energy-optimal formulation

R, that is, the target lifetime, and for different numbers of available cooperators. For the sake of comparison, the optimal completion time calculated by leveraging DLT (thus disregarding energy constraints) is also reported (horizontal lines). From the inspection of such results we can observe that, considering a fixed number of cooperators, the achievable speedup is characterized by a curve which is constant up to a specific lifetime constraint value, and then suddenly drops. When the lifetime constraint is loose, the time-optimal formulation always returns the optimal completion time, equal to the one computed by the DLT. Conversely, as the the lifetime constraint becomes tighter and tighter, our solution is able to trade speed-up for energy by computing a proper offloading schedule. In general, the achievable speedup increases as the number of available cooperators increases.

Figure 5 illustrates the results obtained for the energy-optimal formulation, when different values of the target speedup (i.e., t^*/t_0) are set as constraints and different numbers of cooperators are used. As clear from the figure, the network lifetime improves for a given target speedup value as the cooperators' set becomes larger. Also in this case, our formulation is able to trade network lifetime for speedup by reducing the lifetime of the network as the constraints on the target speedup become more demanding.

VI. Conclusions

We have addressed here the problem of minimizing the completion time of object recognition tasks in the context of visual sensor networks by leveraging processing offloading techniques. Namely, we have proposed a mathematical programming framework to compute the optimal offloading schedule subject to energy consumption constraints at the sensor nodes. The numerical results obtained by applying the aforementioned framework to representative visual sensor network demonstrate that the proposed offloading techniques are indeed effective in speeding up the visual task further capturing the trade-off between task speedup and consumed energy.

Acknowledgements

The present work has been supported by project GreenEyes (FP7- FET-Open grant number:296676) and by project MESH-WISE (FP7- PEOPLE-2012-IAPP: 324515)

References

[1] A. Redondi, L. Baroffio, A. Canclini, M. Cesana, and M. Tagliasacchi, "A visual sensor network for object recognition: Testbed realization," *IEEE/EURASIP Digital Signal Processing Conference*, pp. n/a–n/a, 2013. [Online]. Available: http://dx.doi.org/10.1002/dac.2378

[2] A. Canclini, M. Cesana, A. Redondi, M. Tagliasacchi, J. Ascenso, and R. Cilla, "Evaluation of low-complexity visual feature detectors and descriptors," in *Digital Signal Processing (DSP), 2013 18th International Conference on*, July 2013, pp. 1–7.

[3] S. Leutenegger, M. Chli, and R. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *ICCV*, 2011, pp. 2548–2555.

[4] S. Bataineh, T. Hsiung, and T. G. Robertazzi, "Closed form solutions for bus and tree networks of processors load sharing a divisible job," in *Parallel Processing, 1993. ICPP 1993. International Conference on*, vol. 1, aug. 1993, pp. 290 –293.

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

[6] L. Anand, D. Ghose, and V. Mani, "Analysis of the drop-out rule in probabilistic load sharing in distributed computing systems," *Int. J. Systems Science*, vol. 28, no. 5, pp. 457–465, 1997. [Online]. Available: http://dx.doi.org/10.1080/00207729708929407

[7] M. Moges and T. Robertazzi, "Wireless sensor networks: scheduling for measurement and data reporting," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 42, no. 1, pp. 327–340, 2006.

[8] X. Li, X. Liu, and H. Kang, "Sensing workload scheduling in sensor networks using divisible load theory," in *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, 2007, pp. 785–789.

[9] A. Canclini, L. Baroffio, M. Cesana, A. Redondi, and M. Tagliasacchi, "Comparison of two paradigms for image analysis in visual sensor networks." in *SenSys*, 2013, p. 62.

[10] M. Khan, G. Dan, and V. Fodor, "Characterization of surf interest point distribution for visual processing in sensor networks," in *Digital Signal Processing (DSP), 2013 18th International Conference on*, July 2013, pp. 1–7.

[11] E. Eriksson, G. Dan, and V. Fodor, "Prediction-based load control and balancing for feature extraction in visual sensor networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, May 2014, pp. 674–678.

[12] ——, "Real-time distributed visual feature extraction from video in sensor networks," in *Distributed Computing in Sensor Systems (DCOSS), 2014 IEEE International Conference on*, May 2014, pp. 152–161.

[13] V. Bharadwaj, T. G. Robertazzi, and D. Ghose, *Scheduling Divisible Loads in Parallel and Distributed Systems*. Los Alamitos, CA, USA: IEEE Computer Society Press, 1996.