A Visual Sensor Network for Object Recognition: Testbed Realization

A. Redondi, L. Baroffio, A. Canclini, M. Cesana, M. Tagliasacchi
Dipartimento di Elettronica e Informazione
Politecnico di Milano
Milan, Italy
{*redondi, baroffio, canclini, cesana, tagliasa*} *@elet.polimi.it*

***Abstract*—This work describes the implementation of an object recognition service on top of energy and resource-constrained hardware. A complete pipeline for object recognition based on the BRISK visual features is implemented on Intel Imote2 sensor devices. The reference implementation is used to assess the performance of the object recognition pipeline in terms of processing time and recognition accuracy.**

***Keywords*-visual sensor networks; object recognition; feature extraction**

I. Introduction

The recent advances in manufacturing of electronic system have led to the production of hardware devices featuring an embedded microprocessor and radio chip on board. The optimization of the electronic design and the definition of suitable networking protocols are key elements to enable low-power data sensing, computation and transmission, i.e., the fundamental requirements of a wireless sensor network. As such, during the last decade, the interest of the scientific community has experienced a steady growth, as witnessed by the numerous publications, standardization activities and events dedicated to WSNs.

As a parallel research line, efforts in the field of visual communication have led to the development of several image and video coding architectures. The main objective is to maximize the quality of the reconstructed pixel-domain representation under constrained resources expressed, e.g., in terms of bandwidth and processing power. Naturally, scientists have started to envision a long-standing marriage between WSNs and visual communication technology, with the promise of empowering sensor nodes with vision. Possible applications for wireless visual sensor networks include: i) video surveillance of public places, traffic, parking lots or remote areas, where no power lines are available or the deployment of traditional sensors is difficult or time consuming; ii) environmental monitoring of remote and inaccessible areas (hazardous areas or animal habitats); iii) smart homes that provide continuous monitoring of persons (quite often elderly people) reporting any unusual behaviour or emergency.

Thus, initial work has started to address the main challenges related to wireless multimedia sensor networks, including the study of ad hoc networking protocols and advanced coding solutions. The main research challenges and consequently efforts in the design of wireless multimedia sensor networks can be summarized as follows [1], [2], [3], [4]

- All the layers of the communication stack need to be re-adapted since most of the existing algorithms and protocols originally developed for traditional wireless sensor networks are not suited to support multimedia communication;
- Cross-layered design approaches are preferable to minimize latency thus guarantee the application-specific Quality of Experience, and further reducing protocol-overhead;
- In the signal processing area, multimedia encoding techniques need to be efficiently applied in order to achieve high coding efficiency and robustness.
- Implementations on commercial hardware are preferable, in order to test effectively the performance of the proposed techniques throughout extensive experiments on such testbeds.

However, these efforts have not yet achieved the expected results, mainly because of the mismatch of computational resources between offer (sensor node capabilities) and demand (coding complexity requirements). As a matter of fact, conventional visual communication systems only achieve significant coding efficiency at the cost of high computational complexity, especially at the encoder (transmitter) side. Thus, wireless sensor networks requirements in terms of node complexity (very low) and transmission bandwidth (as low as possible to have wide coverage and long battery life) are hard to achieve with resource-limited hardware and WMSN requirements cannot be efficiently satisfied by the traditional video compression followed by image analysis paradigm, even in a distributed way as demonstrated in recent works [5], [6].

On the other side, a set of visual analysis applications can

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

978-1-4673-5807-1/13/$31.00 ©2013 IEEE

be effectively carried out based on a succinct representation of the image, which disregards the underlying pixel-level representation, whilst capturing only global and local features. From the network perspective, this allows to reduce the required bandwidth to support the target application since only image features and not the entire image (or video) are delivered to the final destination(s) in order to enable higher level visual analysis tasks.

Research in realizing effective platforms for multimedia support in wireless sensor network is active. In [7], the authors describe the realization of a smart camera platform with multiple processors, variable number of digital signal processing units (DSPs) for multimedia processing. The reference architecture is used to support surveillance applications such as vehicle tracking.

SenseEye, described in [8], is a multi-tier multimedia sensor networks which adopts heterogeneous hardware. The different tiers of the network however, share the same architecture, featuring an integration platform coupled with video sensors including CMUCam, Cyclops and Logitech Web Cams. Object detection applications are developed on the reference platform through background subtraction and leveraging the color distribution of the objects to be recognized.

A similar architecture is proposed in [9], which describes a visual sensor platform to perform person/object recognition. The reference hardware is a Beagleboard connected to a Logitech QuickCam S5500 and to a an RA- Link 802.11b/g WiFi adapter as well as a SunSPOT mote providing 802.15.4 wireless connectivity.

The work in [10] presents a distributed image search system on a network of iMote2 sensor nodes equipped with extended flash storage. The system is based on SIFT [11] local features, which are extremely slow to compute on such hardware.

The authors of [12] propose a multi-processor architecture connected to an Omnivision OV7660 VGA to perform feature extraction tasks. The interested reader is pointed to the survey in [13] for a more comprehensive review on the realization of visual sensor networks testbeds and prototypes. Most of the work in this research track shares the very same functional architecture, which features a vision sensor usually equipped with a dedicated micro-processor or DSP, a motherboard platform and one (or multiple) communication interface(s). What differentiates the different proposal is the "power" of the reference hardware components, which obviously reflects in the energy consumption and the reference realization cost.

Along this line, this work also describes the realization of a visual sensor network supporting object recognition tasks. As in [10], the hardware building blocks which we decide to adopt are, to the best of our knowledge, much less powerful than the reference hardware in the literature. The reference testbed is built on commercial hardware and entails the complete pipeline of object recognition visual tasks including fast feature detection and feature description implemented on sensor nodes, features delivery to a sink through a multi-hop communication protocol, and features matching, implemented at a central controller. The work widely discusses the critical issues in the practical realization on such limited hardware and further provides a performance assessment of the object recognition pipeline in terms of accuracy and processing time.

The manuscript is organized as follows: in Sec. II, we briefly revise the fundamentals of the reference feature extraction algorithms; Section III describes the functional building blocks of the testbed. In Section IV, the testbed is adopted to evaluate the performance of the implemented object recognition task, and finally Section V concludes the paper.

II. Background

The process of object recognition based on local features proceeds as follows. First, the input image is processed in order to detect a number of salient keypoints that correspond to highly distinctive locations of the underlying image. Next, an algorithm transforms the patch around each key point in a feature descriptor. Finally, descriptors extracted from the input image are matched with a set of descriptors extracted from a database of reference images, and a ranked list with the most relevant results is returned.

A large number of detector and descriptor algorithms have been proposed in the literature. Starting from early works in the 80's [14], there have been a great effort in producing detectors which are sensitive to corners, edges, blobs or T-junctions while being invariant to different viewing conditions, i.e., identifying the same set of keypoints under scale and rotation transformations.

A lot of research has been done in order to identify robust descriptors as well, resulting in schemes which are invariant to changes in scale, rotation and illumination. The "gold-standard" descriptor, widely accepted for its performance, is the Scale Invariant Feature Transform (SIFT) [11], which combines a DoG detector with a 128-dimensional descriptor obtained analyzing orientation histograms with 8 bins each from a 4x4 pixel neighborhood around each keypoint.

For both detectors and descriptors, in the last few years a lot of attention has been given to fast algorithms tailored to low-power architectures. This resulted in the development of high-speed corner [15] and blob detection [16]schemes. For what concerns descriptors, binary descriptors such as BRIEF [17] have been recently proposed in order to speedup both computation and matching, while at the same time producing a rather compact representation. Working with binary descriptors is advantageous from different perspectives. First, they are built by concatenating results of binary tests on smoothed pixel intensities, which are fast to compute.

Second, since each descriptor element is a bit (by definition), a binary descriptor size is considerably smaller than traditional real-valued descriptors. Third, matching binary descriptors can be done by means of computation of the Hamming distance, which can be executed in a single XOR operation on modern architectures. For all these reasons, binary descriptors are a natural choice for resource constrained systems such as visual sensor networks.

In this work we focused on the Binary Robust Invariant Scalable Keypoints (BRISK [18]) algorithm, which extends the design of BRIEF by adding invariance to scale and rotation transformations. In the following we provide a brief description of BRISK.

A. BRISK Detector

The BRISK detector is a multi-scale version of the popular FAST (Fast Accelerated Segment Test (FAST) [15]) detector. FAST represented a clear breakthrough in high-speed corner detectors. It classifies a candidate point $\mathbf{p}$ (with intensity $I_{\mathbf{p}}$) as a corner if n contiguous pixels in the Bresenham circle of radius 3 around $\mathbf{p}$ are all brighter than $I_{\mathbf{p}} + t$, or all darker than $I_{\mathbf{p}} - t$, with t a predefined threshold. Each corner is then given a score s, defined as the maximum threshold still classifying $\mathbf{p}$ as a corner. Additionally, the authors presented a machine learning approach to create decision trees that allows to classify a candidate point with only a few pixel tests, thus speeding-up the detection process. It is shown that using this approach requires, on average, less than 2.3 tests per pixel to determine whether or not it is a feature.

B. BRISK Descriptor

The BRISK descriptor uses a pattern of points $\mathbf{p}_i$ for sampling the neighborhood of each keypoint. The pattern defines two sets of sampling point pairs, namely long-distance pairings and short-distance pairings. The long-distance set is composed by all those pairs (i, j) such that $|\mathbf{p}_i - \mathbf{p}_j|_2 > \delta_{min}$ and they are used to estimate the orientation of the keypoint by local gradient averaging. Once the keypoint orientation is estimated, the sampling pattern is rotated accordingly and the short-distance pairings (whose sampling point distance is less than a threshold δ_{max}) are used to build the descriptor. First, for each $\mathbf{p}_i$, an intensity value is obtained by Gaussian smoothing with standard deviation σ_i, proportional to the distance between $\mathbf{p}_i$ and the center of the pattern. To efficiently retrieve pattern points locations and smoothing information, a look-up table of all possible orientations is built offline and stored in memory. As an example, for 1024 pre-computed angles, the look-up table size is as large as 40 MB. To construct the actual descriptor, a binary string is built by concatenating the result of all the short distance intensity comparisons of point pairs $(\mathbf{p}_i, \mathbf{p}_j)$ such that each bit b corresponds to:

$$b = \begin{cases} 1, & I(\mathbf{p}_j, \sigma_j) > I(\mathbf{p}_i, \sigma_i) \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

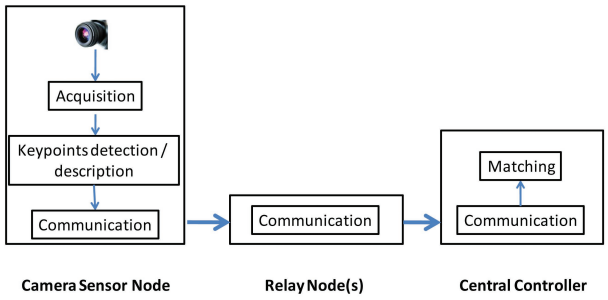

Figure 1. Testbed architecture.

where $I(\mathbf{p}_i, \sigma_i)$ denotes the smoothed intensity value at $\mathbf{p}_i$. In the original implementation of BRISK from the authors, δ_{max} is tuned so that the resulting descriptor has 512 bits.

C. Matching

Matching two BRISK descriptors requires the computation of their Hamming distance, which can be computed very efficiently on today's architectures by means of a bitwise XOR. Two descriptors are said to match if their Hamming distance is smaller than a predefined threshold value θ_H.

III. Prototype Development

As illlustrated in Figure 1, the reference visual sensor network architecture includes a *camera sensor node*, *relaying nodes* and a *central controller*. From the functional point of view, the camera sensor node is responsible for acquiring images, performing key point detection and description and, finally, sending the descriptors to the central controller. The relay nodes only perform communication functionalities, routing the information (stream of descriptors) to the central controller that performs object recognition leveraging the descriptors received from the camera sensor node. In the remainder part of this section the hardware/software characteristics of these three components are described.

A. Camera Sensor Node

The camera sensor, built around the Intel Imote2 platform, is composed of three main building blocks: the low-power PXA271 XScale CPU, the Imote2 Multimedia Board (IMB400), and the TI CC2420 radio transceiver. The CPU can operate in a low voltage (0.85V), low frequency (13MHz) mode, hence enabling very low power operation. The IMB400 board adds multimedia capabilities to the Imote2 platform and allows to capture images and video, through a OmniVision OV7670 camera chip. The CC2420 radio transceiver supports a 250kb/s data rate with 16 channels in the 2.4GHz band. The technical details of the hardware used for realizing the camera sensor node are reported in Table I.

The camera sensor node is operated by Linux, Kernel version 2.6.29. OpenCV libraries are also ported to the

Table I
TESTBED HARDWARE DETAILS

Clock Speed	416 Mhz
Program Flash	32 MB
RAM	256 KB + 32 MB external
Data rate	250 kbps
Image Resolution	640x480

camera node. The camera node also runs the software implementation of BRISK available on the BRISK authors' web site [19].

B. Relay Node

The relay node is responsible for routing the stream of features generated by the camera node to the central controller. In the testbed, the relay nodes are equipped with CC2420 transceivers. The relay node (and the camera node) also implement a simple communication protocol to support multi-hop communication. In details, the camera node generates 802.15.4-compliant packets carrying the descriptors generated by the feature extraction process. The packet size is set to 77 bytes, including 13 bytes for the packet header. The remaining 64 bytes are used to carry exactly one BRISK descriptor. The descriptors are received and forwarded in the same order of generation from the camera node, and then reconstructed at the central controller for matching. The relay nodes also implement a dynamic distance vector routing protocol with the hop count used as routing metrics.

C. Central Controller

The central controller receives the streams of features from the camera and runs the matching task. In the reference testbed, the central controller is an Intel I5 architecture, which implements a conventional matching algorithm. More precisely, visual features received from the visual sensor network are matched against features extracted from each image in a database. Two features are considered to be matching if their Hamming distance is less than a threshold equal to $\theta_H = 90$ bits. After this step, a ranked list of all the image in the database is computed, where images are sorted in descending order of number of matching features. Optionally, a geometric consistency check with RANSAC can be applied, to remove those matches which do not respect the geometric structure between the query image and the images in the database. Given the ranked list of database images, an accuracy measure can be computed. In this work we rely on the Mean of Average Precision (MAP), which is widely used in the field of multimedia information retrieval. For each query q a rank of N documents in the database is obtained. The Average Precision (AP) for the retrieved list of a query q is defined as

$$AP_q = \frac{\sum_{k=1}^{n} P_q(k) r_q(k)}{R_q}, \tag{2}$$

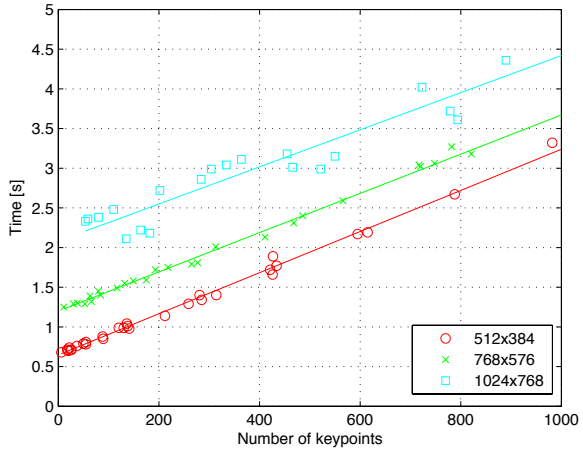

Figure 2. Average time for executing BRISK detector when varying the number of detected keypoints and the resolution of the images.

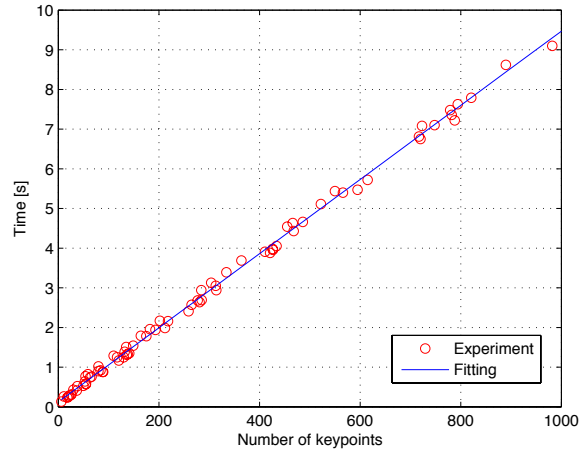

Figure 3. Average time for calculating BRISK descriptors when varying the number of keypoints.

where $P_q(k)$ is the precision (i.e., the fraction of relevant documents retrieved) considering the top-k results in the ranked list; $r_q(k)$ is and indicator function which is equal to 1 if the item at rank k is relevant for the query, and zero otherwise; R_q is the total number of relevant document for the query q and n is the total number of documents in the list. The Mean Average Precision (MAP) for a set of Q queries is the arithmetic mean of the AP across different queries:

$$MAP = \frac{\sum_{q=1}^{Q} AP_q}{Q} \tag{3}$$

IV. PERFORMANCE EVALUATION

In this section we report the results of the performance evaluation of the proposed testbed. We focused our analysis on different performance indicators, including processing

time of both detector and descriptor and accuracy of the matching task with respect to different system parameters (detection thresholds t and memory occupancy) and network conditions.

A. Processing Time Analysis

As a first experiment, we evaluated the processing time needed for keypoint detection and description. For BRISK, the detection time depends on two factors: i) the number of detected keypoints , which can be varied by suitably tuning the threshold t of the detector; and ii) the size of the input image. For the evaluation, we considered L = 10 images of different resolutions (512 x 384, 768 x 756 and 1024 x 768 pixels) randomly selected from the ZuBuD image database [20] and loaded on the considered platform. For each image, we set the BRISK threshold t to the following values [10 30 60 90 120].

Figure 2 and Figure 3 report the average processing time for detecting keypoints and computing descriptors with BRISK for images with variable visual content and different resolutions. From Fig. 2, the detector processing time has an offset which depends on image resolution, and grows linearly with the number of detected keypoints, while the keypoint description time is a linear function of the number of keypoints. Hence, selecting how many points to detect (i.e., by choosing an appropriate value for the detection threshold t) will eventually determine an upper bound on the processing frame rate. As an example, for object recognition, two to three hundred keypoints are enough to saturate performance . This means, in our system, that each image could require 3-5 seconds to be completely processed (i.e. keypoints detection and description). However, for other applications (e.g., object tracking) such low frame rates may not be sufficient to obtain good results.

B. BRISK Memory Issues

As explained in Section II, BRISK leverages a look-up table of discrete rotated and scaled sampling pattern versions to speed up the computation of the descriptor. The original lookup table stores 1024 rotations at 30 scales which calls for a memory space of about 40 MB on the Imote2. Since such space is not available (only 32 MB of memory available), the table sized has been reduced in our tested implementation. We obtained a smaller lookup table by computing only a subset of rotated patterns, while the scale resolution was not changed. This is motivated by the fact that, in most analysis tasks suited for visual sensor networks (e.g. traffic monitoring or face recognition), invariance to scale is much more important than rotation invariance. To evaluate the impact of different lookup table sizes on the feature matching accuracy, the task of feature detection was simulated for different sizes of the lookup table. For each lookup table size, we performed the task of object recognition, based on the 53Obj dataset in [21], which

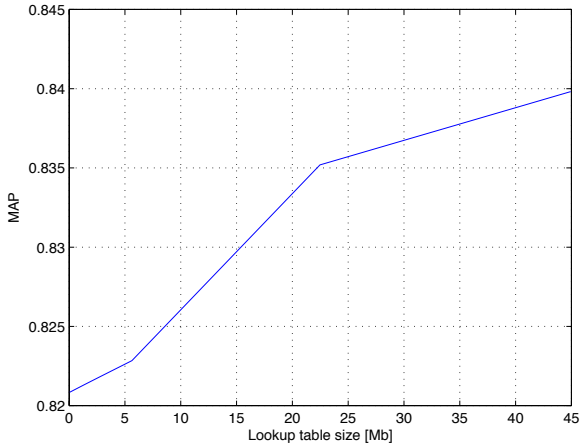

Figure 4. Mean of Average Precision of the object recognition task for the 53Obj dataset under different values of the BRISK lookup table size. For the detection phase we set the threshold t to 60, while for matching descriptors we used $\theta_H = 90$ bits.

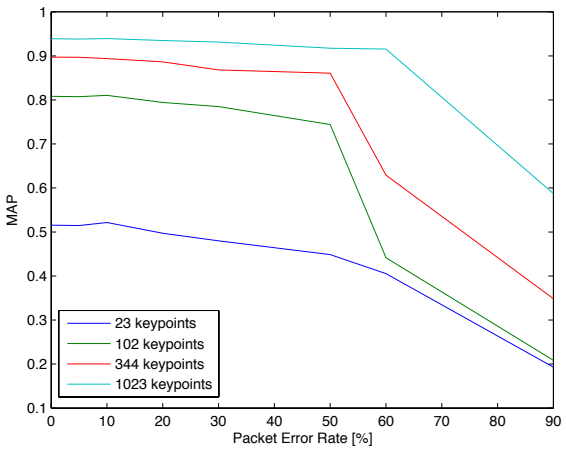

Figure 5. MAP versus packet error rate for BRISK pipeline. The four lines correspond to detection threshold values t of [120 90 60 30], from the bottom up.

contains 213 images of 53 different objects under 4 different viewpoints, together with one query image per object. For each configuration of the lookup table, we computed the Mean of Average Precision (MAP) as accuracy measure. Results are reported in Figure 4 which shows that the MAP is minimally affected by the reduction of the number of rotated patterns.

C. BRISK Robustness to Errors

It is worth analyzing the robustness of the complete object recognition pipeline based on BRISK to transmission and communication errors. To this extent, we artificially inserted packet errors on the wireless channel which cause the loss of BRISK descriptors. The channel errors were emulated by

randomly dropping packets transmitted by the camera node with a given average rate γ. Packets were dropped according to a uniform distribution, and the test was performed 10 times, averaging the results. Figure 5 shows the MAP of the complete BRISK pipeline when varying the packet error rate γ for different values of the number of keypoints per processed image, which can be adjusted by properly tuning the detection threshold t. The BRISK pipeline is extremely robust to channel errors; the MAP remains almost constant for values of γ up to 50% regardless of the number of detected keypoints.

V. Conclusions

This work describes the realization of an object recognition service on resource-constrained camera sensor nodes. The realized testbed was used to assess the performance of the recognition task in terms of processing time and recognition accuracy. For features extraction, 3 to 5 seconds per image were required.

References

[1] I. Akyildiz, T. Melodia, and K. Chowdhury, "Wireless multimedia sensor networks: A survey," *Wireless Communications, IEEE*, vol. 14, no. 6, p. 32, dec. 2007.

[2] S. Misra, M. Reisslein, and G. Xue, "A survey of multimedia streaming in wireless sensor networks," *Communications Surveys and Tutorials, IEEE*, vol. 10, no. 4, p. 18, Fourth Quarter 2008.

[3] S. Soro and W. Heinzelman, "A survey of visual sensor networks," *Adv.Multimedia*, p. 21, 2009.

[4] M. Cesana, A. Redondi, N. Tiglao, A. Grilo, J. Barcelo-Ordinas, M. Alaei, and P. Todorova, "Real-time multimedia monitoring in large-scale wireless multimedia sensor networks: Research challenges," in *Next Generation Internet (NGI), 2012 8th EURO-NGI Conference on*, june 2012, pp. 79 –86.

[5] E. Culurciello, J. H. Park, and A. Savvides, "Address-event video streaming over wireless sensor networks," in *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, may 2007, pp. 849 –852.

[6] S. Paniga, L. Borsani, A. Redondi, M. Tagliasacchi, and M. Cesana, "Experimental evaluation of a video streaming system for wireless multimedia sensor networks," in *Med-Hoc-Net'11*, 2011, pp. 165–170.

[7] M. Bramberger, A. Doblander, A. Maier, B. Rinner, and H. Schwabach, "Distributed embedded smart cameras for surveillance applications," *Computer*, vol. 39, no. 2, pp. 68 – 75, feb. 2006.

[8] P. Kulkarni, D. Ganesan, P. Shenoy, and Q. Lu, "Senseye: A multi-tier camera sensor network," in *Multimedia, 13th ACM International Conference On,*, 2005.

[9] W. Schriebl, T. Winkler, A. Starzacher, and B. Rinner, "A pervasive smart camera network architecture applied for multi-camera object classification," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, 30 2009-sept. 2 2009, pp. 1 –8.

[10] T. Yan, D. Ganesan, and R. Manmatha, "Distributed image search in camera sensor networks," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, ser. SenSys '08. New York, NY, USA: ACM, 2008, pp. 155–168. [Online]. Available: http://doi.acm.org/10.1145/1460412.1460428

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[12] J. Snchez, G. Benet, and J. E. Sim, "Video sensor architecture for surveillance applications," *Sensors*, vol. 12, no. 2, pp. 1509–1528, 2012. [Online]. Available: http://www.mdpi.com/1424-8220/12/2/1509

[13] A. Seema and M. Reisslein, "Towards Efficient Wireless Video Sensor Networks: A Survey of Existing Node Architectures and Proposal for A Flexi-WVSNP Design," *Communications Surveys & Tutorials, IEEE*, vol. 13, no. 3, pp. 462–486, 2011. [Online]. Available: http://dx.doi.org/10.1109/surv.2011.102910.00098

[14] C. Harris and M. Stephens, "A combined corner and edge detector," in *In Proc. of Fourth Alvey Vision Conference*, 1988, pp. 147–151.

[15] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105–119, 2010.

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

[17] M. Calonder, V. Lepetit, M. Özuysal, T. Trzcinski, C. Strecha, and P. Fua, "Brief: Computing a local binary descriptor very fast," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1281–1298, 2012.

[18] S. Leutenegger, M. Chli, and R. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *ICCV*, D. N. Metaxas, L. Quan, A. Sanfeliu, and L. J. V. Gool, Eds. IEEE, 2011, pp. 2548–2555.

[19] [Online]. Available: http://www.asl.ethz.ch/people/lestefan/personal/BRISK

[20] H. Shao, T. Svoboda, and L. Van Gool, "ZuBuD — Zurich buildings database for image based recognition," Computer Vision Laboratory, Swiss Federal Institute of Technology, Tech. Rep. 260, Mar. 2003.

[21] (2003, Mar.). [Online]. Available: http://www.vision.ee.ethz.ch/showroom/zubud/