FAST KEYPOINT DETECTION IN VIDEO SEQUENCES

Luca Baroffio, Matteo Cesana, Alessandro Redondi, Marco Tagliasacchi, Stefano Tubaro

Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano

ABSTRACT

Several computer vision tasks exploit a succinct representation of the visual content in the form of sets of local features. Given an input image, feature extraction algorithms identify keypoints and assign to each of them a descriptor, based on the characteristics of the surrounding visual content. Several tasks might require local features to be extracted from a video sequence, on a frame-by-frame basis. Although temporal downsampling has been proven to be an effective solution for mobile augmented reality and visual search, high temporal resolution is a key requirement for time-critical applications such as object tracking, event recognition, pedestrian detection, surveillance. In recent years, more and more computationally efficient visual feature detectors and descriptors have been proposed. Nonetheless, such approaches are tailored to still images. In this paper we propose a fast keypoint detection algorithm for video sequences, that exploits the temporal coherence of the sequence of keypoints. According to the proposed method, each frame is preprocessed so as to identify the parts of the input frame for which keypoint detection and description need to be performed. Our experiments show that it is possible to achieve a reduction in computational time of up to 40%, without significantly affecting the task accuracy.

Index Terms— Local features, keypoint detection, video.

1. INTRODUCTION

In recent years, ubiquitous computer vision applications are pervading our lives. Smartphones, self-driving terrestrial and aerial vehicles, Visual Sensor Networks (VSNs) are capable of acquiring visual data and performing complex analysis tasks. In particular, VSNs are expected to play a major role in the advent of the *Internet-of-Things* paradigm. Such computer vision tasks usually exploit a concise yet effective representation of the acquired visual content, rather than being based on the pixel-level content. In this context, local features represent an effective solution that is being successfully exploited for a number of tasks such as content-based retrieval, object tracking, image registration, etc. Local feature extraction algorithms usually consist of two distinct components. First, a keypoint detector aims at identifying salient regions (e.g. corners, blobs) within a given image. Second, a descriptor assigns each identified keypoint a descriptor, in the form of a set of values, based on the local characteristics of the image patch surrounding such keypoint. Such information is further processed in order to extract a semantic representation of the acquired content, e.g., by identifying and tracking objects, recognizing faces, monitoring the environment and recognizing events.

As regards visual feature extraction algorithms, SIFT [1] is widely considered as the state-of-the-art for a large number of tasks. It consists in a keypoint detector based on the Difference-of-Gaussians (DoG) algorithm, and in a scale- and rotation-invariant real-valued descriptor, based on local intensity gradients. Besides, SURF [2] is partially inspired by SIFT and aims at achieving a similar level of accuracy at a lower computational cost. More recently, several low-complexity algorithms have been proposed, with the objective of alleviating the computational burden required by both traditional keypoint detectors and descriptors. For example, FAST [3] and AGAST [4] are computationally efficient detectors capable of identifying stable corners. As for descriptors, binary-valued features are emerging as an efficient alternative to traditional real-valued features. BRIEF [5], BRISK [6], FREAK [7] and BAMBOO [8] are instances of such category. For each identified keypoint, they compute a descriptor vector in the form of a sequence of binary values, each of which is obtained by comparing the (smoothed) intensities of a pair of pixels sampled around the keypoint. In some cases, ad-hoc software-based implementations are available for specific hardware architectures [9].

Local feature detection in video sequences has been addressed in the past literature, with the goal of identifying keypoints that are stable across time. For example, Shi and Tomasi [10] propose a widely adopted detector suitable for tracking applications. Zhang et. al propose a complex video-retrieval system based on color, shape and texture features extracted from the key-frames of a video [11]. More recently, Zha et al. propose a method to extract spatio-temporal features from video content [12]. Besides being a key to tasks such as object tracking, event identification and video calibration, temporally stable features improve the efficiency of coding architectures tailored to features extracted from video content [13, 14, 15]. More recently, Girod et al. [16] propose a feature detection and coding algorithm inspired by traditional motion estimation methods. Such algorithm selects a set of features corresponding to canonical image patches whose content is stable across frames, leading to a significant reduction of the transmission bitrate thanks to ad-hoc coding primitives. Although such algorithm represents a good solution for applications that require the efficient transmission of local features for further processing, it might not be the best in terms of computational complexity. Considering low-power devices, computationally intensive operations might significantly reduce the detection frame rate, possibly impairing performance of time-critical tasks or introducing undue delay. In this paper, we introduce a fast detection algorithm based on BRISK [6] and tailored to the context of video sequences, aimed at reducing the computational complexity and thus enabling high frame rates, without significantly affecting performance in terms of accuracy.

The rest of this paper is organized as follows. Section 2 introduces the main concepts behind BRISK. Section 3 illustrates the proposed fast detection architecture. Section 4 defines the experimental setup and presents results. Finally, conclusions are drawn in Section 5.

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.

2. BINARY ROBUST INVARIANT SCALABLE KEYPOINTS (BRISK)

Leutenegger et al. [6] propose the Binary Robust Invariant Scalable Keypoints (BRISK) algorithm as a computationally efficient alternative to traditional local feature detectors and descriptors. The algorithm consists in two main steps: i) a keypoint detector, that identifies salient points in a scale-space and ii) a keypoint descriptor, that assigns each keypoint a rotation- and scale- invariant binary descriptor. Each element of such descriptor is obtained by comparing the intensities of a given pair of pixels sampled within the neighborhood of the keypoint at hand.

The BRISK detector is a scale-invariant version of the lightweight FAST [3] corner detector, based on the Accelerated Segment Test (AST). Such a test classifies a candidate point p (with intensity I_p) as a keypoint if n contiguous pixels in the Bresenham circle of radius 3 around p are all brighter than $I_p + t$, or all darker than $I_p - t$, with t a predefined threshold. Thus, the highest the threshold, the lowest the number of keypoints which are detected and vice-versa.

Scale-invariance is achieved in BRISK by building a scale-space pyramid consisting of a pre-determined number of octaves and intra-octaves, obtained by progressively downsampling the original image. The FAST detector is applied separately to each layer of the scale-space pyramid, in order to identify potential regions of interest having different sizes. Then, non-maxima suppression is applied in a 3x3 scale-space neighborhood, retaining only features corresponding to local maxima. Finally, a three-step interpolation process is applied in order to refine the correct position of the keypoint with sub-pixel and sub-scale precision.

3. FAST VIDEO FEATURE EXTRACTION

Let $\mathcal{I}_n$ denote the n-th frame of a video sequence of size $N_x \times N_y$, which is processed to extract a set of local features $\mathcal{D}_n$. First, a keypoint detector is applied to identify a set of interest points. Then, a descriptor is applied on the (rotated) patches surrounding each keypoint. Hence, each element of $d_{n,i} \in \mathcal{D}_n$ is a visual feature, which consists of two components: i) a 4-dimensional vector $\mathbf{p}_{n,i} = [x, y, \sigma, \theta]^T$, indicating the position (x, y), the scale σ of the detected keypoint, and the orientation angle θ of the image patch; ii) a P-dimensional *binary* vector $\mathbf{d}_{n,i} \in \{0,1\}^P$, which represents the descriptor associated to the keypoint $\mathbf{p}_{n,i}$.

Traditionally, local feature extraction algorithms have been designed to efficiently extract and describe salient points within a single frame. Considering video sequences, a straightforward approach consists in applying a feature extraction algorithm separately to each frame of the video sequence at hand. However, such a method is inefficient from a computational point of view, as the temporal redundancy between contiguous frame is not taken into consideration. The main idea behind our approach is to apply a keypoint detection algorithm only on some regions of each frame. To this end, for each frame $\mathcal{I}_n$, a binary *Detection Mask* $\mathcal{M}_n \in \{0,1\}^{N_x \times N_y}$ having the same size of the input image is computed, exploiting the information extracted from previous frames. Such mask defines the regions of the frame where a keypoint detector has to be applied. That is, considering an image pixel $\mathcal{I}_n(x, y)$, a keypoint detector is applied to such a pixel if the corresponding mask element $\mathcal{M}_n(x, y)$ is equal to 1. Furthermore, we assume that if a region of the n-th frame is not subject to keypoint detection , the keypoints that are present in such an area in the previous frame, i.e. $\mathcal{I}_{n-1}$, are still valid. Hence, such keypoints are propagated to the current set of features. That is,

$$\mathcal{D}_n = \{d_{n,i} : \mathcal{M}_n(\mathbf{p}_{n,i}) = 1 \cup d_{n-1,j} : \mathcal{M}_n(\mathbf{p}_{n-1,j}) = 0\} \quad (1)$$

Note that the algorithm used to compute the *Detection Mask* needs to be computationally efficient, so that the savings achievable by skipping detection in some parts of the frame are not offset by this extra cost. In the following, two efficient algorithms for obtaining a *Detection Mask* are proposed: *Intensity Difference Detection Mask* and *Keypoint Binning Detection Mask*.

3.1. Intensity Difference Detection Mask

The key tenet is to apply the detector only to those regions that change significantly across the frames of the video. In order to identify such regions and build the *Detection Mask*, we exploit the scale-space pyramid built by the BRISK detector, thus incurring in no extra cost. Considering frame $\mathcal{I}_n$ and $\mathcal{O}$ detection octaves, pyramid layers $\mathcal{L}_{n,o}, o = 1, \ldots, \mathcal{O}$ are obtained by progressively smoothing and half-sampling the original image, as explained in Section 2. Then, considering two contiguous frames $\mathcal{I}_{n-1}$ and $\mathcal{I}_n$ and octave o, a subsampled version of the *Detection Mask* is obtained as follows:

$$\mathcal{M}'_{n,o}(k,l) = \begin{cases} 1 & \text{if } |\mathcal{L}_{n,o}(k,l) - \mathcal{L}_{n-1,o}(k,l)| \leq \mathcal{T}_I \\ 0 & \text{if } |\mathcal{L}_{n,o}(k,l) - \mathcal{L}_{n-1,o}(k,l)| > \mathcal{T}_I, \end{cases} \quad (2)$$

where $\mathcal{T}_I$ is an arbitrarily chosen threshold and (k, l) the coordinates of the pixels in the intermediate representation $\mathcal{M}'_{n,o}$. Finally, the intermediate representation $\mathcal{M}'_{n,o}$ resulting from the previous operation needs to be upsampled in order to obtain the final mask $\mathcal{M}_n \in \{0,1\}^{N_x \times N_y}$. Masks can then be applied to detection in different fashions: i) exploiting the mask obtained resorting to each scale-space layer $o = 1, \ldots, \mathcal{O}$ in order to detect keypoint at the corresponding layer o; ii) use a single detection mask for all the scale-space layers.

3.2. Keypoint Binning Detection Mask

Considering two contiguous frames of a video sequence, the amount of features identified in a given area are often correlated [17]. To exploit such information, the detector is applied to a region of the input image only if the number of features extracted in the co-located region in the previous frame is greater than a threshold. Specifically, in order to obtain a *Detection Mask* for the $n-$th frame, a spatial binning process is applied to the features extracted from frame $\mathcal{I}_{n-1}$. To this end, we define a grid consisting of $\mathcal{N}_r \times \mathcal{N}_c$ spatial bins $\mathcal{B}_{i,j}, i = 0, \ldots, \mathcal{N}_r, j = 0, \ldots, \mathcal{N}_c$. Thus, each bin refers to a rectangular area of $S_x \times S_y$ pixels, where $S_x = {}^{N_x}/_{\mathcal{N}_c}$ and $S_y = {}^{N_y}/_{\mathcal{N}_r}$. Then, a two-dimensional spatial histogram of keypoints is created by assigning each feature to the corresponding bin as follows:

$$\mathcal{M}''_n(k,l) = |d_{n-1,i} \in \mathcal{D}_{n-1}| : \lfloor {}^{x_{n-1,i}}/_{S_x} \rfloor = k, \lfloor {}^{y_{n-1,i}}/_{S_y} \rfloor = l, \quad (3)$$

where $(x_{n-1,i}, y_{n-1,i})$ represents the location of feature $d_{n-1,i}$ and $|\cdot|$ the number of elements in a set. Then, a binary subsampled version of the *Detection Mask* is obtained by thresholding such histogram, employing a tunable threshold $\mathcal{T}_H$:

$$\mathcal{M}'_n(k,l) = \begin{cases} 1 & \text{if } \mathcal{M}''_n(k,l) \geq \mathcal{T}_H \\ 0 & \text{if } \mathcal{M}''_n(k,l) < \mathcal{T}_H, \end{cases} \quad (4)$$

Finally, the *Detection Mask* $\mathcal{M}_n$ having size $N_x \times N_y$ pixels is obtained by upsampling the intermediate representation $\mathcal{M}'_n$. Such a detection mask is applied to all scale-space octaves.

4. EXPERIMENTS

Dataset: We evaluated the proposed algorithms with respect to three different test scenarios. First, we exploited the *Stanford MAR dataset* [15], containing the four VGA size, 200 frames long video sequences *Alicia Keys*, *Fogelberg*, *Anne Murray* and *Reba*. Each sequence contains a CD cover recorded with a hand-held mobile phone, under different imaging conditions such as illumination, zoom, perspective, rotation, glare, etc. Furthermore, for each sequence, the dataset contains the ground truth information, in the form of a still image of the corresponding CD cover, having a size of 500×500 pixels.

As a second test, we evaluated the approaches resorting to the *Rome Landmark Dataset*. Such dataset includes a set of 10 query video sequences, each capturing a different landmark in the city of Rome with a camera embedded in a mobile device [18]. The frame rate of such sequences is equal to 24fps, whereas the resolution ranges from 480x360 pixels (4:3) to 640x360 pixels (16:9). The first 50 frames of each video were used as query. On average, each query video corresponds to 9 relevant images representing the same physical object under different conditions and with heterogeneous qualities and resolutions. Then, distractor images randomly sampled from the *MIRFLICKR-1M* dataset [19], so that the final database contains 10k images.

Finally, we tested our method on the *Stanford MAR multiple object* video set [15]. Such a set is made up of 4 video sequences, each consisting of 200 frames at 640x480 resolution. Each video is recorded with a handheld camera and portrays three different objects, one at a time.

Methods: We tested the two detection methods presented in Section 3, that is, *Intensity Difference Detecion Mask* and *Keypoint Binning Detection Mask*. In both cases, we employed the original BRISK implementation from the authors[1], setting the number of octaves to 4 and the detection threshold to 55 and 70 for the *Stanford MAR dataset* and the *Rome landmark dataset*, respectively. As regards *Intensity Difference Detection Mask*, we built the mask testing several different configurations. We tested our algorithm with the 4 layers corresponding to each scale-space octaves. Since the performance was similar when using different layers, we resorted to the top-layer, i.e., the one with the lowest spatial resolution and processing cost. Both *Intensity Difference Detection Mask* and *Keypoint Binning Detection Mask* require a threshold to be set in order to obtain the final detection mask. We tested several different configurations, each representing a tradeoff between computational efficiency and task accuracy.

We compared our algorithms with a *Temporally Coherent Detector* based on non-canonical patch matching [15], which also exploits temporal redundancy in the detected keypoints. Such algorithm aims at propagating stable keypoints across frames, exploiting a pixel-level representation of local features. In details, a traditional keypoint detector is applied to the first frame of a Group Of Pictures of size Δ. Given an identified keypoint, a non-canonical square image patch is extracted from the neighborhood of such a point. Then, considering the following frame, we searched for a matching patch in a window surrounding such a keypoint. Two patches are assumed to be a match if the Sum of Absolute Differences (SAD) between their pixels is below a given threshold $\mathcal{T}_{BM}$. Finally, keypoints for which a match is found are propagated to the next frame, and their position is determined by the aforementioned block matching procedure. In our tests, according to the prescriptions of [15], we employed patches

[1] http://www.asl.ethz.ch/people/lestefan/personal/BRISK

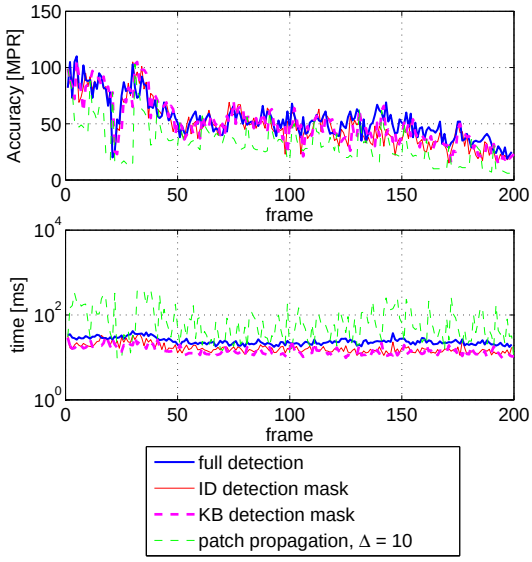

Fig. 1. Accuracy, measured as the number of matches post-RANSAC (MPR), and computational time for each frame of the *Alicia Keys* test sequence.

of 16×16 pixels and we set $\Delta = 10$ and $\mathcal{T}_{BM} = 1800$. Furthermore, to make the procedure faster, we implemented a coarse-to-fine matching algorithm, where the first step consists in a spiral search algorithm with a precision of 2 in a search window of 24×24 pixels, whereas the second step in a spiral search algorithm with quarter-pixel precision in a search window of 1.75×1.75 pixels. Finally, to further speed-up the process, we set an early termination SAD threshold $\mathcal{T}_{ET} = 1000$. This detector was originally proposed with the goal of maximizing coding efficiency, when patches around the detected keypoints need to be compressed and transmitted. To this end, this method can also adopt more sophisticated matching strategies, e.g., based on affine warping. However, in this paper we consider an implementation based on block matching to minimize the computational complexity.

Evaluation methods and measures: In the case of the *Stanford MAR dataset*, for a given video sequence, we extracted a set of features for each frame. Then, the set of features extracted from a frame is matched with the ones extracted from the ground truth frame. A radius match algorithm is used, where the matching threshold is set to $\mathcal{T}_M = 0.18 * 512 \simeq 102$. Finally, geometric coherence of matches is enforced resorting to the RANSAC algorithm. Finally, the number of *Matches-Post-Ransac* (MPR) is employed as the accuracy measure.

In the case of the *Rome Landmark dataset*, the accuracy of the task was evaluated according to the *Mean Average Precision* (MAP). Given an input query sequence q, for each frame $\mathcal{I}_{q,n}$ it is possible to define the *Average Precision* as

$$AP_{q,n} = \frac{\sum_{k=1}^{Z} P_{q,n}(k) r_{q,n}(k)}{R_{q,n}}, \tag{5}$$

where $P_{q,n}(k)$ is the precision (i.e., the fraction of relevant documents retrieved) considering the top-k results in the ranked list of database images; $r_{q,n}(k)$ is an indicator function, which is equal to 1 if the item at rank k is relevant for the query, and zero otherwise; $R_{q,n}$ is the total number of relevant document for frame $\mathcal{I}_{q,n}$ of the

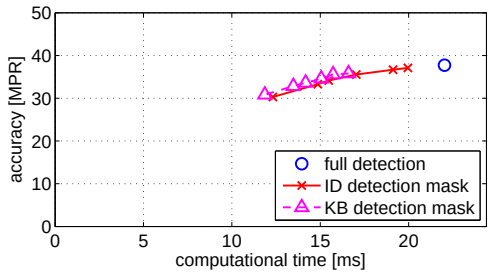

Fig. 2. Energy-Accuracy curves for *Stanford MAR* dataset.

query sequence q and Z is the total number of documents in the list. The overall accuracy for the query sequence q is evaluated according to

$$AP_q = \frac{\sum_{n=1}^{N} AP_{q,n}}{N}, \tag{6}$$

where N is the total number of frames of the query video q. Finally, the *Mean Average Precision* is obtained as

$$MAP = \frac{\sum_{q=1}^{Q} AP_q}{Q}, \tag{7}$$

that is, the mean of the MAP_q measure over all the query sequences.

In the case of the *Stanford MAR multiple object* video set, the accuracy is measured according to a combined detection and tracking precision metric. In particular, for each frame, the goal is to correctly detect the portrayed database object and to identify its position within the frame. Each frame of the video sequences is matched against all the database object. Radius match and geometric verification steps are performed as in the case of *Stanford MAR dataset* scenario. The matching object is the one with the highest number of matches-post-RANSAC. The bounding box for the identified object is obtained by projecting the database object corners according to the homography computed with the RANSAC algorithm at the previous step. Each frame is deemed as correct if the correct object is detected, and if the estimated position is consistent with the ground-truth information. As to the latter, the estimated object position is deemed correct if the displacement between the estimated centroid and the ground truth one is lower than a threshold. We set the value of such a threshold to 10 pixels.

We evaluated the complexity of the feature extraction methods by means of the required CPU time. We performed our tests on a laptop equipped with a 2.5GHz Intel Core i5 processor and 10 GB of RAM.

Results: As an illustrative example, Figure 1 shows the results obtained for the *Alicia Keys* sequence. The charts also report the results obtained when detection is performed independently on a frame-by-frame basis (full detection) to serve as a comparison with the baseline. We observe that the method using the *Intensity Difference Detection Mask* (threshold 20) achieves an accuracy level similar to that of full detection ($MPR = 55$ vs. 56), at a reduced computational time (20.5 ms vs. 24.5 ms). As for *Temporally Coherent Detector*, it leads to a significant loss in terms of accuracy ($MPR = 41$), while being quite computationally intensive (72 ms on average). While accuracy could be further improved by resorting to matching based on affine warping, this would further increase its complexity. This confirms the fact that this detector was originally designed with the goal of maximizing coding efficiency rather than computational cost. Since this is confirmed also on other test sequences, we do not report additional results for this detector.

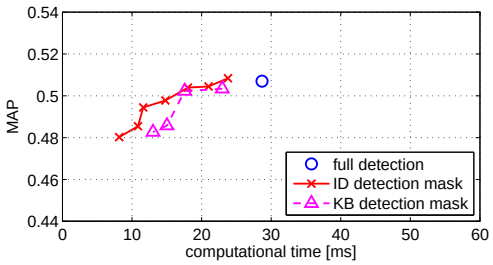

Fig. 3. Energy-Accuracy curve for the *Rome Landmark* dataset, when using the *Intensity Difference Detection Mask* in order to reduce the detection area and with different values for the thresholding parameter. The computational time for each frame can be reduced from 28ms to 18ms, without significantly affecting the accuracy of the task.

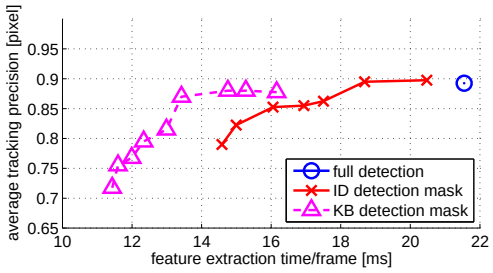

Fig. 4. Energy-Accuracy curves for the *Stanford MAR* multiple object sequences.

It is interesting to observe the energy-accuracy trade-off that can be achieved by varying the threshold used by the algorithms based on detection masks. To this end, Figure 2 compares the performance of *Intensity Difference Detection Mask* and *Keypoint Binning Detection Mask* with that of full detection, averaging the results on the *Stanford MAR dataset*. The two methods based on a detection mask performs on a par, reducing the required computational time by 30% while losing as few as 4 matches.

Furthermore, we tested our approach based on a *Detection Mask* on the *Rome Landmark Dataset*. Figure 3 compares the results of *Intensity Difference Detection Mask* with that of full detection, showing that computational time can be reduced by about 35% without affecting task accuracy. Furthermore, the feature extraction process can be speeded un by 3 times at the cost of 0.03% lower *Mean Average Precision*.

Finally, the results of our fast detection algorithms on the *Stanford MAR multiple object* video set are reported in Figure 4. The computational time can be be reduced up to 40% without significantly impairing object detection and tracking performance.

5. CONCLUSIONS

In this paper we presented a method for fast keypoint detection in video sequences based on *Detection Masks*. Results show that the proposed approach allows for a reduction in terms of computational complexity of up to 35% without significantly impair task performance. In our future investigation we plan to further improve the *Detection Mask* building process, by introducing more sophisticated yet computationally efficient solutions.

6. REFERENCES

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

[2] Herbert Bay, Tinne Tuytelaars, and Luc J. Van Gool, "Surf: Speeded up robust features," in *ECCV (1)*, 2006, pp. 404–417.

[3] Edward Rosten and Tom Drummond, "Fusing points and lines for high performance tracking.," in *IEEE International Conference on Computer Vision*, October 2005, vol. 2, pp. 1508–1511.

[4] Elmar Mair, Gregory D. Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," in *European Conference on Computer Vision (ECCV'10)*, September 2010.

[5] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua, "Brief: Binary robust independent elementary features," in *ECCV (4)*, 2010, pp. 778–792.

[6] Stefan Leutenegger, Margarita Chli, and Roland Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *ICCV*, Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc J. Van Gool, Eds. 2011, pp. 2548–2555, IEEE.

[7] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst, "Freak: Fast retina keypoint," in *CVPR*. 2012, pp. 510–517, IEEE.

[8] Luca Baroffio, Matteo Cesana, Alessandro Redondi, and Marco Tagliasacchi, "Bamboo: A fast descriptor based on asymmetric pairwise boosting," in *IEEE International Conference on Image Processing (ICIP)*, October 2014.

[9] A. Redondi, A. Canclini, M. Cesana, Luca Baroffio, and Marco Tagliasacchi, "Briskola: Brisk optimized for low-power arm architectures," in *IEEE International Conference on Image Processing (ICIP)*, October 2014.

[10] Jianbo Shi and Carlo Tomasi, "Good features to track," 1994, pp. 593–600.

[11] Hong Jiang Zhang, Jianhua Wu, Di Zhong, and Stephen W. Smoliar, "An integrated system for content-based video retrieval and browsing," *Pattern Recognition*, vol. 30, no. 4, pp. 643 – 658, 1997.

[12] Akitsugu Noguchi and Keiji Yanai, "Extracting spatio-temporal local features considering consecutiveness of motions," in *Computer Vision – ACCV 2009*, Hongbin Zha, Rinichiro Taniguchi, and Stephen Maybank, Eds., vol. 5995 of *Lecture Notes in Computer Science*, pp. 458–467. Springer Berlin Heidelberg, 2010.

[13] L. Baroffio, M. Cesana, A. Redondi, M. Tagliasacchi, and S. Tubaro, "Coding visual features extracted from video sequences," *Image Processing, IEEE Transactions on*, vol. 23, no. 5, pp. 2262–2276, May 2014.

[14] L. Baroffio, A. Redondi, M. Cesana, S. Tubaro, and M. Tagliasacchi, "Coding video sequences of visual features," in *20th IEEE International Conference on Image Processing*, Melbourne, Australia, September 2013.

[15] M. Makar, V. Chandrasekhar, S.S. Tsai, D. Chen, and B. Girod, "Interframe coding of feature descriptors for mobile augmented reality," *Image Processing, IEEE Transactions on*, vol. 23, no. 8, pp. 3352–3367, Aug 2014.

[16] Mina Makar, Sam S. Tsai, Vijay Chandrasekhar, David Chen, and Bernd Girod, "Interframe coding of canonical patches for low bit-rate mobile augmented reality.," *Int. J. Semantic Computing*, vol. 7, no. 1, pp. 5–24, 2013.

[17] 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.

[18] L. Baroffio, A. Canclini, M. Cesana, A. Redondi, M. Tagliasacchi, and S. Tubaro, "Rome landmark dataset," http://home.deib.polimi.it/baroffio/romelandmark.

[19] Mark J. Huiskes and Michael S. Lew, "The mir flickr retrieval evaluation," in *MIR '08: Proceedings of the 2008 ACM International Conference on Multimedia Information Retrieval*, New York, NY, USA, 2008, ACM.