SURF Algorithm in FPGA: a Novel Architecture for High Demanding Industrial Applications

N. Battezzati, S. Colazzo, M. Maffione, L. Senepa
Skytechnology
C.so Svizzera 185/bis
10149, Torino, ITALY

Abstract—**Today many industrial applications require object recognition and tracking capabilities. Feature-based algorithms are well-suited for such operations and, among all, Speeded Up Robust Features (SURF) algorithm has been proved to achieve optimal results. However, when high-precision and real time requirements come together, a dedicated hardware is necessary to meet them. In this paper we present a novel architecture for implementing SURF algorithm in FPGA, along with experimental results for different industrial applications.**

I. Introduction

Object recognition and tracking are getting more and more appealing for many embedded applications, from motion-based recognition and automated surveillance, up to vehicle and even spacecraft navigation. To detect an object and track it, interesting features must be identified, such as colors, edges, textures etc. Among all, *interest points* [1], characterized by high local variations of the image intensity and high repeatability, are growing in use because they are characterized by an intrinsic robustness to image transformations, like scale, rotation, illumination and viewpoint changes. They are often called *keypoints*, and their neighborhood, that represents an expressive feature, is summarized by distribution-based *descriptors*, histograms of the surrounding image intensities.

Several algorithms have been developed for detecting keypoints and generating related descriptors. Hessian-based detectors and Scale Invariant Feature Transform (SIFT) descriptors has been proved to outperform other approaches [2]. Bay et al. proposed a combination of Hessian-based detector and a variant of SIFT descriptor, to improve feature detection performance also adding orientation invariance. This solution is called Speeded-Up Robust Feature (SURF) [3].

Many embedded applications that require object recognition and tracking capabilities also have high-precision and real time constraints, that strike against the computation effort needed by the above mentioned techniques. In this scenario, custom hardware approaches are necessary. There are three main possibilities: Digital Signal Processors (DSPs), Field Programmable Gate Arrays (FPGAs) and Graphic Processing Units (GPUs). FPGAs join the parallel computation paradigm to flexibility of custom designed hardware and thus they can achieve very high performance at a relatively low cost.

In this paper we present a novel architecture for implementing SURF algorithm in FPGA, designed for both high-precision and real time requirements. The activity has been partially funded by the STEPS project[1].

II. Architecture description

To devote more attention to industrial applications, in this section we will only outline SURF bottlenecks and describe solutions to overcome them; we refer to [3] for further details.

A. SURF implementation issues

SURF is made of three steps: Fast-Hessian detector, that filtering the original image by 18 scaled box filters calculates 18 different determinants of the Hessian matrix, whose maxima will be the keypoints; SURF descriptor, that calculates the keypoints orientation and generates a 128-dimensions vector to describe the keypoint neighborhood; and finally matching, that looks for correspondences (match-points) between keypoints of two different images by means of the nearest neighbor distance approach. For speeding-up the first two steps, the Integral Image I_Σ [3] is pre-calculated and stored.
To facilitate understanding, we consider an image of 1024x1024 pixels in 8-bit format and every pixel filtered with the 18 box filters (sampling rate SR=1) to improve accuracy. We modified equation (2) in [3], computing an additional absolute value of the determinant, to detect not only *blob* but also *saddle* points, increasing the number of correspondences between two images without adding false matches (*outliers*).

Three main issues limit SURF implementation performance:
Memory requirement: I_Σ requires 32 Mb, under the previous hypothesis, and data of the Hessian matrix other 576 Mb.
Memory access patterns: to filter each point in the original image, 24 access to I_Σ are required, and to evaluate the orientation of each keypoint, 392 access are required. All these memory read operations access data that are neither consecutive nor close to each other, making impossible to fetch bunch of data and thus to perform efficient operations.
Computation effort: the most demanding step in terms of number of computations is matching, that requires $128N^2$ Sum Of Products, where N is the average number of keypoints.

B. High-precision Real time SURF (HR-SURF) architecture

The proposed architecture (HR-SURF), depicted in Fig. 1, relies on three optimizations: an elevated pipeline and

[1]STEPS – Sistemi e Tecnologie per l'EsPlorazione Spaziale is a research project co-financed by Piemonte Region within the P.O.R – F.E.S.R. 2007 – 2013 EC program

parallelism degree, avoid unnecessary data memorization and memory caching for smart access.

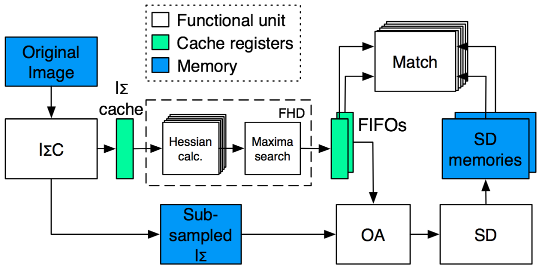

Fig. 1. HR-SURF architecture.

I_Σ Calculation ($I\Sigma C$) block is a full pipelined unit that processes several points in parallel per clock cycle. A *Sub-sampled* I_Σ is used to save memory. Intermediate points that are not saved are then re-calculated starting from the stored samples.

A set of registers works as a cache for the generated I_Σ points, from which the Fast-Hessian Detector (*FHD*) unit fetch data. We exploit the footprint of the memory access to the I_Σ for the filtering calculation. The cache allows the FHD to achieve a throughput of 6 Hessian points per clock cycle (*Hessian calc.*). They are not stored, but pipeline registers directly forward them to *Maxima search* that detects keypoints. These ones are stored in another *FIFO* cache, that decouples this first from the second part of the algorithm.

Orientation Assignment (*OA*) unit calculates the angle of the keypoint, using the sub-sampled I_Σ to filter the image in a circular region around the keypoint. The pipeline inside the unit allows for a throughput of one angle every 500 clock cycles. Being the number of keypoints bounded to 4096 for memory reasons, 732 clock cycles are the average time to compute the orientation and the 128 SURF Descriptors (*SD*) for each keypoint, stored in two *SD memories*.

In order to speed-up the most computationally-expensive part of the algorithm, *Match* unit evaluates the euclidean distance of two keypoints exploiting a high parallelism degree to obtain a throughput of one distance every 2 clock cycles.

III. Applications and Results

We evaluated HR-SURF against other FPGA-based implementations and analysed its results for three different applications: object deformations measurement, vehicle tracking and craters matching for smart spacecraft landing.

We implemented a low-performance prototypical version of the algorithm in an XC6VSX240T Xilinx FPGA, that allowed us to estimate the performance of the HR-SURF. Tab. I details the estimated resource usage and execution time of the HR-SURF for each functional module, as if it was implemented using two XC6VSX240T devices, highlighting that the throughput of the entire system is about 340 ms per frame, including the match stage. Percentages of LUTs and FFs are referred to one single FPGA, while memories and caches require 90% of the whole system's memory.

Tab. II compares estimated results of the HR-SURF against other two FPGA-based implementations, that we refer as Bouris-SURF [4] and Flex-SURF [5]. Note that neither Bouris-SURF nor Flex-SURF implement matching, that is the most expensive stage. Moreover, the first solution adopts images with a size of 640x480 and $SR = d/2$, being d the side of each box filter, while the second approach uses $SR = 2$.

TABLE I
SURF Resource usage and execution time

Unit	LUT [%]	FFs [%]	BRAM [%]	Time [ms]
I_ΣC	6.7	2.3	0.0	0.001
FHD	33.7	15.3	0.0	3.021
OA	9.9	3.1	1.2	0.005
SD	19.1	9.9	1.8	0.002
Match	49.0	12.2	0.0	335.000

TABLE II
Comparison with other approaches: execution times

SURF version	Detector [ms]	Descriptor [ms]	Match [ms]
Bouris-SURF	7.50	10.50	—
Flex-SURF	2937.00	4099.00	—
HR-SURF	3.02	0.01	335

Finally, we used the proposed architecture to test SURF on different high-demanding applications. For each analyzed application, Tab. III reports the average number of *Keypoints* and *Match-points* and the percentage of *Outliers* (gross errors, obtained by visual inspection of the images). The number of match-points is about 20% or more of the total number of keypoints and the outliers percentage is always less than 0.5%. *Object deformation* application presents strong geometric transformations of a plastic sheet, that lead to have a slightly reduced number of match-points with respect to the other applications. *Vehicle tracking* and *Crater matching* applications are characterized respectively by a low image definition and highly jagged patterns, that slightly increase the outliers. Both the high number of match-points and the reduced percentage of outliers are a clear index of how HR-SURF achieves optimal results in industrial applications.

TABLE III
SURF applied to high-demanding embedded applications

Application	Keypoints [#]	Match-points [#]	Outliers [%]
Object deformation	4072	19 %	0.3 %
Vehicle tracking	4092	30 %	0.4 %
Crater matching	3729	23 %	0.4 %

In conclusion, we presented a novel architecture for implementing high-precision and real time SURF in FPGA that outperforms previous solutions, and we proved its effectiveness in high-demanding industrial applications.

References

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, pp. 91–110, 2004.
[2] K. Mikolajczyk *et al.*, "A Comparison of Affine Region Detectors," *Intern. Journal of Computer Vision*, vol. 65, no. 1/2, pp. 43–72, 2005.
[3] H. Bay *et al.*, "SURF: Speeded-Up Robust Features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.
[4] D. Bouris *et al.*, "Fast and Efficient FPGA-Based Feature Detection Employing the SURF Algorithm," in *Intern. Symposium on Field-Programmable Custom Computing Machines*, 2010, pp. 3–10.
[5] M. Schaeferling *et al.*, "Flex-SURF: A Flexible Architecture for FPGA-Based Robust Feature Extraction for Optical Tracking Systems," in *Intern. Conference on Reconfigurable Computing and FPGAs*, 2010.