FPGA-based Real time Extraction of visual features

Merwan BIREM, Franois BERRY
LAboratoire des Sciences et Matriaux pour l'Electronique et l'Automatique
UMR 6602 CNRS - Universite Blaise Pascal
24, Avenue des Landais, 63177 Aubiere Cedex, France
firstname.lastname@univ-bpclermont.fr

***Abstract*— Over the past few years, a new category of sensor is emerging. It is named intelligent sensor or smart sensor. One type of this sensor is smart camera. This paper focuses on smart camera used in the field of robotics and more precisely in the extraction of features. Object tracking, autonomous navigation or 3D mapping are some application examples of feature extraction. We have selected interest points feature that can be detected by several algorithms. One of these algorithms is Harris & Stephen, that has a simple principle based on the calculations of multiples derivatives and gives acceptable results. The smart camera also contains an FPGA as a processing device which will be used for the implementation of Harris & Stephen algorithm. Other modules were also added to the system to deliver more appropriate results.**

***Index Terms*— Image Processing - Point of interest - Harris & Stephen - Field-programmable gate array (FPGA) - VHDL - Hardware Implementation.**

I. INTRODUCTION

The term feature extractor is used to describe the combination of feature detector, and a feature descriptor. Detectors are used to find interesting point in an image, after which a descriptor is created that describes the local neighborhood around the points.

Interest points are low level features. They correspond to a local bi-dimensional important change in the intensity function of the image. The signal at these points contains more information as compared to points from edges or flat region [1]. The literature proposes many detectors to extract features from an image. These detectors differ in the method used to extract the features which implies a difference in the algorithm complexity, processing time and resources needed. An overview of the state of the art in feature extractors can be found in [2].

As an example, these points can be the corners (in L form with an obtuse angle) a junctions (in T or Y form) Fig-1.

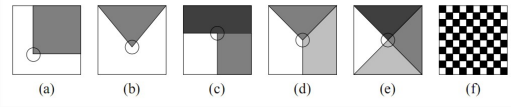

Fig. 1. Different types of interest points: (a) junction in "L", (b) junction in "V", (c) junction in "T", (d) junction in "Y", (e) junction in "X" and (f) junction in "draughtboard"

These features are used in multiples applications like stereoscopic vision, 3D reconstruction and object tracking. and offer many advantages such as :

1) A more reliable source of information than the edges,
2) Robust to occlusions,
3) No need to chain operations,
4) Present in a vast majority of images unlike edges,
5) Stable if the image undergoes a simple transformation (rotation, translation).

In this paper, we focus our work on a "Harris and Stephen"-based detector performing a real time visual features extraction. Our main contribution is proposing several efficient modules in order to select and sort the features in real time. The literature proposes many detectors to extract features from an image. These detectors differ in the method used to extract the feature which implies a difference in the algorithm complexity, processing time and resources needed. Among those detectors, Harris & Stephen [3] is probably the most used method to extract "*interest points*" features. Generally, detection of interest points is done by estimating an **interest value** "R" for each pixel. If the interest value is higher than a predefined threshold "T", the pixel is considered as an interest point. In Harris & Stephen, the interest value for each pixel is calculated using the following formula :

$$R = Det(M) - k * Trace(M)^2 \tag{1}$$

where : $k \in [0.04, 0.06]$

$$Det(M) = AB - C^2 \text{ et } Trace(M) = A + B$$

with :

$$\begin{cases} \bullet\ A = (\frac{\delta I}{\delta x})^2 \otimes w, \\ \bullet\ B = (\frac{\delta I}{\delta y})^2 \otimes w, \\ \bullet\ C = (\frac{\delta I}{\delta x}\frac{\delta I}{\delta y}) \otimes w. \end{cases} \tag{2}$$

where I is a 3×3 window containing the pixel that can be an interest point and w is a gaussian filter 3×3.

As it appears in 2, Harris & Stephen detector is based on calculation of multiple first derivatives along "x" and "y". Most of these operations are SIMD[1](Single Instruction Multiple Data) and therefore highly parallelizable. A major advantage of this detector is that the result is not affected if the image rotates, moves or has a slight change in light intensity. Obviously, this processing is costly in terms of computational load when the image size grows, specially when it is implemented on a CPU. In order to improve the processing time, a classical way is to use an FGPA which is particularly suitable to the SIMD processing. In previous works [4], [5], [6] or [7], the extraction is based on a non-maximum suppression method. In terms of hardware considerations, this method has several disadvantages such as the use of more memories and

[1]Single instruction, multiple data , is a class of parallel processing in Flynn's taxonomy. It describes multiple processing elements that perform the same operation on multiple data simultaneously.

FIFO queues. In addition, this method induces latency due to the buffering of three lines at minimum.

Another important point is the threshold "T" of the interest value. Classically, most of works used a fixed threshold, but this kind of approach does not give efficient results when illumination of the scene changes in large range. In [5], thresholding process was split up by dividing the image into blocks and thresholding each blocks on its own in order to get smoothly distributed features. This solution does not resolve the problem of illumination but it guarantees an homogeneous distribution of interest points.

In this paper, in addition to a real time detection of interest points, we propose 3 modules to extract efficiently the visual features. These modules have been developed in VHDL and are fully compatible with FPGA implementation.

- First module to filter the aggregate of interest points. The result of the detection step provides for each potential feature a set of points. It needs to filter the set in order to select only one points by features (Fig. 2).

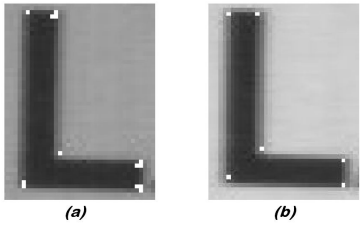

Fig. 2. Exemple of filtering on "L" pattern. (a) Result of the detection step, (b) Filtering. The two images were taken in real time

- The second module proposes to adapt the threshold over the entire image according to the number of interest points needed by the user. This process allows to auto-select a given number of visual features by considering some variations of illumination.
- Last module is a sorting module to sort the interest points in a decreasing order according to their interest value R.

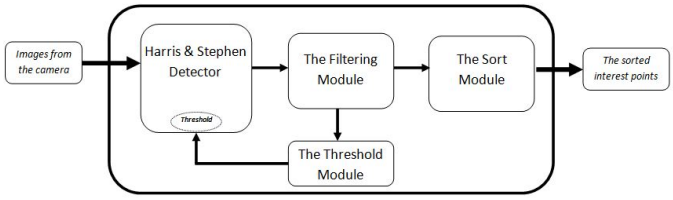

Fig. 3. The Harris & Stephen detector and the three modules

This paper is organized as follows. In the next three sections, each module is explained and an hardware implementation is proposed. Finally, we propose experimental results on one of ours smart cameras (SeeMOS). This smart camera is designed with a CMOS imager and an FPGA STRATIX I from Altera, the FPGA receives the imagery streaming directly from the CMOS imager . More information can be found in [8].

II. Filtering Module

This module uses $n \times m$ window to filter the detected interest points. This module is mainly composed of "m" FIFO queues and one counter (from 1 to "n"). The filtering process is done by keeping only the top left detected interest point when several points are detected at one corner.

Figure 4 shows how the *Filtering Module* works for $n = 2$ and $m = 2$. The green pixels are the detected interest points and the red ones, are the kept points.

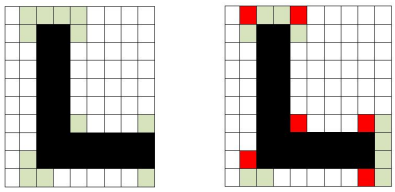

Fig. 4. Filtering the interest points detected in the letter "L"

The Figure 5 represents the block diagram of the contents of the Filtering Module. This module works as follow : The Harris & Stephen detector sends to the Filtering Module "IP_i", which indicates whether the pixel processed is an interest point or not. To validate this detection, the system first checks if there's no other interest point detected in the same row. This is done using the signal "S3" sent by the "counter 1 to n", if $S3 = 1$ it means that no interest point was detected before. Then the system checks if there is an interest point in the rows before the current one using "$FIFO_k$" blocks and "Control IP".

Each "$FIFO_k$" block contains a FIFO queue and a comparator. The i^{th} FIFO queue contains the Y coordinates of the interest points detected in the i^{th} row. The comparator will return "1" if one of these points is closer to the current point processed. The interest point is accepted only if all "$FIFO_k$" blocks return "0", which implies that the "Control IP" returns "1". If so, the Y coordinate of the point processed is added to the FIFO queue "L_i", and the "counter 1 to n" is activated by setting $S3 = 0$. The activation of this counter implies that the next "n" pixels in the current row will not be processed. The "End of the row" is used to shift the FIFO queue "L_i, $FIFO_k$" and to reset L_i and "counter 1 to n".

III. Threshold Module

The use of a fixed threshold in the interest points detector system will have a bad impact on the final results. Because, the variations in illumination implies a variations in the number of the detected interest points. Using an *adaptive threshold* is an efficient way to ensure a proper number of points.

Instead of luminance/contrast information the number of interest point were used to calculate the threshold, because if luminance/contrast information were used other algorithm which can be computationally expensive must be added to the system.

When implementing this module on the FPGA, the question is how to increase or decrease the threshold. To solve this problem, *the inverse cumulative histogram* of the detected interest points is used. The distinctive advantage behind the use of the inverse cumulative histogram is when the number of interest points detected is higher then the one needed, if a traditional technique was chosen like the add of

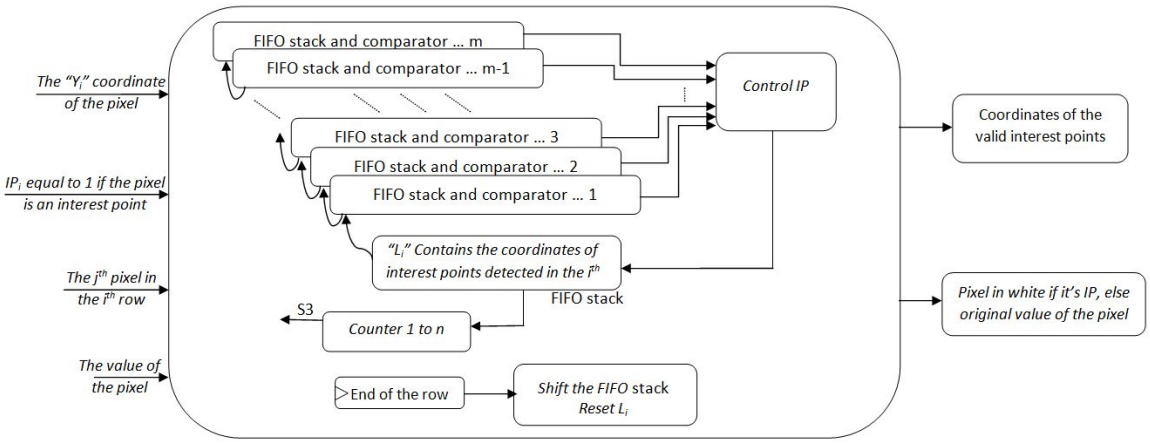

Fig. 5. The Filtring Module

constant T($+1, +10, \ldots$) or the multiplication by a constant T($\times 2, \times 3, \ldots$), the convergence to the desired number may take awhile or can never happen.

The construction of this histogram is done using two memories and several adders (see Figure 6).

If the highest value of the histogram is lower than the requested number of interest points, the threshold is reduced (In the implementation, we choose to reduce the threshold by dividing it by 2). Otherwise, the threshold is increased. The new value of the threshold depends on the old value, and it will be applied on the entire next image.

The Figure 6 represents the block diagram of the contents of the Threshold Module.

IV. Sort Module

The importance and preciseness of point depends on the value of "R", the higher the value of "R", more important and precise is the point.

The used technique to sort the interest points is the one described in the paper [9]. It is done in two steps :

- ***Step 1:*** The ordering process where the order of the detected interest points is decided,
- ***Step 2:*** The rearranging process of the points which puts them in a memory according to their order.

In this method, the sorting of the detected interest points requires the presence of all the points. It explains why the sorting is done only after the image treatment. It means that sorting the points detected in the "i^{th} image" will be done in parallel while processing the "$i + 1^{th}$ image".

The principle of this sorting method is to take an element from the set of detected interest points, then compare its value to the other values, and each time a value higher than the selected one is found, "C_i" is incremented by 1. For each point i, "C_i" represents the number of items in the set of interest points having a value higher than this point, it also represents the order of the point. The rearranging process uses "C_i" as an address to put the points in a memory.

V. FPGA Implementation Results

To implement the Harris & Stephen detector on FPGA, we use our smart camera *SeeMOS* [8]. This implementation

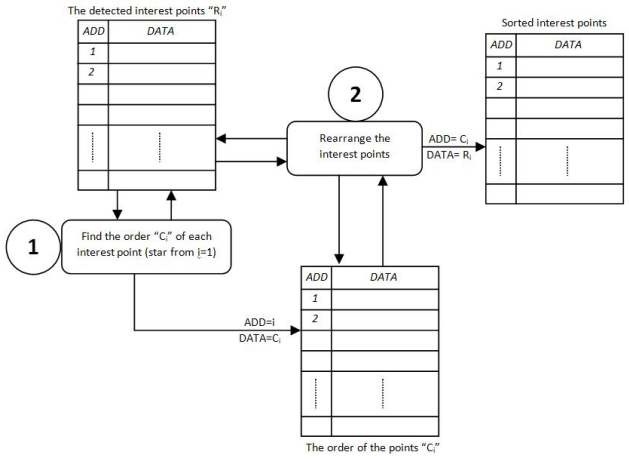

Fig. 7. The Sort Module

consist of three modules :

- Data Acquisition Module : the used imager is a CMOS. In fact, it has more advantages than a CCD imager [10],
- Data Processing Module : which consist of a Stratix FPGA from Altera manufacturer,
- Data Communication Module : between the camera and the host PC that retrieves the data after treatment.

We complete the implementation on the "Stratix-EP1S60F1020C7" of Harris & Stephen detector of interest points and the three modules added to it. The code to describe the Harris & Stephen detector was written using the VHDL Hardware Description Language.

The Figure 8 and Figure 9 show the real-time output results of our implementation working on our smart camera SeeMOS, with and without the *Filtering Module* respectively.

QuartusII SignalTab Logic Analyzer is used to validate the *Sort Module* and the *Threshold Module*. Figure 10 shows the result obtained with this tool for the *Sorting Module*.

The first red line represents the interest values "R" of the points detected and the second represents the order of these points.

Table I shows the percentage of FPGA resources consumption for the implementation of the detection system of interest

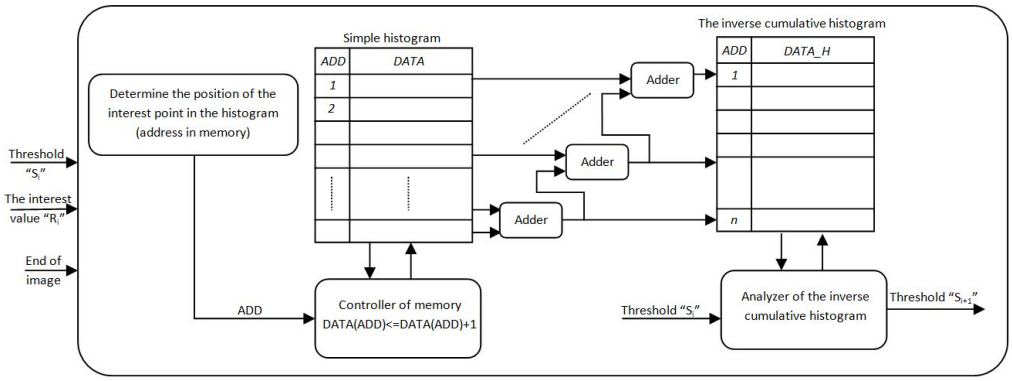

Fig. 6. The Threshold Module

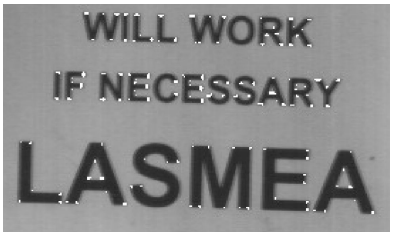

Fig. 8. A simple detector of interest points.(Real time image)

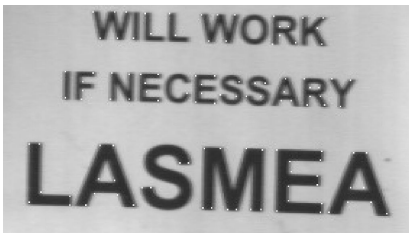

Fig. 9. Detection of Interest Points with the filtering module.(Real time image)

points.

Regarding frames per second capability, this interest points detection system is capable of processing the images at an average speeds of $156fps$ (frame per second), which correspond to a frequency of $41MHz$ when applied to 512×512 $pixel$ images.

VI. Conclusion and Future Work

The hardware implementation of the algorithm detecting the interests points on a dedicated circuit like an FPGA, helped in improving processing time in the order of "ms",even for the 512×512 images. The design of the interest points detection system in a modular fashion, will allow the modification and the improvement of the different modules separately in the future. All the hardware blocks were developed in generic mode, which means the user can change (dimensions of the image, the number of points needed, the lowest threshold allowed, and others parameters) then compile, and synthesize the project to get a new system.

Other modules can also be added to the system, such as the module that returns the interest points and their neighbors in the form of windows centered on the detected interest points.

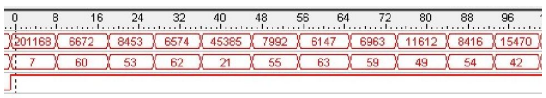

Fig. 10. The result obtained after sorting the interest points detected

Total Logic Element	18'431 / 57'120 (32%)
Total Pins	58 / 782 (7%)
Total Memory Bits	16'208 / 5'215'104 (1%)
DSP Block 9-bits elements	81 / 144 (56%)
Total PLLs	1 / 12 (8%)
Total DLLs	0 / 2 (0%)

TABLE I
FPGA RESOURCES USED

References

[1] J. Louchet, "Introduction à la vision artificielle," 2002.

[2] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, pp. 1615–1630, October 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1083822.1083989

[3] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the 4th Alvey Vision Conference*, 1988, pp. 147–151.

[4] Y. Li, "Fpga implementation for image processing algorithms," *Digital Signal Processing*, no. December, 2006.

[5] D. Benedikt, "Design and implementation of an fpga-based stereo vision system for the eyebot m6," Ph.D. dissertation, University of Western Australia, 2009.

[6] M. Brandon and M. Arin, "Rapid corner detection using fpgas," *National Aeronautics and Space Administration*, dec 2010.

[7] D. Goshorn, J. Cho, R. Kastner, and S. Mirzaei, "Field programmable gate array implementation of parts-based object detection for real time video applications," in *Proceedings of the 2010 International Conference on Field Programmable Logic and Applications*, ser. FPL '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 582–587. [Online]. Available: http://dx.doi.org/10.1109/FPL.2010.114

[8] P. Chalimbaud and F. Berry, "Embedded active vision system based on an fpga architecture," *EURASIP J. Embedded Syst.*, vol. 2007, pp. 26–26, January 2007. [Online]. Available: http://dx.doi.org/10.1155/2007/35010

[9] H. Yasuura, N. Takagi, and S. Yajima, "The parallel enumeration sorting scheme for vlsi," *IEEE Trans. Comput.*, vol. 31, pp. 1192–1201, December 1982. [Online]. Available: http://dx.doi.org/10.1109/TC.1982.1675943

[10] P. Chalimbaud and F. Berry, "Conception d'un capteur de vision active basé sur un imageur cmos," in *RFIA, 14ème congrès francophone (Reconnaissane des Formes et Intelligence Artificielle)*, Toulouse, France, Jan. 2004.