FPGA-based vision perception architecture for robotic missions

Laurent Fiack, Benoit Miramond, Nicolas Cuperlier

FPGA-based vision perception architecture for robotic missions

Laurent Fiack, Benoît Miramond, Nicolas Cuperlier
ETIS Lab - UMR 8051 CNRS / ENSEA / UCP
6 Avenue du Ponceau, 95014 Cergy-Pontoise, France
{firstname.lastname}@ensea.fr

***Abstract*— Most of the robot behaviors are based on visual sensing and perception. This paper describes a smart camera composed of a full-hardware vision architecture coupled with an embedded camera sensor. The hardware architecture corresponds to low-level visual perception processes. The integration of such a system onto the robot enables not only to accelerate the visual processing till real-time behavior but also to compress the data-flow at the output of the camera. The results obtained during indoor robotic missions show an important reduction factor of data communication.**

I. Introduction

Real-time visual processing represents a major challenge for autonomous robots. Most of the robot behaviors are based on visual sensing and perception: navigation, object recognition and manipulation, target tracking, and even social interactions with humans. The visual systems require large computing capabilities which make them hard to embed. Smart cameras are proposed as an alternative to get visual low-level processing back into the robot.

These visual systems are not considered isolated anymore but as part of an architecture integrated in its environment. They take into account several parameters related to the dynamic properties of the systems they belong to (see active vision [1]). These visual processing algorithms strongly depend on the dynamics of interactions between the system and its environment by continuous feedbacks regulating even the low-level visual stages such as the attentional mechanisms in biological systems.

The robotic missions considered in this paper consists in a subset of a complex cognitive system allowing a robot equipped with a charge-coupled device (CCD) camera to navigate and to perceive objects. The global architecture in which the visual system is integrated is biologically inspired and based on the interactions between the processing of the visual flow and the robot movements (Per-Ac architecture [2]). The learning of the sensorimotor associations allows the system to regulate its dynamics [3] and, therefore, navigate, recognize objects, or create a visual compass.

In this paper, we aim at designing an embedded visual processing system in the form of a single chip that could be used for building the CCD-based smart camera of our robot. On one hand, the embedded processing part should be configurable in order to allow a variety of navigation missions. On the other hand, the architecture should also provide intensive computation capabilities to deal with low-level image processing. Finally, the system should reduce the amount of data communication at the output of the camera without loss of information.

Thus, the proposed architecture combines video sensing, perception processing and communication into a System-on-Chip (SoC) embedded onto the robot. This vision system is designed as a full hardware architecture deployed onto a FPGA device. The system is able to process video frames in real-time and communicates intensively with an embedded computer running high-level cognitive tasks that depend on the robot mission. GPU-based solutions provide similar performances but at the cost of important energy consumption that make them unusable in an embedded context.

The rest of the paper is organized as follows. Section II presents the context of this work, especially it describes the different types of robotic missions. For each mission, the environment and the cognitive tasks are different. For that reason the robot needs specific visual features that are provided by the configurable vision SoC. Section III describes the architecture of this system and the parameters that are used to configure the smart camera. Section IV presents the experimental platform and provides the results obtained during robotic missions. Finally, we conclude and outline future works in section V.

II. Context

Among different perceptual modalities, vision is certainly the most important building block of a bio-inspired cognitive robot. Vision enables robots to perceive the external world in order to perform a large range of tasks such as navigation, object tracking and manipulation, and even interaction with humans. This broad range of tasks often relies on different models and architectures. According to a given task, the visual processing often used only a subset of the available algorithms like optical flow, feature points extraction and recognition (over one or several scale spaces). For instance, navigation often relies on low resolution images to only capture the main regularities from the environment, whereas object recognition may need to characterize an object over several scales in order to take into account more details of the object.

The current visual system integrates a multiscale approach to extract the visual primitives. In doing so, it also allows a wider range of applications. Roughly the visual system provides a local characterisation of the keypoints detected on the image flow of an 8-bit gray-scale CCD camera. This local characterisation feeds a neural network which can associate

motor actions with visual information: this neural network can learn, for example, the direction of a displacement of the robot as a function of the scene recognition. The studied visual system can be divided into two main modules:

- a multiscale mechanism for characteristic points extraction (keypoints detection),
- a mechanism supplying a local feature of each keypoint.

A. The multiscale keypoints detection

The multiresolution approach is now well known in the vision community. A wide variety of keypoints detectors based on multiresolution mechanisms can be found in the literature. Amongst them are the Lindeberg interest point detector [4], the Lowe detector [5] based on local maxima of the image filtered by difference of Gaussians (DoGs) or the Mikolajczyk detector [6], where keypoints correspond to those provided by the computation of a 2D Harris function and fit local maxima of the Laplacian over scales. The visual system described here is inspired from cognitive psychology. The used detector extracts points in the neighbourhood of the keypoints, which are sharp corners of the robots visual environment. More precisely, the keypoints correspond to the local maxima of the gradient magnitude image filtered by DoGs. The detector is characterised by a good stability. Following the detector itself, the system provides a list of sorted local features by the way of competition between the corresponding keypoints.

Keypoints are detected in a sampled scale space based on an image pyramid. Pyramids are used in multiresolution methods to avoid expensive computations due to filtering operations. The algorithm used to construct the pyramid is detailed and evaluated in [7]. The pyramid is based on successive image filtering with 2D Gaussian kernels normalised by a factor S.

These operations achieve successive smoothing of the images. Two successive smoothing are carried out by two Gaussian kernels with variance $\sigma^2 = 1$ and $\sigma^2 = 2$. The scale factor doubles (achievement of an octave) and thus the image is decimated by a factor of two without loss of information. The same Gaussian kernels can be reused to continue the pyramid construction. Interestingly, the kernel sizes remain small (half-width and half-height of 3σ) allowing a fast computation of the pyramid. Finally, the images filtered by DoGs in the pyramid can be simply obtained by subtracting two consecutive images. Keypoints detected on the images are the first N local maxima existing in each DoG image of the pyramid. Thus, the keypoint research algorithm orders the N first local maxima according to their intensities and extracts their coordinates. The shape of the neighbourhood for the research of maxima is a circular region with a radius of 21 pixels. This value is one of the system parameters. The number N which parametrises the algorithm corresponds to a maximal number of detections. Indeed, the robot may explore various visual environments (indoor versus outdoor) and particularly more or less cluttered scenes may be captured (e.g., walls with no salience versus complex objects). A detection threshold (γ) is set to avoid nonsalient keypoints. This threshold is based on a minimal value of the local maxima detected. The presence of this threshold is even more important in the lowest resolutions since the information is very coarse at these resolutions. This particularity of the algorithm confers it a dynamical aspect. Precisely, the number of keypoints (and consequently the number of local features) depends on the visual scene and is not known a priori. Furthermore, the threshold γ could be set dynamically through a context recognition feedback but discussing here this mechanism is not our current purpose. However, even if this threshold is considered as a constant value, the number of detected keypoints varies dynamically according to the input visual scene. Consequently, the number of computations (neighbourhood extractions) also depends on the input data.

B. The local image feature extraction

At this stage, the neighbourhood of each keypoint has to be characterised in order to be learnt by the neural network. Existing approaches to locally characterise keypoints are numerous in the literature: local jets, scale invariant feature transform (SIFT) and its variants, steerable filters, and so forth. In the current application, we simply reuse a view-based characterisation where keypoint neighbourhoods are represented in a log-polar space. This representation has good properties in terms of scale and rotation robustness. The local feature of each keypoint is, therefore, a small image resulting from the log-polar transformation of the neighbourhood. The neighbourhoods are extracted from the gradient magnitude image at the scale the keypoint was found by the detector. Each neighbourhood extracted is a ring of radius (5, 16) pixels. Excluding the small interior disc avoids multiple representations of the central pixels in the log-polar coordinates. The angular and logarithmic radius scale of the log-polar mapping are sampled with 16 values. Each feature is thus an image of dimension 16x16 pixels. The sizes of the rings and feature images have been determined experimentally for an indoor object recognition. The given parameters represent a tradeoff between stability and specificity of the features. Finally, the small log-polar images are normalised before their use by the rest of the neural architecture. By associating the data provided by the visual system with actions, the global system allows the robot to behave coherently in its environment [3]. Generally, the visual system must not be considered isolated but integrated in a whole architecture whose modules interact dynamically with each other and through the environment. Hence, the evaluation of the parameters of the visual system depends on the rest of the robot system architecture.

III. HARDWARE ARCHITECTURE

This section describes the hardware implementation of the bio-inspired visual system, which feeds the neural network of the mobile robot. The visual system is implemented into a FPGA device which communicates through Ethernet with an embedded computer running the neural network (Figure 1).

This vision system requires heavy computing power and can't be embedded as software respecting real-time

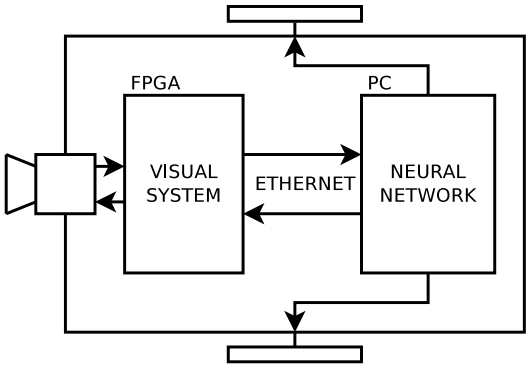

Fig. 1. **Schematic view of the robot**

constraints (the frame-rate of the camera). That's why we realized a hardware implementation through an algorithm/architecture adequacy process [8]. Integrating these first steps of vision perception into the robot itself brings interesting reductions of the data communication. The camera does not act as a primitive sensor anymore but as a smart camera modeling pre-attentional mechanisms that only keeps the most salient visual points. The Gaussian pyramid is presented in figure 2.

A. Architecture of the pyramid

The *gradient* and the *Gaussian filters* are the results of convolutions by a kernel. As often in hardware vision, the pixels of an image come as a flow, and they are processed with the same rhythm. Instead of moving the kernel over the image, the image moves over an architecture which computes the matrix product. A number of pixels must then be stored in first-in first-out (FIFO) memories, depending on the kernel size and the image width. The output pixels are produced with a fixed latency, depending also on these two parameters.

The *DoG* just computes the difference of it's two inputs. Because of the latency of the Gaussian filters, a FIFO is needed for one of its inputs. The *subsampling* decimates the input image by a factor of two, and thus divides the coordinates of the pixels by two.

In the first part of his project, Lefebvre [9] proposed an interface for its processing IPs. With this interface, the input and output buses are composed of the pixels value and their coordinates. A standard interface allows for modularity of these IPs. It is then easy to assemble the processing IPs (*gradient*, *gauss*, *dog* and *subsampling*) to build the hardware *Gaussian pyramid* (Figure 2).

B. Keypoint search and sorting

The next step consists in searching the local maxima of a *DoG* output, to sort them by magnitude, and finally to keep the N most significant points. Each pixel is compared to all the pixels in the neighborhood disc. To follow the pixel rhythm, this comparison must be done in one clock cycle. Once a potential keypoint is found, the exclusion disc avoids two keypoints to be to close.

These points then enter in the *sort* IP. This component consists in a register bank which stores the value, the coordinates, and an index corresponding to the IP that will process the log-polar mapping (Figure 4). These points are stored at a position depending on its value. When a point must be inserted between two registers, the lower part can be shifted down.

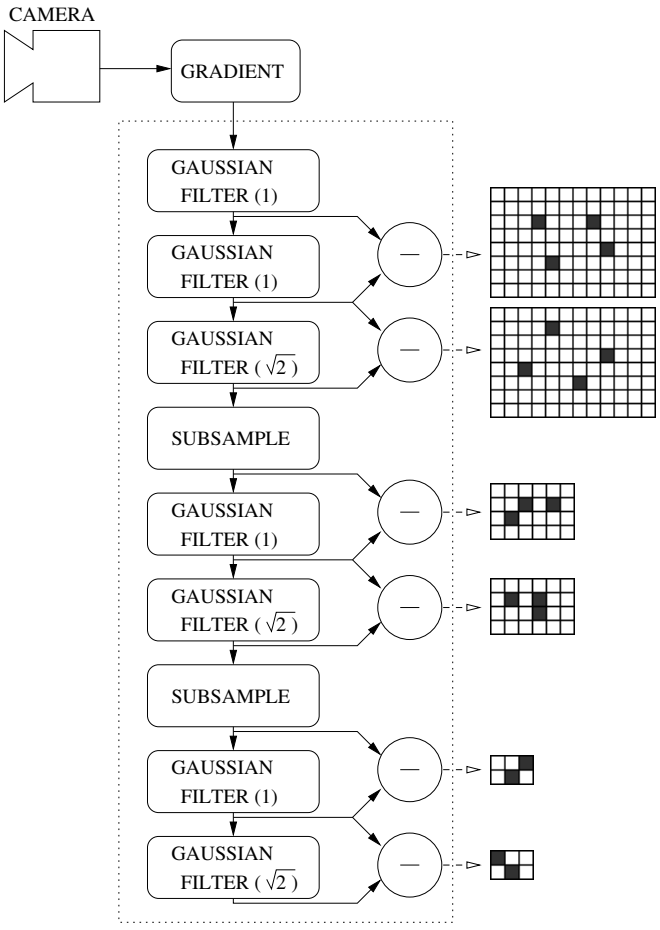

Fig. 2. **Gaussian Pyramid.** Each output corresponds to a frequency band of the input image.

C. Neighborhood extraction and log-polar mapping

Two IPs are responsible for the log-polar mapping (Figure 4), *Address Generator* and *Transform*. The *Address Generator* converts the Cartesian address into the log-polar one. A test ensures that the incoming flow is in the neighborhood disc of the keypoint, and then its coordinates are removed from the coordinates of the flow. These coordinates are the input of a look-up table (LUT) which stores the final log-polar coordinates.

The *Transform* IP reads the incoming pixels, and stores them into a memory at the address computed by the *Address Generator*. This IP is built as a double-buffer memory to compute an image by reading the previous one. The zones of the outer rings containing more than one pixels must be averaged. An accumulator is then added to the memory. As the address conversion is done by a LUT, the values for the average division can also be stored into a LUT.

We need to duplicate these two IPs as many as the number

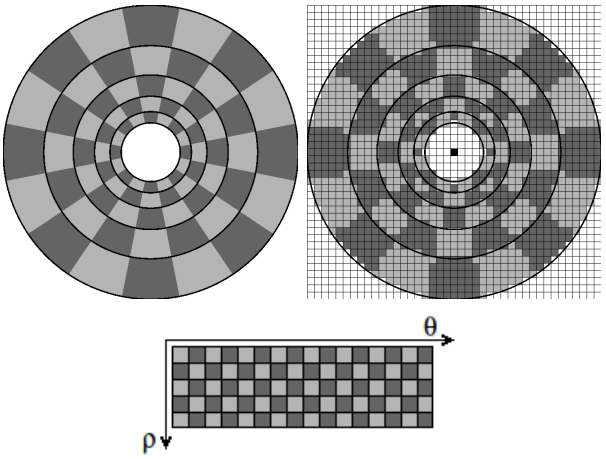

Fig. 3. **Log-polar mapping.** The image on the left represents the log-polar mapping with $\rho = 5$ and $\theta = 16$, the image on the right represents the sampled mapping over the cartesian grid with an extraction radius of 20 and a blind zone radius of 4 pixels, and the third image is the result of the remapping in the (ρ, θ) space.

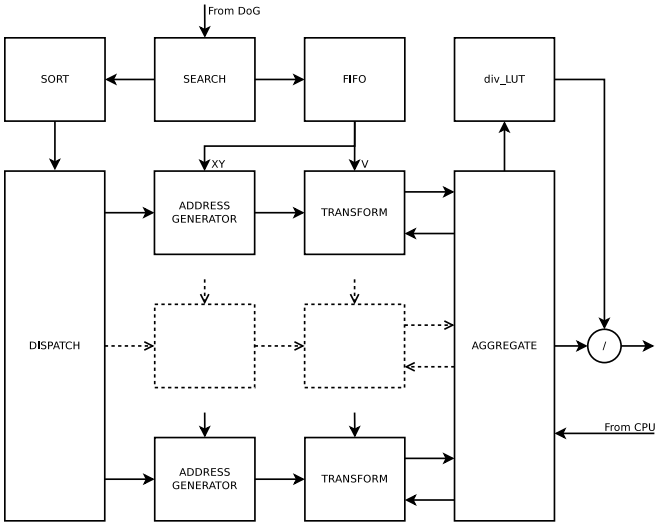

Fig. 4. **Log-polar transform IP.** The N features can be extracted and remapped in parallel.

of keypoints we want to detect (Figure 4). The keypoints are dispatched to an *Address Generator* depending on the index coming from the *sort* IP. As we cannot shift down the content of the entire memory of a *Transform* bloc, we can only overwrite the memory corresponding to the lower keypoint. That's why we need a connection between the rank of the keypoint and its index.

Finally, a CPU can read all the data by selecting an index. A memory (not represented) stores the value of the keypoint, its coordinates, and the relation between its rank and its index, so that the CPU doesn't have to sort the values again. It can read the log-polar feature as a specific RAM memory.

D. Connection with the PC

We choose to use Ethernet and TCP/IP as it offers sufficient throughput, and was already working on the neural network software hosted by the PC. We use a NIOS II soft-processor from Altera to manage the TCP/IP stack. The Ethernet MAC is built has a hardware peripheral. We designed a custom memory-mapped peripheral to read the keypoints values, their coordinates and their associated log-polar features. The global architecture of the smart-camera is depicted in figure 5 .

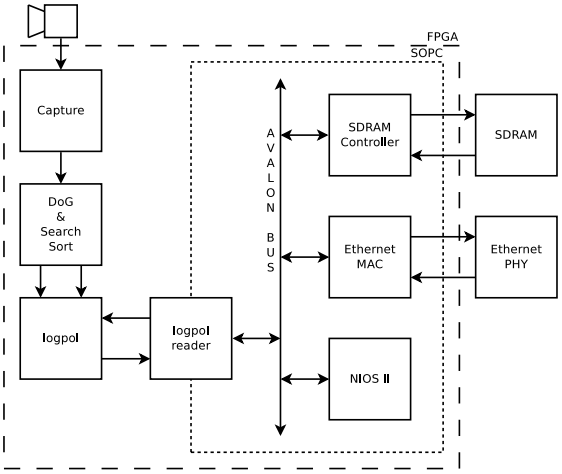

Fig. 5. **SOPC.** The *logpol reader* makes the junction between the visual system and the microprocessor world.

IV. EXPERIMENTS AND RESULTS

A. Vision quality and architecture sizing

The first stage of the *pyramid* have been tested, and works in real-time. The architecture supports the flow of 320×240 images at 22 frames-per-second given by the image sensor.

The synthesis results are given in Figure 6 for the *Altera Cyclone IV EP4CE115* FPGA device of the *Terasic DE2-115* board. The FPGA architecture is composed of 114480 LEs (*Logic Elements*) and 3981312 Memory bits. These results are given for the first stage of the pyramid, for 320×240 images, with a 16-bits bus width. The radius for the research, the exclusion and the extraction is set to 21 pixels, and the radius of the blind zone is set to 5 pixels. The architecture keeps a maximum of $N = 16$ keypoints.

IP	LEs	%	Memory bits	%
Capture	579	< 1	15312	< 1
Gradient	362	< 1	8192	< 1
DoG	4302	4	114688	3
Search	56969	50	344064	9
Sort	1344	1	0	0
Logpol	23110	20	304160	8
SOPC	15465	14	1326953	33
Total	102131	89	2113369	53

Fig. 6. **Synthesis results** given for 320×240, 16-bit pixel bus width, with 21 pixels research radius and $N = 16$ keypoints.

The *Capture* represents all the hardware responsible for reading the images. The *DoG* represents the two Gaussian filters, and the DoG itself. The *Logpol* is the IP shown in Figure 4. The *SOPC* contains the NIOS II, the Ethernet

MAC, the SDRAM controller and the specific peripheral which reads the values from *Logpol*.

The execution of a single stage of the pyramid already uses almost the full FPGA resources. Working with the entire pyramid would need more recent and denser FPGA matrices.

B. Performances and communication

The architecture generates 16 features of 16×16 pixels of 16 bits for each 320×240, 8-bits frame. We can easily compute the compression ratio :

$$r = \frac{320 \times 240 \times 8}{16 \times 16 \times 16 \times 16} = \frac{614400}{65536} = 9.375$$

This architecture allows a much smaller computing time than a classical computer can offer for less power consumption. The visual system itself reduces the amount of data to transmit. The system is currently embedded onto the robot as depicted in figure 7.

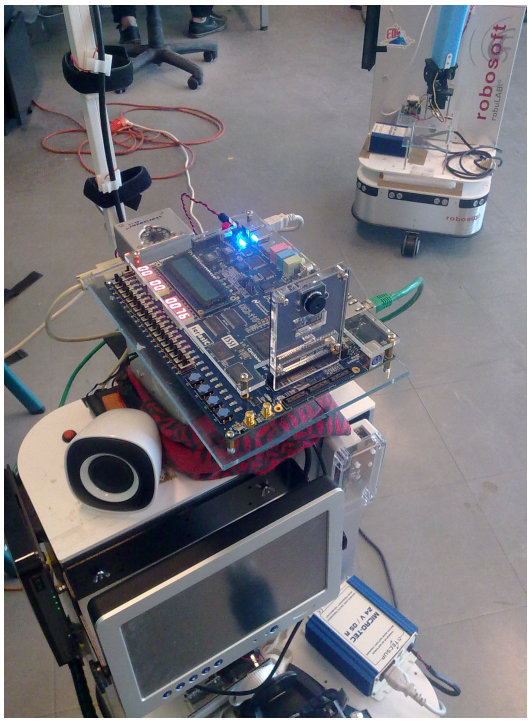

Fig. 7. **The smart camera embedded onto the robot**

The architecture almost fills this FPGA, especially due to the *Search* and the *Logpol* IP, and the *SOPC*. It's already possible to find bigger FPGAs that, in the next generation, will integrate hard wired CPU and Ethernet MAC.

Moreover we are now working at optimizing the system dimensioning, especially the size of the *Logpol* IP. The size of the *Search* can also be reduced since it decreases quadratically with the research radius. But globally there will always be a trade off between the number of scale to compute and the resolution of the features to extract.

V. CONCLUSIONS

Visual processing in the context of autonomous robots is often a challenge both for real-time interactions with the environment and for communication bottlenecks with distant robotic platforms. This paper describes a FPGA-based smart camera executing low-level visual perception tasks directly onto the robot. The system composed of a CCD camera and an FPGA device only outputs a set of keypoints at each frame sampled by the camera. The keypoints detector is implemented as a full-hardware architecture. The system works in real-time at 22 frames per second and enables a compression ratio above 9 compared to a classical camera.

The visual system integrates a multiscale approach to extract the visual primitives. In this paper, we provided results for the first stage of the pyramid (high frequencies). We are now working at integrating the full system into more recent FPGA families with denser logic matrix.

REFERENCES

[1] D. H. Ballard, "Animate vision," *Artificial Intelligence*, vol. 48, no. 1, p. 5786, 1991.

[2] S. Z. P. Gaussier, "Perac: a neural architecture to control artificial animals," *Robotics and Autonomous Systems*, vol. 16, no. 24, p. 291320, 1995.

[3] M. Maillard, O. Gapenne, L. Hafemeister, and P. Gaussier, "Perception as a dynamical sensori-motor attraction basin," in *Proceedings of the 8th European Conference on Advances in Artificial Life (ECAL 05)*, vol. 3630, 2005, p. 3746.

[4] T. Lindeberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision*, vol. 30, no. 2, p. 79116, 1998.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, p. 91110, 2004.

[6] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors," *International Journal of Computer Vision*, vol. 60, no. 1, 2004.

[7] J. L. Crowley and O. Riff, "Fast computation of scale normalised gaussian receptive fields," *Springer Lecture Notes in Computer Science*, vol. 2695, 2003.

[8] F. Verdier, B. Miramond, M. Maillard, E. Huck, and T. Levebvre, "Using high-level rtos models for hw/sw embedded architecture exploration : Case study on mobile robotic vision," *EURASIP Journal on Embedded Systems*, 2008.

[9] T. Lefebvre, "Phd report : Exploration architecturale pour la conception d'un systeme sur puce de vision robotique, adequation algorithme-architecture d'un systeme embarque temps reel," 2012, university of Cergy-Pontoise.