Sensors **2014**, *14*, 6247-6278; doi:10.3390/s140406247

OPEN ACCESS

sensors

ISSN 1424-8220

www.mdpi.com/journal/sensors

Review

A Survey on FPGA-Based Sensor Systems: Towards Intelligent and Reconfigurable Low-Power Sensors for Computer Vision, Control and Signal Processing

Gabriel J. Garc á *, Carlos A. Jara, Jorge Pomares, Aiman Alabdo, Lucas M. Poggi and Fernando Torres

Department of Physics, System Engineering and Signal Theory, University of Alicante, San Vicente del Raspeig, Alicante 03690, Spain; E-Mails: carlos.jara@ua.es (C.A.J.); jpomares@ua.es (J.P.); aa77@alu.ua.es (A.A.); lmp23@alu.ua.es (L.M.P.); Fernando.Torres@ua.es (F.T.)

***** Author to whom correspondence should be addressed; E-Mail: gjgg@ua.es; Tel.: +34-965-903-400 (ext. 1356); Fax: +34-965-909-750.

Received: 30 December 2013; in revised form: 20 March 2014 / Accepted: 21 March 2014 / Published: 31 March 2014

Abstract: The current trend in the evolution of sensor systems seeks ways to provide more accuracy and resolution, while at the same time decreasing the size and power consumption. The use of Field Programmable Gate Arrays (FPGAs) provides specific reprogrammable hardware technology that can be properly exploited to obtain a reconfigurable sensor system. This adaptation capability enables the implementation of complex applications using the partial reconfigurability at a very low-power consumption. For highly demanding tasks FPGAs have been favored due to the high efficiency provided by their architectural flexibility (parallelism, on-chip memory, *etc.*), reconfigurability and superb performance in the development of algorithms. FPGAs have improved the performance of sensor systems and have triggered a clear increase in their use in new fields of application. A new generation of smarter, reconfigurable and lower power consumption sensors is being developed in Spain based on FPGAs. In this paper, a review of these developments is presented, describing as well the FPGA technologies employed by the different research groups and providing an overview of future research within this field.

Keywords: Field Programmable Gate Arrays (FPGAs); rapid prototyping; reconfigurable systems; programmable architectures; low-power sensor systems; smart sensors

---

1. Introduction

Processing capabilities in sensor nodes are typically based on Digital Signal Processors (DSPs) or programmable microcontrollers. However, the use of Field Programmable Gate Arrays (FPGAs) provides specific hardware technology, which can also be reprogrammable thus providing a reconfigurable sensor system. The partial reconfiguration is the process of modifying only sections of the logic that is implemented in an FPGA. Thus, the corresponding circuit can be modified to adapt its functionality to perform different tasks. This adaptation capability allows the implementation of complex applications by using the partial reconfigurability with very low power consumption. This last feature also represents an important aspect when FPGAs are applied in sensor systems. Nowadays, the sensor systems are required to provide an increasing accuracy, resolution, and precision while decreasing the size and consumption. Additionally, FPGAs and their partial reconfigurability allow us to provide sensor systems with additional properties such as processing capabilities, interfaces, testing, configuration, *etc.* These sensors are typically referred to as smart sensors.

The current capabilities of FPGA architectures allow not only implementation of simple combinational and sequential circuits, but also the inclusion of high-level soft processors. The use of integrated processors holds many exceptional advantages for the designer, including customization, obsolescence mitigation, component and cost reduction and hardware acceleration. FPGA embedded processors use FPGA logic elements to build internal memory units, data and control busses, internal and external peripheral and memory controllers. Both Xilinx and Altera (the two market leaders in the FPGA industry) provide FPGA devices that embed physical core processors built inside the FPGA chip. This type of processors are called "hard" processors. Such is the case for the PowerPC™ 405 inside Virtex-4 FPGA devices from Xilinx and the ARM922T™ inside Excalibur FPGA devices from Altera. On the other hand "soft" processors are microprocessors whose architecture is fully built using a hardware description language (HDL). The advantage of using such a type of processor is that a designer can implement the exact number of soft processors required by the application offering a large amount of flexibility for the designer. The most famous soft processors are the LEON3 soft processor from Aeroflex Gaisler, which is a Very High Speed Integrated Circuit and HDL (VHDL) model of a 32-bit processor compliant with the SPARC V8 architecture, the Nios II soft processor which is a 32-bit embedded-processor architecture designed specifically for the Altera FPGAs devices and the MicroBlaze soft processor core from Xilinx. This last processor is a 32-bit RISC Harvard architecture soft processor core with a rich instruction set optimized for embedded applications.

Hardware resources that are implemented in FPGAs differ greatly depending on the manufacturer and the specific FPGAs. However, a great number of devices include components that make them adequate to be applied in sensor systems. This is the case of the previous described processors and the implemented transceivers. A transceiver is a serializer/deserializer (SerDes) capable of operating at serial bit rates up to 28.05 Gigabit/second on current FPGAs such as Stratix V FPGA devices from Altera and Virtex®-7 HT FPGA devices from Xilinx. They are increasingly used for data communications because they can run over longer distances, use fewer wires, and thus have lower costs than parallel interfaces with equivalent data throughput. Most FPGAs could provide configurable I/O standards in order to allow a wide range of devices to be connected and operated at different voltage levels without the need to use adapter interfaces or voltage converters, significantly

simplifying the design and reducing costs. For example, the Spartan-3 FPGA from Xilinx provides various I/O bank standards like LVCMOS, LVTTL, GTL, HSTL, PCI, SSTL, LDT, LVDS, RSDS, and LVPECL that can operate at different voltage levels from 1.2 to 3.3 V.

Some FPGAs incorporate a large amount of arithmetic blocks that can be low-complexity blocks such as simple multipliers or can be relatively more complex like the Digital Signal Processing (DSP) units which consist of combinations of various components like multipliers, adders, accumulators, shift registers, *etc.* A DSP unit significantly accelerates the FPGA's performance and allows achieving greater productivity and flexibility, while decreasing cost and power consumption. For instance, each Stratix II and Stratix II GX device (from Altera) has two to four columns of DSP blocks that efficiently implement multiplication, multiply-accumulate and multiply-add functions. The number of DSP blocks per column and the number of columns available depends on the device, for example, the EP2S180 device from the Stratix II family has 96 DSP Blocks, 769 9×9 Multipliers, 384 18×18 Multipliers and 96 36×36 Multipliers. Furthermore, internal memories offer very high relative speed compared with external memories. Current FPGAs contain large amounts of internal memory blocks, for instance, up to 34 Mb of internal RAM in the Virtex-6 devices from Xilinx. Other memory types can be found such as Random Access Memory (RAM), Read Only Memory (ROM) or shift registers. In addition the designer can implement other memory structures like First In First Out (FIFO).

The cost reduction of the FPGAs, their increasing capabilities and the possibility of improving the performance of sensor systems with specific hardware technologies have led their use in new application fields related with sensors to clearly increase. This paper presents a state of the art overview of the research on sensor systems based on FPGAs in Spain. A great number of applications are integrated in systems that require high data throughputs. Application fields such as image processing and wireless sensors can take advantage of the increasing density of the chips. Nowadays, it is possible to find not only applications in research laboratories, but also in real sensory systems. It is possible to find new fields with growing demand such as thermal management, automotive, robotics, industrial control, medical, reduction of power consumption, *etc.* All these sensor-based applications employ FPGAs with different purposes, as it will be described throughout the paper.

Due to the great amount of data to be processed, computer vision systems currently represent one of the most important fields of research [1–4]. Computer vision systems can be classified into three main levels [5]. The lower level includes the image capture and simple pixel based functions such as arithmetical operations between images. The intermediate level includes operations such as segmentation, matching, preprocessing, convolution or motion estimation. The upper level typically is applied to recognize, classify or capture scene interpretation. Currently, there are a lot of FPGA-based computer vision systems that mainly employ parallel computing in order to increase the processing speed in these computationally intensive applications [6]. High-resolution images and videos require complex compression and coding algorithms [7]. These applications require substantial processing resources and it is possible to find several works that employs the parallel computing features of FPGAs [8–10]. The term Wireless Sensor Networks (WSNs) is currently applied to refer to a set of applications that collect and distribute sensorial data around the sensors [11,12]. Although we can find WSNs with several kinds of sensors, the use of camera as sensor nodes allows the definition of visual sensor nodes that are employed in application such as surveillance. In these applications, the use in the

nodes of FPGA-based image processing allows one to satisfy requirements such as low power consumption, small circuitry scale, and reconfigurability of the hardware architecture.

In this paper, a review of the different FPGA-based sensor systems in Spain is presented. Although FPGA research reached a level of maturity in the 1990s, however until the last decade this technology was not implemented in a wide range of applications [13]. Software tools became powerful and hardware resources have been improved in fields such as communications [14], signal processing [15] and cost reduction, which have also played a fundamental role in the development of FPGA technology. Nowadays, the research of FPGA-based sensor systems is well established in Spain and the different research is very heterogeneous. However, as it will be described in Section 2, this research can be classified into different topics. The term smart sensors [16,17] is typically employed to refer to sensors which integrate the use of an FPGA to perform several functions in a single portable device. Smart sensors are devices that are optimally designed to measure specific physical phenomena that are normally difficult to measure. This system is optimal for high-speed applications where online measurements are needed and the reconfigurability feature is required by a specific application. Compression and cryptography is another important research field of interest [18,19]. Sensor data compression is a technique employed to reduce redundancies in order to decrease data storage and reduce communication costs. Another topic within FPGA-based sensor systems is the use of these devices in order to implement acquisition boards [20]. As previously described, one of the more active research fields in Spain is the field of WSNs. These networks are composed of several FPGA-based networks made up of one or several processing elements and sensors. The nodes send the obtained measurements between them or to a gateway. The different components of the WSNs, their nodes and the use of FPGAs to improve the communications and processing aspects are nowadays a promising research field. Another topic of interest is the use of FPGAs for signal processing. The use of FPGAs allows one to acquire and perform real-time processing of the obtained sensor signals. Currently, the variety of the designed FPGA-based controllers is large [13]. Controllers can be found in applications such as robotics, power electronics and motors. Finally, it the FPGA-based computer vision systems that have been mentioned previously can be cited.

A great part of the current research in Spain on FPGA-based sensor systems can be included in one of the topics indicated in the previous paragraph. These topics will be studied in greater detail in Section 2 but, due to the great number of current approaches, a specific section is created to describe computer vision systems (Section 3). In Section 4, the main properties of the FPGA devices employed in the current research are indicated. Finally, the main conclusions that can be extracted from the presented state of the art are discussed in Section 5.

2. Sensors Systems Based on FPGAs

This section describes the main research about FPGA-based sensor systems in Spain. These works are classified in the topics shown in the next subsections.

2.1. Control Systems

The use of FPGA in industrial control systems is of great interest due to the increasing level of controllers' requirements [13]. The use of FPGAs allows implementing a dedicated parallel

architecture that can be adapted to the plant needs in runtime. FPGAs have already been used with success in different sensor control systems, which requires the implementations of fuzzy logic controllers [21,22], motion controllers [23,24], neural network [25–28], control of asynchronous motors [29], power converter controls [30], mechatronic systems [31], *etc.*

The hardware implementation of a control system can improve the speed performance. However, the FPGA resources are limited and the control systems' algorithms must be refined. This last aspect is an important research topic devoted to optimize the FPGA resources in the implementation of control systems algorithms. For example, in [21] a model-based design method for the synthesis of embedded fuzzy controllers for the joint development of hardware and software components is proposed. Although it is possible to implement FPGA sensor-based controllers with floating point arithmetic [32], the required recourses are not optimized with respect to fixed-point calculations. Coordinate Rotation Digital Computer (CORDIC) is a well-known algorithm used to approximate iteratively some transcendental functions by using adders/subtractors and shifters. This approach has been used by several authors in order to refine and optimize a control system to be implemented in an FPGA [33]. Consequently, when control systems must be developed in an FPGA, a compromise between control performance and complexity of the hardware architecture must be achieved. In the next three subsections, the main FPGA-based controller applications are classified in image-based controllers, advanced control approaches and monitoring systems.

2.1.1. Image-Based Controllers

As previously described, image information can take advantage of the parallel processing capabilities on FPGAs [4]. This information provides global information about the workspace and is progressively integrated in the control systems. In [34], a neuro-inspired mobile robot with a double spike-based control mechanism for two DC motors is proposed. All the image processing issues are also carried out in an FPGA (capture, processing and line tracking). A similar approach is presented in [35] where an address-event representation is employed for visual sensing, processing and finally actuating a robot. In [36], a hardware/software design and implementation for localization of robot in Mars rover missions is presented. This last paper proposes a system architecture implemented on a Xilinx Virtex-6 FPGA to process the obtained images, perform the visual slam, 3D map reconstruction and to obtain the location of the rover at the map. In [37], a high precision automatic system for liquid level measurement in membrane distillation applications is presented. This approach is based on the laser triangulation principle using two lasers and a camera. The level measurement is obtained by an FPGA that performs the image processing. In [38,39] the Simple Network Robot Protocol (SNRP), which permits the integration of network robots and sensors, is defined. In this case, an FPGA has been used to implement a real-time vision system that provides SNRP services to the network. Using the FPGA computer vision module and the SNRP protocol it is possible to implement visual servoing algorithms for industrial robots.

2.1.2. FPGA in Advanced Control

Currently, FPGAs are being applied to implement not only classical control systems, but also different kind of control systems such as predictive control, fuzzy systems or neural networks.

Predictive control is a well-established control strategy that is being used in an increasing set of application areas. The parallel nature of these controllers fits well with the architecture of the FPGAs [40] and different implementations can be found in the literature to operate on any variable (temperature, speed, pressure, *etc.*). In [30] the use of these controllers and their optimal implementation on an FPGA for application in power converters are described. Another kind of control system that is now implemented with success in FPGAs, are the fuzzy controllers [22]. These systems do not require complex modeling of the plant and the control strategy is defined by using linguistic rules that can be implemented using an FPGA architecture. The high computational load of the fuzzy algorithms can be processed using the parallel FPGA architecture to achieve the desired accuracy in real-time. In [21], a model-based approach to implement fuzzy controllers on an FPGA is proposed. Other specific implementations of these controllers can be found in applications such as automotives [41] or education [42]. The increasing capabilities of the FPGAs have opened a new line of research investigating methodologies for scaling intelligent controllers or architectures into embedded systems based on FPGAs [43]. This can be accomplished by the implementation of neural networks [44]. For example, a neuro-inspired mobile robot controller with two DC motors has been proposed in [34]. An important research line is to optimize neuro-inspired models that simulate the neuron layers in the brain. In [3,24,45] a spike-based Proportional-Integral-Derivative (PID) controller is proposed based on FPGAs. The spike-based codification mimics the neuron functioning. The spiking neurons are excited by streams of pulses (spikes), and their output is just another stream of spikes. In the previous references, this information is employed for process visual information and tracking.

2.1.3. Monitoring Systems and Control

The use of FPGA also allows the reduction of delays in the control system feedback. Highly demanding data throughputs can take advantage of the ever-increasing density of the chips in FPGAs [46]. Several applications require not only the capture of sensor information in the feedback but also to process such information in order to obtain the required data to be compared with the system reference. Within this topic, one can mention the work described in [47] where a monitoring infrastructure based on FPGA is proposed. In [37], a computer vision system is presented for liquid level measurement in membrane distillation applications. Another monitoring system is presented in [48–50]. In this case, thermal sensors are employed and they can be used to detect, for example, if a given device dissipates excessive power or does not work correctly.

2.2. Smart Sensors

The demand for small sized, high accuracy and low consumption smart sensors has grown over time. The term smart sensor is frequently employed for sensors that integrate several functions in a single portable device such as communications capability, self-diagnostics, decision-making and some "intelligence". Therefore, the different topics described throughout this paper can be considered as part of a smart sensor: network sensors, control, signal processing, *etc.* These options are commonly integrated in an embedded FPGA-based device when the term smart sensor is employed. The use of FPGAs and their reconfigurability feature allows the addition of different capabilities such as signal conditioning and signal processing [16,51,52]. Furthermore, a smart sensor not only provides the

sensory information but also performs additional functions for error compensation or for obtaining complex data from that measurement (see e.g., [53] where resistance and capacitance information is extracted from the sensor data or [48,49] where FPGAs are employed to include additional features to thermal sensors [54]).

The term smart camera is currently employed for cameras that combine video sensing, processing, and communication on a single embedded platform [17]. The integration of the hardware and software components of a computer vision system in a single portable smart camera is a challenging task. The capacity of the FPGAs to process large image data has allowed the integration of low and mid-level vision algorithms in an embedded smart camera [55]. In this case, the camera does not provide an image but processes data from the image. This approach is optimal for high-speed applications or those that requires the processing of a large amount of data such as tactile information [56].

2.3. Sensor Networks

A sensor network consists of a set of autonomous devices (sensor nodes) connected to a network and distributed in an area susceptible of study. These devices use sensors to monitor physical or environmental conditions, having restrictions on computing power, communication and energy concerns. The term WSNs, already defined in the Introduction section, refers to a sensor network that employs wireless communication. Currently, the number of applications for WSNs has grown hugely in several areas (automation, image processing, security, telemedicine, robotics, domotics, *etc.*) [11]. The main feature demanded for these applications is reduction of the power consumption because the nodes are usually low-cost sensors operating in an environment with limited processing power and restricted battery autonomy. Therefore, low energy WSNs are needed in engineering fields in order to get the longest lifetime possible. For that end, dynamic reconfigurable devices such as FPGAs allow important improvements concerning energy efficiency, because of their efficient use of the communication channels. Moreover, in this case the FPGAs work as distributed reconfigurable devices that permit the implementation of different functionalities everywhere using remote resources. Most of the contributions in the scientific world try to make the most of the FPGAs in order to reduce the transmission of data among the sensor nodes [57,58], to change dynamically the frequency [59,60] and to turn on the radio transceiver selectively [61]. This subsection describes the main approaches developed in Spain concerning the use of FPGAs in sensor networks, where it will be seen that they are related with the purposes above commented.

In [62], a distributed architecture for integrating micro-electromechanical systems was presented. Each micro-electromechanical system is connected to a smart sensor implemented in an FPGA. The FPGA implementation performs the functions of signal conditioner and communication interface, making the designed nodes small in size, flexible, customizable and reconfigurable. The distributed architecture uses the time-triggered master-slave protocol, where both the master and slave nodes have been developed with the same kind of FPGA. A TX/RX unit, a buffer tri-state to access to the bus, a master controller with the time-triggered protocol integrated and a dual-port memory to supply the information related to the system connected to the network, are used within the master FPGA. Similar components are employed in the slave FPGA with an additional hardware divisor in order to obtain the transmission rate.

As stated, FPGAs allow important improvements concerning energy efficiency because of their efficient use of communication channels. A recent idea to improve power consumption in WSN applications is to be able to switch off the main components of a sensor node. Therefore, the hardware device is only activated to accomplish a given task when it is externally demanded. For that purpose, a low power radio that remains always active is used to activate some needed components of the sensor node. This is known as Wake-up Radio (WuR) and it is employed in on-demand WSNs. This idea has been implemented in [63], where FPGAs are used to implement WuRs for WSNs in order to improve the energy efficiency of the task over a traditional micro-controller architecture.

An important contribution in the field of heterogeneous WSNs is the R-GRID platform [64]. This platform consists of a set of distributed reconfigurable resources that can be integrated according to the grid model. The R-GRID approach was developed to facilitate the implementation of multiple-users and multiple-application-instances, dealing with the complexity of distributed heterogeneous FPGAs. For that end, R-GRID uses virtualization techniques to decouple the behavior of the hardware resource from the physical implementation level. In addition, the approach presented in [65] describes the implementation of a fast decision algorithm for the connectivity of mobile sensors with heterogeneous wireless networks using FPGAs. The FPGA device is installed and embedded in the mobile terminals and adjusts a set of weights to improve the Quality of Service (QoS). Furthermore, another multi-purpose sensor network approach using FPGAs is proposed in [47] for delay-based measurements.

The hardware design of the nodes is critical for WSNs in order to be adapted to the application requirements. A modular sensor node can adapt the hardware platform to different scenarios allowing rapid prototyping and low redesign effort. For that end, dynamic reconfigurable devices such as FPGAs play an important role because they can add flexibility to the sensor node. In this context, an important approach developed in Spain is the Cookie platform [4,19,66,67], a modular device that has an innovative WSN node architecture. This modular platform is divided into four functional layers: communication, processing, power supply and sensing/actuating layer. The heart of this platform, the processing layer, includes a microcontroller and an FPGA device, giving more processing power and flexibility to the platform. This layer carries out the processing of all the information given by the sensors. On the one hand, the microcontroller usually deals with the communication control. On the other hand, the FPGA processes the signals coming from sensors. This platform has been tested in processing, power consumption, communication, and encryption with successful results [19,67].

Another relevant issue for WSNs is to compute the location of the mobile nodes connected to the network. For outdoors, location technologies such as GPS or Galileo can be used. However, for indoor environments, technologies based on RFID, image recognition or ultrasonic must be employed. In this context, in [68], a low cost ultrasonic-based location system for mobile nodes is presented using FPGA devices. This location system is employed to obtain the maximum reachable precision. On the one hand, an FPGA device is used to excite the ultrasonic transmitter, and on the other hand, another FPGA device is employed in the mobile node to identify the time difference between the obtained measurements.

2.4. Signal Processing

Embedded signal processing is another topic of interest in the use of FPGAs. Until the appearance of FPGAs in the electronic world, DSPs were the key devices for signal processing. Currently, for

highly demanding tasks, FPGAs have superseded DSPs due to the high efficiency given by their architectural flexibility (parallelism, on-chip memory, *etc.*) [69], reconfigurability [70] and massive performance in the development of algorithms [71]. This subsection provides a brief explanation about the main Spanish approaches in the use of FPGAs for signal processing in sensor systems.

In most cases, FPGAs are used for the implementation of sensor data processing. In this context, in [14], the design of WSNs to get the data of a set of pulse oximeters is presented. In this paper, pulse and oxygen values are processed in the FPGA and the obtained values are sent in real time to the Database Server via a WSN. Another contribution to mention is the presented in [72], where some spike-based band-pass filters have been synthesized for FPGA devices.

Low-level processing of ultrasonic signals is another issue which is being implemented with FPGA devices in order to increase scan rate, precision, and reliability [15,20,73,74]. In this context, using Time-Of-Flight (TOF) measurements given by the transducers, some drawbacks such as cross-talk problems, specular reflection and echo discrimination can arise and generate errors in the distance computation. In order to solve these problems, multimode techniques such as Golay sequences [15] are employed. The implementation of this algorithm in an FPGA device permits adaptation to the distance of the reflector in the environment, simultaneous emissions and simultaneous reception in all transducers being able to discriminate the emitter of the echo.

2.5. Other Sensors

Data compression is a technique that often improves the data bandwidth requirement of any sensor system. FPGAs can be used to implement different kinds of data compression [7,75]. The design of a compression scheme depends, obviously, on the application. In [18,76] a compression scheme over an FPGA is described. The works presented in these papers can be applied to any multi-sensor that must send and receive data simultaneously from different independent sources. Parallelism properties of FPGAs schemes are exploited to implement different units in charge of each of the signals emitted. Sending more than a data source at the same time is the main idea of these papers. This idea was firstly developed by Hernandez *et al.* in [74], where an ultrasonic sensor is improved by sending data simultaneously. Using both, an FPGA and a DSP, the system is able to receive and process this data in real-time. FPGAs can also contribute to decide which kind of image compression fits better with a given image, as in [77], where the authors apply their compression selector in an FPGA embedded on a satellite. An FPGA is a device that really improves the behavior of parallel algorithms. There are also applications exploiting this parallelism like the FPGA-based web servers shown in [78].

Another application where an FPGA may help is in reconfigurable data acquisition systems [79]. Data acquisition systems are used in vast range of tasks. In [80], an FPGA is used as a thermal sensor to measure the behavior of ring oscillators over different voltages. This sensor measures whether a given device dissipates excessive power in relation to the input voltage. The program is implemented through a Microblaze microprocessor over a Xilinx FPGA. Another example of reconfigurable data acquisition system developed over an FPGA is found in [81]. Errors and interference caused by long wires in certain sensors can be easily solved by adding embedded processing units to the sensor. The case of piezoresistive tactile sensors is worked out in this way by the authors. They come up with the implementation of processing units using standard microcontrollers, Programmable Systems on Chip

(PSoCs) and FPGA. The results show that the performance of FPGA solution is closer to that of Application Specific Integrated Circuits (ASICs) rather than that of the other devices.

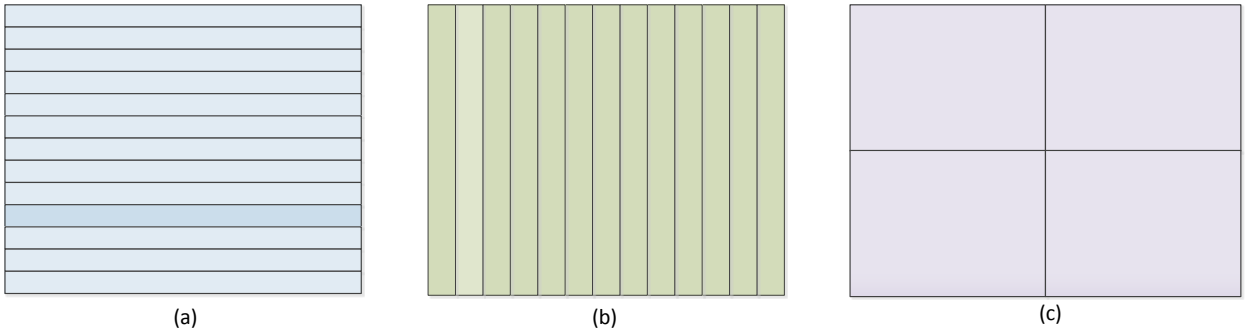
3. Computer Vision Systems Based on FPGAs

Computer vision systems are traditionally based on a sequential architecture. Thus, image processes are run one after another in succession. The program is divided into a sequence of arithmetic and logic operations that are performed by the Arithmetic Logic Unit (ALU). The rest of the CPU is designed to supply the ALU the required data. The algorithm is compiled into a sequence of instructions, which are used to control the operations performed by the CPU and ALU per clock cycle. Therefore, the basic operation performed by the CPU is to seek an instruction in the memory, decode it to determine which operation should perform and execute it.

In contrast, a parallel approach implements any instruction from any algorithm on a separate processor. Nevertheless, if the algorithm were predominantly sequential, with each of its instructions depending on the data from the previous instruction, the gain that could be obtained would be practically zero. In order to get a useful parallel implementation, the algorithm must be susceptible to be split into independent parts and to achieve any significant gain, the portion of the algorithm that can be implemented in parallel must be significant. Luckily, the image processing is inherently parallel, especially in the mid-level and low-level tasks. The complexity in algorithm translation from sequential to parallel opens a very interesting research topic that is reviewed in this section. Works like [82–84] describe some technique to minimize the time required to perform a computer vision algorithm on an FPGA.

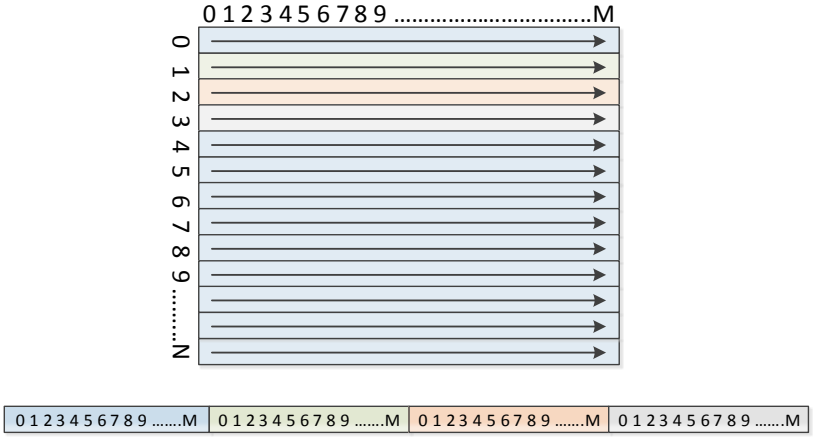
Two different parallelism concepts can be applied to computer vision algorithms: temporary parallelism and spatial parallelism. Image processing algorithms generally implement a sequence of image operations. By assigning each procedure to a different processor, image processing can achieve a temporary fashion of parallelism. This would be a pipelined architecture, where each processor carries out an operation on the data and sends its results to the next entity. Image processing algorithms often contain one or several loops. These loops iterate over all pixels of the image, applying the same operation, regardless of its value. This kind of parallelism is called spatial parallelism. In order to take advantage of spatial parallelism, the image must be partitioned in some fashion (see Figure 1). After this, a different processor can process each part of the image.

Figure 1. (**a**) Row partitioning; (**b**) Column partitioning; (**c**) Block partitioning.



The necessary time (and bandwidth) to read the image from the memory and to store it after being processed is one of the most common bottlenecks in image processing. Converting the spatial parallelism into a temporal parallelism can help minimize this problem. To do so, the image is streamed, *i.e.*, the image is read and written sequentially using a browser frame, usually at a speed of one pixel per clock cycle (see Figure 2). From this image stream, the time spent on the image processing is obtained from the amount of time required to read/write the image and the processing latency. In most operations, latency usually takes much less time than the loading of the whole image itself. So if the algorithm of image processing can be implemented in a flow, the response time will be dominated by the frequency at which the images are provided.

Figure 2. Image stream.



In order to describe the task that a computer vision system must develop, three abstraction levels have been widely assumed in the literature: low-level, mid-level and high-level tasks [5]. Low-level vision tasks consist of pixel-based operations such as filtering, intensity estimation, segmentation and edge detection. In a low-level task, the computer vision system usually deals with a large amount of data. These tasks consist of small neighborhood operations such as segmentation, filtering or basic point operations. However, low-level tasks require, generally, relatively simple operations (such as multiply and add). Mid-level vision consists of pixel grouping operations such as determining object features or region labeling. These tasks are again characterized by local data access, but more complex pixel operations. Finally, high-level vision tasks are more decision-oriented, such as object recognition, face recognition or scene recognition. These tasks involve non-local data access and non-deterministic and complex algorithms. The same task can often be referred as any of the three levels in the literature. However, in this paper the different works about vision and FPGA have been divided into these three categories following the next rules: if the primary purpose task is image enhancement, the task is categorized as low-level; the tasks that operate on the pixels to produce features in the image are mid-level tasks; and finally, decision-making stage is classified as a high-level task.

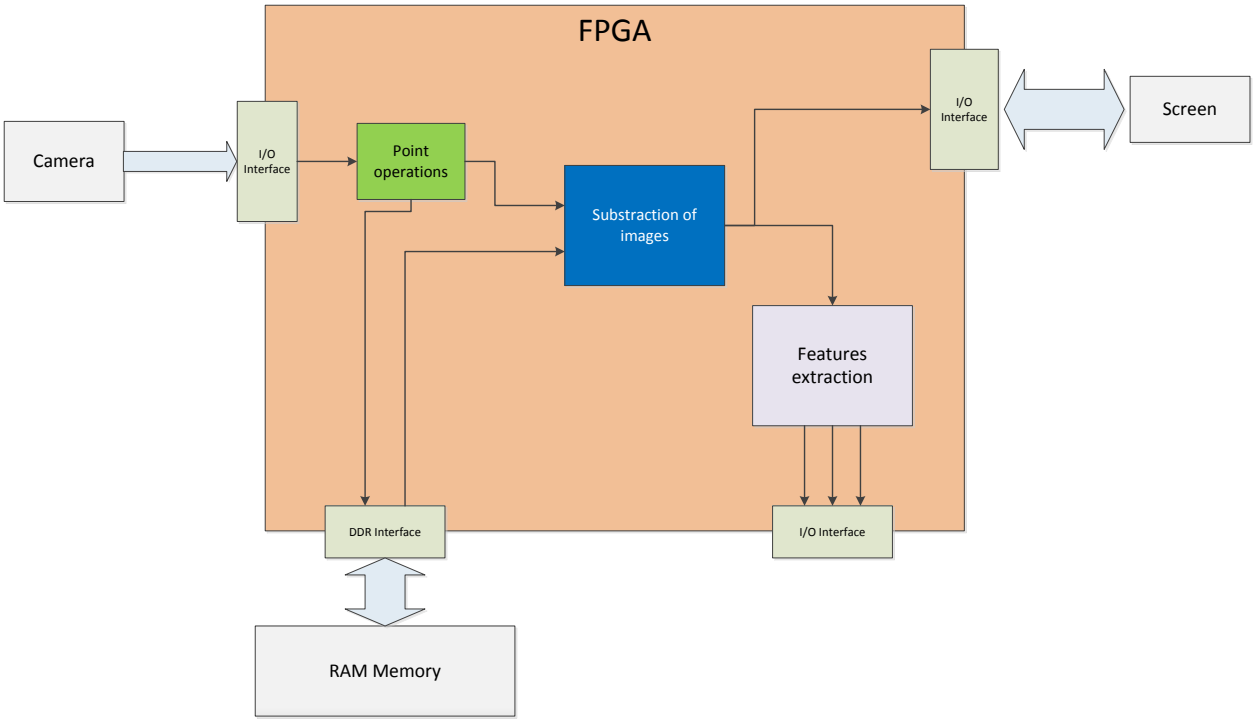
3.1. Low-Level Vision Tasks

FPGAs are ideal for image processing, particularly for low-level and mid-level tasks where parallelism is exploited [9]. Most of the works found in the literature related to computer vision and FPGAs describe a parallelism version of a classical sequential computer vision algorithm [9,10]. For a pipelined architecture, a different hardware block is built for each image processing operation. The block implementing the image processing operation passes its processed data to the next block, which performs a different operation. When the system is not synchronous, intermediate buffers between operations are required. These buffers handle the variations in the data flow. As stated before, building multiple copies of implemented operations and assigning different partitions of the image to each copy can exploit spatial parallelism. A full spatial parallelism can be achieved by building a processor for each pixel. In practice, high image resolution of modern cameras makes this unlikely.

Logical parallelism is the overall parallelism contained in a program, *i.e.*, all the computations that may, according to the semantics of the programming language, be executed in parallel. The logical parallelism within an image processing operation fits into an implementation on the FPGA. This is where most of the image processing algorithms can significantly improve performance. To do so, inner loops are unrolled. Thus, operations are performed in parallel hardware instead of sequentially.

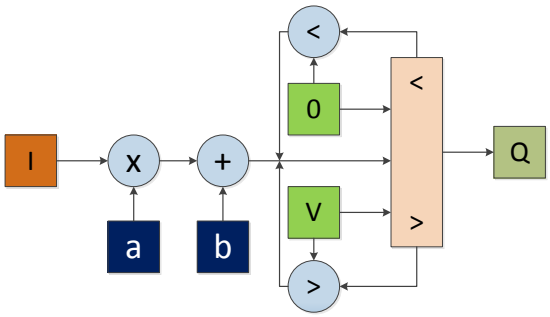
Figure 3 depicts a scheme of a low-level to mid-level vision task implemented over an FPGA. Parallel skills have effects on the construction of the vision system [9]. Implementing a pipelined architecture in an FPGA permits operating at the same frequency pixels are served. Given that power consumption is directly related to the clock frequency, a lower frequency implies a lower power demand by the system. The vision task described in Figure 3 is a typical FPGA approximation to an image processing task.

Figure 3. Scheme of a computer vision system embedded on an FPGA. Four points are tracked, and their center of gravity computed at camera frame rate frequency.



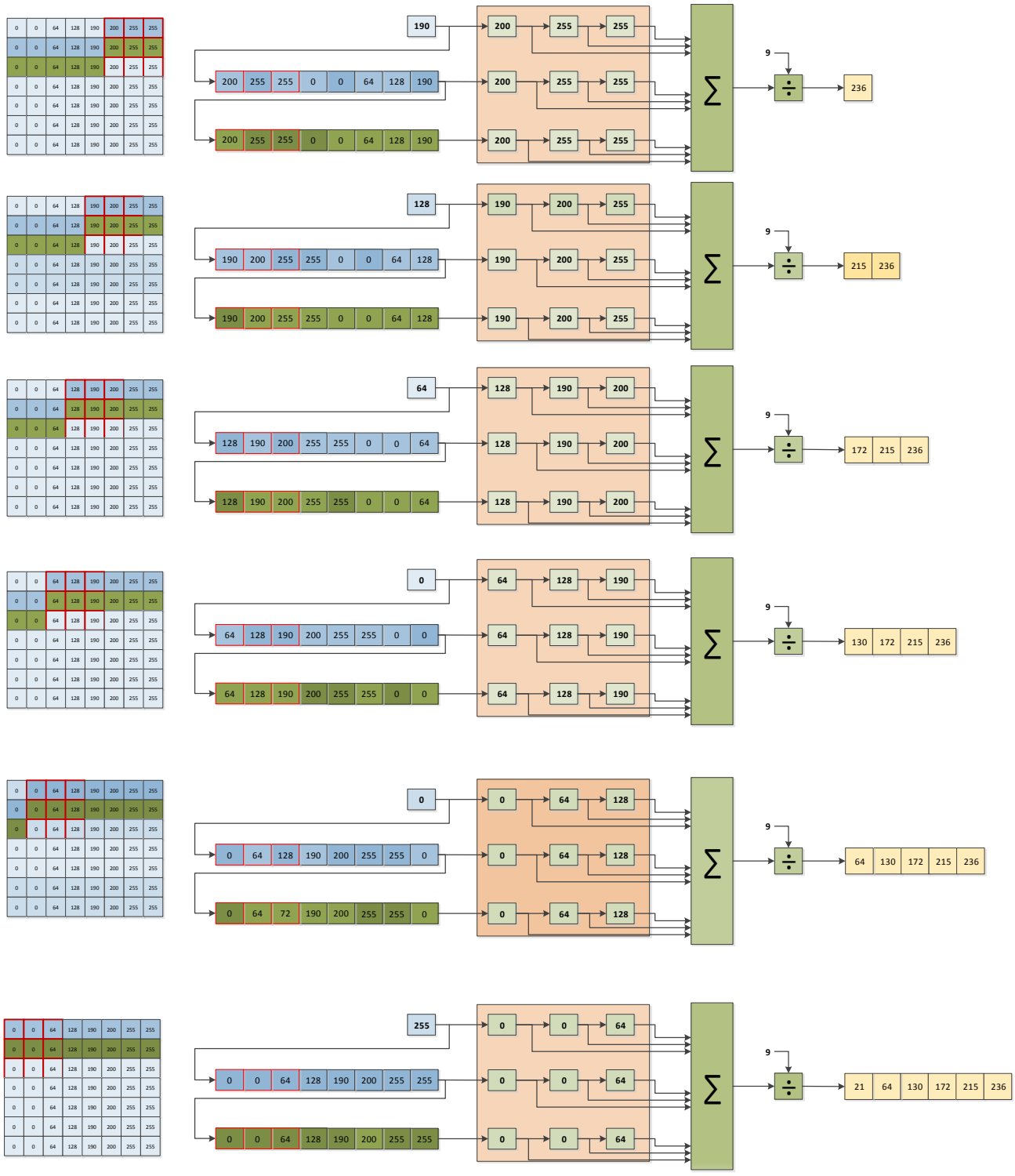
Normally, the image data goes serially, which fits perfectly in a hardware implementation, especially if it is possible to interface directly to the camera. Anyway, a block (represented in Figure 3 as an I/O interface directly connected to the camera) performs the communication with the camera to receive the flow of pixels from the sensor. This block is responsible for implementing the required protocol (I2C, Camera link, *etc.*) to communicate with the capture device, configure it and get the image stream. Once configured and initiated the transmission of data, the flow of pixels is driven into the basic image processing block (Point operations green block in Figure 3). This block represents a low-level vision task block. Point operations have widely used in terms of contrast enhancement, segmentation, color filtering, change detection, masking and many other applications. These operations contain the peculiarity that the output pixel depends only on the value of the input's pixel. Figure 4 depicts an example of this kind of module, where a simple contrast enhancement operation to the input image is performed. The constants *a* and *b* with two simple math operations over input pixel value provide a new luminance value. This value may exceed the range of representable values. Thus, the result must be clipped. In Figure 4, this clipping operation is performed over the output value. Operating with the input value may improve the performance in a parallel scheme because both, math operations and logic comparisons can be processed concurrently with two processors. The result of this module can be stored in some kind of device memory (DDR2 RAM in Figure 3). This last step is not strictly necessary. A buffer storage is required only for system synchronization.

Figure 4. Basic point operation on an FPGA. Input pixel luminance value is multiplied by the constant *a*, and then summed with the constant *b*. The value obtained is clipped before storing the new Q value of the pixel.



Point operations are just the basic low-level vision tasks. Normally, from the enhanced image obtained by a point operation module, the computer vision system performs other low-level operations like an image average filter. Filters or blob tracking operations have in common that they need more information besides the value of the pixel being processed. To do so, providing with the necessary architecture to obtain such information (vector structures, intermediate buffers, *etc.*) is essential. Figure 5 shows some iterations of an image filter computed in an FPGA. On the left the input image is represented for each iteration. The red grid remarks the convolution mask employed to compute the central point in the correspondent iteration, whereas row buffers are depicted in a darker blue and green. Row buffers values are also shown in the right scheme of each iteration. The window mask buffers are represented in orange. Row buffers and window buffers are updated by iterating over the image stream. Parallelism is exploited thanks to these buffers. From window buffer, the simple average can be computed at each iteration. The outcome is a valid pixel value of the output filtered image.

Figure 5. Image average filter on an FPGA. Different successive iterations of the image processing operation.



One of the basic low-level vision tasks is an image convolution. This was also one of the first processing image issues to be implemented in an FPGA [6,33,85]. In this work, images provided by a high-resolution sensor were passed to the FPGA. Then, the program embedded on the FPGA applied a convolution with a mask over the image and afterwards transmitted that preprocessed image to a PC. Recently, this basic operation was used to obtain object's edges of an image provided by a spiking

system [86]. A spike system also called Address-Event-Representation systems (AER) is a camera sensor that computes internally the movement of the objects in the scene. When a pixel changes its luminance, an event is generated and this is the information transmitted by the camera to the computer vision system. In this paper, Linares-Barranco *et al.* present two FPGA implementations of AER-based convolution processors. In [87] a design based of FPGA device is described, used in spiking systems for real time image processing. In this case, the AER device described is a synthetic AER retina emulator, used to simulate spiking retina behavior getting as video source a standard video composite source. This design has been synthesized into synchronous and asynchronous FPGA devices to compare their capabilities. Another project related to AER sensors that uses an FPGA is the described in [45]. In this project, the FPGA can perform five different functions: turn a sequence of frames into AER in real time; histogram AER events into sequences of frames in real time; remap addresses using lookup tables; capture and time-stamp events for offline analysis; and reproduce time-stamped sequences of events in real time. In [88] an FPGA is used to develop a real-time high-definition Bayer to RGB converter. Two image processing operations were parallelized in order to obtain this converter: bilinear interpolation and a new median filter scheme that does not require extra memory and is able to work in real time.

Motion estimation represents a highly descriptive visual cue that can be used for applications such as time interpolation of image sequences, video compression, segmentation from motion or tracking. Optical-flow algorithms have been widely employed for motion estimation using FPGAs [32,89–91]. Different approaches to the subject include image block-matching, gradient constraints, phase conservation, and energy models. In [55] Botella *et al.* present a work developed over a Xilinx board that performs two low-level vision tasks: gradient family optical flow estimation and variant orthogonal moments. These two blocks are then used for a mid-level task (tracking). The system described in [2,32] shows how an optical flow estimation circuit can be implemented using an FPGA platform to achieve real-time computation. The difference in this proposal lies in the fact that authors implement a classical gradient Lucas and Kanade model [92]. They compare different optical flow estimation methods to evaluate the performance of the system.

Adaptive fovea imagers define non-concentric reconfigurable structures for rectangular fields of view. Following procedures used in vision pyramids, from the uniform resolution images supplied by the camera, the upper levels are computed progressively reducing resolution and data volume. In [93,94] adaptive fovea imagers are implemented in an FPGA. Each pixel of the full resolution image is averaged in a low-level vision task. Another interesting image processing application where an FPGA increased the performance is in on-line fingerprint matching [95]. In [96] the FPGA implementation of the structural analysis algorithm consists of a finite state machine core block responsible for managing the neighbourhood analysis. In order to accelerate the computation of distances and angles among minutia points a CORDIC coprocessor is implemented. CORDIC is commonly used when no hardware multiplier is available since the only operations it requires are addition, subtraction, bitshift and table lookup [97].

Stereo correspondence is a low-level vision task. It was not always considered in this group. Nevertheless, no high level information helps the viewer in matching points obtained from the two images of a stereo pair. Stereo vision is a particularly interesting sensor for space vehicles like rovers. Thus, stereo vision matching for embedded systems like FPGAs has been widely researched in the

literature [89,98,99]. Stereo matching algorithms can be classified into two approaches (global and local) based on the strategies used for estimation. Global approaches result in more accurate results but at a higher computational cost. In [100] Barranco *et al.* implement two different alternatives to compute the vector disparity for an active vision system: a gradient-based technique, the local algorithm of Lucas and Kanade and a phase-based one detailed in [101] (also a local algorithm). The first technique estimates small local disparities assuming the intensity or brightness constancy of a pixel between left and right images, while the second one computes the disparity using the phase information for different orientations, in a contrast-independent way. Both methods have been implemented over a Xilinx Virtex4 XC4vfx100 device, and they achieved a working frame rate of 32 fps with a 640 × 480 resolution. Their tests validate the proposal and conclude that Lucas and Kanade algorithm is the best choice. Gil *et al.* [102] describe a mobile robot guiding application, based on an FPGA, which computes stereo correspondence on a pair of images coming from a stereo rig. The stereo algorithm implemented is based on the Census transform, described in [103]. In [104,105] one of the most used global algorithms is implemented in an FPGA: belief propagation. It is a matching algorithm of high precision, but it requires a lot of memory. This memory requirement is even worse for high definition images. The architecture proposed uses a Xilinx Virtex 5 330 VLX FPGA to reduce the execution time required to obtain high definition depth maps.

It is common to find in the literature works that implement only part of the system over an FPGA and the rest is implemented in a PC or another system. Normally, processing image is embebbed on the FPGA, whereas decision tasks (correspondent to high-level vision tasks) are developed over a conventional processor. In [37] the FPGA board allows a concurrent operation as the acquisition of the image and the processing of the data can be performed simultaneously. The processing block includes image filtering, threshold level adjustment and edge detection. The image processed is then passed to a computer in order to measure the high-resolution simultaneous dual liquid level in membrane distillation application.

3.2. Mid-Level Vision Tasks

Normally, the input for a mid-level algorithm is an image processed in a low-level task. Information delivered at this stage corresponds to features of the image itself or of the objects contained in the image. Examples of these are estimation of blobs position, magnification, orientation, corner or edge detection [8], or region labeling.

The great majority of the works including an FPGA have been conducted to obtain a smarter camera sensor. In [17] a binary discrete time Cellular Nonlinear Network (CNN) camera prototype based on an Actel IGLOO FPGA is proposed. The camera is employed to guide a LEGO Mindstorm robot. Embedded sensor must fit the LEGO Mindstorms electronic sensors requirements in terms of device size and power consumption (less than 140 mA). The low-power consumption of the FPGA, and its reconfigurability are exploited in this work to perform corner detection. The work presented in [1] describes the use of an FPGA to compute the relative pose of an underwater robot with reference to a pipe. After an image binarization, FPGA computes the distance between the lines of the pipe appearing in the binary image, the position of the center of the first line and detects if one of the lines has disappeared from the image. From this data a DSP computes parameters like angular displacement

and the distance between the robot and the pipe. Local low and mid-level processing ends here as this information is sent to a host, which is in charge of the final steps concerning scene understanding and interpretation tasks (high-level vision tasks). Blob moving detection over a static background is one of the most common tasks undertaken in computer vision. Navigation, tracking and surveillance applications are directly involved with movement analysis. In [106,107], Principal Component Analysis (PCA) is implemented in an FPGA to detect moving objects within a scene. The complete integrated development of the PCA algorithm on an FPGA was first achieved in these works. In [38,39] an FPGA is employed to implement a remote control system based on networked robot manipulators. The image processing task is divided into two principal data flows. On the one hand, the image taken by the camera located at the robot end-effector is binarized (which represents a low-level vision task) before object descriptors are obtained (a mid-level vision task). Then, FPGA network interface is employed to send the object moments using the SNRP protocol. On the other hand, the image (transformed into a grayscale image) is combined with visual information from the previous data flow. Therefore, the output image of the FPGA is an augmented reality image in which the object centroid position is marked. In order to track an object in the image using a pan-tilt camera, Perez describes an FPGA implementation of a blob's center of gravity computation in real-time [108]. These visual features feed a visual servoing scheme. Recently, a comparison made between three smart camera architectures has demonstrated that FPGA architecture is the better alternative to develop tasks like distance and angle computation of objects relative to camera [109].

Telescopes must deal with several problems related to real time image processing. Modern large telescopes require adaptive optics. Atmospheric turbulence must be on-line compensated, which requires a huge amount of processing power. To solve this, [110] summarizes the early results of a real telescope adaptive optics system based on an FPGA approach. The system has been installed in the OGS telescope at the "Observatorio del Teide" (Tenerife, Spain). This system is embedded on a Xilinx Virtex-4 FPGA. The conceptual design of an FPGA-based slope processor for the wavefront sensors of laser guide stars of extremely large telescopes is presented in [111,112]. The main concepts involved are the use of the subaperture as the finest grain for the parallel processing, the need of a different stream processor for every detector output, and the use of the row of subapertures (or equivalent subset) to determine the reuse of processing hardware. In [113], the same authors develop an FPGA phase recoverer for their CAFADIS camera. The designed phase recoverer carries out the calculations inside the atmospheric characteristic time using really high sampling. A bidimensional Fast Fourier Transform is implemented over the FPGA architecture as nuclei algorithm of the recoverer.

3.3. High-Level Vision Tasks

High-level vision interprets the scene through specific tasks such as relational reasoning, knowledge building, object recognition, *etc.* A task in this group is a decision task based on vision, like face-detection shown in [114]. The most important feature of an FPGA for these operations is low-power consumption. High-level tasks are decision tasks that may reduce sensor data transmission requirements. Adding high-level algorithms to a sensor is a great improvement for very remote sensor like the ones embedded on a satellite.

Hyperspectral imaging is a technique that attempts to identify features on the surface of the Earth using sensors that generally provide large amounts of data. Normally, this data is usually collected by a satellite or an airborne instrument and sent to a ground station that processes it. Thus, the bandwidth connection between the satellite and the station limits the information that can be sent and processed in real time. An on-board system that computes the great quantities of images in real-time increases the system performance [115]. Therefore, the satellite may only send the important information, and not all of the images to be processed in the ground station. The work presented in [115] integrates the Winter's N-FINDR algorithm [116] in an FPGA in order to identify the pixels defining several surfaces. In [117–119] Gonzalez *et al.* implement the Pixel Purity Index (PPI) algorithm over an FPGA to obtain these interesting points in the ground photographed by the satellite. Later, in [119,120], they develop a parallel FPGA-based design of the Image Space Reconstruction Algorithm (ISRA) to sort out the same problem of surface detection using hyperspectral image sensors.

Another high-level vision task related to the satellite photography is described in [77]. The main contribution of this paper is the design of an adviser FPGA approach capable of predicting the reconstruction error of an image when it is compressed with different techniques to a fixed compression ratio, that is, it can advise to the on-board compression system what kind of compression algorithm is more suitable for the satellite requirements. In most cases, this coprocessor will decide whether the on-board JPEG2000 compression system must apply the lossless or lossy algorithm. Sometimes, when high-level vision processing task are required, the hardware design implements a microprocessor embedded on the FPGA (e.g., Xilinx Microblaze) that could run a C-programmed algorithm and be executed without any noticeable restriction from a console application on a desktop PC. In [121] this technique is employed for an embedded vision sensor to track and count people. Sometimes, the FPGA is used in a vision system only to control the image data flow over specific DSP processors. In [122] the hardware architecture of a smart video sensor node was developed using two DSP processors and an FPGA that controls, in a flexible way, the interconnection among processors and the image data flow. The video sensor node processes images locally in order to extract objects of interest, and classify them.

4. FPGAs Employed in Sensor Systems

In this section, the particular characteristics of FPGA devices from several manufacturers are briefly described. The two main FPGA manufacturers in terms of market share are Xilinx and Altera, although there are several others that provide FPGAs like Actel, Atmel, Lattice Semiconductor, *etc.* The characteristics of current products from each of these are described and compared in turn. Of particular interest from a sensor systems' perspective is the power consumption primarily, besides of size of the device in terms of logic resources, embedded memories, embedded multipliers or DSP blocks, and whether or not the device includes a processor core.

4.1. Xilinx

Xilinx was one of the first developers of field programmable gate array technology. It has had a number of devices' families, with the two current families being the Spartan series and the Virtex series. The main difference between the two families is that the Spartan devices are designed primarily

for low cost, and theVirtex devices are designed primarily for high performance. Recently, Xilinx has focused on reducing the power consumption of its devices using integrated optimized hard-core blocks, for instance the Virtex-II Pro family devices have two PowerPC 405 hard-core processors. This processors permit to virtually add any peripheral or create custom accelerators that extend system performance.

The Spartan series is employed for low-power design, cost sensitivity and high-volume; e.g., displays, wireless routers and other applications. The Spartan-6 family is built on a 45 nanometer, 9-metal layer, dual-oxide process technology. The Spartan-6 was marketed in 2009 as a low-cost solution for automotive, wireless communications, flat-panel display and video vigilance applications. Furthermore, most of sensor systems designers employ the Spartan-III or Spartan 6 FPGAs due to its low cost and low energy consumption. In [81], the Spartan 3AN-50 device has been used to implement tactile sensors taking advantage of its numerous I/O pins, compact size and low cost. In [41], the authors used the XC3S2000 device from Spartan-III family to achieve a real-time fuzzy controller. The fuzzy algorithm has been designed with the goal of developing a real-time FPGA-based controller. Therefore, the complexity has been reduced, while keeping a great degree of parallelism. Other works use the computational power of the Spartan III FPGAs to achieve vision and image processing tasks [88,121,122].

The Virtex series of FPGAs integrate features that include FIFO logic, DSP blocks, PCI-Express controllers, Ethernet MAC blocks, and high-speed transceivers. In addition to FPGA logic, the Virtex series include embedded fixed function hardware for commonly used functions such as multipliers, memories, serial transceivers and microprocessor cores. Xilinx's most recently Virtex family, the Virtex 7, is based on a 28 nm design and is designed to deliver a two-fold system performance improvement at 50% lower power compared to previous generation Virtex-6 devices. In addition, Virtex-7 doubles the memory bandwidth compared to previous generation Virtex FPGAs with 1,866 Mbit/s memory interfacing performance and over two million logic cells. Sensor systems designers in Spain tend to use devices like Virtex-E [50,55,73,74], Virtex-II [38,39,107] and Virtex-II Pro [33,106,117] to achieve complex sensing systems and specifically vision systems or wireless sensor networks because such FPGAs are a powerful high performance devices at a reasonable costs.

In the newest generation, the 7 Series devices, the Spartan family is replaced by the Artix and Kintex families. Within each generation, a range of device sizes is available. Table 1 depicts the characteristics and the static power consumption (calculated via the Xilinx Power Estimator tool XPE that is not available for old FPGA families, where it has replaced by "-" in the table) of the different FPGA series of Xilinx.

4.2. Altera

Currently, Altera provides three families of FPGA devices: the Cyclone series (low cost), the Arria series (mid-range) and the Stratix series (high performance). None of these families incorporate a hard-core processor within the logic but Altera has focused its efforts on its soft-core processor called NIOS processor, or NIOS-II in its newest FPGA devices (The last FPGA family from Altera that had a hard-core processor was the Excalibur FPGA family witch integrated a microprocessor subsystem called ARM922T).

Table 1. Characteristics, static power consumption and related works of the different FPGA series of Xilinx.

Family	Process (nm)	LUT Size	Logic Cells	RAM (bits)	Number of DSP Blocks	HC-Processor	Static Power Consumption (mW)	Related Works
Spartan-II	180	4	432–5.3 k	16 K–56 K	-	-	-	[18,34,35,42,45,87]
Spartan-III	90	4	1.5 k–66 k	72 k–1.8 M	4–104	-	27–336	[3,4,14,19,30,35,37,41, 51–53,56,62,65,66, 68,76,81,86,88,121–124]
Spartan-6	45	6	2 k–147 k	144 K–4.8 M	4–180	-	11–94	[24,67]
Artix-7	28	6	11 k–215 k	720 K–13 M	40–740	-	68–122	-
Kintex-7	28	6	19 k–477 k	2.3 M–34 M	120–1920	-	79–216	-
Virtex	220	4	1.7 k–27 k	32 K–128 K	-	-	-	-
Virtex-E	180	4	1.7 k–73 k	65 K–851 K	-	-	-	[2,25,26,32, 48–50,55,73,74]
Virtex_II	120/150	4	512–93 k	72 K–128 K	4–168	-	-	[38,39,102,107]
Virtex-II Pro	90/130	4	2.8 k–88 k	216 K–7.8 M	12–444	PowerPC 405	-	[33,78,106,117–119]
Virtex-4	90	4	12 k–200 k	648 K–9.7 M	32–96	PowerPC 405	128–1278	[64,86,110,112,113, 115,120]
Virtex-5	65	6	12 k–415 k	936 K–18 M	32–1056	PowerPC 405	276–3028	[47,65,80,87,104,105,111]
Virtex-6	40	6	46 k–474 k	5.5 M–37 M	288–2016	-	715–4441	[36]
Virtex-7	28	6	179 k–1954 k	14 M–68 M	700–3600	-	177–1250	-

The Cyclone series was designed for low cost applications, making it well suited for sensor systems including embedded image processing applications. The FPGA family most recent from cyclone series is the Cyclone VI based on 4-input LUT (Look Up Table) with a register on the output. It incorporates dedicated hardware multiplication blocks to achieve a single multiplication of 18-bit numbers or tow multiplications of 9-bit numbers, also it has a configurable-size embedded memory blocks up to 150 Kbits. The performance of the I/O blocks has been improved to support a variety of interface standards like DDR/QDR memories, PCI express and others. In [20], the EP1C6T144CSN device from Altera's Cyclone FPGA family was used to implement an intelligent Front-End Signal Conditioning Circuit for IR Sensors.

In Stratix FPGAs, The basic structure is similar to that of the Cyclone but with much more improvements, where the LUT here has 8 inputs with 28 nm process (for Stratix V devices), furthermore incorporate sophisticated DSP blocks up to 54 × 54 precision and have 20 Kbit Ram blocks that can be configured as dual-port RAM, FIFO or shift registers.

The Arria series based on 8-input LUT, integrate high speed transceiver blocks designed primarily for high performance serial communication applications. The other features of Arria FPGAs are basically the same as that of the Stratix. Table 2 depicts the characteristics and power consumption of Altera FPGA families (power consumption was calculated via PowerPlay Early Power Estimators tool (not available for Excalibur family)).

Table 2. Characteristics and static power consumption of the different FPGA series of Altera.

Family	Process (nm)	LUT Size	Logic Cells	RAM (bits)	Number of DSP Blocks	HC-Processor	Static Power Consumption (mW)
Excalibur	180	4	4 k–34 k	32 K–256 K	-	ARM922T	-
Cyclone	130	4	2.9 k–20 k	58 K–288 K	-	-	48–120
Cyclone II	90	4	4.6 k–64 k	117 K–1.1 M	13–150	-	29–193
Cyclone III	65	4	5.2 k–119 k	414 K–3.8 M	23–288	-	55–150
Cyclone IV	60	4	6.3 k–150 k	270 K–6.3 M	15–266	-	60–152
Arria GX	90	8	8 k–36 k	1.2 M–4.3 M	10–44	-	405–826
Arria II GX	40	8	6 k–102 k	783 K–8.3 M	29–92	-	329–793
Stratix	130	4	10 k–79 k	899 K–7 M	6–22	-	187.5–1,395
Stratix II	90	8	6 k–72 k	410 K–8.9 M	12–96	-	323–1,435
Stratix III	65	8	19 k–135 k	1.8 M–14 M	27–112	-	404–1,255
Stratix IV	40	8	29 k–325 k	6.3 M–22 M	48–161	-	436–1,739
Stratix V	28	8	239 k–1,087 k	29 M–53 M	200–1,840	-	641–1,153

4.3. Other FPGA Providers

Actually, there are various companies that produce and provide FPGA devices like Actel, Lattice Semiconductor, Atmel, Tabula, SiliconBlue, Achronix, QuickLogic, MathStar, Cypress and others.

Actel provides a range of low power FPGAs, making them ideally suited for sensor systems. There are three main families: the Axcelerator, the ProASIC3 and the IGLOO. The IGLOO family is a low

power, based on 3-input LUT and 130 nm process technology, its RAM is a true dual-port memory and the larger devices are able to implement a 32-bit ARMprocessor as a soft-core block. In [17], the authors implement a low-cost camera sensor based on Actel IGLOO FPGA that fits its low power consumption, reprogrammability and cost requirements.

Lattice Semiconductor produces a number of FPGA families. Its current families are the ECP series (low cost), the XP series (non-volatile) and the SC/M family (high performance). The XP FPGAs contain an on-chip no-volatile flash memory that may be used to configure the FPGA on power-up. This saves the need of an external flash memory and consequently reduces design costs.

5. Conclusions

To conclude, FPGA devices have reached a high level of development that puts them in competition with the traditional application specific integrated circuits (ASICs) in terms of performance, power consumption and cost. In just two decades, they have turned from merely simple Hardware-Prototyping tool into an impressive solution for those system designs that require a very high level of accuracy, powerful computational capabilities and real parallel execution. From a technological perspective, the industry of FPGA devices has made great strides from simple FPGA chips for prototyping purposes only that included a few hundred logic cells and small blocks of memory to FPGAs with the 28 nm process technology. These last FPGAs include more than two million logic cells, several types of memories and peripheral interfaces. This is the case of the Virtex-7 FPGA devices family from Xilinx and the Stratix-V from Altera.

As described throughout the paper, research on FPGA-based sensor systems in Spain is well established. The capabilities of the new FPGAs allow providing the sensor systems with different functions such as self-diagnosis, signal processing, communications in a WSN, *etc.* Currently, the term smart sensor is employed to refer these sensor systems with some kind of "intelligence". Furthermore, we have described interesting research related with these FPGA-based sensor systems. Within this research, we can mention several works like the implementation of real time signal processing from the obtained sensory information, the use of parallel architectures to process a great quantity of information, to implement and optimize sensor-based controllers in embedded systems, the use of WSNs to process sensory information using different FPGA-based network nodes, *etc.* Computer vision systems have specially been enhanced by the use of FPGAs. From low-level vision tasks like basic point pixel operations or simple filtering image processing, through mid-level tasks that compute visual features like image moments, until high-level vision tasks where the FPGA allows taking important decisions by processing an image, camera sensors have been considerably enhanced. Different works employ FPGA in sensor systems to implement parallel algorithms in order to process data in a low-power consumption device. These algorithms use FPGA architectures not only to implement simple combinational and sequential circuits, but also to include high-level operations in embedded systems. The optimal implementation of these algorithms using the capabilities of the new FPGAs will suppose an important research field in the near future.

Acknowledgments

The research leading to these results has received funding from the Spanish Government and European FEDER funds (DPI2012-32390), the Valencia Regional Government (PROMETEO/2013/085) and the University of Alicante (GRE12-17).

Author Contributions

Gabriel J. García, Carlos A. Jara and Jorge Pomares reviewed the state-of-the-art and wrote the initial version of the manuscript. Aiman Alabdo developed extensively and finished Section 4, and Gabriel J. García together with Lucas M. Poggi divided and wrote Section 5. Fernando Torres provided their suggestions and corrections during the preparation of this work. All authors contributed extensively to the final version.

Conflicts of Interest

The authors declare no conflict of interest.

References

1. Batlle, J. A New FPGA/DSP-Based Parallel Architecture for Real-Time Image Processing. *Real Time Imaging* **2002**, *8*, 345–356.
2. Díaz, J.; Ros, E.; Mota, S.; Carrillo, R.; Agis, R. Real Time Optical Flow Processing System. In *Field Programmable Logic and Application*; Becker, J., Platzner, M., Vernalde, S., Eds.; Springer-Verlag: Berlin, Germany, 2004; Volume 3203, pp. 617–626.
3. Jimenez-Fernandez, A.; Jimenez-Moreno, G.; Linares-Barranco, A.; Dominguez-Morales, M.J.; Paz-Vicente, R.; Civit-Balcells, A. A Neuro-Inspired Spike-Based PID Motor Controller for Multi-Motor Robots with Low Cost FPGAs. *Sensors* **2012**, *12*, 3831–3856.
4. Tu, Y.W.; Ho, M.T. Design and implementation of robust visual servoing control of an inverted pendulum with an FPGA-based image co-processor. *Mechatronics* **2011**, *21*, 1170–1182.
5. Adelson, E.H.; Wang, J.Y.A.; Niyogi, S.A. Mid-level vision: New directions in vision and video. In Proceedings of the IEEE International Conference Image Processing, Austin, TX, USA, 13–16 November 1994; Volume 2, pp. 21–25.
6. Arias-Estrada, M.; Torres-Huitzil, C. Real-time field programmable gate array architecture for computer vision. *J. Electron. Imaging* **2001**, *10*, 289–296.
7. Reaz, M.B.I.; Mohd-Yasin, F.; Tan, S.L.; Tan, H.Y.; Ibrahimy, M.I. Partial Encryption of Compressed Images Employing FPGA. In Proceedings of the IEEE International Symposium on Circuits and Systems, Kobe, Japan, 23–26 May 2005; pp. 2385–2388.
8. Lu, X.; Song, L.; Shen, S.; He, K.; Yu, S.; Ling, N. Parallel Hough Transform-based straight line detection and its FPGA implementation in embedded vision. *Sensors* **2013**, *13*, 9223–9247.
9. Dubois, J.; Ginhac, D.; Paindavoine, M.; Heyrman, B. A 10 000 fps CMOS sensor with massively parallel image processing. *IEEE J. Solid State Circuits* **2008**, *43*, 706–717.

10. Reza, A.M. Realization of the Contrast Limited Adaptive Histogram Equalization (CLAHE) for real-time image enhancement. *J. VLSI Signal Process. Syst. Signal Image Video Technol.* **2004**, *38*, 35–44.
11. De la Piedra, A.; Braeken, A.; Touhafi, A. Sensor Systems Based on FPGAs and Their Applications: A Survey. *Sensors* **2012**, *12*, 12235–12264.
12. Portilla, J.; Castro, A.; Torre, E.; Riesgo, T. Modular Architecture for Wireless Sensor Network Nodes. *J. Univers. Comput. Sci.* **2006**, *12*, 328–339.
13. Monmasson, E.; Cirstea, M.N. FPGA Design Methodology for Industrial Control Systems—A Review. *IEEE Trans. Ind. Electron.* **2007**, *54*, 1824–1842.
14. Castillo, J.M.; Olivares, J.; Palomares, J.M. Design of a Wireless Pulse Oximeter using a Mesh ZigBee Sensor Network. In Proceedings of the International Conference on Biomedical Electronics and Devices (Biodevices), Rome, Italy, 26–29 January 2011; pp. 401–404.
15. Hernández, A.; Ureña, J.; Hernanz, D.; García, J.J.; Mazo, M.; Derutín, J.P.; Serot, J.; Palazuelos, S.E. Real-time implementation of an efficient Golay correlator (EGC) applied to ultrasonic sensorial systems. *Microprocess. Microsyst.* **2003**, *27*, 397–406.
16. Rodriguez, C.; Morales, L.; Osornio, R.A.; Herrera, G.; Romero, R. FPGA-Based Fused Smart Sensor for Dynamic and Vibration Parameter Extraction in Industrial Robot Links. *Sensors* **2010**, *10*, 4114–4129.
17. Albo-Canals, J.; Ortega, S.; Perdices, S.; Badalov, A.; Vilasis-Cardona, X. Embedded low-power low-cost Camera Sensor based on FPGA and its applications in mobile robots. In Proceedings of the 19th IEEE International Conference on Electronics, Circuits, and Systems (ICECS 2012), Sevilla, Spain, 9–12 December 2012; pp. 336–339.
18. Álvarez, F.J.; Hernández, A.; Ureña, J.; Mazo, M.; García, J.J.; Jiménez, J.A.; Jiménez, A. Real-time implementation of an efficient correlator for complementary sets of four sequences applied to ultrasonic pulse compression systems. *Microprocess. Microsyst.* **2006**, *30*, 43–51.
19. Peter, S.; Stecklina, O.; Portilla, J.; Torre, E.; Langendoerfer, P.; Riesgo, T. Reconfiguring Crypto Hardware Accelerators on Wireless Sensor Nodes. In Proceedings of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks Workshops, Rome, Italy, 22–26 June 2009.
20. De Arcas, G.; Ruiz, M.; Gutierrez, R.; Villamayor, V. Design of an Intelligent Front-End Signal Conditioning Circuit for IR Sensors. *IEEE Trans. Nucl. Sci.* **2008**, *55*, 14–20.
21. Sánchez-Solano, S.; Brox, M.; Toro, E.; Brox, P.; Baturone, I. Model-Based Design Methodology for Rapid Development of Fuzzy Controllers on FPGAs. *IEEE Trans. Ind. Inform.* **2013**, *9*, 1361–1370.
22. Sulaiman, N.; Obaid, Z.A.; Marhaban, M.H.; Hamidon, M.N. FPGA-Based Fuzzy Logic: Design and Applications—A Review. *Int. J. Eng. Technol.* **2009**, *1*, 491–503.
23. Gamazo-Real, J.C.; Vázquez-Sánchez, E.; Gómez-Gil, J. Position and speed control of brushless DC motors using sensorless techniques and application trends. *Sensors* **2010**, *10*, 6901–6947.
24. Perez-Peña, F.; Morgado-Estevez, A.; Linares-Barranco, A.; Jiménez-Fernández, A.; Lopez-Coronado, J.; Muñoz-Lozano, J.L. A FPGA spike-based robot controlled with neuro-inspired VITE. In *Advances in Computational Intelligence*; Rojas, I., Joya, G., Gabestany, J., Eds.; Springer-Verlag: Berlin, Germany, 2013; Volume 7902, pp. 299–308.

25. Gadea, R.; Cerdá, J.; Ballester, F.; Mocholí, A. Artificial Neural Network Implementation on a single FPGA of a Pipelined On-Line Backpropagation. In Proceedings of the 13th International Symposium on System Synthesis, Madrid, Spain, 20–22 September 2000; pp. 225–230.
26. Ortigosa, E.M.; Cañas, A.; Ros, E.; Carrillo, R.R. FPGA Implementation of a Perceptron-Like Neural Network for Embedded Applications. In *Artificial Neural Nets Problem Solving Methods*; Mira, J., Álvarez, J.R., Eds.; Springer-Verlag: Berlin, Germany, 2011; Volume 2687, pp. 1–8.
27. Rosado, A.; Bataller, M.; Guerrero, J. FPGA implementation of Spiking Neural Network. In Proceedings of the 1st IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control, Wurzburg, Germany, 3–5 April 2012; pp. 139–144.
28. Muthuramalingam, A.; Himavathi, S.; Srinivasan, E. Neural Network Implementation Using FPGA: Issues and Application. *Int. J. Inform. Technol.* **2008**, *4*, 86–92.
29. Calmon, F.; Fathallah, M.; Viverge, P.J.; Gontrand, C.; Carrabina, J.; Foussier, P. FPGA and Mixed FPGA-DSP Implementations of Electrical Drive Algorithms. In *Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream*; Glesner, M., Zipf, P., Renovell, M., Eds.; Springer-Verlag: Berlin, Germany, 2002; Volume 2438, pp. 1144–1147.
30. Sanchez, P.M.; Machado, O.; Bueno-Peña, E.J.; Rodríguez, F.J.; Meca, F.J. FPGA-Based Implementation of a Predictive Current Controller for Power Converters. *IEEE Trans. Ind. Inform.* **2013**, *9*, 1312–1321.
31. MacCleery, B.; Kassas, Z.M. New mechatronics development techniques for FPGA-based control and simulation of electromechanical systems. In Proceedings of the 17th IFAC World Congress, Seoul, Korea, 6–11 July 2008.
32. Díaz, J.; Ros, E.; Pelayo, F.; Ortigosa, E.M.; Mota, S. FPGA-Based Real-Time Optical-Flow System. *IEEE Trans. Circuits Syst. Video Technol.* **2006**, *16*, 274–279.
33. Bravo, I.; Jimenéz, P.; Mazo, M.; Lázaro, J.L.; Martín, E. Architecture Based on FPGA's for Real-Time Image Processing. In *Reconfigurable Computing: Architectures and Application*; Bertels, K., Cardoso, J.M.P., Vassiliadis, S., Eds.; Springer-Verlag: Berlin, Germany, 2006; pp. 152–157.
34. Jimenez-Fernandez, A.; Lujan-Martinez, C.; Paz-Vicente, R.; Linares-Barranco, A.; Jimenez, G.; Civit, A. From Vision Sensor to Actuators, Spike Based Robot Control through Address-Event-Representation. In *Bio-Inspired Systems: Computational and Ambient Intelligence*; Cabestany, J., Sandoval, F., Prieto, A., Corchado, J.M., Eds.; Springer-Verlag: Berlin, Germany, 2009; Volume 5517, pp. 797–804.
35. Linares-Barranco, A.; Gomez-Rodriguez, F.; Jimenez-Fernandez, A.; Delbruck, T.; Lichtensteiner, P. Using FPGA for visuo-motor control with a silicon retina and a humanoid robot. In Proceedings of the IEEE International Symposium on Circuits and Systems, New Orleans, LA, USA, 27–30 May 2007; pp. 1192–1195.
36. Lentaris, G.; Diamantopoulos, D.; Stamoulias, G.; Siozios, K.; Soudris, D.; Rodrigalvarez, M.A. FPGA-based path-planning of high mobility rover for future planetary missions. In Proceedings of the 19th IEEE International Conference on Electronics, Circuits, and Systems, Sevilla, Spain, 9–12 December 2012; pp. 85–88.

37. Lorenz, M.G.; Mengibar-Pozo, L.; Izquierdo-Gil, M.A. High resolution simultaneous dual liquid level measurement system with CMOS camera and FPGA hardware processor. *Sens. Actuators A Phys.* **2013**, *201*, 468–476.
38. Marín, R.; León, G.; Wirz, R.; Sales, J.; Claver, J.M.; Sanz, P.J. Remote Control within the UJI Robotics Manufacturing Cell using FPGA-Based Vision. In Proceedings of the European Control Conference, Kos, Greece, 2–5 July 2007.
39. Marin, R.; León, G.; Wirz, R.; Sales, J.; Claver, J.M.; Sanz, P.J.; Fernández, J. Remote Programming of Network Robots Within the UJI Industrial Robotics Telelaboratory: FPGA Vision and SNRP Network Protocol. *IEEE Trans. Ind. Electron.* **2009**, *56*, 4806–4816.
40. Yang, N.; Li, D.; Zhang, J.; Xi, Y. Model predictive controller design and implementation on FPGA with application to motor servo system. *Control Eng. Pract.* **2012**, *20*, 1229–1235.
41. Alvarez, J.; Lago, A.; Nogueiras, A.; Martinez-Penalver, C.; Marcos, J.; Doval, J.; Lopez, O. FPGA implementation of a fuzzy controller for automobile DC-DC converters. In Proceedings of the IEEE International Conference on Field Programmable Technology, Bangkok, Thailand, 13–15 December 2006; pp. 237–240.
42. Alcantara, S.; Pedrett, C.; Vilanova, R.; Moreno, R. An undergraduate laboratory course on fuzzy controller implementation in FPGAs. In Proceedings of the Mediterranean Conference on Control & Automation. MED'07, Athens, Greece, 27–29 July 2007; pp. 1–6.
43. Kassas, Z.M. Methodologies for Implementing FPGA-Based Control Systems. In Proceedings of the 18th IFAC World Congress, Milano, Italy, 28 August–2 September 2011.
44. Zhu, J.; Sutton, P. FPGA Implementations of Neural Networks—A Survey of a Decade of Progress. In *Field Programmable Logic and Application, Lecture Notes in Computer Science*; Cheung, P.Y.K., Constantinides, G.A., de Sousa, J.T., Eds.; Springer-Verlag: Lisbon, Portugal, 2003; Volume 2778, pp. 1062–1066.
45. Serrano-Gotarredona, R.; Oster, M.; Lichtsteiner, P.; Linares-Barranco, A.; Paz-Vicente, R.; Gomez-Rodriguez, F.; Linares-Barranco, B. CAVIAR: A 45k neuron, 5M synapse, 12G connects/s AER hardware sensory-processing- learning-actuating system for high-speed visual object recognition and tracking. *IEEE Trans. Neural Netw.* **2009**, *20*, 1417–1438.
46. Moreno, S.V.; Vera, L.A.; Osornio, R.A.; Dominguez, A.; Stiharu, I.; Romero, R. A field programmable gate array-based reconfigurable smart-sensor network for wireless monitoring of new generation computer numerically controlled machines. *Sensors* **2010**, *10*, 7263–7286.
47. Osuna, C.G.; Marcos, M.S.; Ituero, P.; Lopez-Vallejo, M. A monitoring infrastructure for FPGA self-awareness and dynamic adaptation. In Proceedings of the 19th IEEE International Conference on Electronics, Circuits, and Systems, Sevilla, Spain, 9–12 December 2012; pp. 765–768.
48. Lopez-Buedo, S.; Boemo, E. Making Visible the Thermal Behaviour of Embedded Microprocessors on FPGAs. A Progress Report. In Proceedings of the ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays, Monterrey, CA, USA, 22–24 February 2004; pp. 79–86.
49. Lopez-Buedo, S.; Garrido, J.; Boemo, E.I. Dynamically inserting, operating, and eliminating thermal sensors of FPGA-based systems. *IEEE Trans. Compon. Packag. Technol.* **2002**, *25*, 561–566.

50. Lopez-Buedo, S.; Riviere, P.; Pernas, P.; Boemo, E. Run-Time Reconfiguration to Check Temperature in Custom Computers: An Application of JBits Technology. In *Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream*; Glesner, M., Zipf, P., Renovell, M., Eds.; Springer-Verlag: Berlin, Germany, 2002; Volume 2438, pp. 162–170.
51. Morales, D.P.; Garcia, A.; Palma, A.J.; Olmos, A.M.; Castillo, E. Exploiting Analog and Digital Reconfiguration for Smart Sensor Interfacing. In Proceedings of the International Conference on Field Programmable Logic and Applications, Amsterdam, The Netherlands, 27–29 August 2007; pp. 706–709.
52. Morales, D.P.; Garcia, A.; Palma, A.J.; Olmos, A.M. Merging FPGA and FPAA Reconfiguration Capabilities for IEEE 1451.4 Compliant Smart Sensor Applications. In Proceedings of the 3rd Southern Conference on Programmable Logic, Mar del Plata, Argentina, 26–28 February 2007; pp. 217–220.
53. Ares, L.; Rodríguez-andina, J.J.; Fariña, J. FPGA-Based Direct Resistance and Capacitance Measurements. In Proceedings of the 35th IEEE Annual Conference in Industrial Electronics, IECON'09, Porto, Portugal, 3–5 November 2009; pp. 2837–2841.
54. Poki, C.; Mon, S.; Zhi, Z.; Zi, Z.; Chun, C. A Fully Digital Time-Domain Smart Temperature Sensor Realized With 140 FPGA Logic Elements. *IEEE Trans. Circuits Syst.* **2007**, *54*, 2661–2668.
55. Botella, G.; Martín, H.J.A.; Santos, M.; Meyer-Baese, U. FPGA-Based Multimodal Embedded Sensor System Integrating Low- and Mid-Level Vision. *Sensors* **2011**, *11*, 8164–8179.
56. Oballe-peinado, Ó.; Hidalgo-lópez, J.A.; Sánchez-durán, J.A.; Castellanos-ramos, J.; Vidal-verdú, F. Architecture of a tactile sensor suite for artificial hands based on FPGAs. In Proceedings of the 4th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics, Rome, Italy, 24–27 June 2012; pp. 12–117.
57. Pantazis, N.; Vergados, D. A survey on power control issues in Wireless Sensor Networks. *IEEE Commun. Surv. Tutor.* **2007**, *9*, 86–107.
58. Zhang, D.; Pan, Y.; Hu, X. Design of High-Speed Parallel Data Interface Based on ARM & FPGA. *J. Comput.* **2012**, *7*, 804–809.
59. Berder, O.; Sentieys, O. PowWow: Power Optimized Hardware/Software Framework for Wireless Motes. In Proceedings of the 23rd International Conference on Architecture of Computing Systems (ARCS), Hannover, Germany, 22–23 February 2010; pp. 1–5.
60. Liao, J.; Singh, B.K.; Khalid, M.A.; Tepe, K.E. FPGA based wireless sensor node with customizable event-driven architecture. *J. Embed. Syst.* **2013**, *5*, 1–11.
61. Durante, M.S.; Mahlknecht, S. An Ultra Low Power Wakeup Receiver for Wireless Sensor Nodes. In Proceedings of the 3rd International Conference on Sensor Technologies and Applications, Athens, Greece, 18–23 June 2009; pp. 167–170.
62. Magdaleno, E.; Rodríguez, M.; Ayala, A.J. VHDL Implementation of a communication interface for integrated MEMS. *Microsyst. Technol.* **2008**, *14*, 453–462.
63. Rosello, V.; Portilla, J.; Riesgo, T. Ultra Low Power FPGA-Based Architecture for Wake-up Radio in Wireless Sensor Networks. In Proceedings of the 37th Annual Conference on IEEE Industrial Electronics Society, Melbourne, Australia, 7–10 November 2011; pp. 3826–3831.

64. Dondo, J.; Molina, F.S.; Rincon, F.; Moya, F.; Lopez, J.C. Ubiquitous FPGA Access for Data Intensive Computing. In Proceedings of the International Symposium on Ubiquitous Computing and Ambient Intelligence, Riviera Maya, Mexico, 5–9 December 2011.
65. Jaraíz-Simón, M.D.; Gómez-Pulido, J.A.; Vega-Rodríguez, M.A.; Sánchez-Pérez, J.M. Fast Decision Algorithms in Low-Power Embedded Processors for Quality-of-Service Based Connectivity of Mobile Sensors in Heterogeneous Wireless Sensor Networks. *Sensors* **2012**, *12*, 1612–1624.
66. Portilla, J.; Castro, A.; Abril, A.; Riesgo, T. Integrated hardware interfaces for modular sensor networks. In Proceedings of the SPIE 6590, VLSI Circuits and Systems III, Maspalomas, Spain, 10 May 2007.
67. Valverde, J.; Otero, A.; Lopez, M.; Portilla, J.; de la Torre, E.; Riesgo, T. Using SRAM Based FPGAs for Power-Aware High Performance Wireless Sensor Networks. *Sensors* **2012**, *12*, 2667–2692.
68. Sanchez, A.; Elvira, S.; Castro, A.; Glez-de-rivera, G.; Ribalda, R.; Garrido, J. Low Cost Indoor Ultrasonic Positioning Implemented in FPGA. In Proceedings of the 35th Annual Conference of IEEE Industrial Electronics, Porto, Portugal, 3–5 November 2009; pp. 2709–2714.
69. Atalik, T.; Deniz, M.; Koc, E.; Gercek, C.O.; Gultekin, B.; Ermis, M.; Cadirci, I. Multi-DSP and FPGA-based fully digital control system for cascaded multilevel converters used in FACTS applications. *IEEE Trans. Ind. Inform.* **2012**, *8*, 511–527.
70. Boni, E.; Bassi, L.; Dallai, A.; Guidi, F.; Ramalli, A.; Ricci, S.; Housden, J.; Tortoli, P. A reconfigurable and programmable FPGA-based system for nonstandard ultrasound methods. *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* **2012**, *59*, 1378–1385.
71. Idkhajine, L.; Monmasson, E.; Maalouf, A. Fully FPGA-based sensorless control for synchronous AC drive using an extended Kalman filter. *IEEE Trans. Ind. Electron.* **2012**, *59*, 3908–3918.
72. Domínguez-Morales, M.; Jimenez-Fernandez, A.; Cerezuela-Escudero, E.; Paz-Vicente, R.; Linares-Barranco, A.; Jimenez, G. On the Designing of Spikes Band-Pass Filters for FPGA. In *Artificial Neural Networks and Machine Learning—ICANN 2011*; Honkela, T., Duch, W., Girolami, M., Kaski, S., Eds.; Springer-Verlag: Berlin, Germany, 2011; Volume 6792, pp. 389–396.
73. Hernández, A.; García, J.J.; Mazo, M.; Hernanz, D.; Derutín, J.P.; Sérot, J. Ultrasonic Ranging Sensor using Simultaneous Emissions from Different Transducers. *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* **2004**, *51*, 1660–1670.
74. Hernández, A.; Ureña, J.; García, J.J.; Mazo, M.; Derutín, J.P.; Sérot, J. Ultrasonic Sensor Performance Improvement Using DSP-FPGA Based Architectures. In Proceedings of the IEEE 28th Annual Conference of the Industrial Electronics Society, Sevilla, Spain, 5–8 November 2002; Volume 4, pp. 2694–2699.
75. Sklyarov, V. FPGA-based implementation of recursive algorithms. *Microprocess. Microsyst.* **2004**, *28*, 197–211.
76. Perez, M.C.; Ureña, J.; Hernández, A.; de Marziani, C.; Jiménez, A. Hardware Implementation of an Efficient Correlator for Interleaved Complementary Sets of Sequences. *J. Univers. Comput. Sci.* **2007**, *13*, 388–406.

77. Guzmán, A.; Beltrán, M. Satellite On-Board Image Compression Adviser. In Proceedings of the 4th IEEE International Symposium on Signal Processing and Information Technology, Rome, Italy, 18–21 December 2004; pp. 296–301.
78. Cuenca, S.; Grediaga, A.; Llorens, H.; Albero, M. Performance Evaluation of FPGA-Embedded Web Servers. In Proceedings of the 14th IEEE International Conference on Electronics, Circuits and Systems, Marrakech, Morocco, 11–14 December 2007; pp. 1187–1190.
79. Restelli, A.; Abbiati, R.; Geraci, A. Digital field programmable gate array-based lock-in amplifier for high-performance photon counting applications. *Rev. Sci. Instrum.* **2005**, doi:org/10.1063/1.2008991.
80. León-Franco, J.J.; Boemo, E.; Castillo, E.; Parrilla, L. Ring oscillators as thermal sensors in FPGAs: Experiments in low voltage. In Proceedings of the VI Southern Programmable Logic Conference (SPL), Ipojuca, Brazil, 24–26 March 2010; pp. 133–137.
81. Vidal-Verdú, F.; Oballe-Peinado, Ó.; Sánchez-Durán, J.A.; Castellanos-Ramos, J.; Navas-González, R. Three Realizations and Comparison of Hardware for Piezoresistive Tactile Sensors. *Sensors* **2011**, *11*, 3249–3266.
82. Bouridane, A.; Crookes, D.; Donachy, P.; Alotaibi, K.; Benkrid, K. A high level FPGA-based abstract machine for image processing. *J. Syst. Archit.* **1999**, *45*, 809–824.
83. Donachy, P.; Crookes, D.; Bouridane, A.; Alotaibi, K.; Benkrid, A. Design and implementation of a high level image processing machine using reconfigurable hardware. In Proceedings of the SPIE 3526, Configurable Computing: Technology and Applications, Bellingham, WA, USA, 8 October 1998.
84. Bailey, D.G. *Design for Embedded Image Processing on FPGAs*; John Wiley & Sons: Singapore, Singapore, 2011.
85. Torres-Huitzil, C.; Arias-Estrada, M. FPGA-Based Configurable Systolic Architecture for Window-Based Image Processing. *EURASIP J. Adv. Signal Process.* **2005**, *7*, 1024–1034.
86. Linares-Barranco, A.; Paz, R.; Gómez-Rodríguez, F.; Jiménez, A.; Rivas, M.; Jiménez, G.; Civit, A. FPGA Implementations Comparison of Neuro-cortical Inspired Convolution Processors for Spiking Systems. In *Bio-Inspired Systems: Computational and Ambient Intelligence*; Cabestany, J., Sandoval, F., Prieto, A., Corchado, J.M., Eds.; Springer-Verlag: Berlin, Germany, 2009; Volume 5517, pp. 97–105.
87. Paz-Vicente, R.; Cerezuela-Escudero, E.; Dominguez-Morales, M.; Jimenez-Fernandez, A.; Jimenez-Moreno, G. A perfomance comparison study between synchronous and asynchronous FPGA for spike based systems. In Proceedings of the International Symposium on Performance Evaluation of Computer & Telecommunication Systems, the Hague, The Netherlands, 27–30 June 2011; pp. 38–45.
88. Pérez, J.M.; Sánchez, P.; Martínez, M. Low-Cost Bayer to RGB Bilinear Interpolation with Hardware-Aware Median Filter. In Proceedings of the 16th IEEE International Conference on Electronics, Circuits, and Systems, Yasmine Hammamet, Tunisia, 13–16 December 2009; pp. 916–919.
89. Pauwels, K.; Tomasi, M.; Diaz, J.; Ros, E.; van Hulle, M. A Comparison of FPGA and GPU for Real-Time Phase-Based Optical Flow, Stereo, and Local Image Features. *IEEE Trans. Comput.* **2012**, *61*, 999–1012.

90. Gultekin, G.K.; Saranli, A. An FPGA based high performance optical flow hardware design for computer vision applications. *Microprocess. Microsyst.* **2013**, *37*, 270–286.
91. Browne, T.A.; Condell, J.V.; Prasad, G.; McGinnity, T.M. An Investigation into Optical Flow Computation on FPGA Hardware. In Proceedings of the IEEE International Machine Vision and Image Processing Conference, Coleraine, Northern Ireland, 3–5 September 2008; pp. 176–181.
92. Lucas, B.D.; Kanade, T. An iterative image registration technique with an application to stereo vision. In Proceedings of the Imaging Understanding Workshop, Washington, DC, USA, 6 April 1981; pp. 121–130.
93. Camacho, P.; Coslado, F.; Gonz áez, M.; Sandoval, F. Adaptive Multiresilution Imager Based on FPGAs. In Proceedings of the X European Signal Processing Conference, Tampere, Finland, 5–8 September 2000.
94. Coslado, F.J.; Camacho, P.; Gonz áez, M.; Sandoval, F. Hardware Architecture for Hierarchical Segmentation in Foveal Images. *Int. J. Imaging Syst. Technol.* **2004**, *14*, 153–166.
95. Ratha, N.K.; Karu, K.; Chen, S.Y.; Jain, A.K. Real-time matching system for large fingerprint databases. *IEEE Trans. Pattern Anal. Mach. Intell.* **1996**, *18*, 799–813.
96. Fons, M.; Fons, F.; Cant ó, E. Design of FPGA-based Hardware Accelerators for On-line Fingerprint Matcher Systems. In Proceedings of the Ph.D. Research in Microelectronics and Electronics, Otranto, Italy, 12–15 June 2006; pp. 333–336.
97. Volder, J.E. The CORDIC Trigonometric Computing Technique. *IRE Trans. Electron. Comput.* **1959**, *EC-8*, 330–334.
98. Matthies, L.; Maimone, M.; Johnson, A.; Cheng, Y.; Willson, R.; Villalpando, C.; Goldberg, S.; Huertas, A.; Stein, A.; Angelova, A. Computer Vision on Mars. *Int. J. Comput. Vis.* **2007**, *75*, 67–92.
99. Jeong, H. Real-time Stereo Vision FPGA Chip with Low Error Rate. In Proceedings of the IEEE International Conference on Multimedia and Ubiquitous Engineering, Seoul, Korea, 26–28 April 2007; pp. 751–756.
100. Barranco, F.; Diaz, J.; Gibaldi, A.; Sabatini, S.P.; Ros, E. Vector Disparity Sensor with Vergence Control for Active Vision Systems. *Sensors* **2012**, *12*, 1771–1799.
101. Sabatini, S.P.; Gastaldi, G.; Solari, F.; Pauwels, K.; Hulle, M.M.V.; Diaz, J.; Ros, E.; Pugeault, N.; Krger, N. A compact harmonic code for early vision based on anisotropic frequency channels. *Comput. Vis. Image Underst.* **2010**, *114*, 681–699.
102. Gil, A.; Guti érrez, R.; Alonso, J.L.; Ávila, S.F. Stereo Calculation of significant points using a FPGA. In Proceedings of the 2004 WSEAS International Conference on Electroscience and Technology for Naval and All-electric Ship, Athens, Greece, 12–15 July 2004.
103. Zabih, R.; Woodfill, J. Non-parametric Local Transforms for computing Visual Correspondence. In Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden, 2–6 May 1994; pp. 150–158.
104. Magdaleno, E.; L ike, J.P.; Rodr guez, M.; Rodr guez-Ramos, J.M. Design of Belief Propagation Based on FPGA for the Multistereo CAFADIS Camera. *Sensors* **2010**, *10*, 9194–9210.
105. P érez, J.; S ánchez, P.; Mart nez, M. High-Definition Belief-Propagation based Stereo Matching FPGA architecture. In Proceedings of the Conference on Design of Circuits and Integrated System, Zaragoza, Spain, 18–20 November 2009.

106. Bravo, I.; Mazo, M.; Lázaro, J.L.; Gardel, A.; Jiménez, P.; Pizarro, D. An Intelligent Architecture Based on Field Programmable Gate Arrays Designed to Detect Moving Objects by Using Principal Component Analysis. *Sensors* **2010**, *10*, 9232–9251.
107. Bravo, I. Arquitectura Basada en FPGAs Para la Detección de Objetos en Movimiento, Utilizando Visión Computacional y Técnicas PCA. Ph.D. Thesis, Universidad de Alcalá, Madrid, Spain, July 2007. (In Spanish)
108. Perez, C. Control de Robots Manipuladores Usando Información Visual: Aplicación a la Estimación del Movimiento del Objeto. Ph.D. Thesis, Universidad Miguel Hernández, Elche, Spain, April 2008. (In Spanish)
109. Waheed, A.; Thornberg, B.; Kumar, P. Comparison of Three Smart Camera Architectures for Real-Time Machine Vision System. *Int. J. Adv. Robot. Syst.* **2013**, *10*, 1–12.
110. Rodríguez-Ramos, L.F.; Viera, T.; Herrera, G.; Gigante, J.V.; Gago, F.; Alonso, Á. Testing FPGAs for real-time control of adaptive optics in giant telescopes. In Proceedings of the SPIE 6272, Adaptive Optics II, 62723X, Orlando, FL, USA, 24 May 2006.
111. Rodríguez-Ramos, L.F.; Díaz, J.J.; Piqueras, J.J.; Martin, Y.; Rodríguez-Ramos, J.M. FPGA–based slope computation for ELTs adaptive optics wavefront sensors. In Proceedings of the SPIE 7015, Adaptive Optics Systems, Marseille, France, 23 June 2008.
112. Martin, Y.; Rodríguez-Ramos, L.F.; Martín, Y.; Díaz, J.J.; Piqueras, J.; García-Jiménez, J.; Rodríguez-Ramos, J.M. FPGA-based real time processing of the Plenoptic Wavefront Sensor. In Proceedings of the VI Southern Programmable Logic Conference (SPL), Ipojuca, Brazil, 24–26 March 2010; pp. 87–92.
113. Rodríguez-Ramos, J.M.; Castelló, E.M.; Conde, C.D.; Valido, M.R.; Marichal-Hernández, J.G. 2D-FFT implementation on FPGA for wavefront phase recovery from the CAFADIS camera. In Proceedings of the SPIE 7015, Adaptive Optics Systems, Marseille, France, 23 June 2008.
114. Jin, S.; Kim, D.; Nguyen, T.T.; Kim, D.; Kim, M.; Jeon, J.W. Design and Implementation of a Pipelined Datapath for High-Speed Face Detection Using FPGA. *IEEE Trans. Ind. Inform.* **2012**, *8*, 158–167.
115. González, C.; Mozos, D.; Resano, J.; Plaza, A. FPGA Implementation of the N-FINDR Algorithm for Remotely Sensed Hyperspectral Image Analysis. *IEEE Trans. Geosci. Remote Sens.* **2012**, *50*, 374–388.
116. Winter, M.E. N-FINDR: An algorithm for fast autonomous spectral end-member determination in hyperspectral data. In Proceedings of the SPIE 3753, Imaging Spectrometry V, Denver, CO, USA, 18 July 1999.
117. Gonzalez, C.; Mozos, D.; Resano, J. FPGA support for satellite computations of hyper spectral images. In Proceedings of the International Conference on Field Programmable Logic and Applications, Prague, Czech Republic, 31 August–2 September 2009; pp. 715–716.
118. González, C.; Resano, J.; Mozos, D.; Plaza, A.; Valencia, D. FPGA Implementation of the Pixel Purity Index Algorithm for Remotely Sensed Hyperspectral Image Analysis. *EURASIP J. Adv. Signal Process.* **2010**, doi:10.1155/2010/969806.
119. González, C.; Sánchez, S.; Paz, A.; Resano, J.; Mozos, D.; Plaza, A. Use of FPGA or GPU-based architectures for remotely sensed hyperspectral image processing. *Integr. VLSI J.* **2013**, *46*, 89–103.

120. González, C.; Resano, J.; Plaza, A.; Member, S.; Mozos, D. FPGA Implementation of Abundance Estimation for Spectral Unmixing of Hyperspectral Data Using the Image Space Reconstruction Algorithm. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2012**, *5*, 248–261.
121. Vicente, A.G.; Muñoz, I.B.; Molina, P.J.; Lázaro-Galilea, J.L. Embedded Vision Modules for Tracking and Counting People. *IEEE Trans. Instrum. Meas.* **2009**, *58*, 3004–3011.
122. Sánchez, J.; Benet, G.; Simó, J.E. Video Sensor Architecture for Surveillance Applications. *Sensors* **2012**, *12*, 1509–1528.
123. Magdaleno, E.; Rodríguez, M.; Ayala, A.; Díaz, I. Codiseño hardware/software de un sensor inteligente para medidas de aceleraciones sobre 3 ejes basado en picoblaze. In Proceedings of the 8th Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica, Zaragoza, Spain, 3 July 2008. (In Spanish)
124. Sanchez, A.; Castro, A.; López, V.M.; Azcondo, F.J.; Garrido, J. Single ADC Digital PFC Controller Using Precalculated Duty Cycles. *IEEE Trans. Power Electron.* **2014**, *29*, 996–1005.