A Bottom-up Approach for Visual Object Recognition on FPGA based Embedded Multiprocessor Architecture

Hanen Chenini
Department of Industrial engineering, University of Sfax
Technopole of Sfax 3021 - BP 1164 - Sfax, Tunisia

***Abstract*—This paper presents an object recognition approach of outdoor autonomous systems identifying the nature of the interested object when observing an image. Therefore, seeking for effective and robust recognition method, the proposed approach is performed using a novel saliency based feature detector/descriptor which is combined with an object classifier to identify the nature of objects in an indoor or an outdoor environment. As known, bottom-up visual attention computational models need a considerable computational power and communication cost. A major challenge in this work is to deal with such image processing applications managing a large amount of the information processing and to work within real-time requirements by improving the processing speed.**

Based on interesting approach designing specific architectures for parallelism, this paper presents a solution for rapid prototyping of saliency-based object recognition applications. In order to meet computation and communication requirement, the developed pipelined architectures are composed of identical processing modules which can work concurrently with distributed memories and compute in parallel several sequential tasks with a high computational cost. We present hardware implementations with performance results on an Xilinx System-on-Programmable Chip (SoPC) target. The experimental results including execution times and application speedups as well as requirements in terms of computing resources show that the proposed homogeneous network of processors is efficient for embedding the proposed image processing application.

***Keywords*—Object recognition; Saliency-based feature detector/descriptor; Object classifier; Pipeline architecture; Coarse-grained model**

I. Introduction

Object recognition in autonomous systems (robots, vehicles, UAVs, etc.) is an important task in building a system that can sense, identify the nature of objects around it and afterward react according to this information (exploring unknown environments, obstacle avoiding, computing flight paths, etc.). Generally, the object recognition can be used as a preprocessing operation to classify objects in various applications such as video surveillance [1], Simultaneous Localisation and Mapping (SLAM) [2], Mission Planning [3] and Augmented Reality [4].

In this paper, we consider the problem of searching for only one object of a known class in an unknown environments. In order to search efficiently, the biologically inspired models has a remarkable ability to easily detect and recognise objects under the most complex conditions including variations in lighting, color, orientation or size. This work proposes a novel method for recognising objects based visual attention mechanism [5] to extract complex visual relationships between objects and their surroundings. Our object recognition method is therefore based on a saliency based visual attention approach [6] to distinguish a set of visually conspicuous regions that grab our attention from the rest of a given image without any prior knowledge on its content. In fact, the complete processing can be split up into two main steps: off-line stage and on-line stage as illustrated in Figure 1. During the off-line stage, for each object, a target attentional model is built to represent the characteristics of the interest object from a set of images containing instances of this object. Whereas, the on-line stage can then decide whether or not the instance of a target object is found in the input image. As illustrated in Figure 1, this stage is achieved by performing two main tasks: visual feature detection/description, (2) object classification including matching and comparison between the detected feature from the current image and those from the key image. First, the proposed visual detector/descriptor identifies salient regions in each new image from the video sequence and then describes each one. In order to apply saliency for object recognition, we need to obtain the saliency maps for three distinct features (color, intensity, and orientation). As a result, this method yields an output map containing only the regions that constitute the most salient regions. Furthermore, the feature descriptor then associates those regions with attentional models. In the classification task, attentional models of the input image are compared with the trained attentional model and the the current feature model giving maximum correspondence is considered the best match of the target object. To guide the attention to look for reference objects, each saliency model is classified as container or as non-container of each reference model by computing a dissimilarity score between each extracted model (current object model) and each target model (reference model) via a matching process. Based on dissimilarity scores, we can eliminate the salient regions that don't contain the target objects, and then the result can be used in segmenting the whole color image. Our approach for recognition yields encouraging results for finding a region of interest (ROI) with synthetic and natural input images. Translating our proposed algorithm for real time hardware implementation requires making specific choices so that the design meets the constraints. Some of the main constraints are speed of execution, power dissipation, recognition accuracy of the results. In fact, the image processing applications based saliency computations are naturally

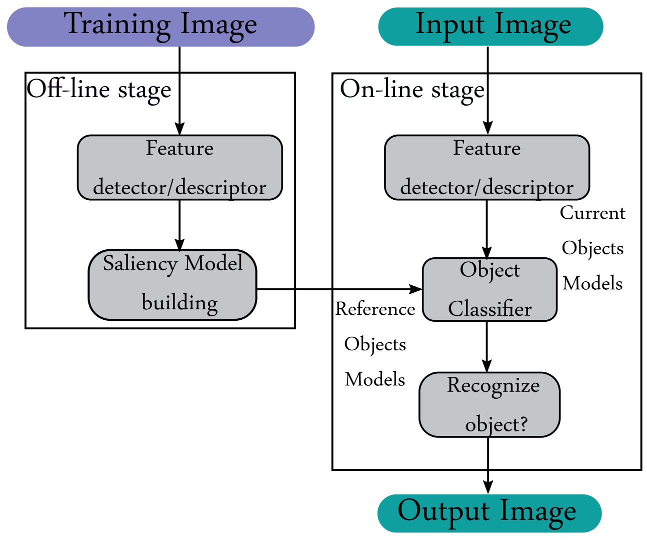

Fig. 1. An overview of the proposed object recognition.

distributed and decentralised since they are organised as a set of sequential pipeline composed by several computing units to process several features (color, intensity, orientation, etc.). Thus, we aim at using them in autonomous embedded systems thanks to their hardware parallel implementation. Therefore, in the second part of the paper, we map an object recognition algorithm that combines saliency detection with a classification method to our proposed coarse-grained architecture based parameterisable softcore microprocessor, extending from the previous work [7]. The proposed methodology might help the designer to rapidly obtain an efficient implementation of complex algorithms. Further, one of the main interest of this paper is to parallelise our proposed application efficiently in hardware, specifically for use in environments that have energy and power constraints. To provide a complete solution for parallel computing, embedding of a real time object recognition application on a dedicated architecture design must identify and exploit the parallelism and pipeline structures in algorithms to match the specific application requirements in term of the computing power and the communication bandwidth. Hence, the developed parallel architecture based homogeneous System-on-Chips (SoC) is comprised of a set of sequential pipeline layers with an embedded communication network to accelerate the software execution time. For the given application, we additionally propose new task parallel skeleton ”data flow skeleton” and its associated communication functions to exploit first the task level parallelism that exist in this application and then to be able to execute algorithms in parallel in hardware. The major findings of the experiment show our FPGA implementations of the saliency models retain a good performance in recognising problems.

The paper is organised as follows; We start in Section 2 with the related work. Section 3 details the algorithm used for saliency based object recognition. Thereafter, Section 4 describes the proposed homogeneous pipeline architectures based softcore processors in response to computational needs and real time performances required by multitasks real time applications and also presents the data flow skeleton for task scheduling of the given application in our pipeline architecture. Section 5 shows the results of the implementation of the proposed visual attention based object recognition into Xilinx FPGA following the proposed approach. The Section 6 concludes the paper and summarises the contributions of this work.

II. Previous Related Work

We confine the related work to biologically-inspired algorithms for object recognition in embedded hardware and real-time architecture. The hardware implementation of object recognition based saliency models in video streams has attracted a large number of research workers. A lot of researchers are interested in optimising hardware accelerators for biologically-inspired algorithms. More specifically, we are interested in object recognition based on bottom-up saliency models accelerated using Field Programmable Gate Arrays (FPGA) programmable devices. Recent work [8] presents a visual saliency model and its hardware real time architecture on FPGA platform to be embedded in a robotic system. Several HMAX (Hierarchical models) accelerators are presented in [9] [10] [11]. The main focus of these papers is to propose variety of purely hardware accelerators designs for some computationally intensive stages in the HMAX model [12]. Unfortunately, they must take into account several problems related especially to area and/or memory occupation dealing with low level hardware. For these reasons, it is desirable to improve performance by employing more powerful reconfigurable hardware accelerators. Thus, some proposals focus on developing an FPGA framework for an end-to-end attention and recognition system using saliency and HMAX accelerators [13] [14] [15]. Particular optimisation efforts have been proposed high performance hardware architectures for bottom-up spatio-temporal visual saliency models. For example, in [16], the authors have suggested a real time implementation of their proposed saliency based algorithm on a highly parallel Single Instruction Multiple Data (SIMD) architecture called ProtoEye, which consists of a 2D array of mixed analog-digital processing elements (PE). Recent efforts were presented in [17] [18], which propose a parallel implementation of this model with multi-GPU and multi-FPGA system reaching real time performance and good recognition accuracy.

Nevertheless, these proposed approaches can be considered, to the best of our knowledge, the first attempt to embed in a single chip a complete real-time visual saliency applications. However, there is no prior work on parallel implementation of saliency-based bottom-up visual attention model applied to visual object recognition tasks in many-core coarse-grained architecture based parameterisable softcore. The processing requirements of such applications can be fulfilled by performing parallel processing on a given image. Our work, extending from the previous work [7], presents the first parallel image processing architecture based parameterisable software and hardware modules. The overarching aim of this work is (1) the development of real-time object recognition in SoPC devices, attaining 94 frames per second (fps) Processing images with size of 256×256, and (2) task scheduling of the recognition algorithm in the proposed multistage architecture for maximum processing throughput.

III. Saliency for Object Recognition

In this section, we address the problem of recognising specific objects of interest from a database. We propose an efficient method for salient region recognition for online image processing.

A. *Off-line Stage*

In this stage, the aim of the work is to build a database of attentional models (Figure 2). An attentional model is based mainly on three components (coordinates within an image, size of the region of interest, and saliency values) associated with each target object has been proposed.

1) The Proposed Feature Detector/Descriptor: To resolve the problem of distinguishing the appearance of the target object under different viewing condition, the proposed feature detector is based on saliency computation method described later. The proposed detector tries to identify salient objects that capture our attention, by virtue of being different from the rest of the image.

When given an image, separate saliency maps are created for intensity, color and orientation at multiple scales in a bottom-up manner and then combined to obtain the final saliency map. In total, 10 feature maps (FM) are generated: 2 for intensity, 4 for color and 4 for orientation. These maps are summed up to 3 conspicuity maps (CM): C_I (intensity), C_O (orientation) and C_C (color) and combined to form the global saliency map SM. In SM, the salient regions SRs within a given image are determined.

2) Saliency Model: Based on the saliency features maps collected from the object, a distinctive model is built for each key object. The output of this stage is therefore several candidate attentional models of the target objects. The representative features of each target object is given by a vector V_{roi}, where its dimension is equal to $2+4+4=10$, denoted as :

$$V_{roi} = (u_i)^T \tag{1}$$

To estimate the contribution of each feature map to its associated conspicuity map, the vector component u_i (i from 0 to 10) is defined as the ratio of the mean saliency in SR for the feature map noted $m_{FM}(i)$ and the mean saliency for the corresponding conspicuity map $m_{CM}(i)$:

$$u_i = \frac{m_{FM}(i)}{m_{CM}(i)} \tag{2}$$

Closely related work was presented in [19], expect that the vector V_{roi} here is composed of 13 elements (10 FM and 3 CM of the VOCUS model) and also here u_i is defined the ratio of the mean saliency in SR to the mean background saliency. Then, the detected the salient region is then kept with its local neighbor and its coordinates in the reference image. With a small rectangular window around our region of interest, we consider that the attentional model of the target object based on its size and location is given by :

$$M_{roi} = \{X, Y, W, H, V_{roi}\} \tag{3}$$

where (X, Y) is the position of upper left-corner of the rectangle and W, H are the width and the height of the rectangle respectively.

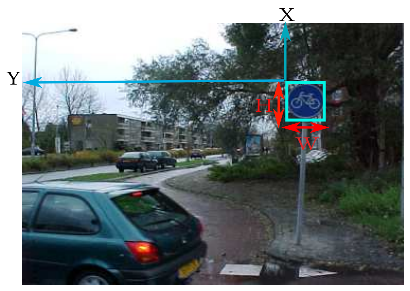

Fig. 2. The proposed attentional model

B. *On-line Stage*

The on-line stage allows to find specific known objects of interest in the input image and then the objects are recognised by comparing the extracted models with the candidate models built at the off-line stage.

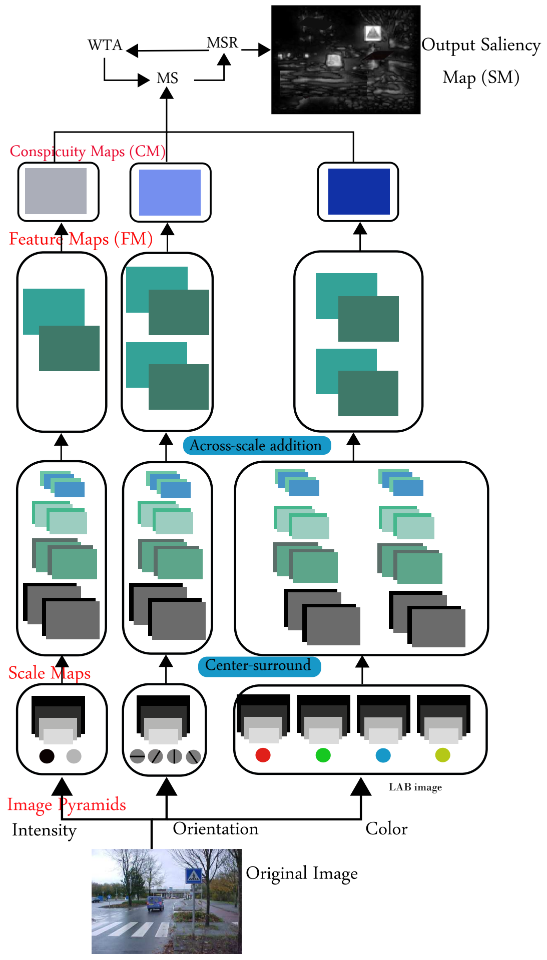

Fig. 3. The bottom-up attentional detector. Saliency maps of three feature channels (intensity, orientation and color) are computed independently and then combined.

1) Feature Detector/Descriptor: See Figure 3 for an overview of the proposed feature detector based bottom-up visual saliency, proposed by Itti et al. [20] and extended by Walther et al. [21]. From an input color image, our approach started by extracting feature maps on three spatial scales with image pyramids for distinct features type: intensity, orientation, and color. For intensity feature, the input image is converted into gray-scale. From the gray-scale image $s0$, a Gaussian image pyramid with three different scales ($s1$ to $s4$) is computed. When compared to the classical attentional models ([22], [23]), the proposed methodology compute separately the on-off and off-on contrasts for intensity feature. For the orientation feature, there are sub-channels which are computed to extract features specific to each orientation (0°; 45°; 90°; 135°). From the gray-scale image, the orientation feature maps are computed using Gabor filters. Finally, for the color feature, the input RGB image is converted into an LAB-image. From LAB-image, a color pyramid is generated for each color red, green, blue, and yellow.

As illustrated in Figure 3, the saliency detection algorithm relies mainly on the principle of center surround contrast and across scale addition. After the feature maps are computed, the scale maps are fused into one multi-resolution feature maps : 2 maps for the intensity feature, 4 multi-scale maps for the color feature (green, blue, red, yellow), and 4 maps for the orientation feature for each orientation. To combine the features maps, the feature maps are normalised to decrease the contribution of less important maps and the resulting maps is then computed as:

$$\overline{M} = \frac{1}{\alpha \times \sqrt{\beta}} \times M \tag{4}$$

where M indicates the map, $\alpha = max(M)$ and β is the number of local maxima that exceed a threshold equal to $max(\frac{M}{2})$. This formula is used for saliency maps by adding them pixel to pixel, the saliency maps is then deduced.

The resulting feature maps $\overline{M}_i$ are then grouped by type of elementary dimensions, and summed into 3 conspicuity maps: C_I (intensity), C_O (orientation) and C_C (color). Again, saliency maps are normalised and summed to form the bottom-up map $MS = \overline{C_I} + \overline{C_O} + \overline{C_C}$. The output image is the saliency map that shows a few region of interest. To determine the most salient location, we select the region with the highest saliency value in the saliency map SM, denoted as Δ_S. Afterward, the regions containing pixels whose average saliency S exceeds a certain threshold ($\frac{\Delta_S}{4}$ in our case) are chosen as salient regions (SRs). From the saliency map, the proposed algorithm iteratively selects salient region and adjusts their weights until identifying the most salient region (MSR). For each iteration, we select salient region with the highest saliency value. the mechanism of a winner-take-all (WTA) network of integrate-and-fire neurons is applied to determine the focus of attention in this map, as well as to implement the property of inhibition of return (IOR). Thus, the saliency in this region is inhibited and then the next SR that has a saliency greater than $\frac{\Delta_S}{4}$ is selected, and so on.

As final step, the processed output saliency map S is characterised by a set of SRs, which are generated by the proposed feature detector. Thus, for each image which contain D different SRs, we can build the global attentional model of the image as $M_{image} = (M_{candidate_m})_{m\in[1,D]} = (X_m, Y_m, W_m, H_m, V_{sr_m})_{m\in[1,D]}$, determining the position, size and the class of an object within an image.

2) Object Classifier: When given the output saliency map of the input image, this step aims to help users found the target objects they seek inside the scenes based on their saliency features. After the image features are extracted with their associated models, we want now to determine whether each current feature vector corresponds to an object found in the candidate models. Current attentional descriptors of the input image are matched with all reference attentional descriptors and then the current model which gives maximum correspondence is considered as the best match of the reference model. In order to do so, each current model is compared to the other reference models by calculating a dissimilarity score and models which are similar have a lower scores.

Processing images with single/multiple objects, varying in color, size and location combinations, SRs that are matched with the target object are those that minimise the distance between the vectors representing the current attentional models with each reference attentional model. In doing so, for each reference model, we compute first the difference of visual lightness $L_{m\in[1,D]}$ between two models based on the $L2$ distance :

$$L_m = \|V_{roi} - V_{sr_m}\|_2 \tag{5}$$

In this work, we are not interested in the salient regions that are fully contained within the image boundary. Consequently, we will consider only the regions that verify the following constraint: $D_{m\in[1,D]} = \|(x, y) - (x_m, y_m)\|_2 < max(D)$ where $max(D)$ denotes the maximum distance between two salient locations which is the diagonal distance of a given image.

As second step, SRs that are matched are those that minimize the difference of visual lightness between the vectors representing the attentional models. To comply with this condition, the similarity scores Sim_m between each current attentional model and a each known model stored in visual memory is defined as:

$$Sim_m = \frac{1}{\sqrt{(V_{roi} - V_{sr_m})^2}} \tag{6}$$

Once similarity scores are computed, we then proceed to find the global minimum and thus each current object model which have a higher similarity value than a specific threshold Th_{obj} represent a given reference object model. The value of Th_{obj} is generally adjusted by user to recognise particular objects.

IV. Parallel Object Recognition based Saliency Algorithm

The proposed application described above can be entirely implemented in a parallel manner. Based on high level MPSoC-methodology [24], this work presents a solution for rapid prototyping of this kind of algorithm based mainly on two essential concepts. The first concept consists of the derivation of a generic architecture based on a homogeneous pipeline architecture where each stage can start as soon it is finished and new data is available, while maintaining low power consumption with much higher throughput. The second one consists in the parallelisation of the sequential code on the different softcores performed using specific communication functions based on parallel skeleton concepts for task/data parallelism.

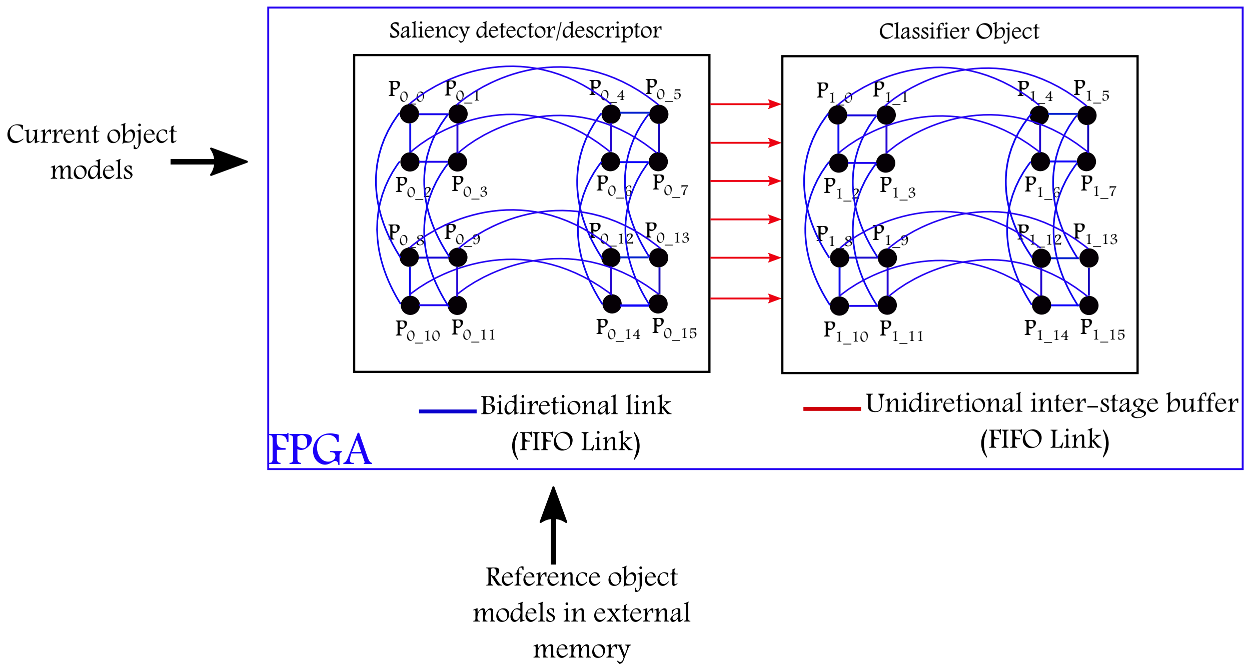
Current object models
Saliency detector/descriptor
Classifier Object
P_{0_0} P_{0_1} P_{0_4} P_{0_5}
P_{0_2} P_{0_3} P_{0_6} P_{0_7}
P_{0_8} P_{0_9} P_{0_12} P_{0_13}
P_{0_10} P_{0_11} P_{0_14} P_{0_15}
P_{1_0} P_{1_1} P_{1_4} P_{1_5}
P_{1_2} P_{1_3} P_{1_6} P_{1_7}
P_{1_8} P_{1_9} P_{1_12} P_{1_13}
P_{1_10} P_{1_11} P_{1_14} P_{1_15}
Bidiretional link (FIFO Link)
Unidiretional inter-stage buffer (FIFO Link)
FPGA
Reference object models in external memory

Fig. 4. Overview of the pipelined two stages architecture

In this work, a novel parameterisable system-on-chip architecture is proposed to handle image acquisition, distribution and processing to embedded multi-tasks applications. To efficiently utilise an increasing number of processing elements, we proceed by first proposing a novel parallel architecture based upon a pipeline of stages with parallel programming patterns and each stage then may exploit parallelism in the most appropriate way.

In order to be tailored to a given application, the proposed multicore design is a parameterisable architecture and thus offers a high degree of flexibility including network dimension of each stage, softcore parametrisation, memory size allocated to each processor, type of communication link, included special IPs for I/O (hand coded blocks to control incoming and outgoing video frames), image size, etc.

A. The Proposed Pipelined Architecture

The proposed architecture is shown in the following Figure 4. The interconnection network of the proposed embedded architecture is based mainly on point-to-point connections between nodes. Depending on the computational requirements of the final application, the proposed pipelined architecture comprises two parallel pipeline stages connected via direct point to point communication links. These parallel pipeline stages are independent and perform divers image-processing tasks and then each stage supplies a new output data to be processed by the next pipeline stage. A set of synchronisations links (FIFO links) allow parallel and pipeline connections between stages depending on the final application. Thus, these links are in charge of control and synchronisation of the different sub-tasks.

The initial step consists in seeking for salient regions in each image that would presumably contain the target objects and then those regions with their associated models will be transmitted over unidirectional signals to the second pipeline stage. In the classification task, current descriptors of each acquired image are matched with all trained objects models based on distance measures to decide whether or not the key objects are present in the current image. In this work, architectural choices were focused on Multiple-Instruction Multiple-Data (MIMD) architecture based on Xilinx's MicroBlaze with distributed memories. In this architecture, each computing node has its own copy of a program and works on different data streams. At any time, different processing nodes may be executing in parallel different pieces of data. The proposed distributed-memory system has an hypercube interconnection scheme.

1) First Stage of the Pipeline Architecture: As illustrated in Figure 5, the first stage of pipeline architecture relies on parallel homogeneous processing nodes. To increase the distribution and processing speed, the proposed "Input Frame Generator module" receives the input signals from the external memory and then transfers the original image to one or more "frame Grabber module" in order to distribute the data among different parallel computing nodes. Each processing node $Node_{0-i}$ (i from 0 to N) then process on an input sub-image supplied by the latter module via point to point connections (FSL links). Thanks to this parameterizable module, the local sub-image to be treated by each node is loaded in its corresponding local memory of each processor and in that case all nodes in this stage have access to the input image at the same time.

As shown in Figure 6, each processing node controls its own memory module. For this reason, $Node_{0-i}$ contains memory unit module, with local memory and frame memories used by the "Frame Grabber module" to store the selected sub-image. In fact, frame memories are used as swap memories when the i^{th} image is written in the frame memory 0, the

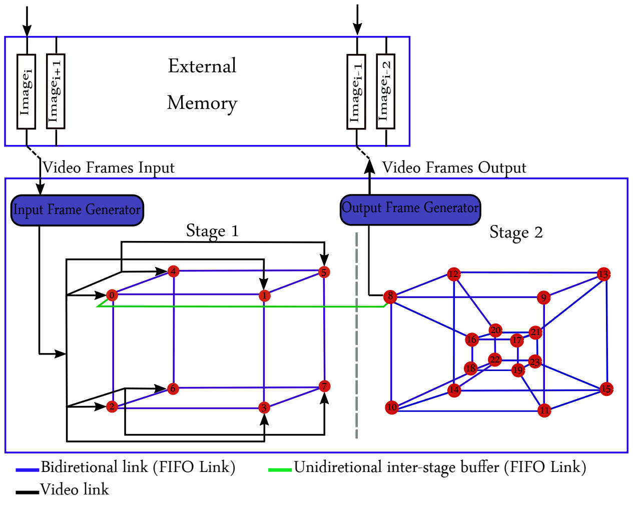

Fig. 5. MIMD-DM architecture based Frame Generators.

$Node_{0-i}$ processed the $(i-1)^{th}$ image in the frame memory 1. Once the data is partitioned, an attentional detection will be computed to highlight visually salient objects with their associated models in the current image. In fact, this step is the most time consuming stage in the sequential version. In its parallel implementation, the input image is first split into $N = 2^{D1}$ homogeneous elements of the same size where $D1$ is the Hypercube dimension of the first pipeline stage. Finally, the result of treatment (i.e. saliency map) is obtained by merging the computed of each elements and then proceed to send attentional models $(X_i, Y_i, W_i, H_i, S_{SR_i})_{i\in[1,D]}$ (D is the total number of detected salient regions in the current image) to the next pipeline stage to perform the complete processing chain.

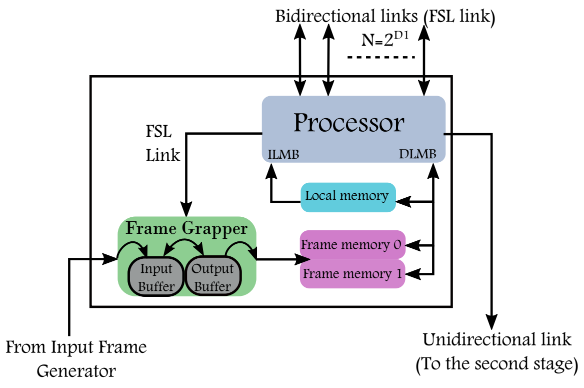

Fig. 6. Basic computing node of the first stage

2) *Second Stage of the Pipeline Architecture:* The second pipeline stage (MIMD distributed memory) as shown in Figure 5 is based on an homogeneous and parallel embedded architecture composed of $M = 2^{D2}$ nodes (where $D2$ is the Hypercube dimension of the network architecture) to satisfy the communication needs while the target system remains relatively inexpensive in term of FPGA occupation, memory size and power consumption, etc. Furthermore, communication between nodes is realised thanks to the well-known message passing communication model using bidirectional communication links (FIFO point to point link) for relatively low implementation costs. Without loss of generality, the basic computing $Node_{1-i}$ (i from 0 to M) is composed of the following modules: soft processor (MicroBlaze processor), with local memory for software program and data, and $D2$ FIFO links for the communication between two MicroBlazes. Only $Node_{0-0}$ in the first stage will communicate with the $Node_{1-0}$ in the second stage (only nodes with index 0 are connected through unidirectional link). Further, the $Node_{1-0}$ node sends the data from the previous stage to the other processors in the same pipeline stage. Once the processing of each node is done, the $Node_{1-0}$ node reaps the results of each node. As illustrated in Figure 5, the $Node_{1-0}$ is also connected to the "Output Frame Generator" module in order to control the output video flow. After the first pipeline stage completes its calculations to detect a set of salient regions in the current image and to describe them, the computations will be then continue for the next stage to classify more than one object at the same time. The target object model stored already in external memory is matched with the extracted attentional models of the current image and then the object model giving maximum correspondence is considered the best match. Finally, the video output is transferred straight away to the "Output Frame Generator" module.

B. *Data Flow Skeleton*

In this section, we are interested in partitioning and pipeline scheduling of the proposed algorithm in the developed pipelined architecture for maximum processing throughput. We aimed to develop parallel algorithms starting from applications composed of several independent parallel data with different degrees of complexity. Thus, to easily map the proposed application onto the multiprocessor system-on-chip, we have focused our attention to provide the parallel structure of the given application which naturally fits into a new developed Data flow skeleton. In a parallel implementation, we must define the parts of the given application that can be done concurrently. In this case, our application can be referring to two independent tasks running concurrently. Data flow skeleton defined as pipeline of skeletons is one of the best choice to exploit task level parallelism that exist in the proposed applications.

Using the data flow skeleton, the overall processing of the proposed application is split into a two of sequential tasks, each task is based on a SCM (Split, compute and merge) skeleton, with synchronisation step at the end of each step as illustrated in Figure 7. Thereafter, the parallel implementation scheme is based on data parallelism (images then lists of attentional models describing each salient region) between the available processors in each stage.

The input image is divided into subsets for parallel processing. The detection module (which is actually the attention mechanism) will concurrently run on different processing nodes of the first hypercube producing as output a list of the most salient regions found on this image. In practice, the split function implemented in each selected node in the first pipeline stage, allows to configure correctly the Frame Grabber module and then recover the subimage in real time from the input image. This process allows all the processing nodes to start the compute step at the same time. Actually, the input Frame

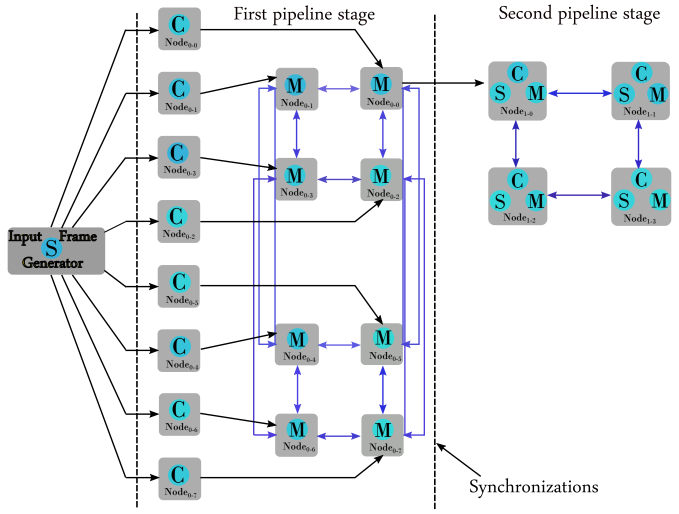

Fig. 7. The proposed data flow skeleton (An example of pipeline architecture with two stages composed of 8 nodes and 4 nodes respectively).

Generator module focuses particularly on distributing the data coming from external memory. Each node then execute the visual detection on the subimage selected. Once the compute step is finished, the result is sent to the node $Node_{0-0}$ through the Merge function.

V. Experimental Results

The experimental section is divided in to two parts. First, we perform experiments demonstrating the properties of our object recognition approach and second we provide experimental results of the pipelined architecture implemented on a Virtex6-LX760T FPGA and compare its performance with two existing HMAX accelerators specifically tailored to saliency based object recognition algorithm.

A. *Evaluation of the Proposed Recognition Method*

To evaluate the performance of the proposed system, we have conducted a large number of experiments on real image sequences. At 256×256 image resolution, we first applied our saliency detection for efficient identifying of bright regions in the input image under large variations on the appearance and shape of the desired object. The recognised object is set using the output of the saliency map obtained. The result of object recognition is shown in Figure 8. The value of Th_{obj} can be determined empirically by human.

Moreover, as we have mentioned before, the target object can be recognised accurately using the proposed algorithm regardless of the position, size . To discard undesired regions from the obtained binary image, a grayscale thresholding based method is applied wherein the recognised salient region is retained according to its coordinates and its size while the rest of image is removed.

B. *FPGA Prototyping Results*

In this section, we present the parallelisation and the embedding of the proposed object recognition algorithm on a SoPC platform. We made several experiments on multicore parallel implementation. Afterward, the performance of the multistage architecture in terms of SoPC resource consumption and the computation time is presented. The entire algorithm is partitioned into sequential tasks and then implemented on our proposed pipelined architecture. The two sequential tasks: (1) visual feature detector/descriptor and (2) object classifier are ported to the embedded architecture. Based on the proposed multi-processor approach, it is possible to implement various parallel FPGA designs in a single chip to investigate the impact of the increasing number of computing nodes on the system performance. The technologies we used to implement our architecture are Virtex FPGAs from Xilinx. The proposed soft multiprocessor is based on 32-bit RISC soft processor MicroBlaze.

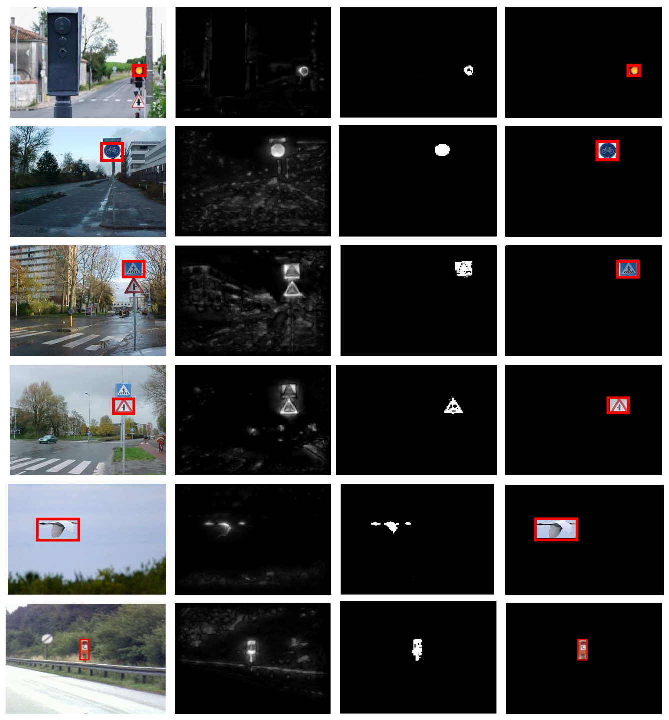

Fig. 8. From left to right: (1) Examples of color images with the target object (red squares), (2) The corresponding saliency map input, (3) The matched salient regions, (4) Recognition Results on 256×256 images.

TABLE I. Synthesis Summary for Point-to-point Based Network Targeted for a Virtex 6 ML 605 FPGA Device

(P_1, P_2)	Slice Registers	Slice LUTs	Block RAM/FIFO
(4,4)	16605 / 301440 (5%)	22545/ 150720 (14%)	142/ 416 (34%)
(8,4)	23414/ 301440 (7%)	37759/ 150720 (25%)	266/ 416 (63%)
(16,8)	50969/ 301440 (16%)	80437/ 150720 (53%)	343/ 416 (83%)
(32,8)	114547/ 301440 (18%)	102489/ 150720 (68%)	405 416 (97%)

According to the processing and communication requirements in our target application, we have created several multi-core architectures based on FSL point-to point links, by varying the number of processing nodes in each pipeline stage. Each node in the first stage has the following configuration: MicroBlaze processor with FPU unit, 32 Kb of local memory for software application and data storage, 32 Kb of frame memory for data of the current image. Whereas, our uniprocessor system in the second stage has the following configuration : MicroBlaze processor, 16 KB of local memory for program and data.

During our experiments, we present the FPGA hardware resource utilisation of various pipelined architectures to perform the two serial processing stages visual detector/descriptor followed by object classifier. The Table I presents the logic synthesis results in terms slice registers, slice LUTs and Block RAM/FIFO using bi-directional point-to-point communications considering different number of processing nodes in the complete system. FIFO depth is configured by 16 bytes. One can see in Table I that the proposed system has been tested with up to 40 processors in a Xilinx Virtex-6 LX760T device.

According to the FPGA implementation results, place and route results of the last network configurations lead to an area occupation of (19%) for Slice Registers, (97%) for RAM blocks and (68%) for Slice LUTs. This pipeline architecture based on FSL links, easily fits a Xilinx Virtex-6 LX760T FPGA. Our resources utilisation is fairly low, which represents 68% on all the resources available on the FPGA.

C. Timing Performance

Based on the experiment results, we have conducted various pipelined FPGA designs with several configurations choices that have direct effect on the processing time of the complete system. However, the computational cost for the calculation of the saliency map is the most time-consuming part of the complete recognition algorithm. We can see in the top Table II that with architecture composed of 40 processing nodes the parallel steps of the complete application executed in much less than $40ms$, leaving more time for execution of the whole algorithm (with the sequential parts) to process more than 25 frame/s (fps). Compared to serial computing, at a system frequency of $100Mhz$, we can see in the top Table II that with architecture composed of 40 processing nodes the parallel steps of the complete application executed in much less than $40ms$, leaving more time for execution of the whole algorithm (with the sequential parts) to process more than 25 frame/s (fps). Moreover, the processing time required for the classification step is more than the time needed to process the visual detection and description.

TABLE II. APPLICATION EXECUTION TIME (MS) (TOP), APPLICATION SPEEDUP (BOTTOM)

Nb of PNs (P_1,P_2)	(1,1)	(4,4)	(8,4)	(16,8)	(32,8)
Detection time	188.921	55.078	27.387	13.694	6.879
Matching time	35.656	10.185	10.185	6.931	6.931
Total time (ms)	188.921	55.078	27.387	13.694	6.931
Nb of PNs (P_1,P_2)	(1,1)	(4,4)	(8,4)	(16,8)	(32,8)
Detection Speed up	1.000	3.430	6.898	13.795	27.463
Matching Speed up	1.000	3.500	3.500	5.144	5.144
Total Speed up	1.000	3.430	6.898	13.795	27.257

Introducing the processing time of the 1^{st}stage (t_{stage_1}) and the 2^{sd}stage (t_{stage_2}) performing the first and the second parallel parts of the given algorithm, the total processing time of the complete design can be modeled by:$\max(t_{stage_1}, t_{stage_2})$. For each new input image, calculation of the output values takes T clock cycles (expressed in milliseconds). The calculation is pipelined: $T_{detection}$ clock cycles is used to extract attentional features from the current image, and T_{match} to match two images with attentional descriptors. When the calculation is finished, the time required to complete this phase is given by: $T = max(T_{detection}, T_{match})$. As result, for last configurations, our proposed method allows recognition calculation in approx. $7ms$ with a frame rate of $94.3fps$ for an image of size 256×256. Thus, we can applied this algorithm as a preprocessing for higher level vision algorithms.

It is then possible to calculate application speed-up from one solution to another depending on the number of processors implemented and run-time of the application. Example speedups is shown in bottom line of Table II with various degree of parallelisms (number of processing nodes) for 256×256 color images. We compute the speedup of the pipeline architecture as the ratio of the execution time $t_{seq}(1,1)$ needed by the sequential algorithm and the execution time $t_{par}(P_1, P_2)$ for the parallel algorithm: $Speed(P_1, P_2) = \frac{t_{seq}(1,1)}{t_{par}(P_1,P_2)} = \frac{t_{seq}(1,1)}{T}$. A speedup of 27 times has been achieved compared to the sequential implementation on a uniprocessor architecture. A very near to linear speed-up and a scalable architecture make it possible to match the processing power with the input image by adjusting the number of processor in each stage.

An advantage of the proposed approach is that the designer can use a set algorithmic skeletons to specify explicitly the communication of data between tasks suitable to be run efficiently on a parallel target architecture. To resolve the problem of efficient implementation of multi-tasks applications, staged computations are required to split the desired application in a number of independent pipeline stages. This can provide an increased performance while minimising execution time and minimising communication costs without affect the global processing time. As a final result, the proposed pipelined system coupled with task decomposition is able to classify objects in the input visual scene and to specify the tasks that can be executed concurrently without an important increase in resources requirements. Additionally, we provide a specific software skeleton suitable to be used to implement a pipeline algorithm.

TABLE III. COMPARISON BETWEEN OUR SALIENCY IMPLEMENTATION AND TWO HMAX MODELS FOR OBJECT RECOGNITION [25] [18].

Hardware	FPGA 2xVirtex6 SX475T[10]	FPGA 2xVirtex6 SX475T[18]	Our FPGA Virtex6 XC6VLX
Resolution	256×256	256×256	256×256
Frequency	100 MHz	100 MHz	100 MHz
Precision	Fixed-point	Fixed-point (24bit)	Floating-point
Computational time	21.81 ms	11.04 ms	6.931 ms

The Table III represents the speedups in execution time gained by our pipeline architecture and two existing HMAX accelerators implementations for 256×256 grayscale images [10] [18]. The initial design of the HMAX accelerator [10] takes about $21.81ms$ per image with a frame rate of 45.85 fps, whereas the second design [18] takes about $11.04ms$ per image with a frame rate of $90.57fps$. Our multi-processor architecture gave an overall speedups of $3.14X$ and $1.52X$ over the initial design and the second design, although it is mapped to a single FPGA only. In the proposed architecture, As seen in the above results, our improved designs is well suited for the object recognition based saliency computations compared to purely hardware implementation.

VI. Conclusion

This paper described a visual saliency based object recognition method, as well as a hardware architecture for pipelined processing, to allow for a more efficient implementation of pipelined embedded applications. This work investigates the contribution of the visual saliency computations for object recognition, and proposes a new saliency detector/descriptor to identify particular objects in unknown environments. Depending on the requirements of the targeted application, we go on to provide the necessary parallel software skeleton to resolve the communication overhead which is widely recognised as the principal obstacle for achieving large speedup using a large number of computing nodes. The results are encouraging and show the potential of the proposed approach to ensure real time processing of multitasks applications by balancing the computation requirement between the pipeline stages. The proposed parallel system was verified experimentally on a Virtex 6 FPGA. A significant speedup of the parallel pipelined architecture has been obtained. Specifically, the pipelined architecture was capable of processing 94 frames per second, demonstrating a $27X$ speedup compared to the original serial implementation.

Future works include implementation of more complex applications that will be embedded using this work to obtain real time neural systems. This will include also the development of the proposed parallel architecture, bringing other benefits such as support of arbitrary network topologies and allowing for dynamic reconfigurability to meet the targeted application requirements. This will allow to show another type of communication devices and parallel software skeletons.

References

[1] J. Wu and Z. Xiao, *Video surveillance object recognition based on shape and color features*, 2010 3rd International Congress on Image and Signal Processing, Yantai, 2010, pp. 451-454.

[2] H. Durrant-Whyte, and T. Bailey: *Simultaneous localization and mapping: Part I*. IEEE Robot. Autom. Mag. 13, 99108 (2006).

[3] M. MUSIAL, U.W. BRANDENBURG, and G. HOMMEL, *Cooperative autonomous mission planning and execution for the flying robot MARVIN*. In : Intelligent Autonomous Systems. 2000. p. 636-643.

[4] J.-Y. Didier, F. Ababsa, M. Mallem: *Hybrid camera pose estimation combining square fiducials localization technique and orthogonal iteration algorithm*. Int. J. Image Graph. 8(1), 169188 (2008).

[5] K.K. Evans, T.S. Horowitz, P. Howe, R. Pedersini, R. Ester, Y. Pinto,Y. Kuzmova, and J.M. Wolfe *Visual attention*. In Wiley Interdisciplinary Reviews: Cognitive Science, vol. 2, no. 5, pp.503-514, 2011.

[6] C. Siagian and L. Itti. *Rapid biologically-inspired scene classification using features shared with visual attention*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29(2):300 312, feb. 2007.

[7] F. Pelissier, H. Chenini, F. Berry, A. Landrault, J.P. Dérutin,*Embedded multi-processor system-on-programmable chip for smart camera pose estimation using nonlinear optimization methods*. J. Real-Time Image Processing 12(4): 663-679 (2016).

[8] F. Barranco, J. Diaz, B. Pino, E. Ros, *Real-time visual saliency architecture for FPGA with top-down attention modulation*, IEEE Trans. on Industrial Informatics, 10 (3), 1726-1735, 2014.

[9] M. DeBole, A. Maashri, M. Cotter, C.-L. Yu, C. Chakrabarti, and V. Narayanan. *A Framework for Accelerating Neuromorphic-Vision Algorithms on FPGAs*. In Computer-Aided Design (ICCAD), 2011. IEEE/ACM International Conference on, nov. 2011.

[10] J. Sabarad, S. Kestur, M. Park, D. Dantara, V. Narayanan, Y. Chen, and D. Khosla. *A Reconfigurable Accelerator for Neuromorphic Object Recognition*. In Proc. of Asia South Pacific Design Automation Conference ASPDAC12, Jan 2012.

[11] M. Park, S. Kestur, J. Sabarad, V. Narayanan, and M. Irwin. *An FPGA-based Accelerator for Cortical Object Classification*. In Proc. of Design Automation and Test Conference and Exhibition DATE12, Mar 2012.

[12] M. Riesenhuber and T. Poggio, *Hierarchical models of object recognition in cortex*, Nature Neuroscience, vol. 2, no. 11, pp. 1019-1025, November 1999.

[13] S. Kestur, M. Park, J. Sabarad, D. Dantara, V. Narayanan, Y. Chen, and D. Khosla. *Emulating Mammalian Vision on Reconfigurable Hardware*. In Intl. Symp. on Field Programmable Custom Computing Machines FCCM12, May 2012.

[14] S. Bae, Y. Cho, S. Park, K. M. Irick, Y. Jin, and V. Narayanan. *An FPGA implementation of information theoretic visual-saliency system and its optimization*. In Intl. Symp. on Field Programmable Custom Computing Machines, FCCM, pages 4148, 2011.

[15] A. Maashri, M. DeBole, M. Cotter, N. Chandramoorthy, Y. Xiao, V. Narayanan, and C. Chakrabarti. *Accelerating neuromorphic vision algorithms for recognition*. In Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE, pages 579 584, june 2012.

[16] N. Ouerhani, H. Hgli, P. Burgi, and P. Ruedi, *A real time implementation of the saliency-based model of visual attention on a SIMD architecture*, in Proc. 24th Symp. Pattern Recognit., 2002, pp. 282289.

[17] A. Rahman, D. Houzet, D. Pellerin, S. Marat, N. Guyader. *Parallel implementation of a spatio-temporal visual saliency model*. Journal of Real-Time Image Processing, Springer Verlag, 2010, 6 special issue (1), pp.3-14.

[18] M.S. Park, C. Zhang, M. DeBole, and S. Kestur. 2013. *Accelerators for biologically-inspired attention and recognition*. In of the 50th Annual Design Automation Conference (DAC '13). ACM, New York, NY, USA

[19] S. Frintrop. *VOCUS : A Visual Attention System for Object Detection and Goal Directed Search*, volume 3899 of Lecture Notesin Computer Science. Springer, 2006.

[20] L. Itti, C. Koch, and E. Niebur, *A Model of Saliency-based Visual Attention for Rapid Scene Analysis*, IEEE Tran. on Pattern Analysis and Machine Intelligence, vol. 20, no. 11, pp. 1254 1259, Nov. 1998.

[21] D. Walther and C. Koch, *Modeling Attention to Salient Proto-objects*, Neural Networks, vol. 19, no. 9, pp. 1395 1407, 2006.

[22] M. Weber, M. Welling, and P. Perona. *Towards automatic discovery of object categories*. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2000.

[23] L. Itti, C. Koch, and E. Niebur. *A model of saliency based visual attention for rapid scene analysis*. IEEE Trans. Pattern Anal. Mach. Intell, pages 1254-1259, November 1998.

[24] H. Chenini, J.P. Derutin, R. Aufrere, R. Chapuis: *Parallel Embedded Processor Architecture for FPGA Based Image Processing using Parallel Software Skeletons*. EURASIP J. Adv. Signal Process (2013).

[25] R.J. Peters and L. Itti. *Applying computational tools to predict gaze direction in interactive visual environments*. ACM Transactions on Applied Perception, 5(2), Article 8, 2008.