High Performance Embedded System for Real-Time Pattern Matching

C.-L. Sotiropoulou[a,b,*], P. Luciano[c,b], S. Gkaitatzis[d], S. Citraro[a,b], P. Giannetti[b], M. Dell'Orso[a,b]

[a]*University of Pisa, Largo B. Pontecorvo 3, 56127, Pisa, Italy*
[b]*INFN-Pisa Section, Largo B. Pontecorvo 3, 56127, Pisa, Italy*
[c]*University of Cassino and Southern Lazio, Gaetano di Biasio 43, Cassino, 03043, Italy*
[d]*Aristotle University of Thessaloniki, 54124, Thessaloniki, Greece*

Abstract

In this paper we present an innovative and high performance embedded system for real-time pattern matching. This system is based on the evolution of hardware and algorithms developed for the field of High Energy Physics and more specifically for the execution of extremely fast pattern matching for tracking of particles produced by proton-proton collisions in hadron collider experiments. A miniaturised version of this complex system is being developed for pattern matching in generic image processing applications. The system works as a contour identifier able to extract the salient features of an image. It is based on the principles of cognitive image processing, which means that it executes fast pattern matching and data reduction mimicking the operation of the human brain. The pattern matching can be executed by a custom designed Associative Memory chip. The reference patterns are chosen by a complex training algorithm implemented on an FPGA device. Post processing algorithms (e.g. pixel clustering) are also implemented on the FPGA. The pattern matching can be executed on a 2D or 3D space, on black and white or grayscale images, depending on the application and thus increasing exponentially the processing requirements of the system. We present the firmware implementation of the training and pattern matching algorithm, performance and results on a latest generation Xilinx Kintex Ultrascale FPGA device.

Keywords: Pattern Matching, Image Processing, Image Filtering, Edge Detection

1. Introduction

In recent years the demands for fast, real-time image processing of massive data volumes have grown significantly. The detector technologies are continuously improving, resulting in higher resolution images and video. The introduction of 3D imaging has also increased the amount of produced data as well as the processing complication. 3D image processing is especially useful in biomedical applications such as PET and MRI. Processing this type and size of data faster requires a clever and efficient pre-processing step that executes filtering and data reduction while maintaining the information that is useful to the next processing steps.

This approach is very similar to the one used for the trigger systems in High Energy Physics (HEP) experiments. In these trigger systems real-time filtering is executed on the massive amounts of data produced by the detectors to select those that contain higher probability for studying and identifying "new" physics.

2. High Energy Physics Background

We have built an Associative Memory (AM) system for the Fast Tracker (FTK) processor [1], a recently approved upgrade for the ATLAS [2] trigger. FTK is a high-performance embedded system based on the combination of two innovative technologies: powerful and flexible FPGAs working with standard-cell ASICs. These ASICs are the Associative Memory (AM) chips [3] that are used to execute the pattern matching algorithm with utmost gate integration density and maximum performance.

The most interesting processes generated at LHC experiments are very rare and hidden in an extremely high level of background. Implementing the most efficient selections in real-time (trigger) is therefore essential to fully exploit the physics potential of experiments where only a very limited fraction of the produced data can be recorded. This is a specific case of Big Data problem whose solution is based on the organization of the trigger system in different levels of selections. At the lower levels, dedicated hardware with a high degree of parallelization is exploited for an extremely efficient preprocessing step.

3. Cognitive Image Processing

Studies have demonstrated that the most convincing models about brain functioning hypotheses are extremely similar to the real time architectures developed for high energy physics [4] and more specifically the trigger organization used in HEP experiments. This led to the idea of exploiting hardware and algorithms that derive from HEP triggers to perform high performance image processing.

[*]Corresponding author
Email address: `c.sotiropoulou@cern.ch` (C.-L. Sotiropoulou)

The embedded system presented in this paper can accelerate neurophysiological studies of the brain. A multilevel model is proposed as appropriate to describe the brain image processing. The method proposed by Del Viva et al. [4] introduces the idea that the brain, when the human subject is under stress or in danger, works by dramatically reducing input information by selecting for higher-level processing and long-term storage only input data that match a particular set of memorized patterns. A double constraint of finite computing power and finite output bandwidth is imposed, very similar to the constraints used in embedded systems design for memory size and data throughput. This double constraint determines the type of information that is meaningful or relevant for following processing steps and becomes part of higher level processing and longer-term memory.

The AM pattern matching process has demonstrated to be able to play a key role in high rate filtering/data-reduction tasks. Simulations [4] have shown the potential of the pattern matching algorithm on static 2D images. Since the needed computational time causes serious limits to the capability to extend these studies to 3D images and movies, we are developing an implementation that can use the AM system based on the AM chip [5] for a real-time pattern selection/filtering of the same type that is studied in these models of human vision. These studies could have an impact in the area of medical imaging for real-time diagnosis or any area where pattern matching is relevant and computing performance is a limiting factor.

Figure 1 shows the results of the simulations of the model described in [4] where pattern matching with the preselected relevant patterns is used to filter the main features of the image. The pictures on the right (b, c) show the quality of the filtered images. The butterfly can be clearly recognized even if the image information is reduced at the level of 10% or less of the original content. The associative memory system works as an edge detector implementation able to extract the salient features of the image.

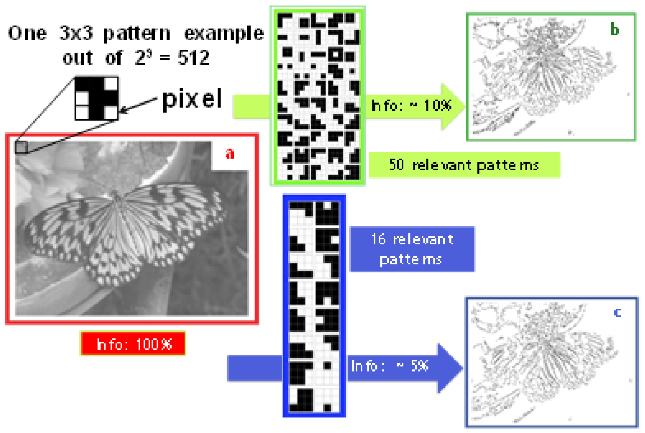

Figure 1: Impact of selected patterns in the output image.

3.1. *Pattern Matching Process*

The pattern is defined as the collection of pixels contained in a 3×3 pixel square, as shown above the butterfly image (a) in Figure 1. Each square is converted in a 9 bits sequence (each bit is '1' for a black pixel and '0' for a white one for B/W processing) or an 18 bits sequence in case of 4 levels of grey (each pixel encoded with 2 bits). The bit sequence is used to identify the pattern. Starting from the left top corner the image is scanned by the 3×3 square moved in steps of one pixel toward the right. When the row is finished, the square is moved one pixel down to scan again the raw from the left to the right. Each pattern detected in the figure during the scan is compared to the set of relevant patterns predefined by a training phase. If there is a match the pattern is saved in the same pixel coordinates. If there is no match the pattern is rejected and the pixel data are filtered. Figure 1 shows two collections of relevant patterns for two different selections. The 16 patterns in the blue box produce a larger image compression than the 50 patterns in the green box. The smaller is the set of chosen patterns the stronger the information reduction that is achieved in the end.

Analysing images with 4 or 8 levels of grey, or using 3D images, increases the number of relevant patterns. The pattern in the 3D case is not a square, but a cube of pixels: a set of three 3×3 squares taken from 3 subsequent frames. Each pattern for B/W is made of 27 bits corresponding to 2^{27} possible patterns. If 4 levels of grey are used the total number of patterns becomes 2^{54}.

3.2. *Training Process*

The training process is executed to select the "relevant patterns". A suitable sample of images need to be processed in order to obtain satisfactory results. The image sample must be large enough (e.g. more than 1000 images for the B/W case) and from the same source as the targeted application. Then the information theory entropy ([4],[6]) of each pattern is calculated. The "relevant patterns" are the ones that their probability of appearance in the sample images correspond to the entropy maximum (maximises the useful information content of the pattern).

4. Hardware Implementation

Our parallel processing architecture is based on hardware currently used in HEP, in the FTK processor [7]. FTK reconstructs the events produced at the ATLAS detector, that can be described as extremely complex images, in few tens of microseconds. It therefore is extremely suitable to tackle Big Data problems. The key role in the novel technology is played by the custom multicore AM system that will intensively be used to filter out the relevant information of the data to be further processed by Field Programmable Gate Arrays (FPGAs) executing higher level algorithms. The AM implements maximised parallelism and offers the best timing performances, since it solves its task in the time data are loaded on it. Additionally our architecture benefits from the FPGA computing power. The FPGA complements the AM task with its flexibility and reconfigurability, adapting its logic to perform any necessary refining post-processing required of the AM output (such as segmentation, clustering [8] or volume and size calculation etc.). In addition the AM has a specific architecture (the pattern matching can be successful on a subset of the tested data) that helps to identify not only perfect features, but also partial, or noisy versions of them.

The implementation is divided in two main parts, like the algorithm: the training phase and the real-time pattern recognition phase, what is referred as the data taking phase. Most of the functions are executed by the FPGA with the only exception of the data taking, that is executed by the AM under the FPGA control. We have estimated the processing latency for the data taking exploiting the long AM experience accumulated in FTK [?]. For the training, instead, we implemented the logic on a Xilinx Kintex Ultrascale XCKU040 of a KCU105 evaluation board, easily connectible to an external PC (or a video camera) and to a set of AM chips [5]. The implementation setup can be seen in Figure 2. In this figure the Xilinx Kintex Ultrascale KCU105 Evaluation Board can be seen. A mezzanine with devices performing pattern matching can be connected to the large connectors on the top of the board. The mezzanine organization offers flexibility: it allows using the system in different configurations and with different AM chip versions (parallel or serial I/O, different pattern densities). We have evaluated the training timing performance directly on the new hardware.

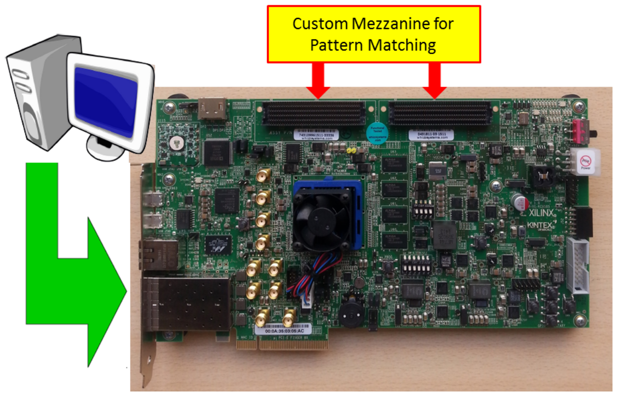

Figure 2: Hardware setup (FPGA Development board with AM mezzanine)

The Training Phase is subdivided in the following steps:

1. Calculation of the pattern appearance frequencies: The embedded system receives the image bit-streams (e.g., data from a PC or a video camera). The FPGA partitions/reorganizes the input data into the small 3×3 pixel patterns. Then, for each possible pattern, the FPGA calculates the occurrence frequency in the processed images/frames, using a large set of training images, to measure the frequencies with precision. When the environment and the lighting conditions change, the training has to be repeated in order to identify the relative patterns set suitable for the new environment. Therefore a continuous real-time training execution is required to allow the device adapt itself autonomously to the different conditions of the images that it observes.
2. Pattern selection: the system must decide which set of patterns is relevant, to be selected for memory storage and later use. We adopt the hypothesis described in [4] to maximize the capability to recognize shapes, i.e., maximum entropy is a measure of optimization. The set of patterns that produces the largest amount of entropy allowed by system limitations (size of the memory to store patterns and output bandwidth) is the best set of relevant patterns. In [4] are described the details of the selection. The selected patterns have to be written inside the AM bank for the following data taking phase.

We have implemented the training for 2D B/W images. The block diagram of the implementation is presented in Figure 3.

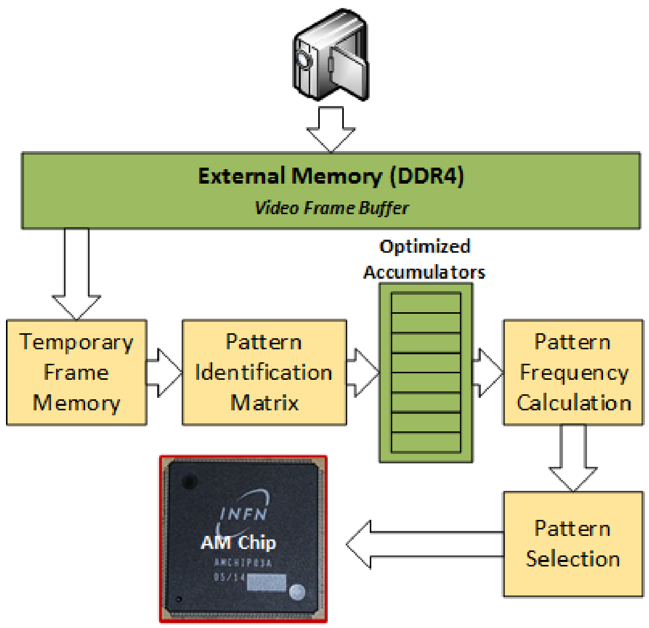

Figure 3: Training implementation block diagram

The FPGA needs to perform training in real-time for demanding streaming video applications. Several optimization techniques are used to achieve the best performance possible in the hardware implementation. The video frames are stored in the external memory before being transferred in an internal frame buffer. As soon as enough data has been transferred for the 3×3 patterns to be formed, a pattern identification matrix begins to be loaded. It identifies and propagates two patterns per clock cycle to the pattern accumulators. The accumulators are specifically designed to facilitate successive accumulation in the same memory location (fall through data logic). As soon as the whole image sample has been read, the pattern frequency is calculated by taking advantage the FPGA DSP slices. The architecture is generic and parametric to allow easier adaptation for the implementations of the more complex 3D and 4 levels of grey cases.

The calculation of the probability of appearance of each pattern is executed by using a radix-2 divider implemented by an FPGA DSP slice. The calculation of the entropy values for each pattern is optimized by using specifically calculated Look-Up-Tables (LUTs). The LUT values are loaded in the implementation during the initialization phase. The LUTs are not ”sample specific”, therefore their values do not depend on the training

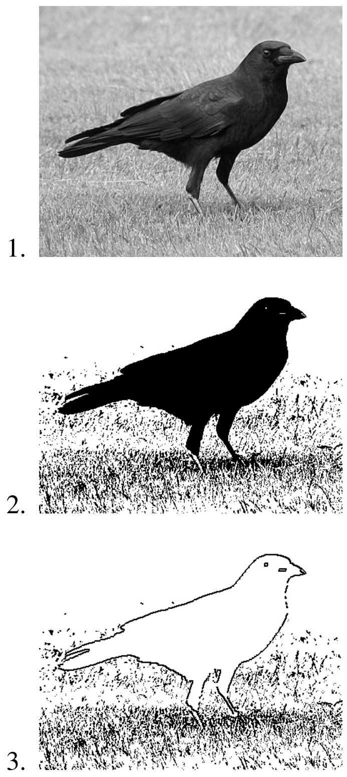

Figure 4: Crow Example

sample of images used. What can vary is the required calculation precision that can change slightly the final pattern selection

The values of the LUTs are calculated by a simulation framework developed to verify the operation of our system. The frameworks operation is threefold: a) It is used as a bit accurate simulation to verify the hardware operation. b) It is used as a benchmark to compare performance between the proposed hardware system and general CPUs. c) It is used to explore the algorithm's potential and confirm the validity of the algorithmic assumptions, before the hardware implementations of the post-processing steps. In Figure 4.1 we can see a 512 × 512 pixel grey scale image. In Figure 4.2 the image is transformed to a black and white binary image of the same size. In Figure 4.3 the output of the simulation framework can be observed by using 50 ”relevant patterns”.

5. Results

The training phase is implemented and tested for 2D B/W images. We use a set of sample images to verify the system operation. The complete training implementation requires less than 2% of the FPGA LUTs and 2.4% of the available BRAMs for processing of black and white images. We use a 250 MHz clock. To process a 512×512 pixel image less than 2.5 ms are required. A third generation i5 processor with 4 Gb RAM requires instead more than 3 s to execute the training algorithm for an image of the same size (performance comparison using the simulation framework).

For the data taking phase we measured 100 ns for each pattern check with the bank where relevant patterns are stored, using the board presented in [**?**]. With future AM versions, not using the serialized I/O that introduces some latency on data transmission, this latency/pattern will be even lower. This time is independent from the number of relevant patterns stored in the AM chip, since the AM chip performs all the comparisons of stored patterns with incoming data in parallel. So the advantage compared to standard CPUs that execute sequentially the comparisons increases enormously when pattern matching is extended on 3D images, especially if 4 levels of grey are used. Our future goal is the exploration of the architecture and pattern bank size needed to store the relevant patterns for a 3D four levels of grey case, where the total amount of patterns is 2^{54}.

6. Conclusions

As image processing requirements for fast and efficient filtering of massive high resolution input data are continuously increasing, we present a high performance embedded system that can execute real-time pattern matching and answer these demands. The system is based on hardware and algorithms developed for the trigger systems of HEP experiments. It is flexible and can be expanded to be used for processing of 3D grey scale input data. A smart training algorithm is used to select the useful patterns, based on cognitive image processing. A current implementation of the training algorithm for 2D black and white image processing can process 512 × 512 pixel images more than 1000 times faster than a normal CPU.

7. References

[1] ATLAS Collaboration, The Fast Tracker (FTK) Techinical Design Report CERN-LHCC-2013-007; ATLAS-TDR-021; available online: https://cds.cern.ch/record/1552953

[2] ATLAS Collaboration, The ATLAS Experiment at the CERN Large Hadron Collider, Journal of Instrumentation 3 S08003, 2008.

[3] M. Dell'Orso and L. Ristori, ”VLSI Structures Track Finding”, Nucl. Instr. and Meth. A, vol. 278, pp. 436-440, 1989.

[4] M. Del Viva et. al Information and Perception of Meaningful Patters, PLoS one 8.7 (2013): e69154.

[5] A. Annovi, et al., VLSI Processor for Fast Track Finding Based on Content Addressable Memories, in IEEE Trans. on Nuclear Science, vol. 53, no. 4, pp. 2428-2433, August 2006.

[6] M. Borda, ”Fundamentals in Information Theory and Coding”, Springer, 2011, ISBN: 978-3-642-20347-3.

[7] A. Andreani, et al, The FastTracker Real-Time Processor and Its Impact on Muon Isolation, Tau and b-Jet Online Selections at ATLAS, IEEE Trans. on Nuclear Science, vol.59, no.2, pp. 348-357, April 2012.

[8] C.-L. Sotiropoulou et. al A Multi-Core FPGA-based 2D-Clustering Implementation for Real-Time Image Processing, in IEEE Trans. on Nuclear Science, vol. 61, no. 6, pp. 3599 - 3606, December 2014.