High-speed image processing with AER-based components

R.Serrano-Gotarredona[1], B.Linares-Barranco[1], T.Serrano-Gotarredona[1], A.J.Acosta-Jiménez[1], A.Linares-Barranco[2], R.Paz-Vicente, F. Gómez-Rodríguez, G.Jiménez-Moreno[2] and A.Civit-Ballcels[2].

[1]Instituto de Microelectrónica de Sevilla (IMSE-CNM-CSIC). Ed. CICA. Av. Reina Mercedes s/n, 41012, Sevilla (SPAIN). Phone: +34955056666, Fax: +34955056686, E-mail: bernabe@imse.cnm.es.

[2]Dpto. Arquitectura y tecnología de computadores. ETSI Informática, Universidad de Sevilla. Av. Reina Mercedes s/n 41012, Sevilla (SPAIN)

Abstract

A high speed sample image processing application using AER-based components is presented. The setup objective is to distinguish between two propellers of different shape rotating at high speed (around 1000 revolutions /sec) to show event-based systems capabilities in high speed applications. Event-based schemes allow the most relevant information to propagate faster through the system layers. So image processing is sped up because a rough result may be available when only a little part of the input has arrived. This setup is much faster than the conventional frame-based image processing systems because they would need to proccess more than 10kFrames/s to do the same task proposed here, whereas only few events are required with the event based technique.

1. Introduction

Image processing is one of the most demanding tasks to be implemented in a computer. To perform, for instance, a single convolution between a 64x64 input image with a 15x15 kernel more than 900 k-MAC's are required using a conventional algorithm. Many researchers are working in improving both the algorithms and the system architectures used [1]-[4]. The problem with image processing is that it is a mainly parallel task while the conventional algorithms and also architectures are intended for performing sequential processing such as mathematical calculus, for example. The major part of the advances in image processing tend to design architectures or algorithms which make a parallel processing. This idea comes from the observation of nature: a human being cannot perform mathematical calculus as fast as a PC, but is further more efficient than it processing images. What is the main cause of that? Essentially, the parallel architecture of the brain. So, maybe, if we are able to design artificial system

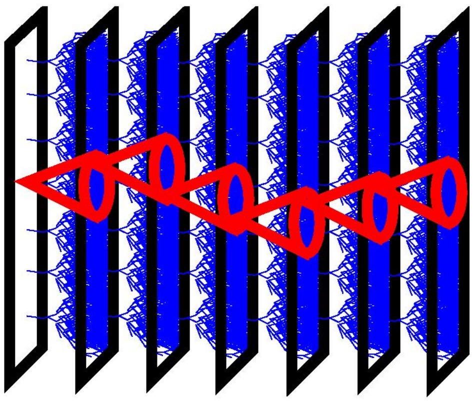

Fig. 1: Brain architecture scheme

whose architecture is inspired in the human brain, *bio-inspired systems*, the gap in performance between them will be reduced.

Which are the main features of the brain architecture?

- It is composed of many processing units, the neurons.
- The massive interconnectivity between neurons.
- Pulsed coding of neuron outputs.

In Fig. 1 a simplified scheme of the brain architecture is shown. It consists of several neuron layers, no more than 10. The neurons are low precision and low speed processing units but their massive interconnectivity makes the processing capability of the brain be much higher than the sum of its components. Neurons in each layer have a receptive field (RF) in the next layer composed of the neurons they are connected to (it is remarked in red in Fig. 1). The pattern of input connections of two neurons trained to detect the same feature is similar. Then passing from one neuron layer to the next could be understood as something similar to perform a convolution. As well, the coding of the output is based on pulses. Several theories about how the information is coded in the pulses are supported [5]-[6]. The information can be coded either in the pulse rate, in their latency, in their simultaneity, etc.

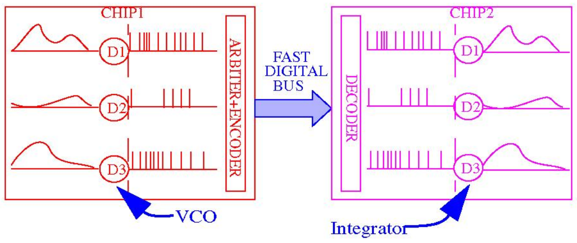

Fig. 2: AER based systems. Point to point communication

But anyhow, it is clear the neuron whose inputs are more intense sends a pulse first and, consequently, the most relevant information propagates first through the successive neuron layers. That way a very fast though not very precise processing can be made just by having into account the first information in arriving, which is the most relevant one.

In the convolution system used here, we have tried to reproduce the brain architecture features to perform convolutions "on the fly". This system is suitable to implement in hardware bio-inspired algorithms such as BCS-FCS [7], which is really computationally costly when running in a PC. To show high speed image processing capability, we present a setup showing two propellers with different shape rotating at high speed as input of the system. It is programmed to detect one of the shapes and not the other and it can distinguish between them even if they are rotating at 1kHz. In the next section, we explain how the system works based on AER (Address Event Representation), an emergent protocol which allows to communicate neuron ensembles whose outputs are coded into pulses. The section 3 explains the setup used in the demo, in the section 4, some results are shown and, finally, some conclusions are commented in the section 5.

2. System Description

The system architecture is based on AER communication. The AER inter-chip communication protocol was first proposed by Sivilotti [8] to reproduce the state of a 2D array of neurons from one emitter chip onto another receiver, continuously and in real time. Fig. 2 shows a basic AER system. The emitter chip contains an array of neurons, each of them generates pulses at a rate according to its activation level. Those pulses are encoded and arbitrated out of the chip through a fast digital bus. Therefore, a neuron address will appear more frequently in the bus as more active is that neuron. The communication between chips is made with a classic four phase handshaking protocol: every time an address is transmitted we say an *"Address Event"* (AE) is sent. The receiver chip takes the AE's from

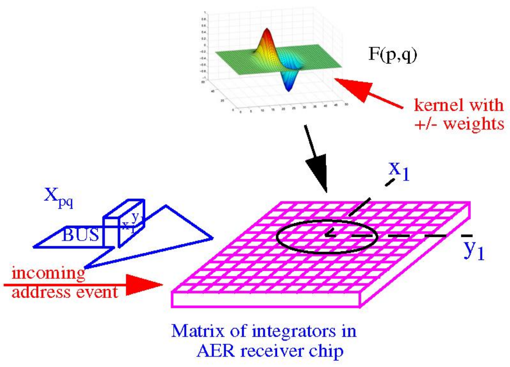

Fig. 3: AER pulses weighting scheme

the bus, decodes the addresses, and sends the pulses to the corresponding neuron to integrate those pulses. That way the receiver neurons can reconstruct the emitter neurons activation state.

At the receiver side, pulses may not be sent only to one neuron but also to a set of them: a *projective field.* Also, those pulses may be weighted depending on the position the receiver neuron occupies in the projective field (see Fig. 3). That way, a convolution will be made at the same time pulses arrive at chip input just by weighting the incoming events with the coefficients of a particular kernel matrix and sending them to the area surrounding the address coded in the AE. Thus, a neuron will integrate pulses coming from its whole neighborhood but weighted by the elements of the programmed kernel matrix. Then, the charge in a particular neuron integration capacitor will be proportional to the value of the convolution between the kernel programmed and the image coded in the incoming AE's. Thus convolutions are performed "on the fly".

How does our system work? An AE-stream feed a 32x32 pixels convolution chip input. That chip, which is object of a forthcoming article (some of its building blocks have been reported in [9]-[10]), is able to perform event-based convolutions, in the way explained in the previous paragraph, with programmable kernels of arbitrary shape and with up to 31x31 values. In this case, the input AE-stream represents two rotating propellers of different shapes. One of the propellers is straight and the other one is curved as shown in Fig. 4.(a). The kernel programmed in the convolution chip has the shape of one of the propellers when it is in horizontal position (Fig. 4.(b)). So, the convolution chip performs a pattern matching task. When the propeller which looks like the kernel is in horizontal position, the chip will output spikes around the position the center of the propeller is. The propeller whose shape is not the adequate will produce much less output spikes and we will be able to distinguish which is the selected

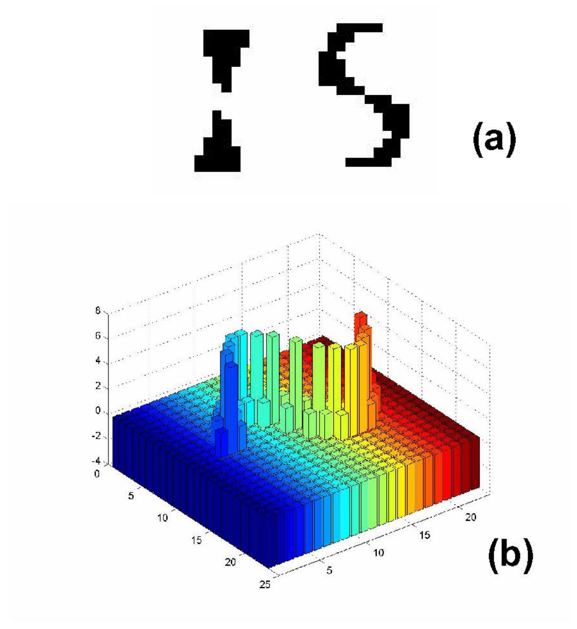

Fig. 4: (a) The two propellers used. (b) Convolution kernel programmed to distinguish between them

propeller although they are rotating so fast that at the input we just see two moving circles. Therefore, the recognition can be made although the propeller is moving or rotating at high speed. We have evidences that says the system can work when the propeller rotates around 1kHz. We used 48x48 input images and 25x25 kernels. larger images can be processed using this system just by tiling several convolution chips in a matrix fashion, since they are intended to do that. Also larger kernels (up 31x31 values) can be programmed on the chips.

3. Demo setup

A scheme representing the system block diagram is shown in Fig. 5. We use a PC to feed the system input and another one to receive its output. The PC's are equipped with a PCI-AER board [11]-[12]. That board receives the list of the AE's we want to send and the times to be sent and sequences the events out satisfying the AER-protocol (four phase handshaking). The board has also an AER-input port. It receives events and make them available for any application running on the PC. In the PC's, we use Matlab both to produce the input events and to receive the output.

The input AE-stream is sent to a splitter board. That board allows us to connect the same input to several (up to four) sinks. One of the splitter outputs will be sent to the convolution chip input. It processes the sequence of AE's and produces its output. The convolution chip output and another output of the splitter board are connected to two inputs of a merger board. That board mixes the input events coming from several input ports into a single output port. That way we have available the input and the output events synchronized. To distinguish the source of the events the convolution chip output passes through a mapper board. That board re-maps the input addresses into different ones ruled by a mapping table stored in a RAM. The re-mapping performed consists of setting one unused bit of the AE's produced by the convolution chip. That is possible because the address space is of 16 bits and the images we use have only 48x48 pixels. Then, mixing the chip input and output with the merger board, we will have available the input and output events synchronized to analyze the system performance, delay, etc.

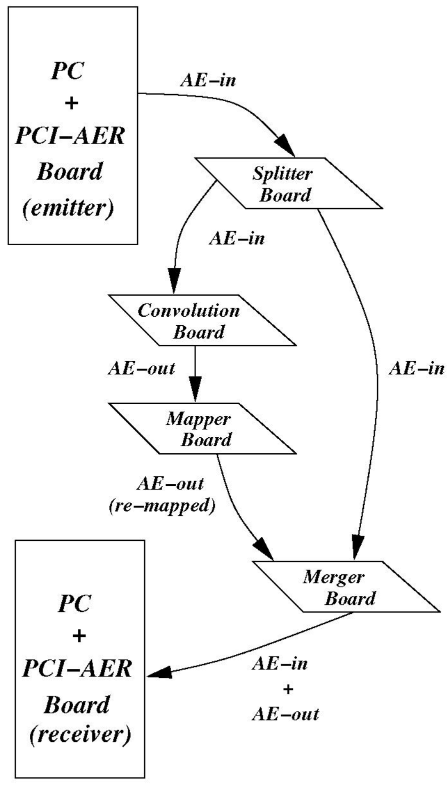

Fig. 5: System block diagram

The receiver PC takes the events from the input AER bus and makes them available for being processed with Matlab.

4. Results

As an example of what we will show in the demo, we present several snapshots taken from the system output AE-stream. The four images in the left side of Fig. 6 show the input and the output of the system when the input was a curved propeller, considering

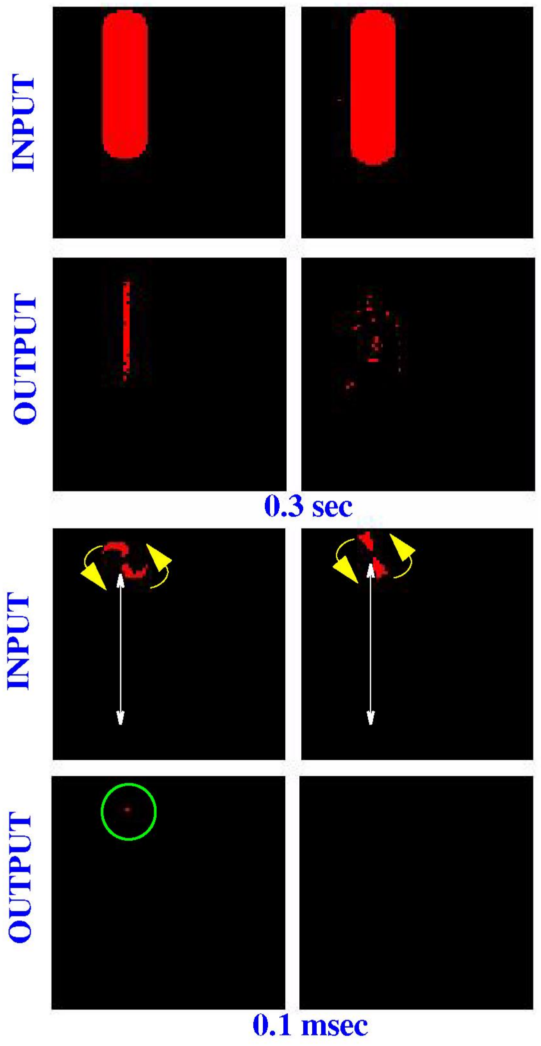

Fig. 6: Demo output snapshots

two different time-windows: 0.3 seconds and 0.1msec. Those images in right hand side of that figure show the input and the output when the input was the straight propeller. The propellers were rotating and also moving linearly. The one which matched the programmed kernel (Fig. 4.(b)), the curved one, produces spikes at the position corresponding to the propeller center (see left hand side of Fig. 6). Very few and sparse AE's are produced when the input is the straight one (right hand side of Fig. 6). The output in the top-left part of the figure shows the path followed by the curved propeller while the bottom-left image shows a specific point of its trajectory, highlighted with a circle because it is difficult to distinguish it.

5. Conclusions

A small image processing system has been presented. Its bio-inspired architecture makes it able to perform efficiently very high speed task which are, at least, very difficult to carry out with conventional processing algorithms running on a modern PC. Here a simple task is performed, but some of the building blocks needed to implement much more complex bio-inspired processing systems are here interfaced.

Acknowledgements

The work presented in this paper was partially supported by EU grant IST-2001-34124 (CAVIAR), and Spanish grants TIC-200-08164-C03-01 (SAMANTA) and TIC-2000-0406-P4 (VICTOR). RSG was supported by an FPU spanish grant from the Ministry of Education & Science.

References

[1] V. Öwall, M. Torkelson, and P. Egelberg, "A Custom Image Convolution DSP with a Sustained Calculation Capacity of >1GMAC/s and Low I/O Bandwidth", Journal of VLSI Signal Processing, vol. 23, pp. 355-349, 1999.

[2] H. Kwon, "A Low-Power Image Convolution Algorithm for Variable Voltage Processors", Proc. of the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), vol. 2, pp. 677-680, 2003.

[3] H. H. Cut, A. Gentile, J.C. Eble, M. Lee, O. Vendier, Y. J. Joo, D. S. Wills, M. Brooke, N. M. Jokerst, and A. S. Brown, "SIMPil: An OE Integrated SIMD Architecture for Focal Plane Processing Applications", Proc. of the IEEE MPPOI, pp. 44-52, 1996.

[4] W. C. Fang, "A SmartVision System-on-a-Chip Design Based on Programmable Neural Processor Integrated with Active Pixel Sensor", Proc. of the IEEE Int. Conf. on Circuits and Systems (ISCAS), vol. 2, pp. 128-131, 2000.

[5] S. Thorpe, A. Delorme and R. Van Rullen, "Spike based strategies for rapid processing", Neural Networks, Vol. 14, pp. 715-725, (2001 Special issue).

[6] W. Maass and C. M. Bishop, "Pulsed Neural Networks", The MIT Press, 1999.

[7] S. Grossberg and E. Mingolla. "Neural dynamics of perceptual grouping: Textures", boundaries, and emergent segmentation. Percept. Psychophys., 38:141--171, 1985.

[8] Sivilotti, "Wiring considerations in Analog VLSI Systems with application to Field Programmable Networks", Ph.D. dissertation, Caltech, Pasadena CA 1991.

[9] B. Linares-Barranco, T. Serrano-Gotarredona, R. Serrano-Gotarredona and J. Costas-Santo, "A new charge-packet mismatch calibrated integrate and fire neuron for processing positive and negative signals in AER-based systems", Proc. of ISCAS'04, Vancouver, vol. 5, pp 744-747, June 2004.

[10] R.Serrano-Gotarredona, T. Serrano-Gotarredona and B. Linares-Barranco, "On events generators for Address Event Representation transmitters" Proc. of SPIE'05. Sevilla, May 2005

[11] A. Linares-Barranco, B. Linares-Barranco, G. Jiménez-Moreno, A. Civit-Ballcels, "Synthetic Generation of Events for Address-Event-Representation Communications", Lecture Notes in Artificial Intelligence, vol. 2451, n. 1, pp.371-379, 2002.

[12] R, Paz-Vicente, A. Linares-Barranco, D. Cascado, S. Vicente, G. Jiménez, A.Civit, "Time-recovering PCI-AER interface for bio-inspired spiking systems", Proc. of SPIE'2005, Sevilla, May 2005.