Neocortical Frame-free Vision Sensing and Processing through Scalable Spiking ConvNet Hardware

L. Camuñas-Mesa, J. A. Pérez-Carrasco, C. Zamarreño-Ramos, T. Serrano-Gotarredona, and B. Linares-Barranco

Abstract– **This paper summarizes how Convolutional Neural Networks (ConvNets) can be implemented in hardware using Spiking neural network Address-Event-Representation (AER) technology, for sophisticated pattern and object recognition tasks operating at mili second delay throughputs. Although such hardware would require hundreds of individual convolutional modules and thus is presently not yet available, we discuss methods and technologies for implementing it in the near future. On the other hand, we provide precise behavioral simulations of large scale spiking AER convolutional hardware and evaluate its performance, by using peformance figures of already available AER convolution chips fed with real sensory data obtained from physically available AER motion retina chips. We provide simulation results of systems trained for people recognition, showing recognition delays of a few miliseconds from stimulus onset. ConvNets show good up scaling behavior and possibilities for being implemented efficiently with new nano scale hybrid CMOS/nonCMOS technologies.**

I. Introduction

Artificial machine vision systems capture and process sequences of frames. For example, a video camera captures images at about 25-30 frames per second, which are then processed frame by frame, pixel by pixel, usually with convolution operations, to extract, enhance and combine features, and perform operations in feature spaces, until a desired recognition is achieved. This frame convolution processing is slow, specially if many convolutions need to be computed for each input image or frame [1]-[2].

Living brains do not operate on a frame by frame basis. In the retina, each pixel sends spikes (also called events) to the cortex when its activity level reaches a threshold. Pixels are not read by an external scanner. Pixels decide when to send an event. All these spikes are transmitted as they are being produced, and do not wait for an artificial "*frame-time*" before sending them to the next processing layer. Besides this frame-less nature, brains are structured hierarchically in cortical layers [3]. Neurons (pixels) in one layer connect to a projection field of neurons (pixels) in the next layer. This processing based on projection-fields is similar to convolution-based processing [4], at least for the earlier cortical layers. For example, it is widely accepted that the first layer of visual cortex V1 performs an operation similar to a bank of 2D Gabor like filters at different scales and orientations [2] whose actual parameters have been measured [5]-[7]. This fact has been exploited by many researchers to propose powerful convolution based image processing algorithms [1]-[2],[5]-[13]. Fig. 1 shows a typical hierarchical structure of a feed forward Convolutional Neural Network. However, convolutions are computationally expensive. It seems unlikely that the high number of convolutions that might be performed by the brain could be emulated fast enough by software programs running on the fastest of today's computers. Although some researchers are providing some interesting bio-inspired solutions for frame-constrained vision systems [14], many researchers believe that a new frame-less hardware technology is required for approaching the processing capability of biological brains.

Authors are with the Instituto de Microelectrónica de Sevilla
(IMSE-CNM-CSIC), Américo Vespucio s/n, 41092 Sevilla, Spain.
E-mail: *bernabe@imse-cnm.csic.es*

Address-Event-Representation (AER) is a promising emergent hardware technology that shows potential for providing the computing requirements of large projection-field based multi-layer systems. AER was first proposed in 1991 in one of the Caltech research labs [15]-[24], and has been used since then by a wide community of neuromorphic hardware engineers. AER has been used fundamentally in image sensors, for simple light intensity to frequency transformations [16], time-to-first-spike coding [17]-[18], foveated sensors [19], spatial contrast [20]-[21] and more elaborate transient detectors [22]. But AER has also been used for auditory systems [25]-[26], competition and winner-takes-all networks [27]-[28], and even for systems distributed over wireless networks [29]. Some AER convolution processing chips with hardwired kernels (slightly tunable) have also been proposed [43]-[44]. However, it was not until arbitrary-shape-kernel convolution chips became available (with [45] or without kernel symmetry restrictions [30]) that their potential for building large scale AER ConvNets for arbitrary pattern and object recognition applications became apparent [31]-[40]. Several AER fully-programmable-kernel convolution chips have been reported. Either mixed-mode based on pixel-level charge packet integration [30]-[31], or fully digital with in-pixel accumulator and adder to emulate leaky integrate-and-fire neurons [46].

These chips, which can perform large arbitrary kernel convolutions (32x32 in [30]) at speeds of about $3x10^9$ connections/sec/chip, can be used as building blocks for larger cortical-like multi-layer hierarchical structures, because of the modular and scalable nature of AER based systems. AER (convolutional) modules can be interconnected through nearest neighbor LVDS links, either within a single chip (using Network-on-Chip technology [32], with several tens of modules per chip) or within a surface mount PCB. Consequently, present day technology could make it possible to assemble several thousands of such convolutional modules, allowing a reconfigurable inter connectivity for a variety of applications.

II. History of ConvNets and Spiking ConvNets

In 1959 Hubel and Wiesel reported their findings on projection field processing in early stages of visual cortex, receiving the 1981 Nobel prize. Based on this, Convolutional Neural Networks (ConvNets) were originally proposed by Fukushima in 1969 [8]-[9] and further developed by Yann LeCun [10] and other groups, as a type of continuous-time gradient-based learning neural paradigm, with great success in a variety of

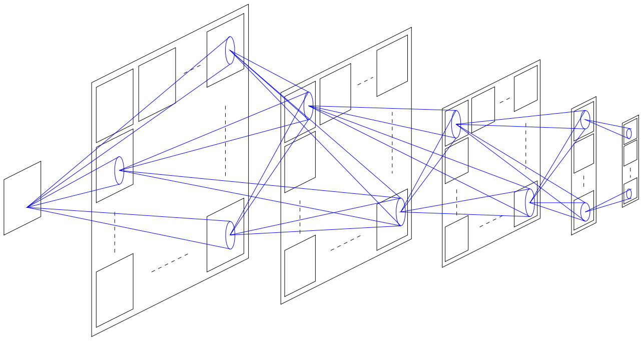


Fig. 1: Typical Feed Forward ConvNet Structure

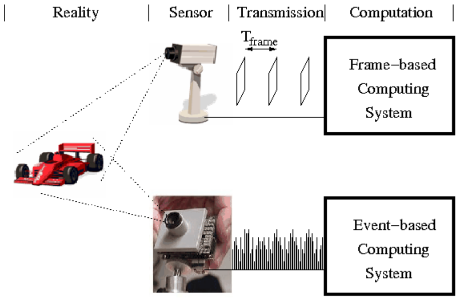


Fig. 2: Conceptual illustration of Frame-constraint (top) vs. a Frame -free Event-based (bottom) Vision sensing and processing system.

(industrial) applications as well as research. Examples of industrial applications and developments are, to mention a few: (1) NEC with products for face/person detection, age and gender recognition for vending machines, as well as prototypes for cancer cell detection or mobile phone imaging applications, (2) France Telecom/Orange with face detection and recognition, text detection and recognition, various mobile phone applications, (3) Vidient Technologies with products for video surveillance, human detection and tracking, (4) Canon with cameras with embedded video surveillance, (5) Microsoft with handwriting recognition, (6) AT&T/Lucent-Technologies/ NCR with products for check recognition. Examples of state-of-the-art research exploiting ConvNets are (1) Poggio at MIT with object recognition and scene analysis [2], (2) Seung at MIT with image segmentation, and biological image analysis (brain circuit reconstruction) [34], (3) NEC Labs with natural language processing and understanding [35], (4) NYU with biological image analysis, object recognition and visual navigation for robots [36].

On the other hand, in 1996, Thorpe demonstrated that the human visual system is capable of performing object recognition tasks at such speeds that any neuron involved only had time to fire one spike [38]. Based on this finding, he developed a Framework for spiking ConvNets, which is presently being exploited commercially for high speed object recognition software [39].

In the field of VLSI circuit design we have witnessed, during the past years, important developments in the field of spiking neural hardware, and specifically spiking hardware for ConvNet processing [30]-[40] of visual information sensed by highly efficient spiking sensors [22]. It is now becoming apparent that the combination of ConvNets theories and knowledge, the framework of spiking information sensing and processing, with state-of-the-art hardware technologies such as Networks-On-Chip [32] and emergent nano scale CMOS and hybrid CMOS/non-CMOS nanotechnologies [42], will result in highly efficient systems for sophisticated cognitive tasks, similar to the human brain. In this paper we review scaling properties for ConvNets hardware, discuss spike signal coding, explain spiking convolution chips and how to use them for assembling large scale modular cortex-like structures for object recognition. We present some behavioral simulation results of such structures for human detection and tracking. At the end we discuss the potential of spiking ConvNets systems for being implemented with new coming nanotechnology devices.

III. Frame-constraint vs. Frame-free Event-based Vision Sensing and Processing

Fig. 2 illustrates the conceptual difference between a Frame- and an Event-based sensing and processing system. Each use a camera sensor to capture reality. In the top row, a frame-constraint camera captures a sequence of frames, each of which is transmitted to the computing system. Each frame is processed by sophisticated image processing algorithms for achieving some recognition. The Computing system needs to have all pixel values of a frame before starting any computation. In the bottom row an event-based vision sensor operates without frames. Each pixel sends an event (usually its own *x,y* coordinate) when it senses some property (change in intensity [16], contrast with respect to neighboring pixels [21], ...).

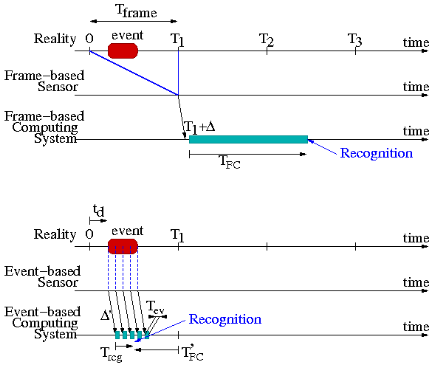


Fig. 3: Comparison of timing issues between a (top) Frame-constraint and a (bottom) Frame-free Event-based Sensing and Processing System.

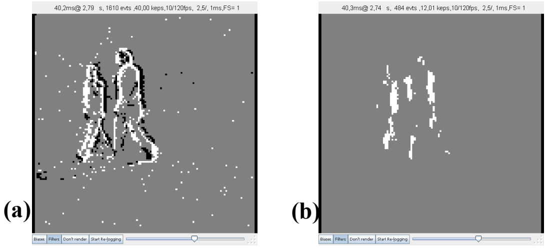

40,2ms@ 2,79 s, 1610 evts ,40,00 keps,10/120fps, 2,5/, 1ms,FS= 1
Biases
Filters
Don't render
Start Re-logging
(a)
40,3ms@ 2,74 s, 484 evts ,12,01 keps,10/120fps, 2,5/, 1ms,FS= 1
Biases
Filters
Don't render
Start Re-logging
(b)

Fig. 4: Illustration about the hardware implementation of the method: (a) Two persons walking captured with a 128x128 temporal contrast (motion) retina [22]. Pixels sensing a positive time derivative in light intensity send a positive event (white), while those sensing a negative time derivative send a negative event (black). Grey pixels are silent. The figure shows the events captured during an interval of about 80ms with a total of about 1500 events. (b) As these pixel events are generated asynchronously by the motion retina, they are received and processed one by one by a receiver convolution chip programmed with a 7x7 vertical Gabor 2D spatial filter. The computation delay in the convolution chip is about 150ns per event. The figure shows about 300 output events produced during the same 80ms by the convolution chip.

Events are sent out to the Computing System as they are produced, without waiting for a Frame Time. The Computing System updates its state after each event. Fig. 3 illustrates the inherent difference in timings between both concepts. In the top (Frame-constraint), reality is binned into compartments of duration T_{frame}. During the first frame T_1 an event happens (such as a flashing shape), but the information produced by this event does not reach the computing system until the full frame is captured (at T_1) and transmitted (with an additional delay Δ). Then the computing system has to process the full frame, handling large amount of data and requiring a long “Frame Computation Time” T_{FC} before the “recognition” information is available. In the bottom of Fig. 3, pixels “see” directly the event in reality and send out their own events with a delay Δ' to the computing system. Events are processed as they flow with an Event Latency T_{ev} (in the order of *ns*). For performing recognition not all events are necessary. Actually, more relevant events usually come out first or with higher frequency. Consequently, recognition time T_{rcg} can be smaller than the total time of the events produced. Note that recognition is possible before frame time T_1, resulting in a negative T'_{FC} when compared to the recognition delay of a Frame-constraint system [54].

Fig. 4 provides an illustration of a typical operation of an AER based hardware [46]. In this case the hardware is composed of one temporal contrast (motion) sensing retina of 128x128 pixels [22] that is sending its output events to a 2D convolution chip programmed with a 7x7 pixel vertical Gabor filter. A pixel in the retina sends out an event (which usually consists of its *x,y* coordinate) every time its incident light intensity changes a relative amount of at least 2.5%. Fig. 4(a) shows the 1500 events generated by the retina during about 80ms when observing two persons

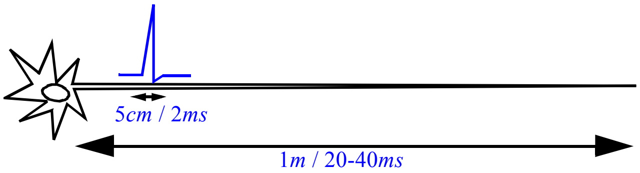


Fig. 5: Biological action potential travelling through a nerve

walking. The receiver convolution chip processes each event as it comes in with a delay of about T_{ev} = 150ns. Pixels in the 2D array of integrators of the convolution chip will generate their own output events. Fig. 4(b) shows the 300 output events produced by the convolution chip during the same 80ms. This 7x7 kernel typically requires between 5 and 20 spatio-temporal correlated input events to produce an output event. As soon as these events are fed to the convolution chip, the corresponding output event appears with a delay of 100-200ns. Consequently, in practice, input and output event flows are simultaneous.

IV. Scaling Properties of ConvNets Hardware

Interestingly, AER hardware sensing or processing modules can be assembled into large hierarchical structures, as if one assembles bricks [40]. This is because of the robustness and asynchrony of the AER communication links between the modules, and the availability of “glue” modules such as AER splitters, mergers, and mappers [40]-[41]. A typical ConvNet architecture (see Fig. 1) usually contains a reduced number of sequential layers (4-10), each of which performs several 2D filtering operations in parallel. Early stages extract simple features (such as edge orientation and scale), which are progressively combined into more complex shapes and figures at later stages. Early stages usually operate with small but dense kernels, while later stages use longer range but sparser ones [2]. To increase the knowledge (dictionary of shapes and figures) of the system one simply adds more 2D filters in later layers. Example ConvNets systems for face and character recognition applications may have several tens to hundreds filters per layer. What is interesting about ConvNets, compared to other neural networks, is their graceful scaling capability. To increase knowledge one simply has to increase the number of filters in a layer. Thus, number of neurons (pixels) scales linearly with the number of modules. There is a fixed number of synapses per filter (the convolutional kernel weights). Consequently, number of synapses also scales linearly with the number of filters. On the other hand, the latency of the computing structure (if implemented as parallel hardware) is determined mainly by the number of sequential layers, which is a reduced number and does not change for a given application. Therefore, speed does not degrade by adding more modules per layer (more knowledge). In other neural network architectures the number of synapses scales quadratically with the number of neurons. Consequently, ConvNets seem very appealing for configurable, modular and scalable spiking hardware implementations.

V. Energy/Information Efficiency of Spike Signal Coding

Biological brains code and transmit information as spiking signals. This is because spike coding is highly energy efficient as well as information processing efficient. Fig. 5 shows a typical 2*ms* “action potential” neural spike travelling about 1*m* from the brain to a finger muscle in about 20-40*ms*. In space, such spike only charges about 5*cm* of nerve, but not the whole line. Thus, the spike sender only needs to provide charge for this 5*cm* travelling nerve segment. This contrasts with present day electronic systems where digital information (bits) are transmitted by fully charging and later discharging wires. Furthermore, when using transmission lines for high speed, there is usually a 50Ω resistive component, besides the capacitive load, which is permanently dissipating energy (even in the absence of information transmission). Spike encoding in biology is therefore highly energy efficient. It is true that in present day AER systems, although information is encoded in spikes, it is still transmitted by fully charging and later discharging interconnect wires. However, researchers should start looking into more efficient approaches, such as soliton technology [47], because the energy dissipation of interconnects is going to be major problem when scaling up neural systems to brain complexities, while scaling down sizes using new nano technologies.

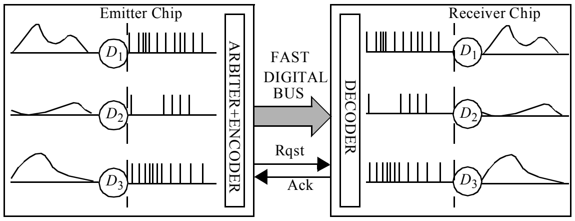


Fig. 6: Rate-coded point-to-point AER inter-chip communication link

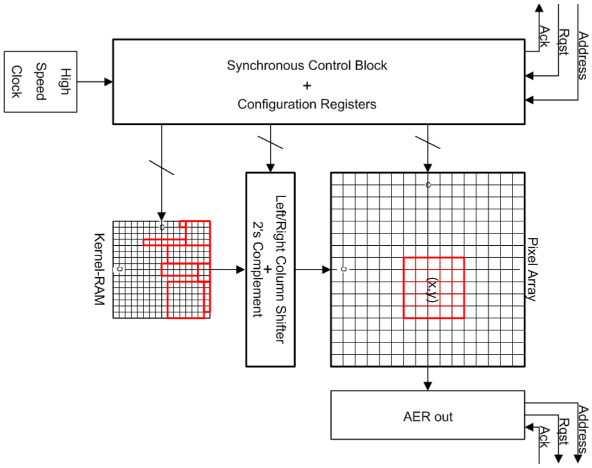


Fig. 7: AER multi-kernel Convolution Chip Diagram

Regarding information coding, spikes allow for very efficient schemes. Thorpe demonstrated in 1996 [38] that, in the human brain, fast recognition of sophisticated figures (such as animal detection in a photograph) is performed in a feed forward manner in such a way that the neurons involved only generate one spike. Thorpe later developed spike-processing convolutional architectures and rank-order coding schemes capable of performing this type of recognition efficiently in software [39].

VI. AER Convolution Chips

Fig. 6 illustrates event communication in a point-to-point AER link [48], where pixel intensity is coded directly as pixel event frequency[1]. The continuous-time states of pixels D_i in an emitter chip are transformed into sequences of fast digital pulses (spikes or events) of minimal width (in the order of *ns*) but with much longer inter-spike intervals (typically in the order of *ms*). Each time a pixel generates a spike, its *x,y* address is written on the inter-chip digital bus, after proper arbitration. This is called an "Address Event". The receiver chip reads and decodes the addresses of the incoming events and sends spikes to the corresponding receiving pixels for reconstruction or further processing. In an AER convolution receiver chip, incoming events are sent to a neighborhood of pixel *x,y* onto which the 2D kernel is added. Fig. 7 shows the conceptual diagram of a fully digital AER Convolution chip. It contains a pixel array, where each pixel includes an adder/accumulator where incoming events are accumulated. When the accumulator reaches a positive (negative) threshold, the pixel is reset and generates a positive (negative) output event. The convolution kernels are stored in the kernel RAM. The controller copies the kernel line by line from the kernel

[1] Other more efficient coding schemes have been proposed, such as rank-order coding [53] where the order of the events carries the information, instead of pixel frequency.

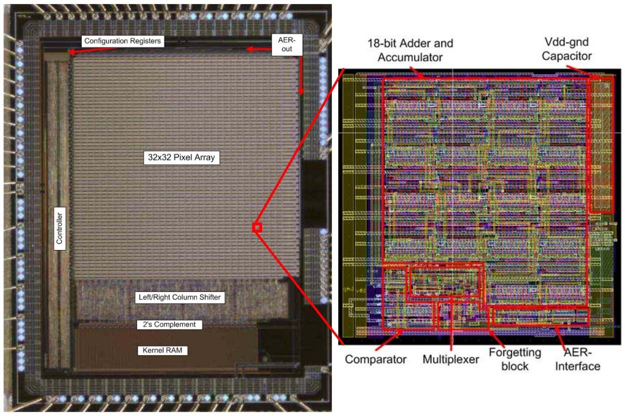


Fig. 8: Chip photograph and pixel layout details

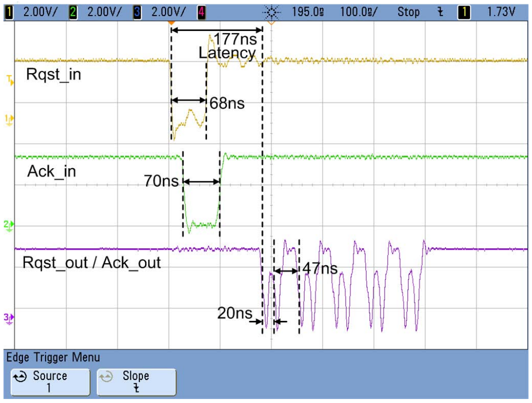
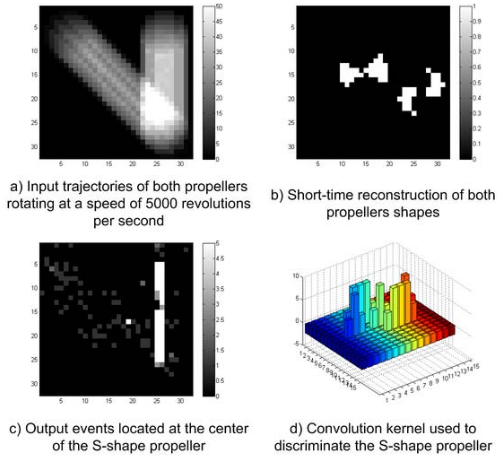


Fig. 9: Latency between input and output events in an AER convolution chip


a) Input trajectories of both propellers rotating at a speed of 5000 revolutions per second

b) Short-time reconstruction of both propellers shapes

c) Output events located at the center of the S-shape propeller

d) Convolution kernel used to discriminate the S-shape propeller

Fig. 10: Recognition of propellers rotating at 5000 revolutions per second

RAM to the pixel array. Kernels are shifted left/right depending on the incoming event coordinate. In parallel, the controller subtracts a fixed number from each pixel accumulator at a fixed rate, to emulate a leak. This way, leaky integrate-and-fire neurons [30] are implemented with fully digital circuitry.

A 4.3x5.4mm^2 prototype chip has been fabricated in the AMS 0.35μm CMOS process. A die photograph is shown in Fig. 8,. The largest block is the 32x32 array of pixels, with an approximate area of 3x3.2mm^2. The synchronous controller consumes around 4500x300μm^2, the static kernel-RAM of 32x32 6-bit words 600x2700μm^2, and the left/right column shifter 600x3100μm^2. The rest of the circuits, like the AER-arbiters, 2's complement and clock generator, consume much less area. The pixel layout, with an area of 95.6x101.3μm^2, is also shown in Fig. 8. Most of this area is consumed by the 18-bit adder and accumulator. The rest of the circuits are: the forgetting block, the multiplexer, the comparator and the AER interface. Although the chip resolution is 32x32 pixels, it can address an input space of 128x128. Chip power consumption depends both on the input throughput and the kernel size and varies between 66mW and 198mW.

Fig. 9 shows oscilloscope captures of the input and output ports handshaking signals of the convolution chip. The chip was programmed with a 3x5 kernel and configured in such a way that one input event (see the 68-70ns pulses in Rqst_in and Ack_in) would generate 10 output events (Rqst_out and Ack_out are shorted and show 10 pulses). The delay between the onset of the incoming event and the onset of the first outgoing event is 177ns.

As an illustration of the high speed performance of this kind of chips, Fig. 10 shows the recognition results of propellers rotating at 5000 rps (revolutions per second). The convolution chip is fed with an input event flow representing two propellers of different shapes (one rectilinear and one S-shaped) rotating at 5krps and moving across the field of view. The convolution chip is programmed with a kernel that performs template matching on the S-shaped propeller. As can be seen, the output of the chip follows correctly the trajectory of the center of the S-shaped propeller.

VII. Modular Systems

Reported (software) ConvNets need tens, hundreds, or even thousands of convolutional filters to perform properly on real-world pattern recognition tasks. Consequently, if we want to provide a realistic hardware infrastructure for real-world applications, it will be essential to assemble hundreds or thousands of AER convolutional modules like the one shown in Fig. 7. Furthermore, the infrastructure needs to offer a good degree of reconfigurability and programmability, so that arbitrary ConvNet architectures could be implemented and tested easily.

Fig. 11 shows a conceptual solution to achieve this. It consists of a 2D array of modular convolutional units. Each unit includes a programmable-kernel convolution module (like the one in Fig. 7) plus a local router. Each module can receive input events from any four neighbors, and sends its own output events to one or more of its four neighbors. Each local router is programmed with a local routing table. The router detects, for each incoming event, the input port and decides to either process it by the local convolution module, or transfer it to one or more of its output ports. The events produced by the local convolution module are also processed by the local router who sends them out through the proper output port(s). It is fairly easy to show that any arbitrary multi-filter architecture netlist can be implemented in this 2D structure by generating proper local routing tables for each module in the 2D array. Furthermore, such process can be automated by a compiling software which, given a ConvNet netlist, would generate automatically all local routing tables. The modular 2D structure in Fig. 11 can be implemented physically using surface mount PCBs with miniature individual convolution chips with local router, or could be implemented with large chips of the type called Network-on-Chip (NoC), capable of hosting tens to hundreds of

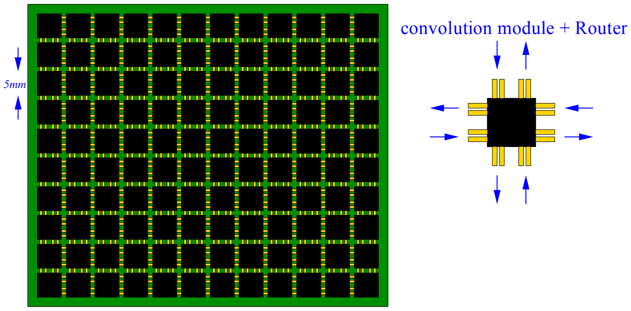


Fig. 11: Concept of modular ConvNet Structure

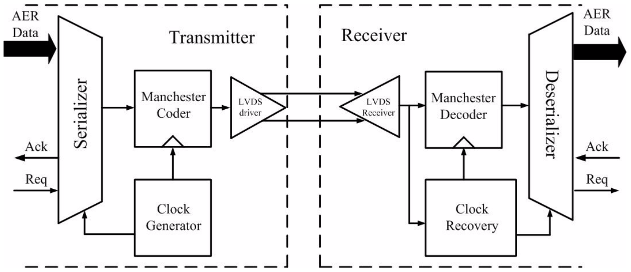


Fig. 12: LVDS Transmitter and Receiver block diagram with Manchester Encoding

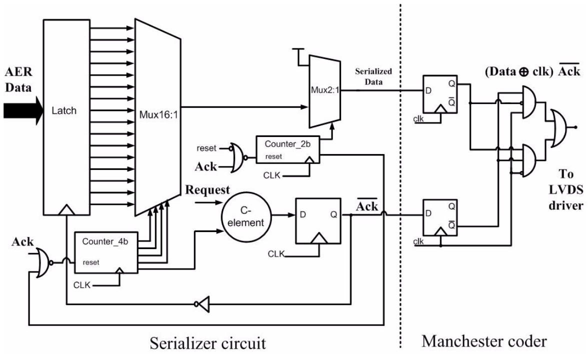


Fig. 13: Serializer circuit block diagram

such modules per chip [32]. Multiple boards (or NoCs) can be further assembled hierarchically to scale up such systems.

VIII. AER LVDS Serial Links

In Fig. 11 each AER link is represented by 2 wires. This symbolizes a serial link of the type LVDS (low voltage differential signalling). In LVDS bits are sent serially on a differential wire at low voltage excursions. Lines are terminated with 100Ω impedance. LVDS is an industrial standard [49] and many commercial products use this serial communication internally. However, present day LVDS links need to maintain a continuous flow of information permanently to keep sender and receiver synchronized. When no information needs to be transmitted, meaningless symbols (called 'commas') are sent. If sender and emitter loose synchronization, long wait times are required to recover synchronization.

We propose to use a different approach, where there can be silent periods, and sender and receiver synchronize quickly. The idea is to exploit Manchester encoding [50] to transmit data and clock. This allows for very simple sender and receiver circuitry and fast locking between sender and receiver after silent periods, at the cost of transmitting information at half the speed. Fig. 12 shows the block diagram a Manchester encoding sender receiver pair for AER communication [51]. The sender, shown in Fig. 13 contains a serializer triggered by the AER Request signal, and a Manchester coder. The receiver is shown in Fig. 14. It includes a clock recovery circuit containing 5 delay inverters, whose delay tunes to the incoming stream during data transmission. In the absence of data, the state of the inverter delay is memorized, which allows to quickly read new incoming data when it arrives. Since the receiver is idle during the silent periods, it is possible to devise schemes where the power consumption of both sender and receiver is made negligible during these periods, while allowing instant recovery to the data transmission state [52].

IX. System-Level Behavioral Simulations

So far, it seems feasible to provide a hardware technology for spiking ConvNets based on AER. However, it is true that at present such large scale hardware systems have not been reported yet. Probably the largest AER system reported so far is the CAVIAR systems [40], which uses four custom made AER chips (motion retina, convolution chip, winner-take-all chip, and learning chip) plus a set of FPGA based AER interfacing and mapping modules. The CAVIAR system includes 45k neurons, emulates up to 5 million synapses, performs an equivalent of 9 giga-connects-per-second, and can sense, identify and track objects with a 3*ms* delay. However, this system only has 4 convolution modules. Obviously, present-day AER hardware state-of-the-art is still not at the level of what is shown in Fig. 11 (with about 10^7 neurons emulating about 10^{11} synapses). In order to estimate the performance and evaluate the limitations one may encounter when assembling larger scale ConvNet with AER hardware, we have developed an event-based AER system simulator [54]. Fig. 16 shows an example circuit and netlist description used in this simulator. AER links are represented as "channels". At the end of the simulation, each channel would contain a list of all the events that have travelled through this channel including event information (such as its *x,y* address) and timing information (time at which the event was generated inside

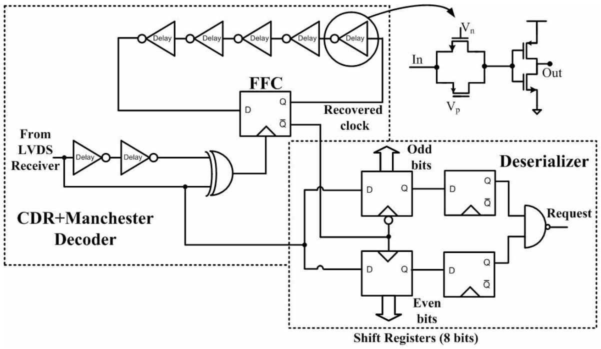


Fig. 14: Deserializer with clock recovery and Manchester decoder

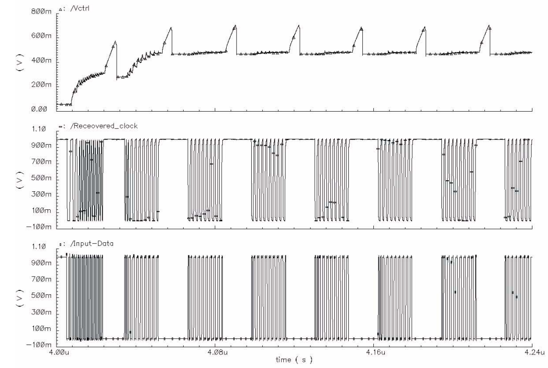
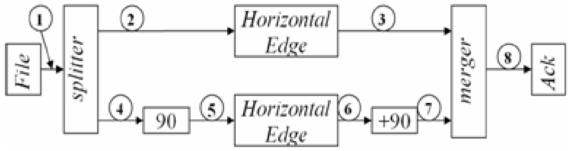


Fig. 15: Post-layout simulations of LVDS AER link



```
%First, we declare sources to the system
%         SOURCES   SOURCES DATA
sources   {1}       {data1}

%Next, we declare priorities
priorities {0.9,0.8,0.7,0.6,0.5,0.4,0.3,0.2}

%Next, we declare blocks
%NAME          IN-CHANNELS   OUT-CHANNELS   PARAMS        STATES
splitter       {1}           {2,4}          {params1}     {state1}
h_sobel        {2}           {3}            {params2}     {state2}
imrotate90     {4}           {5}            {params3}     {state3}
h_sobel        {5}           {6}            {params4}     {state4}
imrotate90     {6}           {7}            {params5}     {state5}
merger         {3,7}         {8}            {params6}     {state6}
ack            {8}           {}             {params7}     {state7}
```

Fig. 16: Example circuit (top) and its netlist description (bottom) of a 2-convolution (horizontal edge) ConvNet System.

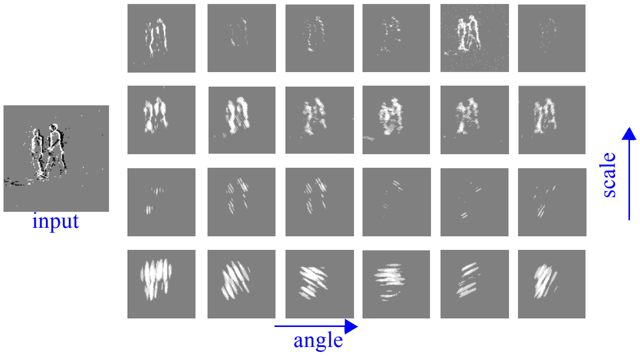


Fig. 17: Behavioral simulation results of a bank of AER Gabor filters of different scales and orientations over a physical sensory input obtained with an AER motion retina.

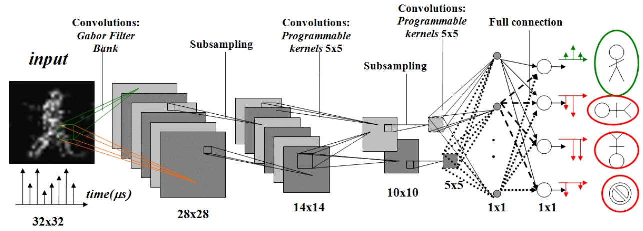


Fig. 18: Structure of spiking ConvNets trained for recognizing people from visual data captured from an AER motion retina

the emitter module, and time of physical use of channel). The input to the system (like “File” in Fig. 16) can be real physical events captured from a real AER retina and recorded as a file. Individual blocks in the netlist are behaviorally modelled, including all timing delays of handshaking signals, parasitic effects, finite precision effects, etc. This way, we can use real performance figures of already physically available AER chips/modules to model them in the simulator. The combination of real sensory event-format data with performance figures of physical AER hardware allows to estimate reasonably well the performance of scaled-up systems.

As an illustrative example, Fig. 17 shows the simulation results of a bank of 24 Gabor filters with 4 scales and 6 orientations. The input is a 4 second 128x128 pixel motion retina recording of two persons walking from right to left, totaling about 130k events. Convolution modules compute their outputs as the events flow in. Each convolution module/chip needs about 100-200*ns* of computation time per input event [30]. After a few input events (about 10, depending on kernel) a convolution module provides its own output events. Consequently, the delay between input and output flow of events in a convolution module is of the order of microseconds (or fraction), making both flows in practice simultaneous. Fig. 17 shows for each convolution module and retina, the events captured during the same 40*ms*.

A bank of Gabor type filtering is usually the first stage of visual processing, like in the human brain [2]. For pattern and object recognition more stages are required. Fig. 18 shows an example ConvNet trained to recognize humans recorded with an AER motion retina. The input visual flow was captured with a physical temporal contrast (motion) AER retina [22] when observing people walking. A person walking produces about 3keps (kilo events per second). Visual pixel array was down sampled to 32x32. The spiking convolutional network has 7 layers. The first layer is a Gabor filter bank, second layer is subsampling, third layer is a trainable 5x5 kernels filter bank, fourth layer is subsampling, fifth layer is again a trainable 5x5 kernel filter bank, and sixth and seventh layers are fully connected trainable perceptrons. The system was trained off-line through back propagation learning to categorize inputs as vertical humans, up side down humans, horizontal humans, or other objects. After training, it was tested with new retina recordings, showing a correct recognition rate of above 86%. Correct recognition was performed after receiving only between 50-80 retina events (18-27ms of input stimulus) with a negligible throughput delay of a few microseconds.

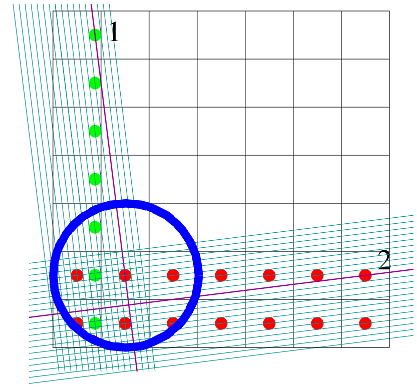


Fig. 19: CMOS neurons underneath a nano scale device fabric. Each neuron has one input and one output node. A grid of nano wires is fabricated on top of CMOS. At each nano wire intersection there is a nano scale synapse device. Each horizontal nanowire connects to one neuron output only, and each vertical nanowire connects to one neuron input only.

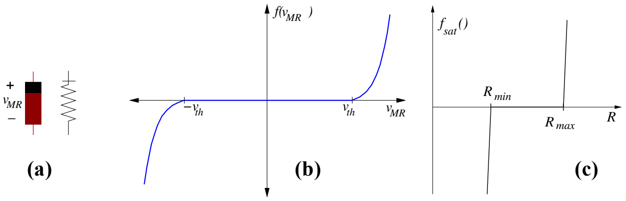


Fig. 20: Memristor (a) symbol, (b) characteristic learning function, and (c) saturating function for restricting *R* to the interval [R_{min},

X. NanoTechnology Implementation Potential

Present-day AER convolution chips compute by scanning row-wise the kernel over its array of pixels. This is a sequential operation which introduces delays in the order of hundreds of nano-seconds, depending on kernel size. In present days we are witnessing a new trend micro and nano technologies, and new nano scale devices, such as the memristor, are being reported with high potential to be used as compact and trainable synapses [55]. The memristor, whose symbols are shown in Fig. 20(a), is an adaptive 2-terminal resistive device, postulated in 1974 [59] but not available until recently [55]. Our objective is to exploit such device as the synaptic element of a neural perceptron. Neurons can be designed using available CMOS VLSI technology, while synapses (which are required in much larger quantities) can be fabricated as nano-scale devices arranged on top of a silicon chip using some post-CMOS fabrication technique, in a CMOL-like arrangement [56], as shown in Fig. 19. The synaptic devices require two modes of operation: (1) a computational mode in which they contribute to a neuron’s integral with a characteristic weight, and (2) an adaptation mode in which they change its characteristic weight when their terminal voltages meet some requirement. In the first mode we will use the devices as resistors, while in the second we want to change its conductance when some of their terminal voltage difference exceeds a threshold v_{th}. For example, Fig. 20 shows symbols and characteristics learning function of a voltage controlled memristor which can be defined by the following equation [61]-[62]

$$i_{MR} = G(w)v_{MR} \qquad \dot{w} = f(v_{MR}) \tag{1}$$

If $|v_{MR}| < v_{th}$ its conductance does not change, otherwise it changes according to eq. (1), where w is a parameter controlling conductance $G \in [G_{min}, G_{max}]$. Fig. 21 shows a macro model circuit that can be used in an electric circuit simulator [60]. It includes a variable resistor, a nonlinear transconductor NOTA implementing the function in Fig. 20(b), a capacitor to implement the derivative and store the actual weight w, and a saturating

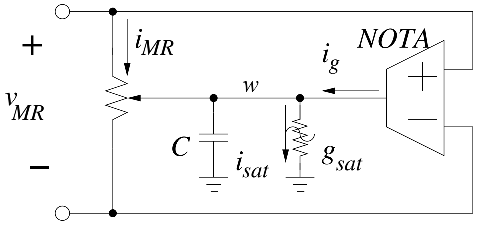


Fig. 21: Memristor macro model circuit for electric simulations

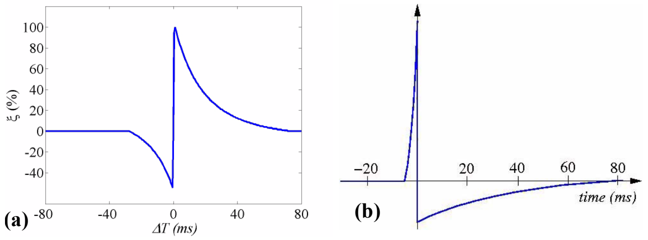


Fig. 22: (a) STDP characterization in biological synapses. Vertical axis is synaptic strength change and horizontal axis is time delay between pre- and post-synaptic spikes. (b) Action potential waveform.

element g_{sat}, described by the curve in Fig. 20(c), to restrict $G \in [G_{min}, G_{max}]$. By combining memristors with spiking signals, Spike-Time-Dependent-Plasticity arises naturally [57].

Spike-time-dependent-plasticity (STDP) is a neural learning mechanism originally postulated [63] in the context of artificial machine learning algorithms (or computational neuroscience) exploiting spike-based computations (as in brains) and has evolved to powerful algorithms [64]-[67]. Astonishingly, experimental evidences of biological STDP have later been reported by several neuroscience groups worldwide [68]. In STDP the change in synaptic weight Δw is expressed as a function ξ of the time difference between the post-synaptic spike at t_{pos} and the pre-synaptic spike at t_{pre}. Specifically, $\Delta w = \xi(\Delta T)$, with $\Delta T = t_{pos} - t_{pre}$. The shape of the STDP function ξ can be interpolated from experimental data from Bi and Poo [68] as shown in Fig. 22(a). For positive ΔT there will be a potentiation of synaptic weight $\Delta w > 0$, which will be stronger as $|\Delta T|$ reduces. For negative ΔT there will be a depression of synaptic weight $\Delta w < 0$, which will be stronger as $|\Delta T|$ reduces. We recently demonstrated [57]-[58] that if a memristor (as defined in eq. (1)) is stimulated on its two terminals by two asynchronous spiking signals of the shape shown in Fig. 22(b) separated by a time ΔT, and attenuating the post-synaptic one by α_{pos}<1, then the weight update function shown in Fig. 22(a) is mathematically obtained, which is identical to the one obtained by Bi and Poo from physiological experiments. This opens the possibility that in biological synapses there might be a memristive type of mechanism responsible for biological STDP [57]. Also, it turns out that the action potential shape strongly influences the resulting STDP function [58].

Using these concepts we can propose a crossbar architecture using memristors as synapses and spiking neurons that send back a replica. Fig. 23 shows a possible arrangement for implementing an STDP spiking memristive feed forward perceptron. Depending on the polarity of the memristors, the neural spikes need to be inverted or not. The inset shows a conceptual block diagram of the neuron circuit required. Neurons are made of integrators whose input node is maintained at virtual ground by a high-gain differential amplifier with a capacitor connected at its negative feedback input, thus acting as an integrator. At the output of the differential amplifier appears the accumulated integral (signed reversed) of the in-flowing current. Whenever this integral reaches threshold V_{REF}, an action potential spike as in Fig. 22(b) will be triggered. During the time of the action potential spike (including fast positive spike plus longer negative tail), a reset pulse will also be provided to short the integrating capacitor. This has three reasons: (a) to reset the accumulated integral, (b) to buffer the output spike to send it back through the input collecting line, (c) and to avoid any further input signal integration during spike production. An attenuated version of the spike voltage is sent forward to the next layer synapses. This architecture avoids cross-coupling of spikes between rows and columns. Using this arrangement with the memristor macro model of Fig. 21 we performed intensive behavioral simulations in Cadence-Spectre to test the concept on the 4x4 feed forward array shown in Fig. 23. Only the first 2 column synapses are stimulated with 200*ms* period spikes (of 45*ms* duration) with a 25*ms* relative delay between the two columns. The result was that only synapses at the first two columns change their resistance, while those on the other two columns do not, confirming the correct operation of STDP, without any crosstalk between columns nor rows [60]. Since STDP works correctly for a feed forward crossbar array, it can be extended to CMOL like networks wired with a connectivity compatible with ConvNets.

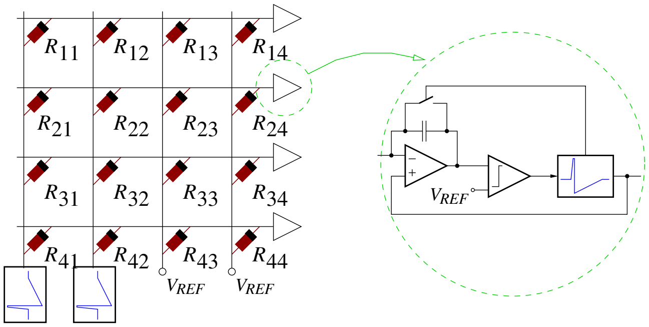


Fig. 23: Feed forward synaptic memristive array simulated behaviorally with Cadence Spectre

XI. Conclusions

We have shown how to implement ConvNets with spiking hardware to perform sophisticated pattern recognition task. Large scale systems have been emulated using a behavioral simulator, but using performance figures of already available hardware, together with real stimuli obtained with physical AER retina chips.

XII. Acknowledgements

This work was supported by EU grant 216777 (NABAB), Spanish grants TEC2006-11730-C03-01 (SAMANTA2) and TEC2009-10639-C04-01 (VULCANO), and Andalucian grant P06TIC01417 (Brain System). JAPC was supported by an Andalucian scholarship and CZR by an FPU scholarship.

XIII. References

[1] S. Grossberg, E. Mingolla, and J. Williamson, ``Synthetic Aperture Radar Processing by a Multiple Scale Neural System for Boundary and Surface Representation," *Neural Networks*, vol. 8, No. 7/8, pp. 1005-1028, 1995.

[2] T. Serre, L. wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, “Robust Object Recognition with Cortex-Like Mechanisms,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, No. 3, pp. 411-426, March 2007.

[3] Gordon M. Shepherd, *The Synaptic Organization of the Brain*, Oxford University Press, 3rd Edition, 1990.

[4] E. T. Rolls and G. Deco, *Computational Neuroscience of Vision*, Oxford University Press, 2002.

[5] R. DeValois, D. Albrecht, and L. Thorell, “Spatial Frequency Selectivity of Cells in Macaque Visual Cortex,” *Vision Research*, vol. 22, pp. 545-559, 1982.

[6] R. DeValois, E. Yund, and N. Hepler, “The Orientation and Direction Selectivity of Cells in Macaque Visual Cortex,” *Vision Research*, vol. 22, pp. 531-544, 1982.

[7] P. H. Schiller, B. L. Finlay, and S. F. Volman, “Quantitative Studies of Single-Cell Properties in Monkey Striate Cortex. Spatial Frequency,” *J. Neurophysiology*, vol. 39, no. 6, pp. 1334-1351, 1976.

[8] K. Fukushima, “Visual Feature Extraction by a Multilayered Network of Analog threshold Elements,” *IEEE Trans. Syst. Sci. and Cybernetics*, vol. SSC-5, No. 4, pp. 322-333, Oct. 1969.

[9] K. Fukushima and N. Wake, “Handwritten Alphanumeric Character Recognition by the Neocognitron,” *IEEE Trans. Neural Networks*, vol. 2, No. 3, pp. 355-365, May 1991.

[10] Y. LeCun and Y. Bengio, “Convolutional Networks for Images, Speech, and Time Series,” in *The Handbook of Brain Science and Neural Networks*, M. Arbib (Ed.), Cambridge, MA: MIT Press, pp. 255-258, 1995.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based Learning Applied to Document Recognition," *Proc. of the IEEE*, vol. 86, No. 11, pp. 2278-2324, November 1998.

[12] C. Neubauer, "Evaluation of Convolution Neural Networks for Visual Recognition," *IEEE Trans. on Neural Networks*, vol. 9, No. 4, pp. 685-696, 1998.

[13] S. Lawrence, C. L. Giles, A. Tsoi, and A. Back, "Face Recognition: A Convolutional Neural Network Approach," *IEEE Trans. on Neural Networks*, vol. 8, No. 1, pp. 98-113, 1997.

[14] H. Yamasaki and T. Shibata, "A Real-Time Image-Feature-Extraction and Vector-Generation VLSI Employing Arrayed-Shift-Register Architecture," *IEEE Journal of Solid-State Circuits*, vol. 42, pp. 2046-2053, 2007.

[15] M. Sivilotti, *Wiring Considerations in Analog VLSI Systems with Application to Field-Programmable Networks*, Ph.D. Thesis, California Institute of Technology, Pasadena CA, 1991.

[16] E. Culurciello, et al., "A biomorphic digital image sensor," *IEEE J. of Solid-State Circuits*, vol. 38, pp. 281-294, 2003.

[17] P. F. Ruedi, et al., "A 128x128, pixel 120-dB dynamic-range vision-sensor chip for image contrast and orientation extraction," *IEEE J. of Solid-State Circuits*, vol. 38, pp. 2325-2333, 2003.

[18] C. Shoushun and A. Bermak, "Arbitrated Time-To-First Spike CMOS Image Sensor with On-Chip Histogram Equalization," *IEEE Trans. VLSI Systems*, vol. 15, No. 3, pp. 346-357, March 2007

[19] M. Azadmehr, et al."A foveated AER Imager Chip," *Proc. of the IEEE Int. Symp. on Circ. and Syst.* (ISCAS2005), pp. 2751-2754, Kobe, Japan, 2005.

[20] K. A. Zaghloul and K. Boahen, "Optic nerve signals in a neuromorphic chip," *IEEE Trans. Biomed. Eng.,* vol. 51, no. 4, pp. 657-675, Apr. 2004.

[21] J. Costas-Santos et al., "A contrast retina with on-chip calibration for neuromorphic spike-based AER vision systems," *IEEE Trans. Circ. Syst. I, Reg. Papers,* vol. 54, no. 7, pp. 1444-1458, July 2007.

[22] P. Lichtsteiner, C. Posch, and T. Delbrück, "A 128x128 120dB 30mW Asynchronous Vision Sensor that Responds to Relative Intensity Change," *IEEE J. Solid-State Circ.*, vol. 43, No. 2, pp. 566-576, Feb. 2008.

[23] K. Boahen, "Retinomorphic Chips that see Quadruple Images,"*Proc. Int. Conf. Microelectronics for Neural, Fuzzy and Bio-Inspired Systems* (Microneuro99), pp. 12-20, Granada, Spain, 1999.

[24] J. Lazzaro, et al., "Silicon Auditory Processors as Computer Peripherals," *IEEE Trans. on Neural Networks*, vol. 4, pp. 523-528, May 1993.

[25] G. Cauwenberghs, N. Kumar, W. Himmelbauer, and A.G. Andreou, "An analog VLSI Chip with Asynchronous Interface for Auditory Feature Extraction," *IEEE Trans. Circ. Syst. Part-II*, vol. 45, pp. 600-606, May 1998.

[26] V. Chan, S.-C. Liu, and A. van Schaik, "AER EAR: A Matched Silicon Cochlea Pair With Address Event Representation Interface," *IEEE Trans. Circ. Syst. Part-I*, vol. 54, pp. 48-59, Jan. 2007.

[27] Chicca et al., "A Multichip Pulse-Based Neuromorphic Infrastructure and Its Application to a Model of Orientation Selectivity," *IEEE Trans. Circ. Syst. Part I*, vol. 54, No. 5, pp. 981–993, May 2007.

[28] M. Oster, et al., "Quantification of a spike-based winner-take-all VLSI network," *IEEE Trans. Circ. Syst. 1*, vol. 55, No. 10, pp. 3160-69, 2008.

[29] T. Teixeira, et al., "Event-Based Imaging with Active Illumination in Sensor Networks," *Proc. IEEE Int. Symp. Circ. Syst.*, pp. 644-647, ISCAS 2005.

[30] R. Serrano-Gotarredona, et al., "A Neuromorphic Cortical-Layer Microchip for Spike-Based Event Processing Vision Systems," *IEEE Trans. Circ. Syst. I*, vol. 53, No. 12, pp. 2548-2566, December 2006.

[31] R. Serrano-Gotarredona, et al., "On Real-Time AER 2D Convolutions Hardware for Neuromorphic Spike Based Cortical Processing," *IEEE Trans. on Neural Networks*, vol. 19, No. 7, pp. 1196-1219, July 2008.

[32] S. Vangal et al, "An 80-Tile 1.28 TFLOPS Network-on-Chip in 65nm CMOS," *IEEE Int. Solid-State Circ. Conf.,* ICCSC07, pp. 98-99, Feb. 2007.

[33] L. Bottou and Y. LeCun, "Graph Transformer Networks for Image Recognition," Proc. of ISI

[34] Jain V, et al., "Supervised learning of image restoration with convolutional networks," *IEEE Int. Conf. Comp.t Vis.* (ICCV), 2007.

[35] R. Collobert and J. Weston, "A unified architecture for natural language processing: deep neural networks with multitask learning," *Proc. Int. Conf. on Machine Learning* (ICML 08), pp. 160-167, 2008.

[36] R. Hadsell, P. Sermanet, M. Scoffier, A. Erkan, K. Kavackuoglu, U. Muller and Y. LeCun, "Learning Long-Range Vision for Autonomous Off-Road Driving," *J. Field Robotics*, 26(2):120-144, February 2009.

[37] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, V. Vapnik, "Parallel Support Vector Machines: The Cascade SVM,"

[38] S. Thorpe, D. Fize, C. Marlot, "Speed of processing in the human visual system," *Nature* 381: 520-2, 1996.

[39] S. Thorpe, et al., "SpikeNet: Real-time visual processing with one spike per neuron," *Neurocomputing* 58-60: 857-64, 2004.

[40] R. Serrano-Gotarredona, et al., "CAVIAR: A 45k-Neuron, 5M-Synapse, 12G-connects/sec AER Hardware Sensory-Processing-Learning-Actuating System for High Speed Visual Object Recognition and Tracking," *IEEE Trans. on Neural Networks*, vol. 20, No. 9, pp. 1417-1438, September 2009.

[41]F. Gomez-Rodriguez, R. Paz-Vicente, A. Linares-Barranco, M. Rivas, L. Miro, S. Vicente, G. Jiménez, and A. Civit, "AER Tools for Communications and Debugging," *Proc. of the IEEE Int. Symp. Circ. and Syst.*, pp. 3253-3256, (ISCAS 2006), Kos (Greece), May 2006.

[42] D. B. Strukov and K. K. Likharev, "CMOL FPGA: a configurable architecture for hybrid digital circuits with two-terminal nanodevices," *Nanotechnology* 16, pp. 888-900, 2005.

[43] P. Vernier, et al., "An integrated cortical layer for orientation enhancement," *IEEE J. solid-State Circ.,* vol. 32, No. 2, pp. 177-186, Feb. 1997.

[44] T. Y. W. Choi, et al., "Neuromorphic implementation of orientation hypercolumns," *IEEE Trans. Circ. Syst. I,* vol. 52, No. 6, pp. 1049-60, 2005.

[45] T. Serrano-Gotarredona, et al., "AER image filtering architecture for vision processing systems," *IEEE Trans. Circ. Syst. II, Analog Dig. Signal Proc.,* vol. 46, No. 9, pp. 1064-1071, Sep. 1999.

[46] L. Camuñas-Mesa, et al., "Fully digital AER convolution chip for vision processing," Proc. 2008 IEEE Int. symp. Circ. & Syst. (ISCAS08), pp. 652-655, May 2008.

[47] D. S. Ricket, X. Li, N. Sun, K. Woo, and D. Ham, "On the Self-Generation of Electrical Soliton Pulses," *IEEE J. Solid-State Circ.,* vol. 42, No. 8, Aug. 2007.

[48] K. Boahen, "Point-to-Point Connectivity Between Neuromorphic Chips Using Address Events," *IEEE Trans. Circ. Sys. II*, vol. 47, No. 5, pp. 416-434, 2000.

[49] ANSI/TEIA/EIA-644-1995, *Electrical Characteristics of Low Voltage Differential Signalling (LVDS) Interface Circuits*. Telcom. Ind. Asoc., Nov. 15, 1995.

[50] Williams, F.C., and Kilburn, T., *J. Inst. Elect. Eng*., 96, Pt. III, No. 40, March 1949.

[51] C. Zamarreño et al., "LVDS interface for AER links with burst mode operation capability," *Proc. of the IEEE Int. Symp. on Circ. and Syst.*, pp. 644 - 647, 2008

[52] C. Zamarreño et al., "Low power LVDS transceiver for AER links with burst mode opration capability," *Proc. DCIS* 2009.

[53] Delorme A, et al., "Networks of integrate-and-fire neurons using Rank Order Coding B: Spike timing dependent plasticity and emergence of orientation selectivity," *Neurocomputing* 38-40, pp. 539-45, 2001.

[54] J. A. Pérez-Carrasco, et al., "On the Computational Power of AER Vision Processing Hardware," *Proc. Int. Conf. Circ. Design & Integ. Syst.*, 2007.

[55] D. B. Strukov, et al., "The Missing Memristor Found," *Nature,* vol. 453, pp. 80-83, 1 May 2008.

[56] D. B. Strukov and K. K. Likharev, "CMOL FPGA: a configurable architecture for hybrid digital circuits with two-terminal nanodevices," *Nanotechnology* 16, pp. 888-900, 2005.

[57] B. Linares-Barranco and T. Serrano-Gotarredona, "Memristance can explain Spike-Time-Dependent-Plasticity in Neural Synapses," *Nature Precedings* <http://hdl.handle.net/10101/npre.2009.3010.1> 31st March, 2009.

[58] B. Linares-Barranco and T. Serrano-Gotarredona, "Exploiting Memristance in Adaptive Asynchronous Spiking Neuromorphic Nanotechnology Systems," *Proc. IEEE NANO*, July 2009.

[59] L. O. Chua, "Memristor-the missing circuit element," *IEEE Trans. CAS*, vol. CT-18, no. 5, pp. 507-519, 1971.

[60] J. A. Pérez-Carrasco et al. "On Neuromorphic Spiking Architectures for Asynchronous STDP Memristive Systems," *Proc. of the IEEE Int. Symp. on Circ. and Syst.*, 2010.

[61] L. O. Chua and S. M. Kang, "Memristive Devices and Systems," *Proc. IEEE* vol. 64, No. 2, pp. 209-223, 1976.

[62] G. S. Snider, "Spike-Timing-Dependent Learning in Memristive Nanodevices," *IEEE Int. Symp. Nano Architectures,* pp. 85-92, June 2008.

[63] W. Gerstner, R. Ritz, J. L. Hemmen, "Why spikes? Hebbian learning and retrieval of time-resolved excitation patterns," *Biological Cybernetics,* 69, 503-515, 1993.

[64] T. Masquelier and S. Thorpe, "Unsupervised Learning of Visual Features through Spike Timing Dependent Plasticity," *Plos Comp. Biology,* 3(2): e31. doi:10.1371/journal.pcbi.0030031, 2007.

[65] T. Masquelier, R. Guyonneau, and S. Thorpe, "Spike Timing Dependent Plasticity Finds the Start of Repeating Patterns in Continuous Spike Trains," PLoS ONE 3(1): e1377. doi:10.1371/journal.pone.0001377, 2008.

[66] T. Masquelier, R. Guyonneau, and S. Thorpe, "Competitive STDP-Based Spike Pattern Learning," Neural Comp. 21, 1259-1276, 2009.

[67] T. Masquelier, E. Hugues, G. Deco, and S. Thorpe, "Oscillations, Phase-of-Firing Coding, and Spike Timing-Dependent Plasticity: An Efficient Learning Scheme", J. of Neuroscience, 2009. 29 (43): 13484-93.

[68] G. Bi and M. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type," *J. Neurosci.,* 18(24), 10464-10472, 1998.