Chapter 22

BIOINSPIRED STIMULUS ENCODER FOR CORTICAL VISUAL NEUROPROSTHESES

Leonel Sousa, Pedro Tomás
Department of Electrical and Computer Engineering, IST/INESC-ID, Portugal
las@inesc-id.pt, pfzt@sips.inesc-id.pt

Francisco Pelayo, Antonio Martinez, Christian A. Morillas, Samuel Romero
Department of Computer Architecture and Technology, University of Granada, Spain
fpelayo@ugr.es, {amartinez, cmorillas, sromero}@atc.ugr.es

Abstract This work proposes a real-time bioinspired visual encoding system for multielectrodes stimulation of the visual cortex supported on Field Programmable Logic. This system includes the spatio-temporal preprocessing stage and the generation of time-varying spike patterns to stimulate an array of microelectrodes and can be applied to build a portable visual neuroprosthesis. It only requires a small amount of hardware thanks to the high operating frequency of modern FPGAs, which allows to sequentialize some of the required processing. Experimental results show that, with the proposed architecture, a real-time visual encoding system can be implemented in FPGAs with modest capacity.

Keywords: Visual cortical-neuron stimulation, retina model, FPL implementation, visual information processing

1. Introduction

Nowadays, the design and the development of visual neuroprostheses interfaced with the visual cortex is being tried to provide a limited but useful visual sense to profoundly blind people. The work presented has been carried out within the EC project "Cortical Visual Neuroprosthesis for the Blind" (CORTIVIS), which is one of the research initiatives for developing a visual neuroprosthesis for the blind [1; 2].

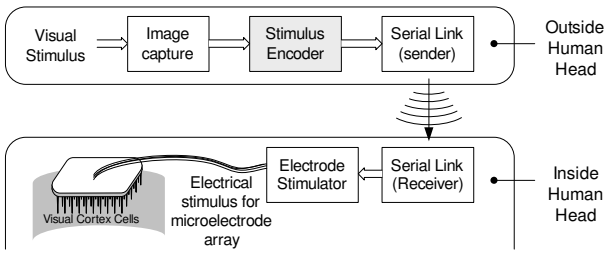


Figure 22.1. Cortical visual neuroprosthesis

A block diagram of the cortical visual neuroprosthesis is presented in fig. 23.1. It includes a programmable visual stimulus encoder, which computes a predefined retina model, to process the input visual stimulus and to produce output patterns that approximate the spatial and temporal spike distributions required for effective cortical stimulation. These output patterns are represented by pulses that are mapped on the primary visual cortex and are coded by using Address Event Representation [3]. The corresponding signals are modulated and sent through a Radio Frequency (RF) link, which also carries power, to the electrode stimulator. This project uses the Utah microelectrode array [4], which consists on an array of 10×10 silicon microelectrodes separated by about 400 μm in each orthogonal direction (arrays of 25×25 microelectrodes are also considered). From experimental measures on biological systems, it can be established a time of 1 ms to "refresh" all the spiking neurons, which means an average time slot of 10 μs dedicated to each microelectrode. The RF link bandwidth allows communication at a bit-rate of about 1 Mbps, which means an average value of 10 kbps for each microelectrode in a small size array of 10×10 electrodes or about 1.6 kbps for the 25×25 microelectrode array.

The work presented herein addresses the design of digital processors for implementing the block shown shaded in fig. 23.1 and extended in figure 23.2 in Field Programmable Logic (FPL) technology. The adopted retina model is a complete approximation of the spatio-temporal receptive fields characteristic response of the retina ganglion cells. It also includes two other blocks. A leaky integrate-and-fire block, that generates the spikes to stimulate the cortical cells, i.e. the neuromorphic pulse coding. A spike channel interface block which uses the Address Event Representation (AER) protocol to communicate information about the characteristics of spikes and addresses of target microelectrodes without timestamps. The architecture of the system has been designed taking into account the specifications of the problem described above and the technical characteristics of modern Field Programmable Gate Array (FPGA) devices. Experimental results show that a complete artificial model that generates neuromorphic pulse-coded signals can be implemented in real-time even in FPGAs with low-capacity.

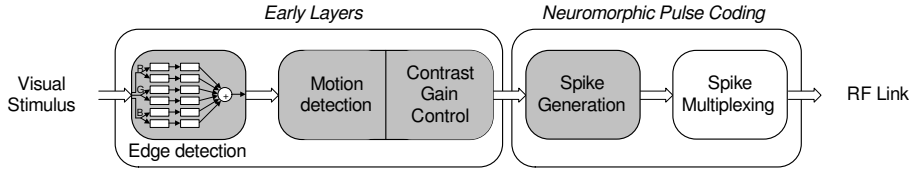


Figure 22.2. Retina early layers

The rest of this paper is organized as follows. The architecture for modelling the retina and for coding the event lists that will be carried out to the visual cortex is presented in section 2. Section 3 describes the computational architectures designed for implementing the retina model in FPL, discussing their characteristics and suitability to the technology. Section 4 presents experimental results obtained by implementing the stimulus encoder on a FPGA and section 5 concludes the presentation.

2. Model Architecture

The neuron layers of the human retina perform a set of different tasks, which culminate in the spiking of ganglion cells at the output layer [5]. These cells have different transient responses, receptive fields and chromatic sensibilities. The system developed implements the full model of the retina, which includes all the processing layers, plus the protocol for carrying the spikes to the visual cortex in a serial way through a RF link.

The visual stimulus encoder includes two bio-inspired blocks which are shown in grey in fig. 23.2. The first is an edge detection block based on a set of weighted Gaussian spatial filters dependent on time through low pass filters. The second block consists of a local contrast gain controller (CGC) which attempts to compensate for the biological delay of the visual signal processing and transmission. The spike generation block produces the spike events which will stimulate the visual cortex.

2.1 Retina early layers

The presented retina computational model is based on research published in [5; 6; 7; 8] and was extended to the chromatic domain by considering independent filters for the basic colors.

The first filtering element, the retina early layers (see fig. 23.2) is an edge detector composed of a set of two Gaussian spatial filters per color channel, parameterized by a gain a_{ij} and a standard deviation σ_{ij}, as shown in equation 23.1.

$$g_i(\mathbf{r}) = \frac{\mathbf{a_i}}{\sqrt{\mathbf{2}\pi}\sigma_\mathbf{i}} \mathbf{e}^{-\frac{\mathbf{r^2}}{\mathbf{2}\sigma_\mathbf{i}^\mathbf{2}}} \tag{22.1}$$

In order to detect edges the Gaussian filters of each color channel are parameterized with opposite signs and different standard deviations. The output is then processed by a set of temporal first-order low pass filters:

$$h_{ij}(t) = H(t) \cdot B_{ij} e^{-B_{ij} \cdot t} \tag{22.2}$$

where $H(t)$ represents the Heaviside step function and B_{ij} is the decay rate. The output of the edge detection block is:

$$m(\mathbf{r}, t) = \sum_{i=R,G,B} s_i(\mathbf{r}, t) * \sum_{j=1}^{2} g_{ij}(\mathbf{r}) \cdot h_{ij}(t) \tag{22.3}$$

where $*$ denotes the convolution operator and $\{s_R, s_G, s_B\}$ are the color components of the visual stimuli. The bilinear approximation was applied to derive a digital version of the time filters represented in the Laplace domain [9]. It leads to first order Infinite Impulse Response (IIR) digital filters with input $v_{ij}[\mathbf{q}, t]$ and output $w_{ij}[\mathbf{q}, t]$ which can be represented by equation 23.4.

$$w_{ij}[\mathbf{q}, n] = b_{LPij} \times w_{ij}[\mathbf{q}, n-1] + c_{LPij} \times (v_{ij}[\mathbf{q}, n] + v_{ij}[\mathbf{q}, n-1]) \tag{22.4}$$

The second filtering block of the visual stimulus encoder is a motion detector based on a temporal high pass filter defined by a pole in $-\alpha$ whose impulse response is represented in equation 23.5.

$$h_{HP}(t) = \delta(t) - \alpha H(t) e^{-\alpha t} \tag{22.5}$$

The filter was also represented in the digital domain by applying the bilinear approximation. The result is a first order Infinite Impulse Response (IIR) digital filter which can be represented by equation 23.6 where $m[\mathbf{q}, n]$ is the filter input and $u[\mathbf{q}, n]$ is the output.

$$u[\mathbf{q}, n] = b_{HP} \times u[\mathbf{q}, n-1] + c_{HP} \times (m[\mathbf{q}, n] - m[\mathbf{q}, n-1]) \tag{22.6}$$

The resulting activation u(**r**,t) is multiplied by a Contrast Gain Control (CGC) modulation factor $g(\mathbf{r}, t)$ and rectified to yield the firing rate of the ganglion cells response to the input stimuli.

The CGC models the strong modulatory effect exerted by stimulus contrast. The CGC non-linear approach is also used in order to model the "motion anticipation" effect observed on experiments with a continuous moving bar [7]. The CGC feedback loop involves a low-pass temporal filter with the following impulse response:

$$h_{LP}(t) = B e^{-\frac{t}{\tau}} \tag{22.7}$$

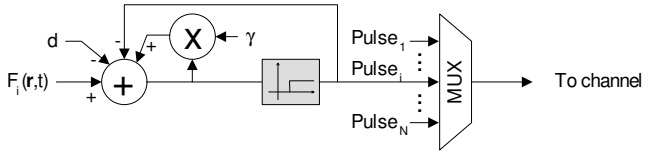


Figure 22.3. Neuromorphic pulse coding

where B and τ define the strength and constant time of the CGC. The filter is computed in the digital domain, by applying the same approximation as for the high-pass filter, by equation 23.8.

$$z[\mathbf{q}, n] = b_{LP} \times z[\mathbf{q}, n-1] + c_{HP} \times (y[\mathbf{q}, n] + y[\mathbf{q}, n-1]) \tag{22.8}$$

and the output of the filter is transformed into a local modulation factor (g) via the non-linear function:

$$k(t) = \frac{1}{1 + [v(t) \cdot H(v(t))]^4} \tag{22.9}$$

The output is rectified by using the function expressed in eq. 23.10:

$$F_i(\mathbf{r}, t) = \psi H(y(\mathbf{r}, t) + \theta)[y(\mathbf{r}, t) + \theta] \tag{22.10}$$

where ψ and θ define the scale and baseline value of the firing rate $f_i(\mathbf{r}, t)$.

The system to be developed is fully programmable and typical values for all parameters of the model are found in [6]. The model has been completely simulated in MATLAB, by using the *Retiner* environment for testing retina models [10].

2.2 Neuromorphic pulse coding

The neuromorphic pulse coding block converts the continuous-varying time representation of the signals produced in the early layers of the retina into a neural pulse representation. In this new representation the signal provides new information only when a new pulse begins. The model adopted is a simplified version of an integrate-and-fire spiking neuron [11]. As represented in fig. 23.3, the neuron accumulates input values from the respective receptive field (output firing rate determined by retina early layers) until it reaches a given threshold. Then it fires and discharges the accumulated value. A leakage term is included to force the accumulated value to diminish for low or null input values. The pulses are then generated accordingly to equations 23.11 and 23.12

$$P_{acc}[\mathbf{q}, n] = F[\mathbf{q}, n] + \gamma \cdot P_{acc}[\mathbf{q}, n-1] - pulse[\mathbf{q}, n-1] - d \tag{22.11}$$

$$pulse[\mathbf{q}, n] = H(P_{acc}[\mathbf{q}, n] - \phi) \tag{22.12}$$

where P_{acc} is the accumulated value, F is the early layer's output, $\gamma < 1$ sets the decay rate of P_{acc} ($decay = 1 - \gamma$) and H represents the Heaviside step function. The circuit fires a pulse whenever the accumulated value P_{acc} is bigger then the threshold ϕ.

The amplitude and duration of the pulses are then coded using AER, which is represented in a simplified way in fig. 23.3 by a multiplexer. This information is then sent to the corresponding microelectrodes via the RF link. An event consists of an asynchronous bus transaction that carries the address of the corresponding microelectrode and is sent at the moment of pulse onset (no timestamp information is communicated). An arbitration circuit is required at the sender side and the receiver has to be listening to the data link with a constant latency.

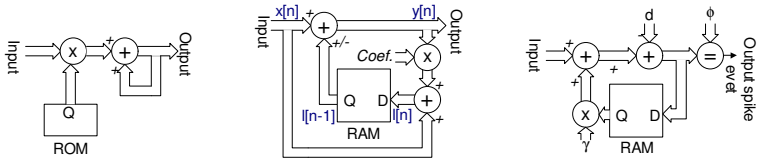
3. FPL Implementation

This section discusses the implementation in FPL technology of the visual encoding system presented in the previous section. The usage of FPL technology will allow changing the model parameters without the need of designing another circuit. This has special importance since different patients have different sight parameters therefore requiring adjustments to the artificial retina. FPL may also allow changing model blocks if new information on how visual stimuli is processed reveals the need to introduce new filtering blocks.

For the design of the system, different computational architectures were considered both for the retina early layers and for the neuromorphic coding of the pulses. These architectures lead to implementations with different characteristics, in terms of hardware requirements and speed. The scalability and programmability of the system are also relevant aspects that are taken into account for FPL implementations.

3.1 The retina early layers

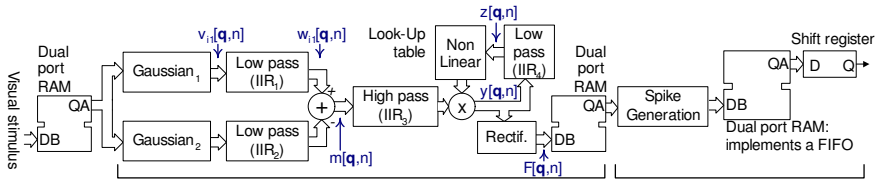
There can be multiple approaches to implementing the retina early layers each of which involve different hardware requirements, so the first step would be to analyze the necessary hardware to build the desired processor. Assuming a typical convolutional kernel of 7×7 elements for the Gaussian spatial filters and that high-pass and low-pass temporal filters are computed by using the difference equations 23.4, 23.6 and 23.8, respectively, then 104 multiplications and 107 additions per image cell are required for just one spatial channel. For a matrix of 100 microelectrodes, the hardware required by a fully parallel architecture makes it impractical in FPL technology. Therefore, the usage of the hardware has to be multiplexed in time, but the temporal restriction of processing the whole matrix with a frequency up to 100Hz must also be fulfilled. This restriction is however wide enough to consider the processing of cells with



(a) Gaussian filter block.

(b) Infinite Impulse Filter (IIR) block.

(c) Spike generation block.



(d) Global architecture.

Figure 22.4. Detailed diagram for the visual encoder computational architecture.

a full multiplexing schema inside each main block of the architecture model presented in section 2: edge detection and motion detection and anticipation.

The architectures of these blocks can then be described as shown in figure 23.4, where each main block includes one or more RAM blocks to save processed frames in intermediate points of the system in order to compute the recursive time filters. At the input, dual-port RAMs are used for the three color channels, while a ROM is used to store the coefficient tables for the Gaussian spatial filters. The operation of the processing modules is locally controlled and the overall operation of the system is performed by integrating the local control in a global control circuit. The control circuit is synthesized as a Moore state machine, because there is no need to modify the output asynchronously with variations of the inputs. All RAM address signals, not shown in the figure 23.4, are generated by the control circuit.

The Gaussian filter module (see fig. 23.4(a)) calculates the 2D convolution between the input image and the predefined matrix coefficients stored in the ROM. It operates in a sequential method, where pixels are processed one at a time in a row major order and an accumulator is used to store the intermediate and final results of the convolution. The local control circuit stores the initial pixel address, row and column, and then successively addresses all the data, and respective filtered coefficients stored in the ROM, required for the computation of the convolution. After multiplying each filter coefficient the resulting value is successively accumulated. When the calculation is finished for a given pixel (see fig. 23.4(d)), the value is sent to the low pass filter. The filter (see fig. 23.4(b)) is computed in a transposed form where the output is calculated by adding the

input $x[n]$ to the stored value $l[n]$ which was previously computed as

$$l[n-1] = b_{LPij} \cdot y[n-1] + x[n-1] \tag{22.13}$$

The input scaling by C_{LPij} as shown in equation 23.4 is made with the Gaussian filter coefficient matrix. The output of the low pass filters of the different color channels is then added accordingly to eq. 23.3 and the result, $m[\mathbf{q}, n]$, is processed by a temporal high pass filter module also implemented with the IIR module of fig. 23.4(b) before being processed by the CGC block. This module consists on a low pass filter (eq. 23.8) and a non-linear function (eq. 23.9) which is computed by using a lookup table stored in a RAM block. The low-pass filter circuit is again implemented with the IIR module of fig. 23.4(b). The last processing module is a rectifier which clips the signal whenever its value decreases below a predefined threshold. It is implemented by a simple binary comparator whose output is used to control a multiplexer.

The overall circuit operates in a 4 stage pipeline corresponding to each one of the main processing blocks. Only the first pipeline stage requires a variable number of clock cycles depending on the dimension of the filter kernel–49 cycles for a 7×7 kernel. Each of the two other pipeline stages is solved in a single clock cycle.

3.2 Neuromorphic pulse coding

Fig. 23.4(d) shows two processing modules associated with the neuromorphic pulse coding: *i)* for pulse generation and its representation and *ii)* to arbitrate the access to the serial bus (the input to the serial link in figure 23.1). This block is connected to the retina early layers through a dual port RAM (CGC block in fig. 23.4) where one writes data onto one port and the other reads it from the second port.

The pulse generation circuit, which converts from firing rate to pulses, can be seen as a simple Digital Voltage Controlled Oscillator (DVCO) working in a two clock cycle stage pipeline. In the first stage the input firing rate is added to the accumulated value. In the second stage a leakage value is subtracted and, if the result is bigger than the threshold ϕ, a pulse is fired and the accumulator returns to the zero value (see fig. 23.4(c)). Note that in this architecture the accumulator circuit is made with a RAM, since it corresponds to a single adder and multiple accumulator registers for the different microelectrodes.

AER was used to represent the pulses while the information is serialized. In a first approach, the architecture consisted of an arbitration tree to multiplex the onset of events (spikes) onto a single bus. In this tree, we check if one or more inputs has a pulse to be sent and arbitrate the access to the bus through the request/acknowledge signals in the tree [3]. This tree consists of multiple sub-trees, where registers can be used for buffering the signals between them. This architecture is not scalable, since it requires a great amount of hardware and is

not a solution when arrays with a great number of microelectrodes are used (see section 4). To overcome this problem, and since the circuit for pulse generation is intrinsically sequential, a First In First Out (FIFO) memory is used to register the generated spikes for the microelectrodes. Only one pulse is generated in a clock cycle and information about it is stored in the FIFO memory because requests may find the communication link busy. This implementation has the advantage of not increasing the hardware resources required with the increase in the number of microelectrodes, but its performance is more dependent on the channel characteristics since it increases the maximum wait time to send a pulse.

The FIFO is implemented by using a dual-port data RAM, where input is stored in one port and, at the other port, the pulse request is sent to the channel. Also to avoid overwriting when reading data from the FIFO a shift register is used to interface the data output of the RAM with the communication channel.

4. Experimental Results

The visual encoding system was described in VHDL and exhaustively tested on a Digilent DIGILAB II board based on a XILINX SPARTAN XC2S200 FPGA [12] with modest capacity. The synthesis tool used was the one supplied by the FPGA manufacturer, the ISE WebPACK 5. This FPGA has modest capacities, with 56 kbit of block-RAM and 2352 slices, corresponding to a total of $200,000$ system-gates. The functional validation of the circuits was carried out by comparing the results obtained with MATLAB Retiner [10].

For the test and experimental results presented in this section, only one color channel is considered and the input signal is represented in 8-bit grayscale. However, internally 12-bit signals are used in order to reduce discretization errors. The dimension of the microelectrode array is 100.

Analyzing the results in table 23.1 it is clearly seen that the retina early layers do not require many FPGA slices but consumes a significant amount of memory. The synthesis of the circuit for an array of 1024 microelectrodes shows a similar percentage of FPGA slice occupation but it uses all the 14 RAM blocks supplied by the SPARTAN XC2S200. In terms of time restrictions, the solution works very well since it can process the array of input pixels in a short time, about 0.1ms for 100 microelectrodes, which is much lower than the specified maximum of 10ms. In fact this architecture allows to process movies at frame rate of 100 Hz with $10,000$ pixels per image without increasing the number of slices used.

The analysis of the AER translation method has however different aspects. For a typical matrix of 100 microelectrodes, the tree solution (with or without intermediate registers) is a valid one as the resource occupation is low. However, the increase in the number of microelectrodes implies the usage of a great

Table 22.1. Retina encoding system implemented on a SPARTAN XC2S200 FPGA

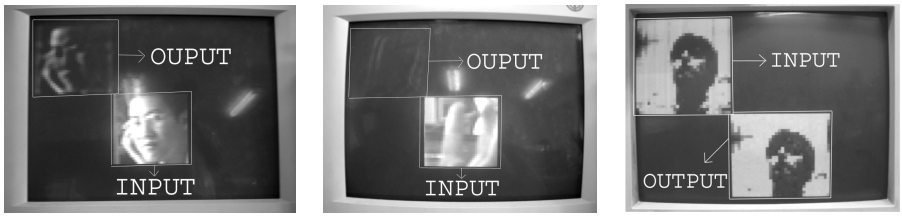
Block	*Number of microelectrodes*	*Slice Occupation*	*Maximum clock frequency*	*Number of RAM blocks used*
Retina early layers	100	21%	47MHz (49 cc*)	5
	1024	21%	47MHz (49 cc*)	10
Spike Generation	100	2%	51MHz	1
	1024	2%	51MHz	3**
AER	100	9%	99MHz	0
Registered Tree	256	17%	98MHz	0
	512	37%	93MHz	0
AER	100	10%	64MHz	0
Unregistered Tree	256	21%	63MHz	0
	512	45%	57MHz	0
FIFO AER	***	5%	75MHz	1

* 49 clock cycles are required for processing a pixel
** implemented as distributed RAM in the overall system occupying 33% extra slices
*** not dependent of the number of micro-electrodes

amount of hardware and this solution becomes impracticable for FPL technology. In that case, the proposed FIFO based solution has the great advantage of accommodating a great number of electrodes with a small amount of hardware and just one RAM block. With a channel bandwidth of about 1 Mbps, the FIFO based AER circuit does not introduce any temporal restrictions in the firing rate. Even when operating at a lower clock frequency than that admitted by the retina early layers circuit, e.g. 40 MHz, the FIFO based circuit is able to process 100 requests in about 2.5 μs which is much less than the value of 1 ms initially specified and also much less than the time required to send the information through the channel.

In table 23.1 results are individually presented for the retina early layers and for the neuromorphic pulse coding circuits. The overall system was also synthesized and implemented in a SPARTAN XC2S200 FPGA. In this case the Neuromorphic Pulse Coding RAM block had to be implemented in distributed RAM due to the low memory capacity of the FPGA . A clock frequency greater than 40 MHz is achieved by using about 28% of the slices and 12 of the total of 14 RAM-blocks, for 100 microelectrodes.

In order to test the visual encoding module a system prototype was developed based on two XILINX XC2S200 FPGAs (one to implement the visual encoder and the other to implement the capture of image frames and visualization of the system output), one digital output color camera module, model C3188A [13]



(a) Edge detecting. (b) Early Layers. (c) Spike generation.

Figure 22.5. Testing of the visual encoding module for a microelectrode array of 32×32.

which captures images in QVGA at a frame rate up to 60Hz and one standard monitor. Figure 23.5 shows some of the experimental results of the Visual Encoding implementation: fig. 23.5(a) show the result of the edge detecting block, fig. 23.5(b) shows the output of the Early Layers module, where moving fingers caused a non-zero output. Fig. 23.5(c) shows the output of the neuromorphic pulse coding module after image restoration with a second order low pass filter with poles at 6Hz.

5. Conclusions

The present work proposes a computational architecture for implementing a complete model of an artificial retina in FPL technology. The system is devised to stimulate a number of intra-cortical implanted microelectrodes for a visual neuroprosthesis. It performs a bio-inspired processing and encoding of the input visual information. The proposed processing architecture is designed to fulfill the constraints imposed by the telemetry system to be used, and with the requirement of building a compact and portable hardware solution.

The synthesis results demonstrate how the adopted pipelined and time multiplexed approach makes the whole system fit well on a relatively low complexity FPL circuit, while real-time processing is achieved. The use of FPL is also very convenient to easily customize (re-configure) the visual pre-processing and encoding system for each implanted patient.

Acknowledgments

This work has been supported by the European Commission under the project CORTIVIS ("Cortical Visual Neuroprosthesis for the Blind", QLK6-CT-2001-00279).

References

[1] Cortical visual neuro-prosthesis for the blind (cortivis): http://cortivis.umh.es.

[2] Ahnelt P., Ammermüller J., Pelayo F., Bongard M., Palomar D., Piedade M., Ferrandez J., Borg-Graham L., and Fernandez E. Neuroscientific basis for the design and development of a bioinspired visual processing front-end. In *Proc. of IFMBE*, pages 1692–1693, Vienna, 2002.

[3] Lazzaro J. and Wawrzynek J. A multi-sender asynchronous extension to the address event protocol. In *Proc. of 16th Conference on Advanced Research in VLSI*, pages 158–169, 1995.

[4] Maynard E. The Utah intracortical electrode array: a recording structure for potential brain-computer interfaces. *Elec. Clin. Neurophysiol.*, 102:228–239, 1997.

[5] Wandell Brian. *Foundations of Vision: Behavior, Neuroscience and Computation*. Sinauer Associates, 1995.

[6] Wilke S., Thiel A., Eurich C., Greschner M., Bongard M., Ammermüler J., and Schwegler H. Population coding of motion patterns in the early visual system. *J. Comp Physiol A*, 187:549–558, 2001.

[7] Berry M., Brivanlou I., Jordan T., and Meister M. Anticipation of moving stimuli by the retina. *Nature (Lond)*, 398:334–338, 1999.

[8] Markus Meister and Michael J. Berry II. The neural code of the retina. *Neuron*, 22:435–450, March 1999.

[9] Oppenheim A. and Willsky A. *Signal and Systems*. Prentice Hall, 1983.

[10] F. Pelayo, A. Martínez, C. Morillas, S. Romero, L. Sousa, and P. Tomás. Retina-like processing and coding platform for cortical neuro-stimulation. In *Proc. of 25th Annual IEEE International Conference of the Engineering in Medicine and Biology Society*, IEEE Catalog number 03CH37439C, pages 2023–2026, Cancun, Mexico, September 2003.

[11] Gerstner W. and Kistler W. *Spiking Neuron Models*. Cambridge University Press, 2002.

[12] XILINX. *Spartan-II 2.5V FPGA Family: Functional Description*, 2001. Product Specification.

[13] Quasar Electronics Ltd. C3188A - Digital Output Colour Camera Module. http://www.electronic-kits-and-projects.com/kit-files/cameras/dc3188a.pdf.