Inter-spikes-intervals exponential and gamma distributions study of neuron firing rate for SVITE motor control model on FPGA

Fernando Pérez-Peña [a], Arturo Morgado-Estévez [a], Alejandro Linares-Barranco [b]

[a] *Computer Architecture and Technology Area, Universidad de Cádiz, School of Engineering, Calle Chile, 1, Cadiz 11002, Spain*
[b] *Robotic and Technology of Computers Lab (RTC), Universidad de Sevilla, ETSI Informática, Avd. Reina Mercedes s/n, Seville 41012, Spain*

A B S T R A C T

This paper presents a statistical study on a neuro-inspired spike-based implementation of the Vector-Integration-To-End-Point motor controller (SVITE) and compares its deterministic neuron-model stream of spikes with a proposed modification that converts the model, and thus the controller, in a Poisson like spike stream distribution. A set of hardware pseudo-random numbers generators, based on a Linear Feedback Shift Register (LFSR), have been introduced in the neuron-model so that they reach a closer biological neuron behavior. To validate the new neuron-model behavior a comparison between the Inter-Spikes-Interval empirical data and the Exponential and Gamma distributions has been carried out using the Kolmogorov–Smirnoff test. An in-hardware validation of the controller has been performed in a Spartan6 FPGA to drive directly with spikes DC motors from robotics to study the behavior and viability of the modified controller with random components.

The results show that the original deterministic spikes distribution of the controller blocks can be swapped with Poisson distributions using 30-bit LFSRs. The comparative between the usable controlling signals such as the trajectory and the speed profile using a deterministic and the new controller show a standard deviation of 11.53 spikes/s and 3.86 spikes/s respectively. These rates do not affect our system because, within Pulse Frequency Modulation, in order to drive the motors, time length can be fixed to spread the spikes. Tuning this value, the slow rates could be filtered by the motor. Therefore, this SVITE neuro-inspired controller can be integrated within complex neuromorphic architectures with Poisson-like neurons.

Keywords:
Bio-inspired
Neuro-inspired
AER
LFSR
Poisson
FPGA

1. Introduction

The main goal of the neuromorphic engineering research field is to develop hardware devices based on the principles of the human nervous system [1]. The term “Neuromorphic Engineering” was first coined by Caver Mead in the late eighties [2]. He started mimicking the behavior of neuron cells by using Very-Large-Scale-Integration (VLSI) chips. Then, Sivilotti [3] defined the commu-nication protocol to be used between those devices: Address Event Representation (AER). AER enables the communication of thou-sands of neurons from one chip to another. Within AER, each neuron is given an address to identify along the architecture. All the devices were expected to be connected with that AER bus.

The most popular options to implement the end neuromorphic devices into hardware are these three: the use of a full custom VLSI design application-specific integrated circuit (ASIC) [4], a Field Programmable Gate Array (FPGA) [5,6] or a Field-Programmable Analog Array (FPAA) [7]. Up to these days, many neuron models [8], large scale architectures [9] and sensory devices such as vision [10] or cochlea sensors [11] have been developed. However, the challenge today is to bridge the gap between sensors and large architectures to reach an accurate spiking motor control.

An FPGA design based on neurons known as Integrate and Generate (I&G) which is described in [12] was used in this study. I&G neuron includes one pre and postsynaptic connection and an integrator which computes the ongoing spikes. It models the activity level of the neuron; its firing rate depends on the integrated value. In its basic model, the integrated value is updated for each incoming spike by an increment or a decrement of the membrane potential (MP), modeled by a counter, depending of the polarity of the received spike. If the spike is positive MP is incremented, but it is decremented when the spike is negative. Current MP is continuously translated into an output stream of spikes. The distribution of these output stream is deterministic in the basic model, but it can be easily modified to obtain new distributions. This model is similar to LIF model but it allows to modify the distribution over time of the outgoing spikes.

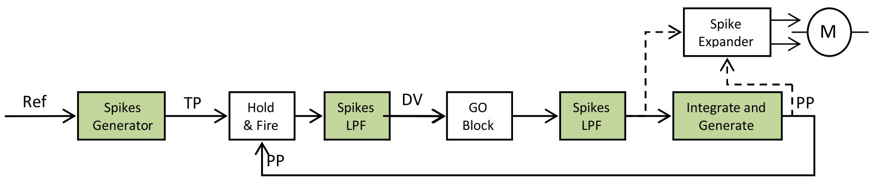

Fig. 1. Block diagram of the neurocontroller where the Spike-based Vector Integration To-End Point (SVITE) algorithm is included. All the information flows as spikes. The Hold & Fire block computes the subtraction of two spike streams: current commanded position and target position (reference in spike domain). Green blocks represent where pseudo-random behavior has been included using LFSR. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The behavior is currently described in VHDL and implemented for FPGA. This model has a completely deterministic output firing rate (Eq. 4), without any random component, which fits with the aim of spike based motor controlling [13].

In principle, for motor controlling it does not seem feasible to have random spikes distributions inside the controller. However, in this paper we present a study where pseudo-random distributions of spikes have replaced deterministic ones in the motor controller. Statistical models are presented and the results show that random distributions can be used for spike-based motor control.

There are *in-vivo* experiments that show variability at the firing rate pattern if a constant stimulus is presented within different tests at different times [14]. This behavior is extremely problematic if the current trend is to match the neuron response per each stimulus presented to elaborate a map between stimuli and firing patterns. Regarding this view, [15] points out that if the stimulus presented is fluctuating, the neuron will produce a precise spike timing response.

For a deterministic spike distribution motor controller, if we consider this controller as an isolated part of the architecture, there would not be any noisy or random spikes. However, if we consider this controller as part of large neuromorphic architectures, we must face these different possible distributions of spikes and try to minimize their effects or at least predict the behavior of the motor controller and act in accordance with them.

The next paragraphs include a review on the current spike-based neuro-inspired motor controller.

The neurocontroller used in this study for deterministic behavior is presented in [13]. It is based on an algorithm proposed by Professor Grossberg [16], the Vector Integration To-End Point (VITE). It was translated into the spikes paradigm by using existing building blocks in the literature [17,18]. The block diagram is shown in Fig. 1, where a digital reference signal is converted into spikes (Spikes Generator). These spikes arrive to a Hold&Fire block, which is in charge of subtracting those spikes coming from the motor behavior from this reference translated into a spike stream. The resulting spike stream is low pass filtered by a spike-based low pass filter (Spikes LPF) block. Then, a special GO block is used to enable and speed up the movement. Finally, another spike-based low-pass filter and an Integrate and Generate neuron (I&G) offer a final uniformly distributed spike stream output, as explained deeply in [13]. It must be pointed out that there are two signals available for motor control: the one at the output of the second LPF that represents the speed profile, or the one at the output of the last block (I&G), which is the commanded position to reach. It will depend on the desired parameter to be controlled (speed of a motor or position of a robotic joint). Another important point is the GO block; it triggers the movement and allows controlling the speed. It injects spikes according to a growing rate fixed with time (like the slope of a ramp). The behavior of this GO block evidences the dependence of the firing rate with time.

As described in Ref. [17], and shown in Fig. 2, each LPF block is composed by an I&G block with a feed-back loop using another

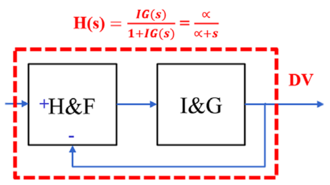

Fig. 2. Spikes Low-Pass-Filter decomposition into I&G neuron and its Laplace Transfer function [17]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

H&F block. In such a configuration, the Laplace transfer function of the system correspond to a low-pass-filter.

The advantages of this neuro-inspired controller are that it is very efficient regarding power and hardware resources consumption (3.4 W and 2.5% of the FPGA[1] respectively) and it uses only five neurons. It can also be replicated: one Spike-based VITE for each motor or degree of freedom you want to control.

In contrast, the disadvantages are that it applies an open-loop control. Thereby, the problem of motor inertia is present and this could be a mismatch between the order commanded and the position reached. Both issues are being solved within the ongoing research.

However, the main advantage of the neuro-motor-controller is that it uses the spikes to drive the motors. There are many modulation protocols to control the motors, though the most used one is the Pulse Width Modulation (PWM). In our research, and in this study, we used the Pulse Frequency Modulation (PFM), where pulses transmitted to the motor always have the same width, but the frequency will be the parameter, in order to achieve an accurate control.

This way of motor control seems to be the most neuro-inspired one because it is possible to deliver the spikes directly to the motors. Neither computation nor translation is necessary to have motion; this can be achieved just by extending the time length, to the fixed quantity, in order to avoid the spike filter of the motor (this is the reason for including the upper “Spike Expander” block at the block diagram of Fig. 1).Considering a digital clock frequency of 50 MHz, the pulses will last 20 ns with a rate that could be up to mega events per second. This time length and rates have to be adapted before going into the power stage of a DC motor. The power stage fixes the maximum switching frequency and the motor fixes the minimum according to their response. Thus, in theory, the entire system should produce spikes within this range, which will change within each motor.

This method has been established for a low level control of DC motors without delay from the new motor command from the controller to the actuation to the motor. Otherwise with Pulse-Width-Modulation (PWM) for DC motor actuation, or using servomotors, the actuating signal send to the motor/servo has a

[1] The FPGA selected is from Spartan 6 family; device XC6S150LX with 23,038 slices.

fixed period in the order of 20 ms, so we should integrate the output spikes of the neuro-motor controller during a fixed time (those 20 ms) before actuating. Therefore, you were setting a delay from the side of the controller in the system. This delay can have some side effects as over dumped responses [12].

Once the controller is defined, we now propose how we can include the random components inside the blocks of the algorithm to get closer to a more realistic model or at least how they can be included as the edge of large neuromorphic architectures.

As we used deterministic firing patterns, if we include some random elements to enforce the variability at the firing rates, we need to manage the expected spike signal. Several previous studies established that neuron firing rates can often be modeled by a homogeneous Poisson distribution [19,20]. In such kind of process, the average firing rate is constant over time and its times between spikes fit in an exponential distribution. A Poisson distribution is described by the following formula:

$$P_n(t=T)=\frac{(\lambda T)^n}{n!}e^{\lambda T} \tag{1}$$

However, that homogeneous Poisson process is at the front line of our case, in which the algorithm injects spikes in a time dependent manner [15] and a feedback is included. For these cases, there are two options: an inhomogeneous Poisson process where the firing rate varies with time or a renewal process where also time dependence with only the previous spike fired is included [19]. Looking through the inter spike interval, it should follow an exponential or gamma distribution, respectively (2) and (3). We present a methodology to check this.

$$f(x)=\ \lambda e^{-\lambda x} \tag{2}$$

$$f(x)=\frac{\beta^{\alpha}x^{\alpha-1}e^{-\beta x}}{\Gamma(\alpha)} \tag{3}$$

The rest of the paper is structured as follows: In the methodology section, the hardware implementation is briefly explained and the tests performed are exhaustively described. Then, the results and discussion section shows and discusses the results obtained. Finally, the conclusion section sets up the novelties and advantages shown in this study.

2. Methodology

2.1. Introduction

Two methodologies were used: (1) a statistical hypothesis test to check if the inter spike interval (ISI) of the information-carrying signals could be fitted into any well-known and largely studied neuronal distributions like gamma or exponential [19,20]; and (2) a comparison between the signals used for motor controlling [15] to check how long they can be used for their original purpose: motor controlling.

The base line is set as the SVITE algorithm running within a robotic platform [15] shown in Fig. 1. As long as the algorithm uses only deterministic neurons, it is necessary to include random components [21] to reach different behaviors along the tests.

2.2. Setup components: hardware

The algorithm was implemented using the AER Node board. This board includes a Xilinx Spartan-6 LXT 1500 FPGA. It was developed by RTC lab under the VULCANO project[2] and it allows high speed serial AER communications over Rocket IO transceivers, and adaptation to particular scenarios through daughter boards connected on the top. For these tests, the setup includes a daughter board with an USB microcontroller that communicates with the FPGA over Serial Peripheral Interface (SPI). This interface is used to send the parameters needed for each block [12] and to manage the tests.

This main board also runs a massive spikes monitor [22] that addresses each block of the algorithm and does the handshake to communicate, using the AER protocol, with a monitor board [23]. The monitor board receives the spikes and allows them to be processed by the computer using jAER [24] or MATLAB.

2.3. Running tests

Once the system with pseud-random behavior is implemented on the Spartan 6 board, a PC with MATLAB is used to deliver the target position to the algorithm. This position can arrive to the controller in the form of AER coming from a PC or from an hardware AER visual system (cortex inspired) [25]. It is a spatial coordinate, for instance, $x=5$ (we used only one dimension, although it can be replicated to control more complex robotic structures by replicating one SVITE motor-controller per joint in the robot).

This coordinate is feed into the "Spikes Generator" (first block on the left in Fig. 1) and it plays the role of reference. A spike stream is then produced by this block for stimulating the whole controller. The firing behavior produced here is very important since it will determine the spike distribution. Original firing rate (using a deterministic "Spikes Generator" [26]) is described by Eq. (4),

$$r=\frac{f_{CLK}}{2^{NBITS-1}}\times input \tag{4}$$

where f_{CLK} is the clock frequency (50 MHz in our tests), *NBITS* is the amount of bits used to implement the internal spikes counter of the "Spikes Generator" and *input* is the parameter received through the SPI interface (16 bits). For this clock frequency we define as time bin the period of the clock (20 ns) as the minimum update period for neurons' state.

The output of this block has a constant firing rate for a fixed input reference. Thus, this was the main block to be modified in order to include a non-deterministic behavior. The other green blocks of Fig. 1 must also be modified because they include a firing element like the Integrate and Generate (I&G) or LPF block. The firing pattern of these blocks is the same as Eq. (4), but *input* becomes the spikes fired by the previous block.

So, to modify that behavior, we have included a linear feedback shift register (LFSR) for both: "Spikes Generator" and I&G. This component, with the proper XOR feedback, generates a uniform sequence of different pseudo-random values within a range that depends on the number of bits used to implement it. Once all the possible numbers have been generated, this LFSR will repeat the same random sequence over and over again. Also, a time-slice or bin to trigger the output can be configured. With this component, the probability to fire a spike at T will be:

$$P\,(spike=1|T)=\frac{input}{2^{NBITS_{LFSR}}-1} \tag{5}$$

Some specific details depend on the type of block

a. Spikes Generator
   When the time bin is reached the value from the LFSR will be compared with the reference and only if it is lower, the block will fire a spike. Usually, the digital input reference for the spikes generation has 16 bits, so we took this initial number of

[2] VULCANO Project: Ultra-Fast Frame-less Vision by Events. Application to Automation and Anthropomorphic Cognitive Robotics. (TEC 2009-10639-C04-02).

bits to build the LFSR register. From Eq. (5), one can see that there is an inversely proportional relationship between the reference (*input*) and the number of bits of the LFSR ($NBITS_{LFSR}$). A trade-off should be reached. Later on we will measure different behaviors for different LFSR lengths.

b. Integrate & Generate (I&G)
In this case, the comparison with LFSR will be made with the number of spikes counted and only if it is lower, the block will fire a spike.

If we consider Eq. (5) and the algorithm architecture, one can see that if we fix the same LFSR register for each I&G neuron included at the algorithm (one at each spikes low-pass-filter and last I&G), the firing rate will decrease along the blocks because the events (firing a spike) are not independent from the previous one, it is a conditional probability. Furthermore, we have to manage an adequate amount of spikes to allow the motors to run. To reach this behavior, we considered a gradual increase of the LFSR resolution which provokes higher rates of spikes for the useful signals.

2.4. Data processing

At the end of the tests, we have a set of firing rates or spike signals belonging to each block of Fig. 1 for all the combinations performed. First, they are split and each *inter spike interval* (ISI) is calculated. Second, for each ISI, we get the empirical *cumulative distribution function* (CDF) that can be compared with the ideal distribution's CDF. This is the first comparison, the CDF plotted one.

On the other hand, the second comparison is carried out with the Kolmogorov–Smirnov test (KS test) and chi-square test to validate the distribution fitness. We have applied this test to compare how well the observed distribution of ISIs follows the theoretical exponential and gamma distributions. These tests give the *p*-value that allows rejecting or not the null hypothesis. We have considered a *p*-value between 0.05 and 0.1 as reasonable and above 0.1 to accept the hypothesis. Additionally, if the statistical test result is below 5% one can assume that the sample data fitness is good enough. All these statistical studies were conducted using the Statgraphics Centurion software package.

We performed these two comparisons because the K–S test does not show a good performance at Null-hypothesis and *p*-value when the CDF for the theoretical distribution is made up from empirical data [27]. In such cases, the K–S is an alternative statistic available in the test results. It shows the biggest difference between both distributions. Furthermore, with this value and the plotting comparison, a right conclusion can be drawn.

In view of this comparison, it cannot be forgotten that these signals are used for motor control purposes. Therefore, for each modification done to include the random element, we must check if the motor controller performance and accuracy are not lost.

3. Results and discussion

All the results shown in this section were measured on real hardware. There are no simulation results.

3.1. Spikes Generator

If we were considering a bigger time bin, i.e., 0.1 ms, instead of the agreed 20 ns, we would only have an average of 67.4 spikes/s with the maximum digital input reference of 7000 and 20 bits LFSR to reach good distribution fitness for the "Spikes Generator". Therefore the time bin has been fixed to 20 ns. Fig. 3 shows the relation between the digital input references, the size (bits) used for the LFSR register and the goodness of fit for a gamma or exponential distribution for the "Spikes Generator", first block of the control model.

Fig. 3 shows that for small sizes of the LFSR register (such as from 16 to 20 bits), the results of the test are far from the statistical threshold of 5%. If a deterministic source were used, the KS minimum values would be 0.6321 and 0.5243 for the Exponential and Gamma distributions respectively. As the size increases, the results improve. Besides, the higher the reference, the higher the spike rate, and so, the better the result. That matches the probability definition we made in Eq. (5). So, in order to make a decision on which value should be selected, we are going to look at the *p*-value for each approach.

From the point of view of the Kolmogorov–Smirnoff test, the best approximation is got with a 30-bit LFSR, for a Gamma distribution. Looking through the statistical analysis, the *p*-value also indicates that the best fitness was obtained for the Gamma approach (Table 1) and for a 30-bit LFSR. One can think that an Exponential approach is more accurate, because it involves a Poisson process since there is a time independent probability

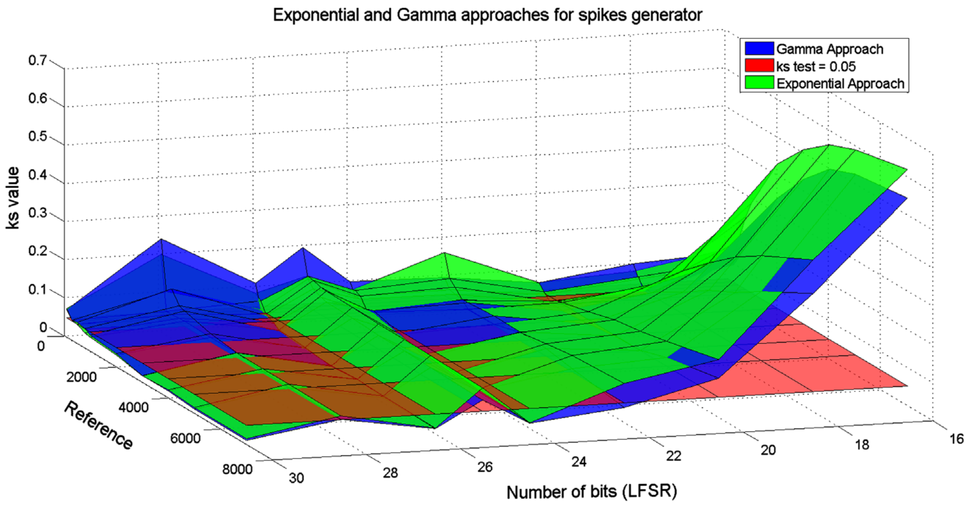

Fig. 3. 3D representation of the KS-value for the gamma and exponential distributions. The red plane is fixed at 0.05 (threshold of the K–S test: passed if the value is below the threshold). The green surface represents the Exponential approach and the blue surface represents the Gamma one. It can be observed that for LFSR resolutions of 24, 26 and 30 bits the test is passed for both approaches (the red plane is visible). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

Table 1
P-value and average firing rate for the exponential and gamma approaches when the LFSR is implemented with 24, 26 and 30 bits.

Ref.	*p*-value (exp)			*p*-value (gamma)			Average(exp) [spikes/s]			Average(gamma) [spikes/s]		
	24 Bits	26 Bits	30 Bits	24 Bits	26 Bits	30 Bits	24 Bits	26 Bits	30 Bits	24 Bits	26 Bits	30 Bits
100	0	0	0.219638	0	0	0.258212	50.57	37.25	2.33	130.47	32.2	3.03
300	0	0	0.316934	0	4.69E-08	0.314826	447.27	111.68	6.94	387.76	125.84	6.62
500	0	0	0.0539985	0	0	0.0239918	745.17	186.29	11.59	718.45	191.92	10.86
800	0	0	0.228294	0	0	0.0519524	1191.95	297.94	18.43	1201.62	300.08	17.15
1000	0	1.23361E-05	0.0189995	0	7.79468E-05	0.00416151	1487.61	372.81	23.32	1527.63	363.5	22.63
2000	0	5.30E-07	0.00278128	0	1.24587E-05	0.000501784	2978.14	745.2	46.55	3119.73	762.55	44.72
3000	0	0.0108252	0.223811	1.6556E-06	0.00204732	0.189354	4462.33	1118.03	69.86	4802.4	1102.85	69.42
4000	1.2414E-06	6.0598E-06	0.0162689	0.00267274	3.18256E-06	0.0587112	5940.60	1489.4	92.95	6422.73	1481.73	89.04
5000	1.7624E-05	0.000336263	0.441025	0.0012794	0.015823	0.8107	7435.44	1861.94	116.29	8013.46	1905.94	113.88
7000	0.00764807	0.00480427	0.134655	0.0197746	0.0138585	0.535872	10409.17	2605.19	163.02	10959.4	2668.41	159.20

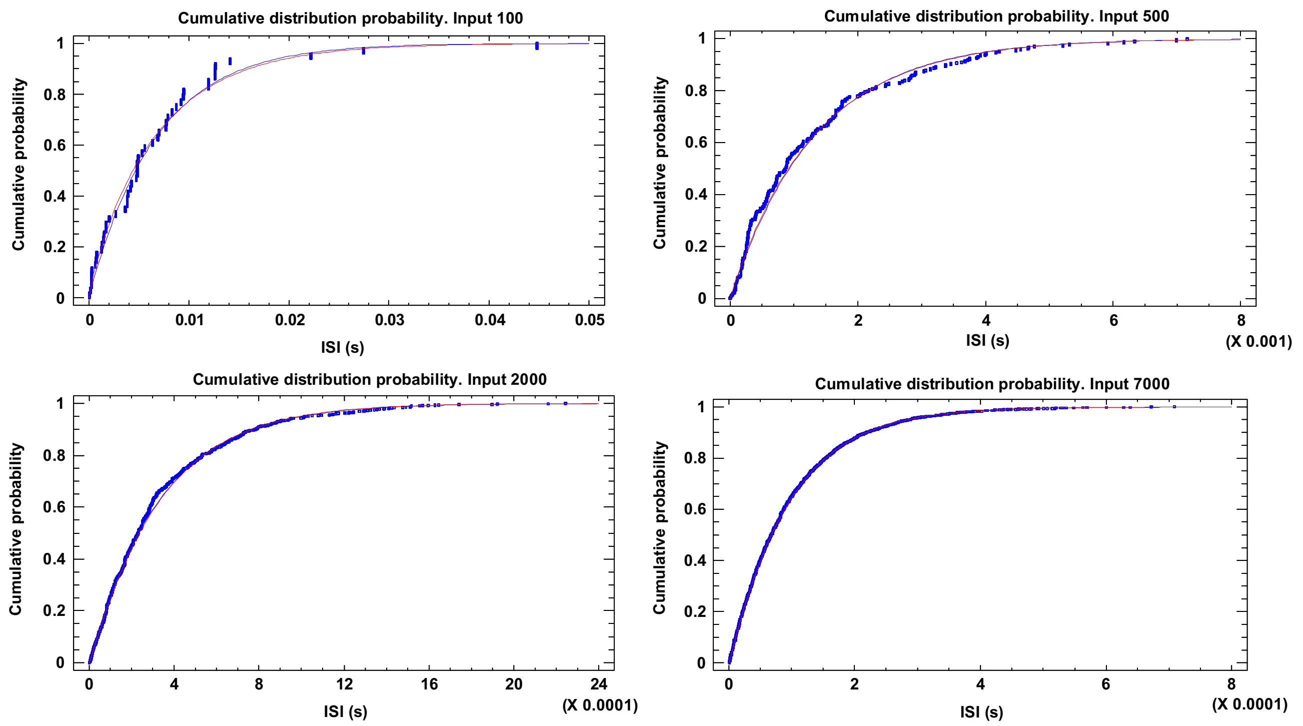

Fig. 4. Cumulative distributions function for gamma and exponential approaches when a 24 bit LFSR and constant digital input reference values of 100, 500, 2000 and 7000.

definition for each event (spike in our case). However, the results show that the value produced by the LFSR register is not purely random, but pseudo-random. Therefore, both approaches are quite similar, as shown in Fig. 3. The final decision is to take a 30-bit LFSR based on the p-value and KS statistical value. Fig. 4 shows also the accuracy for a 24-bit LFSR.

If we go across the table and compute the average value, it results in 0.16 for Exponential and 0.22 for Gamma, for 30 bits value. For 24 and 26 bits, the average *p*-value does not reach 0.01. Looking at columns where the average firing rate is shown, one can see that for 24 bits there are bigger differences between the gamma and exponential approaches and if LFSR size is higher, the differences are lower, reaching at 30 bits value nearly the same rates.

Considering a 30-bit LFSR register, there is a linear relationship between the reference delivered to the block and the output firing rate for the Gamma approach. The maximum firing rate within a 16 bit width reference (65535) is 1.5 K spikes/s.

With the “Spikes Generator” characterized, it is time to use it inside the neuro-motor controller, study the behavior and compare it with the previous deterministic one. Fig. 5 shows the output signals for each block of the algorithm (blue lines). It can be observed that if we are using a random source, the commanded signal does follow the input signal multiplied by 2. That is the reason why there are three different representations in the graph. Eventually, it is possible to use a random “Spikes Generator” to provide the reference to the algorithm.

3.2. *Integrate and generate (I&G)*

We have started using the same LFSR for each I&G neuron. Fig. 6 shows the KS-values for the useful blocks from the motor control point of view, that is the speed profile signal and the commanded position signal. It is demonstrated that adding the same LFSR to all the neurons is not a good option; Fig. 7 shows how the algorithm goes into a non-stable situation and starts oscillating around the goal; eventually, the goal is not reached. The reason for this is that with such a small number of spikes flowing across the algorithm, the inhibited connections (understood as spikes which decrease the integrated value) can have higher and so damaging effects on the whole algorithm. The response of some

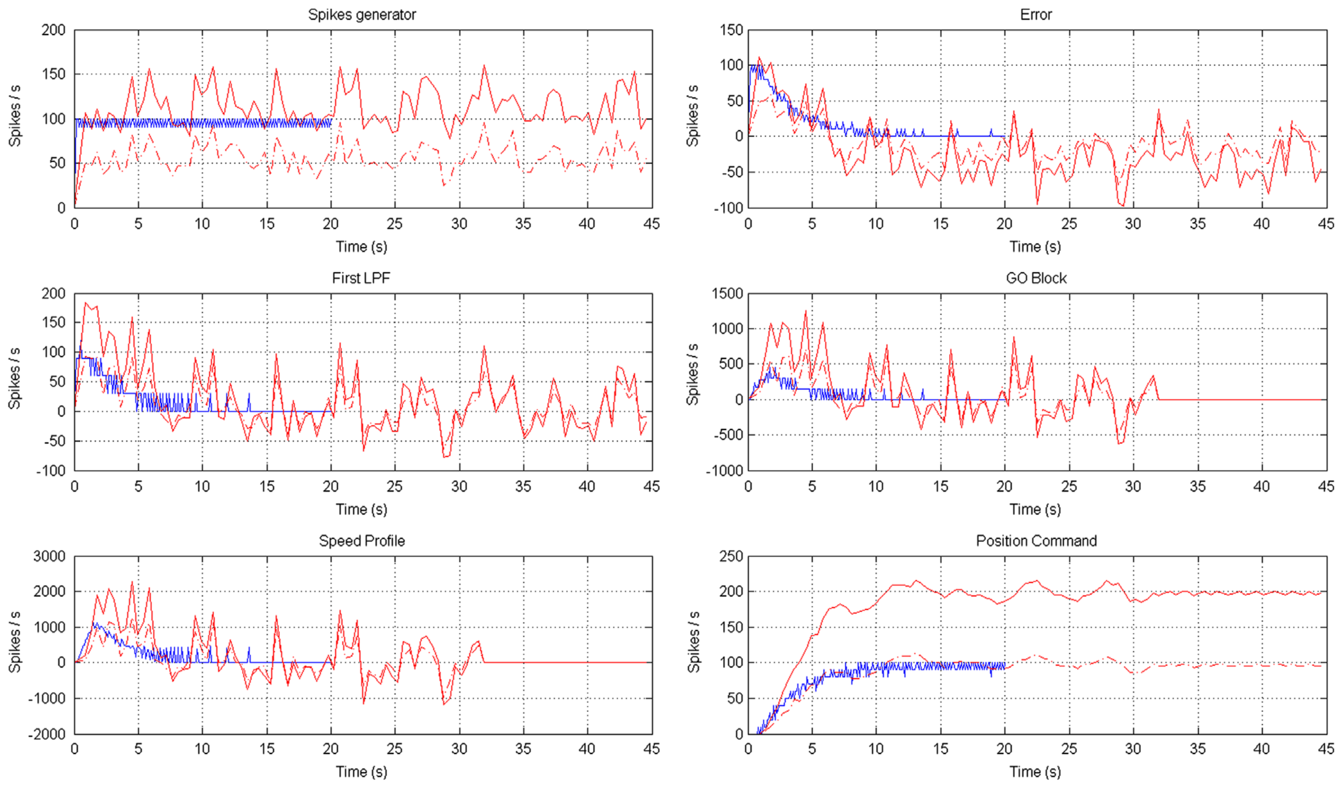

Fig. 5. Output signals of each block of the algorithm. They were obtained by integrating a fixed period of spikes in a similar way as the kernel density estimation [28]. Blue lines represent the behavior used in the running neuro-motor-controller presented in [13] versus the random source, represented by red lines. The dotted red lines represent a 50 spikes/s input reference and the solid red lines represent a 100 spikes/s input. The standard deviation calculated from comparing the sources was 7.4 spikes/s; 100 spikes/s for the speed profile and 3.86 spikes/s for the commanded position. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

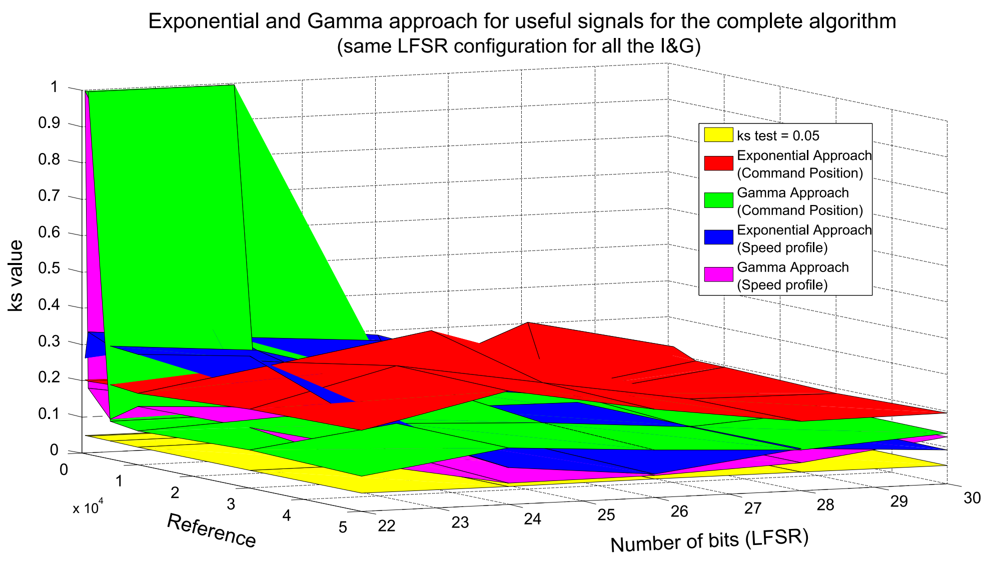

Fig. 6. The yellow surface represents the plane for $KS = 0.05$. The green surface represents the KS-value for the gamma approach and the red surface represents the ks-value for the exponential approach; both for the commanded position signal. The blue surface represents the ks-value for the exponential approach and the pink surface represents the ks-value for the gamma approach; these two for the speed profile signal. As it is easy to see, none of the combinations passed the test, neither any of the possibilities between the range (16, 32) bits for the LFSR got a p-value different from zero. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

neurons may start oscillating due to this negative contribution. Also, it seems that if the GO block injects a large number of spikes, those damaging effects will be amplified.

To increase the firing rates as it was defined in the methods, some of the tested options have been: 10-15-20, 15-20-25 or 20-25-30 bit LFSR (each LFSR of the triplets for each I&G component: both spike low-pass-filters and last I&G block); other configurations do not provoke any spikes at the useful signals (speed profile or position). The obtained results open up discussions:

– The first two configurations do not pass the KS-test for any digital reference or any useful signal. Therefore, although the

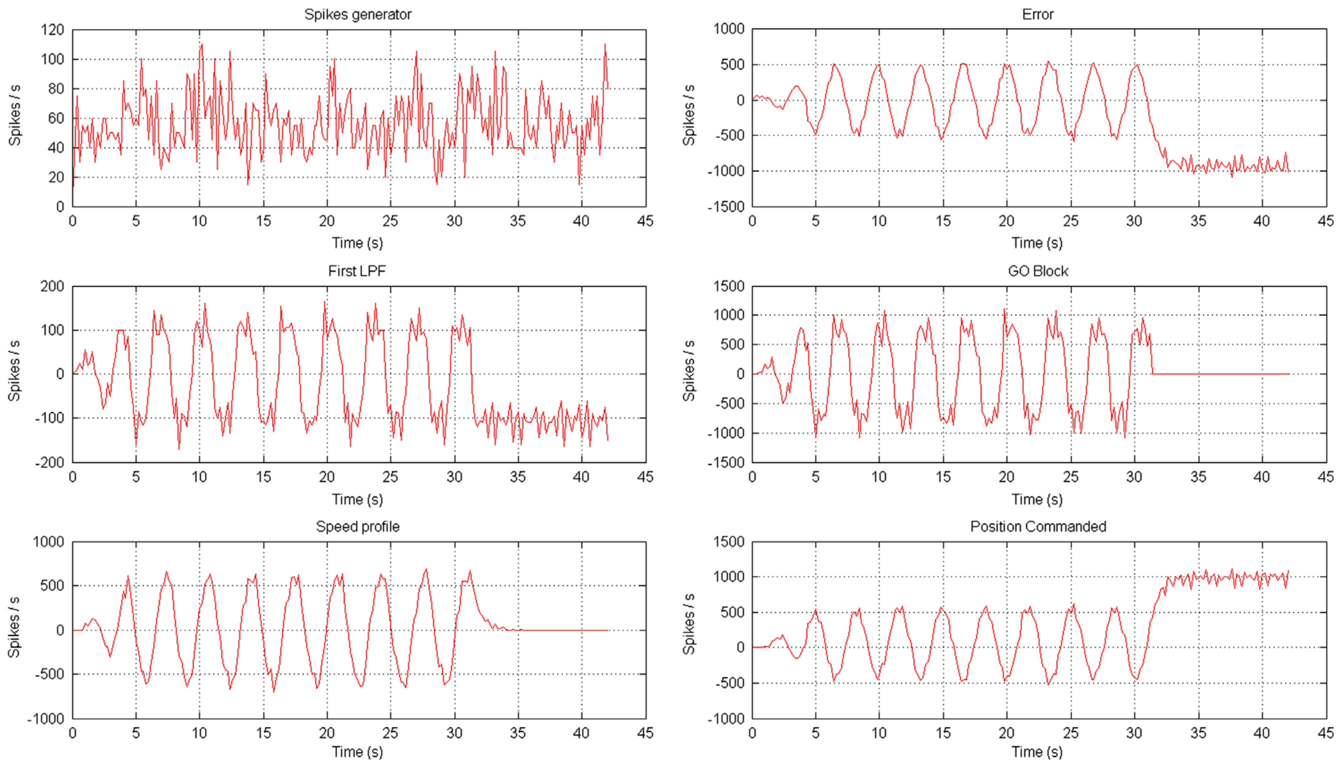

Fig. 7. Output signals of each block of the algorithm. These signals are for a 25 bit LFSR register for all the neurons and 50 spikes/s as the input reference to the algorithm.

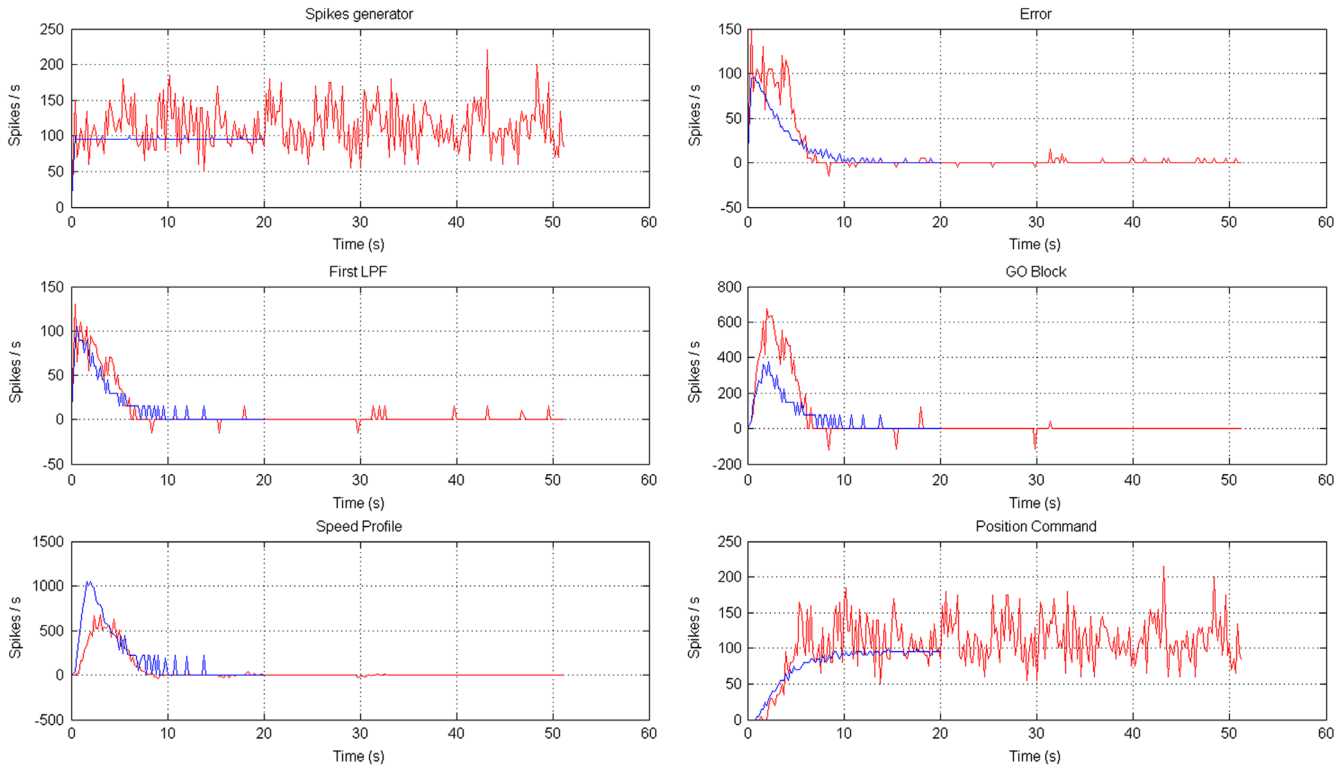

Fig. 8. Output signals of each block of the algorithm. The blue lines represent the entire deterministic algorithm and the red lines represent the one which includes the random “Spikes Generator” and the 20-25-30 LFSR configuration. The average behavior is nearly the same. The standard deviation calculated from comparing the sources was 21.38 spikes/s; 157 spikes/s for the speed profile and 11.53 spikes/s for the commanded position. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article).

signals nearly match the deterministic behavior, we cannot predict them with any distribution.

– However, the last configuration (20-25-30) gives us a chance because if we compare the signals behavior with the complete deterministic situation, the result is an absolute equality (Fig. 8). Thus, these signals (speed profile and command position) are useful for motor control. But, can we predict them? Do they follow any well-known distribution?

The statistical study states that position commanded signals can be approximated by an exponential distribution for references higher than 1600 (since 16 bits are used for the input, the range is very large).

4. Conclusions

In this study, we assessed the effect of including a random spike distribution in a neuro-inspired motor controller with the aim of approaching to a pure neuron model, which is supposed to have variations at its firing pattern between tests with the same stimuli presented. Also, this random component could be seen as it were generating some noisy spikes.

A Poisson "Spikes Generator" can be used to convert the digital input reference into spikes for a neuro-inspired neuro-motor-controller without losing accuracy. The key point is to correctly match the digital input reference delivered to the block with the desired firing rate produced. The spike timing of the source can be predicted by a gamma or exponential approach.

In the results section, output motor control signals have quite a considerable ripple. It means that a small number of spikes are fired in a random way above or below the average. These spikes do not affect the system´s functionality; the motor will filter these spikes and they will not provoke or disturb the motion. This is one of the advantages of delivering spikes straightforward to the motor: if there is any situation where there are some noisy spikes, they will not provoke any movement.

It is reasonable to think that in a massively connected neuronal network, one of these neurons with multiple pre-synaptic connections belonging to different systems could reach its threshold due to mixed contributions or noise and finally fire a spike. In such case, this spike would not provoke any change in the system because the surrounding neurons will 'filter' it or not reach their threshold. In our neuro-inspired motor controller, there would not be any action done if noisy or even deliberate spikes are fired; again, the motor will filter them.

To sum up, this study showed that it is possible to include a Poisson spike distribution as the input reference of the algorithm and even a random component in all the neurons of the algorithm without making disturbances on the final movement accuracy. Besides, the position commanded signals produced will follow an Exponential distribution for their ISI if all the firing blocks of the algorithm are fine-tuned with a random element (20-25-30 bit LFSRs configuration) and the motor controller works properly with them.

References

[1] Janardan Misra, Indranil Saha, Artificial neural networks in hardware: a survey of two decades of progress, Neurocomputing 74 (2010) 239–255.

[2] C. Mead, Analog VLSI and neural systems, Addison-Wesley, Reading, MA, 1989.

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

[4] G. Indiveri, E. Chicca, R. Douglas, A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity, IEEE Trans. Neural Netw. 17 (1) (2006) 211–221.

[5] A. Cassidy, A.G. Andreou, J. Georgiou, Design of a one million neuron single FPGA neuromorphic system for real-time multimodal scene analysis, in: Proceedings of the 45th Annual Conference on Information Sciences and Systems (CISS) 23–25 March, 2011 pp. 1–6.

[6] L.P. Maguire, et al., Challenges for large-scale implementations of spiking neural networks on FPGAs, Neurocomputing 71 (2007) 13–29.

[7] M. Sekerli, R.J.Butera, An Implementation of a Simple Neuron Model in Field Programmable Analog Arrays, in: Proceedings of the 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, San Francisco, CA, USA, 1–5 September, 2004 pp. 4564–4567.

[8] Gerstner Wulfram, Werner M. Kistler, Spiking neuron models: Single neurons, populations, plasticity, Cambridge UniversityPress, 2002.

[9] R. Serrano-Gotarredona, M. Oster, P. Lichtsteiner, A. Linares-Barranco, R. Paz-Vicente, F. Gómez-Rodríguez, L Camuñas-Mesa, et al., CAVIAR: A 45 k neuron, 5 M synapse, 12 G connects/s AER hardware sensory–processing–learning–actuating system for high-speed visual object recognition and tracking, IEEE Trans. Neural Netw. 20 (9) (2009) 1417–1438.

[10] E. Culurciello, R. Etienne-Cummings, K.A. Boahen, A biomorphic digital image sensor, IEEE J. Solid State Circuits 38 (2003) 281–294.

[11] V. Chan, S.C. Liu, A Van Schaik, AER EAR: a matched silicon cochlea pair with address event representation interface, IEEE Trans. Circuit Syst. 54 (1) (2006) 48–59.

[12] A. Jimenez-Fernandez, G. Jimenez-Moreno, A. Linares-Barranco, M.J. Dominguez-Morales, R. Paz-Vicente, A. Civit-Balcells, A neuro-inspired spike-based pid motor controller for multi-motor robots with low cost FPGAs, Sensors 12 (2012) 3831–3856.

[13] F. Perez-Peña, A Morgado-Estevez, A. Linares-Barranco, A. Jimenez-Fernandez, F. Gomez-Rodriguez, G. Jimenez-Moreno, J. Lopez-Coronado, Neuro-inspired spike-based motion: from dynamic vision sensor to robot motor open-loop control throughspike-VITE, Sensors 13 (2013) 15805–15832.

[14] Softky, R. William, C. Koch, The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs, J. Neurosci. 13 (1) (1993) 334–350.

[15] Z.F. Mainen, T.J. Sejnowski, Reliability of spike timing in neocortical neurons, Science 268 (5216) (1995) 1503–1506.

[16] D. Bullock, S. Grossberg, Neural dynamics of planned arm movements: emergent invariants and speed-accuracy properties during trajectory formation, Psychol. Rev. 95 (1988) 49–90.

[17] A. Jimenez-Fernandez, M. Domínguez-Morales, E. Cerezuela-Escudero, R. Paz-Vicente, A. Linares-Barranco, G. Jimenez, Simulating building blocks for spikes signals processing, in: Proceedings of the 11th International Work-Conference on Artificial Neural Networks, Torremolinos-Málaga, Spain, 8–10 June, 2011 vol. 6692, pp. 548–556.

[18] F. Perez-Pena, A. Morgado-Estevez, C. Rioja-Del-Rio, A. Linares-Barranco, A. Jimenez-Fernandez, J. Lopez-Coronado, J.L. Muñoz-Lozano, Towards AER VITE: BuildingSpikeGate Signal, in: Proceedings of the 19th IEEE International Conference on Electronics, Circuits, and Systems, Seville, Spain, 9–12 December, 2012 pp. 881–884.

[19] P. Dayan, L. Abbot, Theoretical Neuroscience, MIT Press, Cambridge, MA, 2001.

[20] F. Rieke, D. Worland, R. De Ruyter van Steveninck, W. Bialek, Spikes: Exploring the Neural Code, MIT Press, Cambridge, 1999.

[21] A. Linares-Barranco, M. Oster, D. Cascado, G. Jiménez, A. Civit, B. Linares-Barranco, Inter-spike-intervals analysis of AER Poisson-like generator hardware, Neurocomputing 70 (16-18) (2007) 2692–2700.

[22] E. Cerezuela-Escudero, M.J. Dominguez-Morales, A. Jiménez-Fernández, R. Paz-Vicente, A. Linares-Barranco, G. Jiménez-Moreno, Spikes Monitors for FPGAs, an Experimental Comparative Study, in: Proceedings of the 12th International Work-Conference on Artificial Neural Networks, Puerto de la Cruz-Tenerife, Spain, 12–14 June, 2013 7902, 179–188.

[23] R. Berner, T. Delbruck, A. Civit-Balcells, et al., A 5 Meps $100 USB2.0 address-event monitor-sequencer interface, in: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), New Orleans, LA, 2007 pp.2451–2454.

[24] jAER Open Source Project, URL: ⟨http://jaer.wiki.sourceforge.net/⟩, 2014 (accessed 24.01.14).

[25] F. Perez-Peña, A. Morgado-Estevez, T. Serrano-Gotarredona, F. Gómez-Rodríguez, V. Ferrer-García, A. Jimenez- Fernandez, A. Linares-Barranco, Live demonstration: spike-based vite control with dynamic vision sensor applied to an arm robot, in: Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS), Melbourne, AU, 2014.

[26] A. Linares-Barranco, G. Jimenez-Moreno, B. Linares-Barranco, A. Civit-Ballcels, On algorithmic rate-coded AER generation, IEEE Trans. Neural Netw. 17 (3) (2006) 771–788.

[27] Mathworks Documentation Center, URL: ⟨http//www.mathworks.es⟩, 2013 (accessed: 16.12.13).

[28] G. Sonja, S. Rotter, Analysis of parallel spike trains, Springer, 2010.