


From Vision Sensor to Actuators, Spike Based Robot Control through Address-Event-Representation

View metadata, citation and similar papers at core.ac.uk

brought to you by  CORE

provided by idUS. Depósito de Investigación Universidad de Sevilla

A. Jimenez-Fernandez, C. Lujan-Martinez, R. Paz-Vicente, A. Linares-Barranco,
G. Jimenez, and A. Civit¹

Departamento de Arquitectura y Tecnología de Computadores,
Universidad de Sevilla,
Av. Reina Mercedes s/n, 41012-Sevilla, Spain
angel@us.es

Abstract. One field of the neuroscience is the neuroinformatic whose aim is to develop auto-reconfigurable systems that mimic the human body and brain. In this paper we present a neuro-inspired spike based mobile robot. From commercial cheap vision sensors converted into spike information, through spike filtering for object recognition, to spike based motor control models. A two wheel mobile robot powered by DC motors can be autonomously controlled to follow a line drawn in the floor. This spike system has been developed around the well-known Address-Event-Representation mechanism to communicate the different neuro-inspired layers of the system. RTC lab has developed all the components presented in this work, from the vision sensor, to the robot platform and the FPGA based platforms for AER processing.

1 Introduction

Humanoids are robots that mimic the human body, emulate the behaviour of human movements and try to reproduce the human mental algorithms. Neuromorphic engineers work in the study, design and development of neuro-inspired artefacts developed with artificial mechanism, like VLSI chips for sensors[5][6], neuro-inspired processing, filtering or learning [9][10][11][12], neuro-inspired control-pattern-generators (CPG) [13][14][15], neuro-inspired robotics [16] and so on.

One of the problems that neuromorphic VLSI engineers had to face was the viability to implement several thousand of neuro-inspired cells (spiking neurons) in a chip, and be able to communicate them with a few available pins. This matter was solved multiplexing in time the neurons behaviour using a very high speed digital bus. The solution was called Address-Event-Representation (AER). AER was proposed by the Mead lab in 1991 [1] for communicating between neuromorphic chips with spikes (Figure 1). Each time a cell on a sender device generates a spike, it communicates with the array periphery and a digital word representing a code or address for that pixel is placed on the external inter-chip digital bus (the AER bus). In the receiver chip, spikes are directed to the pixels whose code or address was on the bus. In this way, cells with

¹ This work was supported by Spanish grant SAMANTA 2 (TEC2006-11730-C03-02), and by the Andalusia Council with the BrainSystem project (P06-TIC-01417).

the same address in the emitter and receiver chips are virtually connected by streams of spikes. Cells that are more active access the bus more frequently than those less active. There is a growing community of AER protocol users for bio-inspired applications in vision, audition systems, and robot control, as demonstrated by the success in the last years of the AER group at the Neuromorphic Engineering Workshop series [3]. The goal of this community is to build large multichip and multi-layer hierarchically structured systems capable of performing massively-parallel data-driven processing in real time [2][4][8].

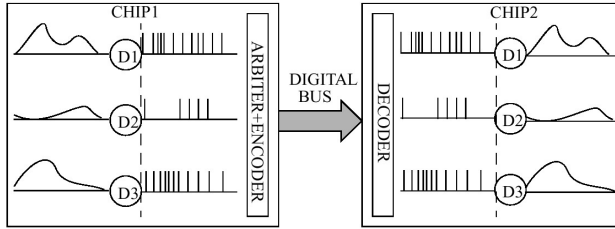


Fig. 1. Rate-coded AER inter-chip communication scheme

2 Neuro-inspired Sensing, Filtering and Actuating Model

In this paper we describe a neuro-inspired mobile robot with a double spike-based control mechanism for two DC motors. This control model works with pulse frequency modulation (PFM), just like the spikes, to power directly the actuators with spikes. Visual information comes from a spike-based sensor to the control model through a spike filter, extracting the direction and speed references for the robot control layer. The spike filter has the aim to obtain the geometric centre of the object detected by the vision sensor as fast as possible. This filter only requires several events for accurate results. The vision sensor converts the analogue luminosity of the CCD camera pixels into spikes after applying a pre-processing of the visual information, extracting the temporal difference information of the sensor. The mobile robot has been constructed around reconfigurable components (FPGA and microcontrollers). Two AER-tools [25] have been used: the USB-AER, for the visual information pre-processing and spike conversion, and the AER-robot, for the spike filtering and the spike based motor control operation for the two wheels of the robot. Figure 2 shows system blocks. A commercial low cost CCD is sending the information in RAW format to a FPGA through an ADC converter. These frames are pre-processed to extract temporal differences and converted into AER information by the Exhaustive AER generator [24] using the USB-AER tool [25]. The complete neuro-inspired implementation allows processing the information with very low time delays compared with the delay of the motor dynamic changes. Furthermore, by changing our low-cost AER-visual sensor by an AER retina, all the processing will be no-frame dependant, compared with the completely digital visual processing that has to process the complete frame before extracting an order for the mobile platform.

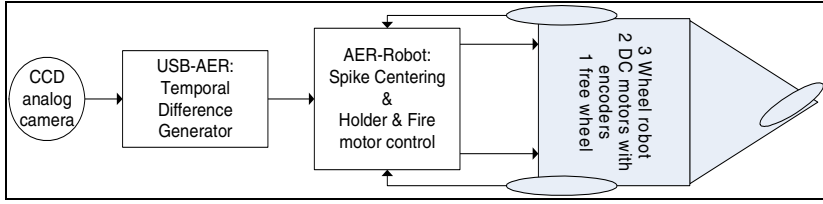


Fig. 2. Neuro-inspired system for mobile robot control

3 Visual Stage

AER image sensors are devices that capture moving images from real world and convert image information to spiking events, according to the AER neuromorphic approach. Different AER retinas are developed using CMOS based ASIC chips as image sensor that directly generates AER streams [5]. In this paper we propose a cheaper alternative image sensor based in a traditional image video camera (CCD sensor with composite video) and a FPGA board that captures frame information from a video composite signal generated by a standard web cam or camcorder, and convert this frame information in an AER stream. Main advantage of Synthetic Retina is for AER chains testing and debugging. As AER events production is controlled by an FPGA based design, written in VHDL, the behavioral of the retina can be easily modified by reprogramming the FPGA circuits.

As Synthetic Retina uses an inexpensive Xilinx Spartan II FPGA and an inexpensive CMOS web cam, it is presented also as a cheap alternative to traditional ASIC AER retinas for AER testing when these retinas are not available. Synthetic Retina has some limitations derived from video composite signal limitation and sensor used. Traditional video sensor has an automatic gain control built-in, so image brightness is modified by video sensor for accommodating to lighting conditions. Therefore, AER generated does not describe exactly captured image, but a bright-compensated version of it. Also maximum change speed is limited by frame-time period. Using high speed video camera can improve this limitation.

3.1 Brightness Retina and Contrast Retina

Two different kind of AER retinas are commonly used in AER chains: brightness, and contrast retinas. Both of them operate in a very similar way, so image is captured and converted to AER events. Brightness retina [24] generates an AER stream describing captured information. Frequency of event for each pixel is a function of the amount of light received by this pixel sensor. For that, last captured frame is maintained on memory and used to generate events. In Derivative retinas [5], AER information does not describe image brightness, but image brightness changes for each pixel. An invariant luminosity pixel does not generate any events. High events activity is present when pixel evolves from dark to light or opposite. For that, Synthetic Retina maintains two copies of captured frames instead of just one. One copy stores last frame, and the other the subtraction of this last frame with previous one. Second frame buffer can otherwise store previous frame, and subtraction is computed at AER generation process. Result of this subtraction produces two sign results, positives ones when a pixel evolves

from dark to bright, and negatives ones when opposite, a brilliant pixel darkens. Derivative retinas can be useful because the AER traffic generated is lower than Brightness retina. Only those parts of the images that changes generates AER events.

3.2 Synthetic Retina Implementation

Figure 3 diagram describes hardware implementation blocks in VHDL. An analogue to digital converter (ADC) is used for digitalizing composite video signal. Actually all known AER retina works with gray-levels so we don't need to decode colour subcarrier. A VHDL module waits for vertical and horizontal syncs to capture video information. An arbiter is used to share external memory access between video captured and AER events generation. As shown in Figure 3, external RAM is divided in two different frame buffers, one storing last frame information, and the other that stores image difference between last frame and previous one. Choosing which frame is used to feed the AER generation, we obtain a Brightness or Derivative Retina.

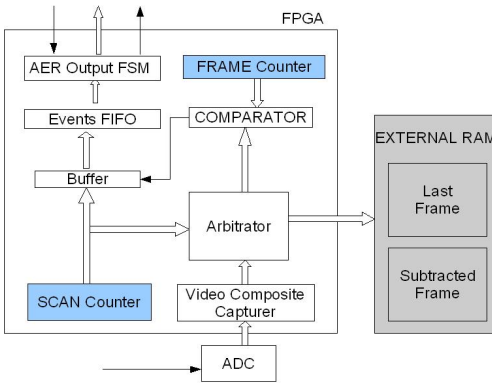


Fig. 3. Synthetic retina block diagram

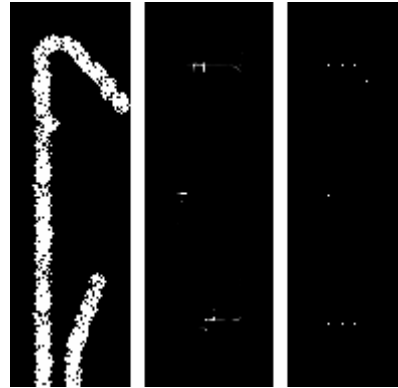


Fig. 4. Centre from events algorithm applied to a path

To generate AER events, exhaustive method has been implemented [23]. A binary counter is used to scan captured frames and generate events for each pixel if comparison between pixel value and slice counter gives that slice counter is below pixel value. Slice counter divides a frame time in slices. To transmit pixel information of 255 (maximum allowed value), 255 events needs to be transferred, and so all slots are occupied. A pixel with a value of 100 only needs to transfer 100 events of the 255 given slots. Bit order in slice counter is reversed to avoid event concentration in the first slices of each frame. AER generation method can be modified in order to test other method of Frame-to-AER conversion [22][23].

4 Centre Detection and Movement Information Translation

In order to command the robot, for line tracking, there is need to determine line centre from an AER synthetic retina generated events. The idea is to calculate locations, nearest as possible to centre of an object in motion or a path, which let to infer the

object's location or path's direction. Centring from events is based on these ideas: only active information is present in the AER bus, transmitting the cell addresses allows performing extra operations on the events while they travel from one chip to another and operations must consume a short period of time.

Highest and lowest values for each coordinate are calculated at each event arrival from its address. Then, centre is inferred dividing by 2 (shifting right) the addition of the corresponding limits from each coordinate. Periodically, these limits values are forgetting, making them to achieve their initial values progressively if no event is received for a while. Forgetfulness is defined by the input event rate. A higher one means more information, possibly redundant, so forgetfulness can be applied more frequently. This guarantees that object in motion does not saturate the limits, which would produce bad centres at the output. The inferred address is always contained in the object in motion area; as only contrast change information is received. In figure 4 experimental results is shown after applying cantering algorithm. The first column represents the frames. The second one represents each inferred centre (on each event arrival). Finally, the third one shows sent centres as outputs. The inferred centre is the desirable position to achieve from the robot reference position. Y-difference from the robot reference one to the y-centre coordinate is used to increase or decrease the speed. If this difference is so far, speed can be increased. The situation is under control when tracking a line, or object motion is going faster than the robot and it may move faster. On the other hand, if it is short, speed may be decreased. It could be a closed, or hairpin, bend or the robot is approaching to the object. Turn sense and its intensity are defined by the difference of x-centre coordinate and the reference position. Turn direction is defined by the difference's sign and turn intensity depends on the absolute value of it. The AER DC Motor Control stage receives exciting and inhibitory spikes for both motor turn and speed. Spike frequency controls intensity for each one, from 1Kevent/s to 260Kevent/s, and exciting and inhibitory spikes mean sense, coded as positives and negatives addresses. Spike output is generated by a synthetic random generator [17][23]. Y-difference and X-difference are mapped to AER DC motor control stage frequency (from 1Kevent/s to 256Kevent/s) and corresponding address event, turn sense and speed, are sent.

5 Motor Control Stage

In biology, one of the last steps of the information processing is to drive a limb. Trying to mimic the biology, we use spikes to drive actuators in robotic platforms, for example, driving DC-motors with spikes. We have developed closed-loop speed control for DC-motors, and we have used them to control a mobile robot. All these controls has been implemented in a FPGA and tested in the AER-Robot board [9]. Digital systems usually drive DC-motors using a Pulse Width Modulation (PWM). PWM signals have a fixed period, and variable high time or duty-cycle [18]. In the neuronal model presented, spikes are codified following a Pulse Frequency Modulation (PFM) where information resides in the pulse frequency. In a PFM scheme the pulse's high time is fixed, and the frequency variable, just opposite to a PWM scheme. We use the PFM as other way to drive DC-motors, applying spikes almost directly to the DC-motor. Spikes are so short in time (e.g. 20ns) that they will be filtered by the DC-motor. The solution is to increase them in time (e.g. 1us) for a fixed time. As we can found in [18], the spike width affects to the DC-motor static gain. The longer spike width is the higher static gain will be.

Table 1. *Robot Features*

| Parameter | Value | Units |
|-------------------------|-------|--------|
| Motor nominal voltage | 12 | V |
| Motor start current | 4.8 | A |
| Max motor ang. Speed | 133.5 | r.p.s. |
| Motor rise time | 17.6 | ms |
| Gear box relation | 1:19 | n.a. |
| Max. gear box speed | 7.02 | r.p.s. |
| Encoded channels | 3 | Ch |
| Encoder counts per turn | 500 | p.p.r. |
| Max. encoder freq. | 100 | kHz |
| Max. ref. spikes rate | 263 | kSp/s |

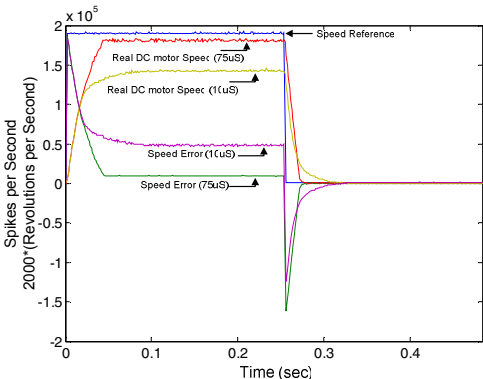


Fig. 5. Closed-loop spiking-speed control response for divers spikes width

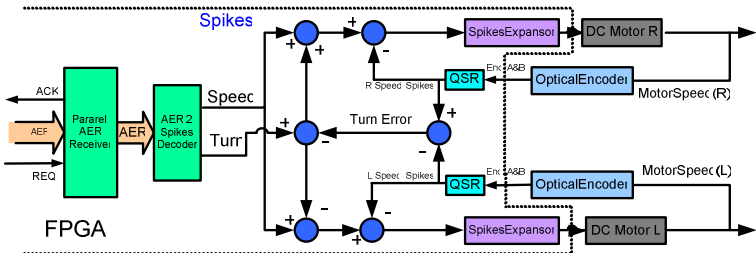


Fig. 6. Control blocks for Eddie

5.1 Closed-Loop Spiking Speed Control

Another step to drive DC-motors with spikes is to implement a closed-loop control, like a speed control. For this purpose there is needed an element that subtracts two spiking signals, the speed reference and the feedback information, the real DC-motor speed, being this element output the error between the desired speed and the real speed. This is the function of the Spike Hold and Fire (SH&F) component. The SH&F subtracts in real-time two incoming spike streams (U and Y) being the output a spike stream whose spike rate is the difference between the U and Y spike rate. Another question is how to measure the feedback information (DC-motor speed), and translate it to a spike stream. In our case, we have used an optical encoder, which outputs are two signals (A&B). These signals frequency represents the DC-motor speed, following just a PFM scheme, and the phase between A&B, the speed direction. To translate them to spikes we use a finite state machine (FSM) that determines the speed direction (sign). Then it fires a signed spike for every edge in the A&B signals.

Using the AER-Robot board and an USB2AER monitor/sequencer [20], we have measured the motor responses. Inside the AER-Robot lies the VHDL components that implements the closed-loop control: an AER decoder (for the references from the AER

input port), a SH&F (who gives the error), a spikes expensor (to apply the spikes to the DC-motor), and a massive spikes monitor (who gives an address to every spikes, and send them through an AER parallel port). Connected to these ports there is an USB2AER monitor/sequencer, which generates the AER events for speed reference, and sends the spikes information to a PC, through a full speed USB2.0 port. In Figure 5 we can see the DC-motor speed response when we apply a constant spike stream, just like a step, for two different spikes width. We show the information recovered from the AER bus, after integrating the spikes for a fixed time. In the plot there are 5 signals, the two first one is the speed reference, with a constant spike rate. The next two ones are the DC-motors speed for each spike width, read from the optical encoder's spikes. And finally there are the speed errors, which start with a high spike rate, and start to decrease when the motor speeds are increasing. As a typical closed-loop control system, we can see both static gains below one, which introduces the static errors. We have also decreased the rise time, from 17.6ms to 0.73ms, and achieve a low ripple (including the oscillation introduced by the spikes integration).

5.2 Eddie, the Robotic Platform

Finally, we have tested these controls in a robotic platform, Eddie. This platform is a differential robot, with two independent DC-motors in the rear, and a free wheel in the front [19]. The control have been implemented in VHDL and loaded in the AER-Robot board. In Figure 6 we can find control blocks for Eddie. Where inputs are: the speed and turn references, and both motors speed feedback. Outputs are the spikes applied to the DC-motors. For this application, the input spikes will come from a parallel AER port, and it will be send to a reference input (speed or turn) after decode them (from an address event to a simple spike). Figure 6 also shows the control loops for each motor. Spikes references flows from left to right, the turn reference are added, or subtracted, to the speed reference (using two SH&F), obtaining the individual speed reference for each DC-motor. Both individual references are subtracted to DC-motor real speed; this is the individual speed error. Finally, the speed error spikes, are increased in time, and applied to the DC-motor. We also measure the turn speed (the difference between both motors speed) and implement a third control loop for the turn. This one helps to keep the same speed for both motors.

6 Conclusions

Joining the three AER layers (sensing, centre filtering and actuating) it is possible to follow a path drawn on the floor using a mobile robot. The speed response of the whole system depends on the motors dynamic response, because the AER system is able to change the motor commands with a few events (some microseconds). The principal advantage of this control model is that no frame processing is required to detect the line and then act to the motors, while the actuation to the motors is done in real time while the visual information is received and processing during the transmission.

References

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

- [2] Serrano-Gotarredona, T., et al.: AER Image Filtering Architecture for Vision-Processing Systems. *IEEE Transactions on Circuits and Systems. Fundamental Theory and Applications* 46(9) (September 1999)
- [3] Cohen, A., et al.: Report to the National Science Foundation: Workshop on Neuromorphic Engineering, Telluride, Colorado, USA, June-July (2004), <http://www.ini.unizh.ch/telluride>
- [4] Mahowald, M.: VLSI Analogs of Neuronal Visual Processing: A Synthesis of Form and Function. PhD. Thesis, California Institute of Technology Pasadena, California (1992)
- [5] Lichtsteiner, P., Posch, C., Delbruck, T.: A 128X128 120dB 15 us Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal on Solid-State Circuits* 43(2), 566–576 (2008)
- [6] Chan, V., Liu, S.C., van Schaik, A.: AER EAR: A Matched Silicon Cochlea Pair with Address-Event-Representation Interface. *IEEE Transactions on Circuits and Systems-I* 54(1), 48–59 (2007)
- [7] Linares-Barranco, A., et al.: AER Neuro-Inspired interface to Anthropomorphic Robotic Hand. In: *IEEE World Conference on Computational Intelligence, IJCNN*, Vancouver (July 2006)
- [8] Serrano-Gotarredona, R., et al.: AER Building Blocks for Multi-Layer Multi-Chip Neuromorphic Vision Systems. In: *NIPS 2005* (2005)
- [9] Serrano-Gotarredona, R., et al.: On Real-Time AER 2-D Convolutions Hardware for Neuromorphic Spike-Based Cortical Processing. *IEEE Transactions on Neural Networks* 19(7), 1196–1219 (2008)
- [10] Serrano-Gotarredona, R., et al.: A Neuromorphic Cortical-Layer Microchip for Spike-Based Event Processing Vision Systems. *IEEE Transactions on Circuits and Systems-I* 53(12), 2548–2566 (2006)
- [11] Hafliker, P.: Adaptive WTA with an Analog VLSI Neuromorphic Learning Chip. *IEEE Transactions on Neural Networks* 18(2), 551–572 (2007)
- [12] Indiveri, G., et al.: A VLSI Array of Low-Power Spiking Neurons and Bistables Synapses with Spike-Timing Dependent Plasticity. *IEEE Transactions on Neural Networks* 17(1), 211–221 (2006)
- [13] Gomez-Rodríguez, F., et al.: AER Auditory Filtering and CPG for Robot Control. In: *ISCAS 2007* (2007)
- [14] Vogelstein, R.J., et al.: Phase-Dependent Effects of Spinal Cord Stimulation on Locomotor Activity. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 14(3), 257–265 (2006)
- [15] Vogelstein, R.J., et al.: Dynamic Control of Spinal Locomotion Circuits. In: *ISCAS 2007* (2007)
- [16] Linares-Barranco, A., Gomez-Rodríguez, F., Jimenez-Fernandez, A., Delbruck, T., Lichtensteiner, P.: Using FPGA for visuo-motor control with a silicon retina and a humanoid robot. In: *ISCAS 2007* (2007)
- [17] Linares-Barranco, A., et al.: Inter-spike-intervals analysis of AER Poisson-like generator hardware. *Neurocomputing* (2007)
- [18] Jiménez-Fernández, A., et al.: Civit. AER and dynamic systems co-simulation over Simulink with Xilinx System Generator. In: *ICECS 2008* (2008)
- [19] Jiménez-Fernández, A., et al.: AER-based robotic closed-loop control system. In: *ISCAS 2008* (2008)
- [20] Berner, R., et al.: A 5 Mips \$100 USB2.0 Address-Event Monitor-Sequencer Interface. In: *ISCAS 2007* (2007)
- [21] Gomez-Rodríguez, F., et al.: Two Hardware Implementation of the Exhaustive Synthetic AER Generation Method. *LNCS*. Springer, Heidelberg (2005)
- [22] Gomez-Rodríguez, F., et al.: AER tools for communications and debugging. In: *ISCAS 2006* (2006)
- [23] Linares-Barranco, A., et al.: On Algorithmic Rate-Coded AER Generation. *IEEE Transaction on Neural Network* 17(3), 771–788 (2006)
- [24] Culurciello, E., et al.: A biomorphic digital image sensor. *IEEE Journal of Solid-State Circuits* 38, 204–281 (2003)