Neuromorphic Real-Time Objects Tracking Using Address Event Representation and Silicon Retina

F. Gómez- Rodríguez, L. Miró-Amarante, M. Rivas, G. Jimenez, and F. Diaz-del-Rio

Robotics and Computer's Technology Lab.
University of Seville
Seville, Spain
gomezroz@us.es

Abstract. This paper presents a hierarchical neuromorphic system for tracking objects. We use AER (Address Event Representation) for transmitting and processing visual information provided by an asynchronous temporal contrast silicon retina. Two AER processing layers work in cascade for firstly detecting different objects, and secondly tracking them even with crossing trajectories. The output of the system offers not only the position of the tracked object but also the speed in pixels per second. The system is fully hardware implemented on FPGA (Spartan II 200), which is part of the USB-AER platform developed in part by authors. A 97.2% of the Spartan II is used for 128x128 pixels input resolution and 6 maximum objects recognition and tracking.

Keywords: object tracking, real-time, Address Event Representation, AER, neuromorphic, neuro-inspired, FPGA.

1 Introduction

This paper presents a neuromorphic [1] real time object tracking using visual information provided by a silicon retina[1]; this information is processed and transmitted using Address Event Representation (AER) [2].

There are two previous works in the field of object tracking which also use AER for processing and transmitting the visual information. Our system has two important differences from these two previous approaches: first, we present a fully hardware system, described in VHDL and implemented in a FGPA, while in [3] and [4] a computer and a DSP were used for events computing, respectively. Second, our system uses only two events for getting the object position and with only two object positions it is possible to estimate the object velocity.

Followed in this section we review the *Address Event Representation* (AER) communication protocol; the hardware platform used, the *USB-AER Board* [5], developed by the Robotics and Computer's Architecture Lab; and the Silicon Retina [6] used, developed by the Institute of Neuroinformatics (INI) of the University of Zurich.

[1] Authors would like to thank to Tobias Delbruck and his group for their silicon retina, and to VULCANO project (TEC2009-10639-C04-02) for supporting this work.

Section 2 is devoted to system's description; in section 3 we present some experiments to demonstrate the system's capabilities; finally, in section 4 we present some conclusions.

1.1 The AER Communication Protocol

The AER protocol was proposed for neuro-inspired information transmission from one neuro-inspired chip to another. The basics of AER consist in assigning an address to each cell (neuron) in a chip. Cell activity is transmitted showing the cell's address in a common bus; two flow control signals are commonly needed (REQ and ACK), to start and stop the transmission.

As a result, the activity of every cell will appear in the common bus. Usually the activity is frequency coded. In this way, if the cell's activity is high, its address will appear in the bus more frequently than other with lower activity. Each address occurrence, in the bus, is known as an *event.* So, we can say that the activity of each cell is coded in events frequency. Since the AER bus multiplexes all the events in a common bus, an arbiter is needed in the transmitter. Fig. 1.a shows the organization of AER communication. To transmit each event, a simple handshake protocol is normally used (see Fig. 1.b).

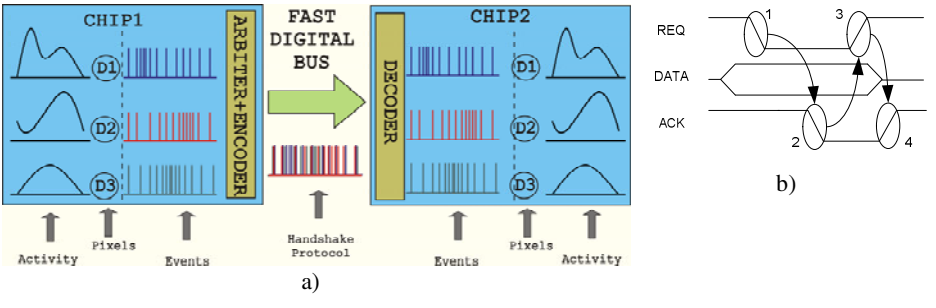


Fig. 1. AER protocol: a) AER transmission overview, b) AER Handshake protocol

1.2 USB-AER Board

The USB-AER board was developed by the Robotic and Computer's Technology Lab of the University of Seville, to give support to AER-based systems developers. The USB-AER board was initially developed for AER-based systems testing and debugging. This board implements several modules for AER stream sequencing, monitoring, mapping (to change *on the fly* the address space); depending on the FPGA firmware.

The flexibility of USB-AER board allows implementing any others functionalities such as the one presented in this paper.

A picture of the USB-AER board is shown in Fig. 2. USB-AER board core is a Spartan-II 200 Xilinx FPGA, connected with a 12ns, 512Kwords of 32 bits SRAM memory bank. The board uses a Silicon Laboratories C8051F320 microcontroller to implement the USB and the MMC/SD interfaces.

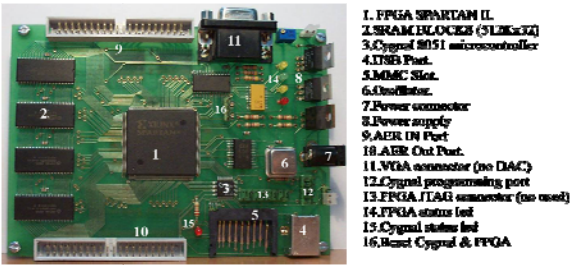
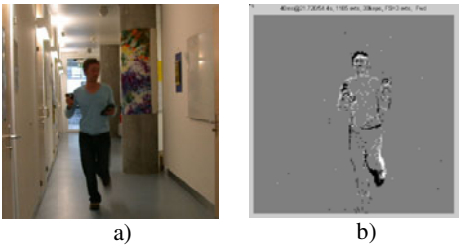


Fig. 2. USB-AER Board

1.3 The Asynchronous Temporal Contrast Silicon Retina

The silicon retina used was developed by Tobias Delbruck and his group at INI of University of Zurich [5]. Concretely, it is an asynchronous temporal contrast silicon retina, with an AER interface. So the retina output consists of an AER sequence that corresponds to the movement in the scene that the retina is "seeing".

Fig. 3.a shows a real scene; and Fig. 3.b. a frame-like reconstruction of the AER sequence, dark dots represent pixels where positive contrast changes happened and light dots where negative changes happened.



a) b)

Fig. 3. Retina functionality: a) Scene, b) Retina Output (pictures extracted [6])

2 Real Time Object Tracking System

The system that we propose consists of several cells interconnected, like neurons that process the AER sequence. Every cell tries to track one object obtaining its position and estimating its velocity. We propose a new architecture where, instead of using a classical fully parallel cell architecture, we use cascade architecture.

2.1 Cascade Architecture

The biological neural system architecture consists in a fully parallel interconnected network distributed in layers; fig.4.a shows a scheme of this network.

Fig. 4.b shows the proposed architecture, in each layer the first cell receives the complete AER sequence and extracts, retains and processes several events; which and how many events are extracted depend on the application; the rest of the events are resent to the second cell in the layer. This procedure is repeated in the rest of the cells in a layer and in the rest of the layers.

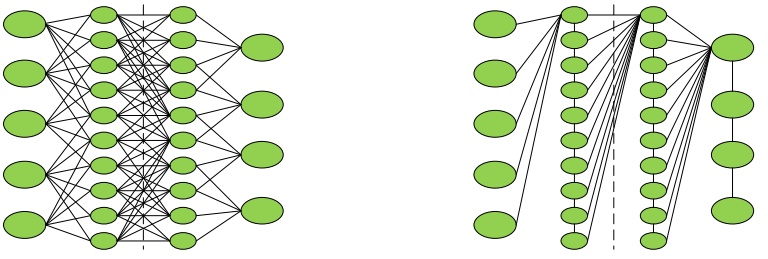


Fig. 4. Neural architectures: a) classical neural parallel network, b) cascade neural network

The most important advantage of this architecture is that each cell only processes the extracted events. Furthermore, this scheme presents an implicit inhibition mechanism, that is, the processed events by one cell are eliminated from the AER sequence and do not interfere with the computation. This scheme allows reducing the AER sequence complexity from one cell to the next.

In this system all cells use AER for information transmission, so each cell has 3 AER ports: one AER input and two AER outputs. The first output gives the result of events *computation* and the second one resends the refused events.

2.2 Objects Tracking Procedure

Following, we present the object tracking procedure used by the architecture presented above. We only use one layer, where each cell, called TrackCell, consists of two sub-cells: one is devoted to object's position determination, called CMCell, and the other, called VCell, is devoted to object's velocity estimation. Fig. 5.a shows these cells.

From now on, we suppose that the input is the visual information provided by the silicon retina, so events correspond to the movement in the scene.

The first cell in the layer receives all the events. Just after the first event is received, the CMCell only extracts and computes events from a small area around this first event (area of interest), resending the rest of the events to the second cell. If during a period of time CMCell does not receive enough events, typically 10 events, it will reset. On the contrary, if during this period of time CMCell receives enough events from the area of interest, it computes the object position as the mean value between the last positive event location and the last negative event location. After the object position computation, CMCell moves the center of the area of interest to the object position. This procedure is repeated after each event is received. If CMCell

receives events near the area of interest (a few pixels around) it will change the area size allowing adapting it to the object size. Fig. 5.b shows the CMCell state machine diagram.

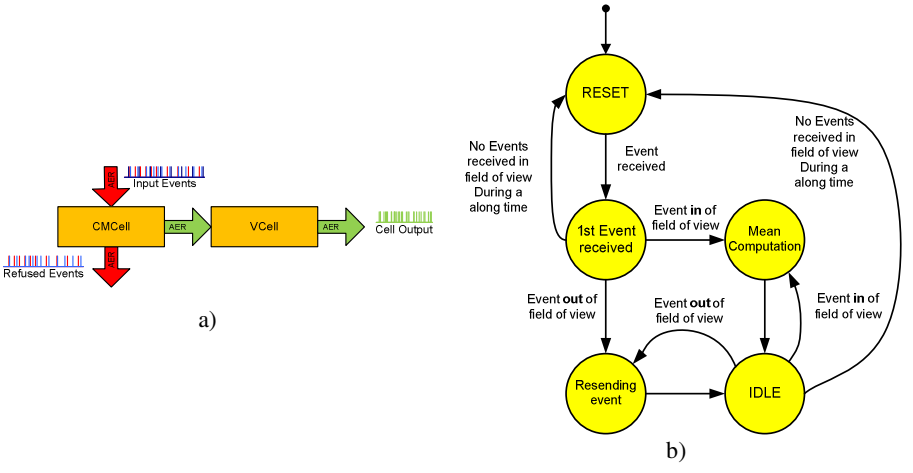


Fig. 5. a) CMCell+VCell and b) CMCell state machine diagram

The output of the CMCell consists of an AER sequence that encodes the object position. This information is used by VCell to estimate the object's velocity.

VCell takes the object position periodically and computes the velocity as the mean value of the last two object positions. Initially, the period used is 100ms, but it changes depending on the velocity computed. If the velocity is high the period will be reduced. On the contrary, if the velocity is low the period will be increased. The velocity is also transmitted using AER.

With this scheme it is possible to track as many objects as TrackCells can be synthesized in a FPGA in cascade. In our case, due to USB-AER board FPGA capacity only 6 objects can be tracked simultaneously.

The key point of this procedure is that events are processed as soon as they are received, without frame integration. Therefore the response time of each cell is very short; in fact it is the delay time (order of ns).

3 Experiments and Capabilities

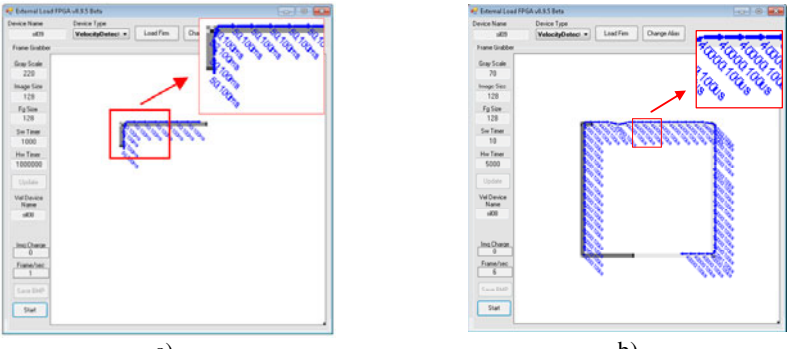
In this section we present some experiments to demonstrate the system's capabilities and performance. The hardware configuration consists of a silicon retina and a USB-AER board.

In order to see what happened in the system the results are stored in the USB-AER SRAM and periodically they are downloaded to a PC and displayed on the screen.

3.1 Synthetic Stimulus

In these two first experiments we use a USB-AER board for AER sequence generating (no retina used). The USB-AER generates an AER sequence emulating the retina behavior but without noise and impreciseness. The AER sequence emulates a 4x4 pixels object describing a square. In the first experiment the object's velocity is 50 pixels/s and in the second 40000 pixels/s.

Fig. 6.a shows AER sequence fused with the system's output, both accumulated during 1s when the object's velocitiy is 50 pixels/s. The gray dots are the AER sequence reconstruction and the system's output are drawn in blue. The first value, near to the row, is the velocity modulus and the second is the VCell period. Fig.6.b shows the system's output when the object's velocity is 40000 pixels/s.



a) b)

Fig. 6. System's output for synthetic stimulus fused with the reconstruction of the AER sequence: a) 50 pixels/s b) 40000 pixels/s

With this experiment we demonstrate the system's capability for determining the position and estimating the object velocity, even when the object velocity is very high.

3.2 Synthetic Objects

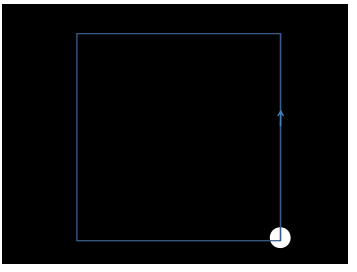
In these experiments we use a silicon retina observing a TFT monitor which is displaying objects moving in the scene at different location and velocities.

Circle at different velocities

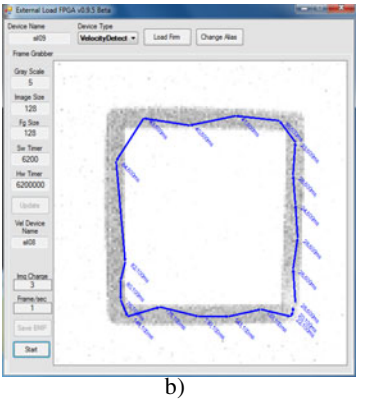
In this experiment a circle starts moving slowly (26.6 pixels/s) from the right bottom corner to the right up corner, it then moves faster to the left up corner (40 pixels/s), then much faster to the left bottom corner (80 pixels/s) and then fastest to the right bottom corner again (160 pixels/s). Fig 7.a shows a scheme of the scene.

Fig. 7.b shows the system's output fused with the reconstruction of the retina output, both accumulated during 6.2s. It can be observed how the system determines the object position and the velocity in every moment. The velocity estimation presents some discrepancies from the true object velocity (up to 25%). This error is due to the

retina imprecision and the speed of the estimation. In this system, real time is more important than accuracy. Furthermore, biological systems do not obtain a quantitative object velocity value, but a qualitative value. This system obtains a very good approximation to the real velocity.



a)



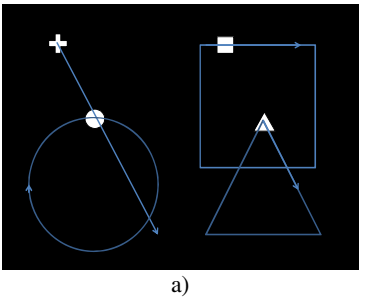
b)

Fig. 7. Second experiment: a) experiment configuration, b) system output fused with silicon retina output

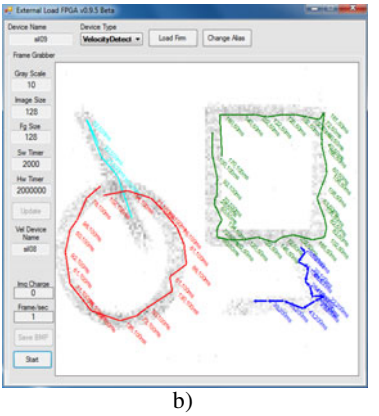
With this experiment we demonstrate the system's capability for adapting to object velocity in real time, that is, the system can track objects even when the object's velocity is changing.

Four objects at different velocities with crossing pathway

In this third experiment we present a more complex scene: 4 objects moving at different velocities with crossing pathways. Fig 8.a shows the experiment's configuration.



a)



b)

Fig. 8. Third experiment: a) experiment configuration, b) system output fused with retina output

Fig.8.b shows the system's output fused with the retina's output reconstruction (accumulated during 2s). Each color represents the output of one TrackCell. Each TrackCell uses a different period for velocity estimation. Once again the object velocity estimation presents some discrepancies with the real velocities: the objects velocities are constant, and the system's output does not show constant velocities, the reason for that was explained above.

With this experiment we demonstrate the system capability for several objects tracking even when the pathways are crossing.

4 Conclusion

This paper presents a neuromorphic system for real-time object tracking using new cascade architecture. The system is fully hardware implemented, described in VHDL; each TrackCell (CMCell+VCell) block only requires 315 slices of a Spartan-II 200 Xilinx FPGA (13.4%). The response time is the minimum possible and depends on the sensor, typically the first object position is obtained after 10 events, and a new position after each new event. The maximum object speed for its tracking and speed prediction is 40000 pixels/s.

Results show that the system can determine the position and estimate the velocity of several objects simultaneously, even in difficult situations: velocity change and crossing pathways. And in ideal situations it can estimate the object velocity with 100% accuracy.

References

[1] Indiveri, G., Liu, S.-C., Delbruck, T., Douglas, R.: Neuromorphic Systems. In: Squire, L.R. (ed.) Encyclopedia of Neuroscience, pp. 521–528. Elsevier Ltd., Amsterdam (2009)

[2] Mahowald, M.: VLSI Analogs of Neuronal Visual Processing: Thesis (1992)

[3] Delbruck, T., Lichtsteiner, P.: Fast sensory motor control based on event-based hybrid neuromorphic-procedural system. In: 2007 IEEE International Symposium on Circuits and Systems, pp. 845–848 (May 2007)

[4] Litzenberger, M., Posch, C., Bauer, D., Belbachir, A.N., Schon, P., Kohn, B., Garn, H.: Embedded Vision System for Real-Time Object Tracking using an Asynchronous Transient Vision Sensor. In: 2006 IEEE 12th Digital Signal Processing Workshop & 4th IEEE Signal Processing Education Workshop, pp. 173–178. IEEE, Los Alamitos (2006)

[5] Paz, R., Gomez-Rodriguez, F., Rodriguez, M., Linares-Barranco, A., Jimenez, G., Civit, A.: Test infrastructure for address-event-representation communications. In: Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) IWANN 2005. LNCS, vol. 3512, pp. 518–526. Springer, Heidelberg (2005)

[6] Lichtsteiner, P., Posch, C., Delbruck, T.: A 128x128 120 dB 15 us Latency Asynchronous Temporal Contrast Vision Sensor. IEEE Journal of Solid-State Circuits 43, 566–576 (2008)

[7] Lichtsteiner, P., Posch, C., Delbruck, T.: A 128 X 128 120db 30mw asynchronous vision sensor that responds to relative intensity change. In: IEEE International Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers, pp. 2060–2069. IEEE, Los Alamitos (2006)