An Approach to Distance Estimation with Stereo Vision Using Address-Event-Representation

M. Domínguez-Morales[1], A. Jimenez-Fernandez[1], R. Paz[1], M.R. López-Torres[1], E. Cerezuela-Escudero[1], A. Linares-Barranco[1], G. Jimenez-Moreno[1], and A. Morgado[2]

[1] Robotic and Technology of Computers Lab, University of Seville, Spain
[2] Electronic Technology Department, University of Cadiz, Spain
mdominguez@atc.us.es

Abstract. Image processing in digital computer systems usually considers the visual information as a sequence of frames. These frames are from cameras that capture reality for a short period of time. They are renewed and transmitted at a rate of 25-30 fps (typical real-time scenario). Digital video processing has to process each frame in order to obtain a result or detect a feature. In stereo vision, existing algorithms used for distance estimation use frames from two digital cameras and process them pixel by pixel to obtain similarities and differences from both frames; after that, depending on the scene and the features extracted, an estimate of the distance of the different objects of the scene is calculated. Spike-based processing is a relatively new approach that implements the processing by manipulating spikes one by one at the time they are transmitted, like a human brain. The mammal nervous system is able to solve much more complex problems, such as visual recognition by manipulating neuron spikes. The spike-based philosophy for visual information processing based on the neuro-inspired Address-Event-Representation (AER) is achieving nowadays very high performances. In this work we propose a two-DVS-retina system, composed of other elements in a chain, which allow us to obtain a distance estimation of the moving objects in a close environment. We will analyze each element of this chain and propose a Multi Hold&Fire algorithm that obtains the differences between both retinas.

Keywords: Stereo vision, distance calculation, address-event-representation, spike, retina, neuromorphic engineering, co-design, Hold&Fire, FPGA, VHDL.

1 Introduction

In recent years there have been numerous advances in the field of vision and image processing, because they can be applied for scientific and commercial purposes to numerous fields such as medicine, industry or entertainment.

As we all know, the images are two dimensional while the daily scene is three dimensional. Therefore, in the transition from the scene (reality) to the image, what we call the third dimension is lost. Nowadays, society has experienced a great advance in these aspects: 2D vision has given way to 3D viewing. Industry and

research groups have started their further research in this field, obtaining some mechanisms for 3D representation using more than one camera [14]. Trying to simulate the vision of human beings, researchers have experimented with two-camera-based systems inspired in human vision ([12][13]). Following this, a new research line has been developed, focused on stereoscopic vision [1]. In this branch, researchers try to obtain three-dimensional scenes using two digital cameras. Thus, we try to get some information that could not be obtained with a single camera, i.e. the distance at which objects are.

By using digital cameras, researchers have made a breakthrough in this field, going up to create systems able to achieve the above. However, digital systems have some problems that, even today, have not been solved. A logical and important result in stereoscopic vision is the calculation of distances between the point of view and the object that we are focused on. This problem is still completely open to research and there are lots of research groups focusing on it. The problems related to this are the computational cost needed to obtain appropriate results and the errors obtained after distance calculation. There are lots of high-level algorithms used in digital stereo vision that solve the distance calculation problem, but this implies a computer intervention into the process and it is computationally expensive.

The required computational power and speed make it difficult to develop a real-time autonomous system. However, brains perform powerful and fast vision processing using millions of small and slow cells working in parallel in a totally different way. Primate brains are structured in layers of neurons, where the neurons of a layer connect to a very large number (~104) of neurons in the following one [2]. Most times the connectivity includes paths between non-consecutive layers, and even feedback connections are present.

Vision sensing and object recognition in brains are not processed frame by frame; they are processed in a continuous way, spike by spike, in the brain-cortex. The visual cortex is composed of a set of layers [2], starting from the retina. The processing starts when the retina captures the information. In recent years significant progress has been made in the study of the processing by the visual cortex. Many artificial systems that implement bio-inspired software models use biological-like processing that outperform more conventionally engineered machines ([3][4][5]). However, these systems generally run at extremely low speeds because the models are implemented as software programs. Direct hardware implementations of these models are required to obtain real-time solutions. A growing number of research groups around the world are implementing these computational principles onto real-time spiking hardware through the development and exploitation of the so-called AER (Address Event Representation) technology.

AER was proposed by the Mead lab in 1991 [8] for communicating between neuromorphic chips with spikes. Every time a cell on a sender device generates a spike, it transmits a digital word representing a code or address for that pixel, using an external inter-chip digital bus (the AER bus, as shown in figure 1). In the receiver the spikes are directed to the pixels whose code or address was on the bus. Thus, cells with the same address in the emitter and receiver chips are virtually connected by streams of spikes. Arbitration circuits ensure that cells do not access the bus simultaneously. Usually, AER circuits are built with self-timed asynchronous logic.

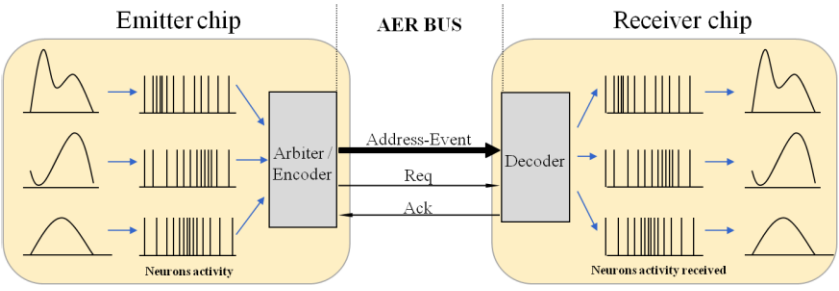

Fig. 1. Rate-coded AER inter-chip communication scheme

Several works are already present in the literature regarding spike-based visual processing filters. Serrano et al. presented a chip-processor able to implement image convolution filters based on spikes that work at very high performance parameters (~3GOPS for 32x32 kernel size) compared to traditional digital frame-based convolution processors (references [6],[7],[5]).

There is a community of AER protocol users for bio-inspired applications in vision and audition systems, as evidenced by the success in the last years of the AER group at the Neuromorphic Engineering Workshop series. One of the goals of this community is to build large multi-chip and multi-layer hierarchically structured systems capable of performing complicated array data processing in real time. The power of these systems can be used in computer based systems under co-processing.

First, we describe element by element of our processing chain until obtain the complete system. Then we propose an AER algorithm, which can be developed in AER systems using a FPGA to process the information; and that is able to obtain differences from both retinas and calculate a distance estimation of the object in movement. Finally, we present distance estimation results of the whole system, and compare it with the real distance.

2 System Description

In this section we will describe in detail the system used and each one of the components that form our system. We can see from figure 2 a block diagram of the whole system.

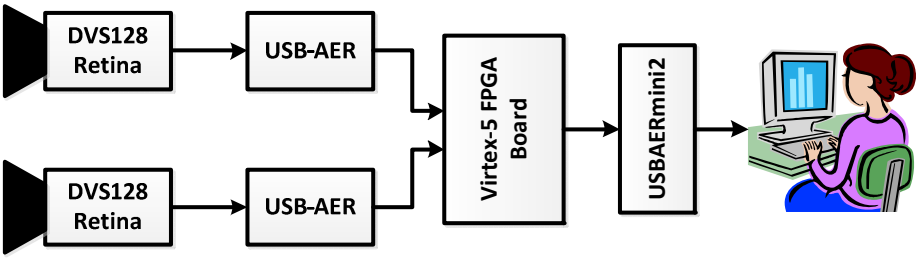

Fig. 2. Complete system with all the elements used

All the elements that compose our system are these (from left to right): two DVS128 retinas [10], two USB-AER, a Virtex-5 FPGA board, an USBAERmini2 [9] and a computer to watch the results with jAER software [11]. Next, we will talk about the USB-AER and the Virtex-5 FPGA board (Fig. 3).

USB-AER board was developed in our lab during the CAVIAR project, and it is based on a Spartan II FPGA with two megabytes of external RAM and a cygnal 8051 microcontroller.

To communicate with the external world, it has two parallel AER ports (IDE connector). One of them is used as input, and the other is the output. In our system we have used two USB-AER boards, one for each retina. In these boards we have synthetized in VHDL a filter called Background-Activity-Filter, which allows us to eliminate noise from the stream of spikes produced by each retina. This noise (or spurious) is due to the nature of analog chips and since we cannot do anything to avoid it in the retina, we are filtering it. So, at the output of the USB-AER we have the information filtered and ready to be processed.

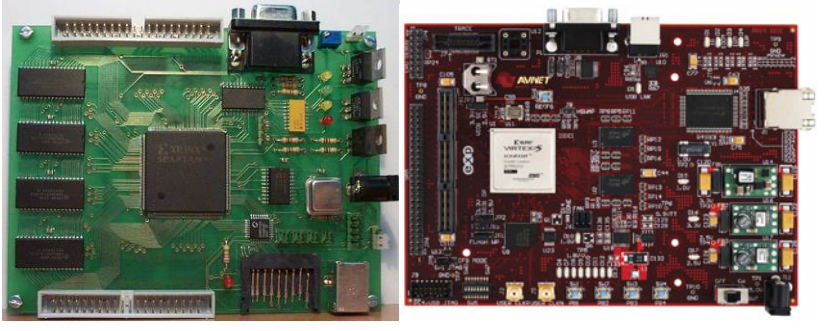

Fig. 3. Left, USB-AER board; right, Virtex-5 FPGA board

The other board used is a Xilinx Virtex-5 board, developed by AVNET [17]. This board is based on a Virtex-5 FPGA and mainly has a big port composed of more than eighty GPIOs (General Purpose Inputs/Outputs ports). Using this port, we have connected an expansion/testing board, which has standard pins, and we have used them to connect two AER inputs and one output.

The Virtex-5 implements the whole processing program, which works with the spikes coming from each retina, processes them and obtains the differences between both retinas and the spikes rate of these differences. The whole program block diagram is shown in figure 4. The system behavior and its functionality are shown in the following sections.

3 Multi Hold and Fire Algorithm

Once we have all the elements of our chain, we can start thinking about the algorithms used. In this work we receive the traffic from both retinas and calculate the differences between them. To do that we have used the idea of the Hold&Fire building block [16] to obtain the difference of two signals. With this block working

correctly, we have extrapolated this to a 128x128 signals system (one for each pixel of the retinas) and obtained a Multi Hold&Fire system that allows us to calculate the differences between both retinas' spikes streams.

The Hold&Fire subtracts two pulses, received from two different ports. When it receives an event, it waits a short fixed time for another event with the same address. If it does not receive a second event and the fixed time is over, it fires the pulse. If, otherwise, it receives another event with the same address, then, if the new event comes from the other retina, the event is cancelled and no event is transmitted, but if this second event comes from the same retina, the first event is dispatched and this second event takes the role of the first event and the system waits again the short fixed time. This Hold&Fire operation for subtracting or cancelling two streams of spikes is described in depth in the paper indicated before.

To summarize, if both events with the same address have the same polarization and come from different ports they are erased and the Hold&Fire block does not fire anything. In this case it fires only when both events have different polarizations. On the other hand, when both events come from the same port, then the opposite happens: it only fires when both events have the same polarization. There are other cases to complete the truth table.

Our algorithm is based on the Hold&Fire block, as it is said before, but it has one Hold&Fire block for each pixel of the retina. It treats each pixel separately and obtains the difference between this pixel in the left retina and the same pixel in the right retina. At the end, we have the difference of both retinas in our system output.

The complete VHDL system, which is completely implemented on the Virtex-5 FPGA board, consists of:

- One handshake block for each retina: these blocks establish the communication protocol with the retina. They are based on a state machine that waits for the retina request signal, receives the AER spike and returns the acknowledge signal.
- Two FIFOs: to storage a great amount of spikes and lose none.
- One Arbitrator: select spikes from both FIFOs depending on the occupation of them.
- Multi Hold&Fire module: applies the algorithm explained before to the stream of spikes received. To storage the information of the first pulse of each sequence, this block uses a dual-port RAM block.
- Distance estimation block: this block will be explained in the next section.
- Handshake out: this block establishes the communication between our processed information and the out world.

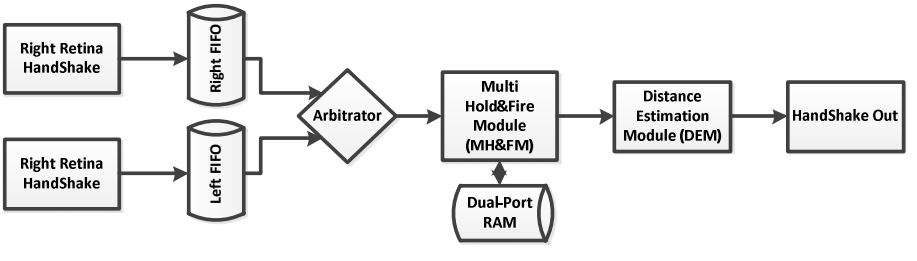

Fig. 4. VHDL block diagram inside Virtex-5 FPGA

4 Calculating Distances

In this section we will talk about the schematic block that was only named in the previously section: distance estimation module.

The algorithm explained is a first approximation to the distance calculation using spikes. Nowadays, there is no algorithm that can solve this problem using only spikes in a real-time environment. That is why we are trying to focus on this field.

Existing algorithms in digital systems extract features from both cameras, process them, and try to match objects from both cameras frame by frame [15]. This process involves high computational costs and does not work in real time.

We want to do this same thing in real time using AER. As a first step to achieve this goal we propose an algorithm based on the spikes rate of the Multi Hold&Fire output.

Theoretically we have to explain the meaning of what we are going to obtain and, after that, we will show the practical results and we will test if both match.

In our system, both retinas are calibrated with a certain angle to obtain a focus distance of 1 m. To do that, we have put our retinas in a base, separated 13'5 cm. We have obtained the system shown below (Fig. 5).

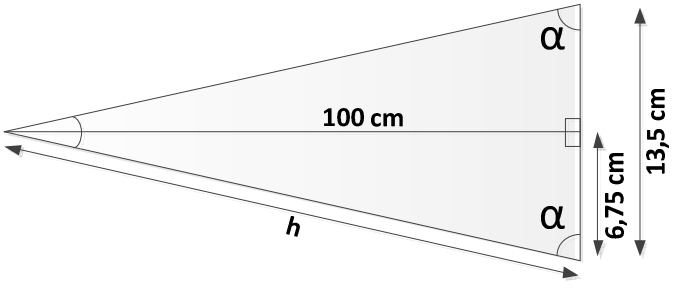

Fig. 5. Retinas situations and trigonometry

Applying Pythagoras and trigonometric rules, we can obtain:

$$h^2 = 6'75^2 + 100^2 \rightarrow h = 100'22755 \text{ cm}$$

$$\sin \alpha = 100 / 100'22755 = 0'99773$$

$$\arcsin 0'99773 = 86'1387º$$

So, our retinas are calibrated with an angle of 86'1387º to obtain a focal distance of one meter. After that, we have measured the spikes rates at the output of our Multi Hold&Fire algorithm using a recorded video about one object in movement. This video has been played many times at different distances from the retinas and we have annotated the number of spikes fired during this video. The resulting spike stream was recorded using jAER software [11]. After measurements, we have recorded all the results and have made a graph with all of them. This graph indicates the number of spikes versus distance.

It is logical to think that, at the central match point of the focal length of each retina, the Multi Hold&Fire acts like a perfect subtractor and do not fire any spike at all (except for the retinas' spurious), so the spike rate at this point is near zero. If we approach the retinas, the spike rate will be increased because the object becomes bigger and each retina sees it from a different point of view, so the subtractor will not act so perfectly.

Otherwise, if we put the video further from the retinas, spike rate will be slightly increased due to the subtraction result (different points of view of the retinas), but the object becomes smaller, so it compensates this failure: as the further the object is, the smaller it is and therefore, the lower the spike rate is; because less spikes are fired, but the subtraction acts worse and fires more spikes. That is why the second aspect is balanced with the first one.

So, in theory, we will see a graph where the number of spikes is greatly increased near the retinas and is slightly increased as we move away the object from the focal collision point. We can see the experimental results in the next figure. We have stimulated our system using two types of stimuli: an oscillating pendulum and a vibrating ruler. Measurements were taken from the starting point of 10 centimeters to 150 centimeters. They were taken every 10 centimeters.

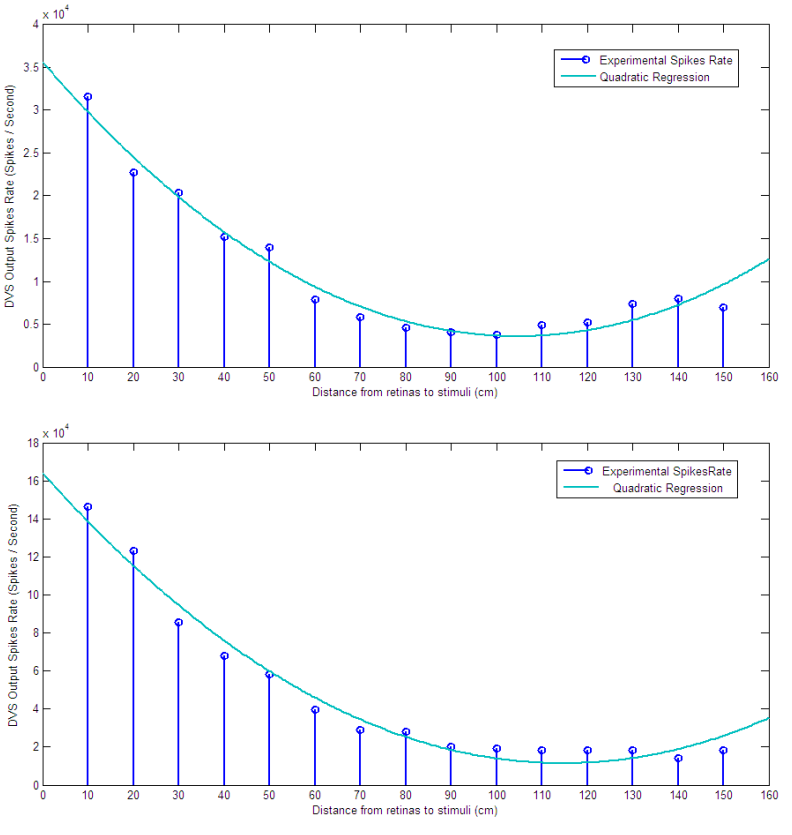

Fig. 6. Spike rate versus Distance. Up: Pendulum. Down: Ruler.

In figure 6 we can see the experimental results obtained. It is interesting to remark that, approximately, at a distance of 100 centimeters (focal collision of both retinas) we obtained the lowest spike rate. If we see measurements taken closer, it can be seen that spike rate increases, and far away from the focal collision point, the spike rate is increased a little. Finally, we obtained the results we expected. In the first case, with the pendulum, it can be seen better than the other one; but, what is true is that, in a short distance, we can estimate the distance of the object in movement with a quadratic regression (line in blue shown in figure 6).

It is very interesting to see that our system behaves similarly to human perception: a priori, without knowing the size of the object, we cannot give exact distances, only an approximation. This approximation depends on our experience, but the system proposed cannot learn. However, we can measure the distance qualitatively, and interacting with an object in a nearby environment.

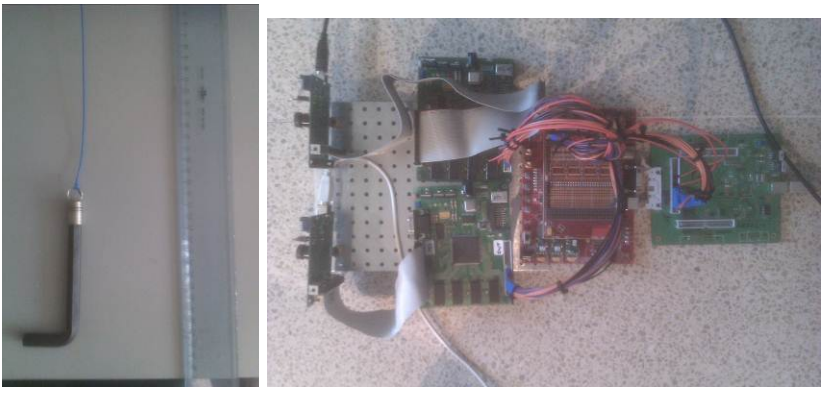

Fig. 7. Hardware used to test our system (right) and stimulus used (left)

5 Conclusions

The existing difficulties to calculate distances in digital systems have been shown. That is why a biological approach (Address-Event-Representation) to work with has been presented. We have introduced the Address-Event-Representation notation to communicate neuro-inspired chips as a new paradigm in Neuromorphic Engineering. We have evaluated the advantages of this method and explained why we work with it.

In this work we propose a first approximation to distance estimation using spikes in a close environment. To do that, a stereoscopic vision system with two DVS retinas has been used, working with VHDL over a Virtex-5 FPGA.

We have described and shown the whole system used and each one of its elements required to obtain the distance estimation. With the hardware system described, we have explained the algorithms used. The first algorithm uses a method to obtain differences between both retinas in real time and without sampling. With these differences the second algorithm was explained, which works with the spike rate obtained in our system after the differences calculation.

With the results of these two algorithms, we have been able to model the spike rate versus the distance of the object. The simulation results are very encouraging, because we can see in the graphs shown that there is a relationship between distance and the spike rate after our processing and that this system works quite similar to human perception.

Acknowledgements. First, we want to thank the contribution of Tobias Delbruck, whose retinas are used in this work; as well as Raphael Berner, whose USBAERmini2 was used too. Without their contributions this work could not have been done. At last, but not least, we want to thank the Spanish government, which has supported the project VULCANO (TEC2009-10639-C04-02).

References

1. Barnard, S.T., Fischler, M.A.: Computational Stereo. Journal ACM CSUR 14(4) (1982)
2. Shepherd, G.M.: The Synaptic Organization of the Brain, 3rd edn. Oxford University Press (1990)
3. Lee, J.: A Simple Speckle Smoothing Algorithm for Synthetic Aperture Radar Images. Man and Cybernetics SMC-13 (1981)
4. Crimmins, T.: Geometric Filter for Speckle Reduction. Applied Optics 24, 1438–1443 (1985)
5. Linares-Barranco, A., et al.: AER Convolution Processors for FPGA. In: ISCASS (2010)
6. Cope, B., et al.: Implementation of 2D Convolution on FPGA, GPU and CPU. Imperial College Report (2006)
7. Cope, B., et al.: Have GPUs made FPGAs redundant in the field of video processing? In: FPT (2005)
8. Sivilotti, M.: Wiring Considerations in analog VLSI Systems with Application to Field-Programmable Networks. Ph.D. Thesis, Caltech (1991)
9. Berner, R., Delbruck, T., Civit-Balcells, A., Linares-Barranco, A.: A 5 Meps $100 USB2.0 Address-Event Monitor-Sequencer Interface. In: ISCAS, New Orleans, pp. 2451–2454 (2007)
10. Lichtsteiner, P., Posh, C., Delbruck, T.: A 128×128 120dB 15 us Asynchronous Temporal Contrast Vision Sensor. IEEE Journal on Solid-State Circuits 43(2), 566–576 (2008)
11. jAER software: `http://sourceforge.net/apps/trac/jaer/wiki`
12. Benosman, R., Devars, J.: Panoramic stereo vision sensor. In: International Conference on Pattern Recognition, ICPR (1998)
13. Benosman, R., et al.: Real time omni-directional stereovision and planes detection. In: Mediterranean Electrotechnical Conference, MELECON (1996)
14. Douret, J., Benosman, R.: A multi-cameras 3D volumetric method for outdoor scenes: a road traffic monitoring application. In: International Conference on Pattern Recognition, ICPR (2004)
15. Dominguez-Morales, M., et al.: Image Matching Algorithms using Address-Event-Representation. In: SIGMAP (2011)
16. Jimenez-Fernandez, A., et al.: Building Blocks for Spike-based Signal Processing. In: IEEE International Joint Conference on Neural Networks, IJCNN (2010)
17. AVNET Virtex-5 FPGA board: `http://www.em.avnet.com/drc`