

# Embedded neuromorphic vision for humanoid robots

Chiara Bartolozzi, Francesco Rea, Charles Clercq  
Italian Institute of Technology  
via Morego 30, Genova, Italy  
chiara.bartolozzi@iit.it

Daniel B. Fasnacht, Giacomo Indiveri  
University of Zurich, ETH Zurich  
Winterthurerstr. 190, Zürich, Switzerland  
fasnacht@ini.phys.ethz.ch

Michael Hofstätter  
Austrian Institute of Technology  
Donau-city Straße 1, Wien, Austria  
Michael.Hofstaetter@ait.ac.at

Giorgio Metta  
Università degli Studi di Genova  
Viale F. Causa 13, Genova, Italy  
pasa@liralab.it

## Abstract

*We are developing an embedded vision system for the humanoid robot iCub, inspired by the biology of the mammalian visual system, including concepts such as stimulus-driven, asynchronous signal sensing and processing. It comprises stimulus-driven sensors, a dedicated embedded processor and an event-based software infrastructure for processing visual stimuli. These components are integrated with the existing standard machine vision modules currently implemented on the robot, in a configuration that exploits the best features of both: the high resolution, color, frame-based vision and the neuromorphic low redundancy, wide dynamic range and high temporal resolution event-based sensors. This approach seeks to combine various styles of vision hardware with sensorimotor systems to complement and extend the current state-of-the-art.*

## 1. Introduction

The mainstream computational paradigm in embedded vision is founded on the concept of digitized frames, obtained by sampling the world at a regular fixed rate and by processing sequences of “static snapshots”. Visual data transfer, storage and processing are therefore time and CPU consuming tasks, even though in most situations the amount of information within each frame and across consecutive frames is highly redundant. Using this approach, tasks that seem effortless for animals such as object categorization [20], motion estimation and direction of attention (to name a few) require vast amounts of computing resources. Even then, the outcome is hardly robust and often results are obtained with latencies that do not allow efficient interactions with the real world, in real-time. These problems are further complicated by the fact that in sensory perception

tasks the data being sensed and processed is typically noisy and ambiguous. “Frame-based” time sampling and quantization artifacts present in conventional sensors are particularly problematic for robust and reliable performance.

Conversely, in the nervous system computation is predominately stimulus driven, triggered by significant events (spikes) which are produced either by the periphery – i.e. the sensors – or simply by the system’s internal states. Neural computation is often characterized by primitives that enhance variations and discontinuities, and discard redundancies. For example, visual signals are typically encoded by temporal and spatial contrast operations [18]; this, combined with forms of automatic gain control, such as adaptation and learning, allow the system to respond to a wide dynamic range in the input signals, and to produce appropriate behaviors in a wide variety of different conditions [24, 17, 20].

Understanding the computational principles of biological systems allows researchers to engineer these principles into bio-inspired technology that can take advantage of the strength of brain-like sensing and computing [22, 5]. This approach is crucial for robots to robustly and reliably interact with the real world in real-time and for the development of fully autonomous robotic systems.

In this paper we describe the application of some of these ideas in the development of a biomorphic vision system for the Open Source humanoid platform iCub [31] (see also <http://www.icub.org/>). In particular, we use the “Dynamic Vision Sensor” (DVS) [19], which is a VLSI implementation of the transient signal pathway of the mammalian visual system that encodes visual scenes in an asynchronous event-driven fashion. For the computing infrastructure we developed a framework for processing the sensed signals and producing motor outputs with event-driven algorithms and procedures.

The use of asynchronous vision in humanoid robotics offers the advantage of robust and adaptive signal encoding. This in turn enables more efficient operation in different environments and conditions (e.g. lighting conditions, fast moving objects and so forth). In particular, the use of sparser temporal and spatial information extracted already at the input sensor level can lead to faster and lighter computation, which will be eventually implemented on custom compact and low-power devices, ideal for autonomous robotics.

In turn, the use of a sophisticated humanoid robot gives the advantage of challenging the vision system with complex tasks and movements, that would otherwise be more limited, as is the case for experiments with desktop setups and simple actuators. We exploit for example the six degrees of freedom of the head for generating saccades, vergence, head and synchronized torso movements.

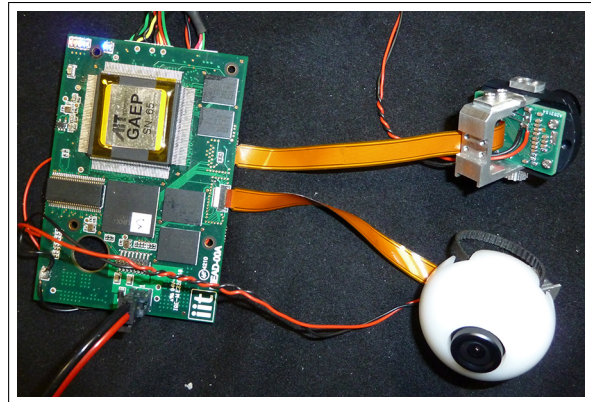
In the following sections we describe the iCub’s hybrid visual system which merges a pair of asynchronous sensors with the traditional frame-based vision<sup>1</sup>. After describing the main hardware components we will then show a case study with the implementation of logpolar visual attention.

## 2. Embedded asynchronous vision

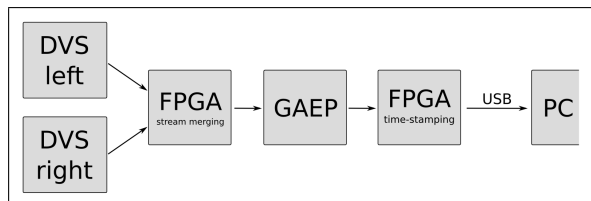
The current implementation of the visual system comprises two DVS sensors, an embedded dedicated processor for asynchronous data processing, the General Address Event Processor (GAEP) [14], and a Field Programmable Gate Array (FPGA) used to interface the sensors to the GAEP and to a CPU. These custom hardware components are designed to communicate asynchronous data with a spike based “address-event representation” (AER) protocol [23, 10]: the sensors produce events that are sent on the communication bus in form of digital pulses associated with the identity (or address) of the active pixel. The information is self-encoded in the timing of the events, it is explicitly added to the address in form of a time-stamp only when communication to a synchronous digital processor is needed.

A software module collects streams of asynchronous events that are then available for further processing and visualization. The asynchronous eyes replace the standard frame-based cameras inside the eyeballs of the iCub, while the standard cameras are placed on top of the robot’s head. This second visual input is used in parallel to the asynchronous cameras to complement the visual information with color-based images currently needed for some visual modules such as object-based attention, object segmentation and recognition.

<sup>1</sup>Dragonfly<sup>®</sup> 2, IEEE-1394a FireWire digital camera, on-board digital processing, 1/3” Sony CCDs, from PointGrey



(a)



(b)

Figure 1. (a) Picture of the miniaturized iHead board that hosts the GAEP and the FPGA, connected with the asynchronous eyes. (b) Data flow (the two functionalities of the FPGA board are implemented on the same chip).

### 2.1. Vision Sensor

The asynchronous sensor mounted on the iCub is the well known DVS, an event-driven AER sensor that detects changes of contrast in its visual field and encodes them in a sequence of digital pulses (events) that convey information about the visual input in their relative timing. Each pixel independently responds to variations of contrast in the visual field, achieving a wide intra-scene dynamical range, that makes the sensor capable of responding to a large interval of contrast over different illuminations.

The DVS has been successfully used for fast robotic applications [6], vehicle tracking and classification [13] among others (see [9] for a review). Currently two DVS cameras are mounted on the iCub and used to develop procedures and algorithms for asynchronous vision.

### 2.2. Embedded hardware

Figure 1 and 2 show the vision sensors and the embedded hardware used for the first processing steps of asynchronous data.

The event streams from the two DVS are tagged as left/right and then brought together in the FPGA. This stream is then forwarded to the GAEP where a first processing step can be performed on the visual input. The data

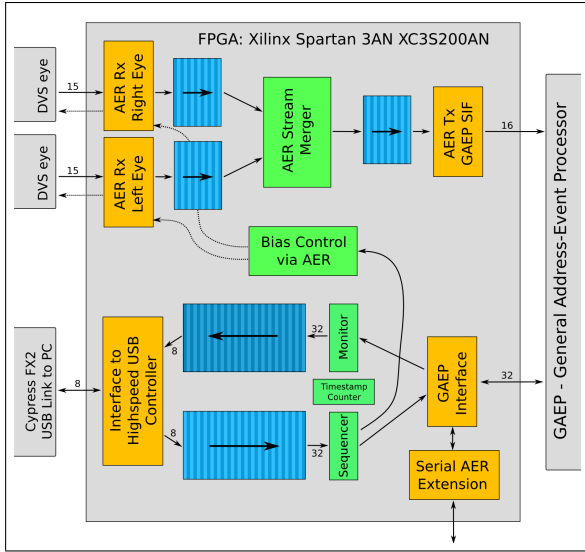


Figure 2. FPGA block diagram. Orange: Interface logic, blue/striped: FIFOs, green: internal logic & control. Solid arrows: AER data-path, dashed arrows: bias control for the DVS.

is then returned to the FPGA where it is time-stamped and forwarded over a high-speed USB interface.

Figure 2 shows the FPGA internals that were developed for this application. In the top part the AER streams from the left and right DVS are merged and forwarded to the *Sensor Interface* of the GAEP.

The lower part shows the GAEP to USB data-path, where address-events processed by the GAEP are time-stamped and then sent off via high-speed USB. This part of the data-path is fully bidirectional such that the PC can also send data to the GAEP, e.g. to configure its operations.

Another use of this reverse data-path is shown in the center of Fig. 2: The FPGA can tune the DVS operating conditions by means of on-chip programmable bias generators [8]. Initial setup of the bias values and on-line reconfiguration are done via configuration-address-events sent from the PC to the Bias Control unit.

The block “Serial AER Extension” can optionally be used to reroute the data coming from the GAEP to another AER processing stage in order to extend the functionality and processing done in the asynchronous domain.

### 2.2.1 General Address-Event Processor

The GAEP is based on a SPARC-compatible LEON3 core with a custom data interface for asynchronous sensor data. The GAEP responds to the need for transferring the inherently precise timing information of asynchronous events into the synchronous domain at high temporal resolution (at the input of the GAEP the maximal time resolution is 10 ns), combined with local processing capability functional-

ity. The main functional hardware blocks of the GAEP sensor interface include data transfer and protocol handshake, data rate measurement, data filtering, time-stamp assignment and input data buffer management.

As the asynchronous sensors respond to variations in the visual input, the GAEP input data rate is highly variable, hence the data rate measurement is used to control the processor and the software execution, optimizing the use of resources on the basis of the needs. This mechanism is intended to bridge the gap between asynchronous and traditional frame based synchronous data processing.

In the final visual system implementation, the GAEP will perform preprocessing of visual data, conveying information about visual features (e.g. edge orientation) to the robot’s core computing unit, where it will be used for extracting high-level information about the environment.

## 2.3. Software

The software procedures for handling and computing with events are built using the YARP middleware [11]. Designed to implement high degree of parallelism by facilitating concurrent processing and distributed computing, YARP provides libraries that allow nodes to easily communicate in a generic network. This approach is bio-inspired, as parallel processing is one of the key features of brain computation accounting for fast response time [32]. In YARP, parallel processing could be implemented by distributing computation over many different processors that communicate on a common network. Figure 3 shows the specific instantiation of the network on the system described in this paper (in general the robot can support various network and device configurations). The iCub is placed on a mobile holonomic platform that enables navigation on a flat terrain. A PC104 standard CPU unit is located inside the head of the robot. The FPGA of the iHead board is directly connected via a USB connection to this CPU unit, where a device driver provides the possibility of reading and writing events to the board. From this point on the events arriving from the visual sensors are made available on the local network via Gigabit Ethernet. An i7 CPU PC is placed on the mobile base and is connected to the Gigabit Ethernet. Given its computational power and the high performing connection with the PC104, most of the real-time processing is allocated in this node. Other nodes can be connected to the network through a WiFi connection, with potentially unlimited number of nodes for other tasks.

### 2.3.1 Event-based modules

The *aexGrabber* module collects events from the FPGA and broadcasts them to the YARP network, minimizing the overhead that could come from multi-point communication between computing nodes. This module can also send events

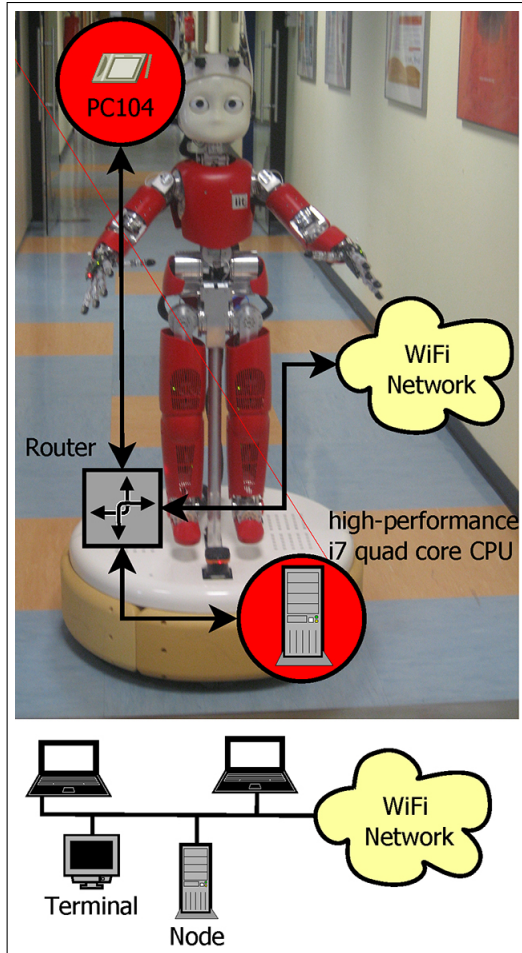
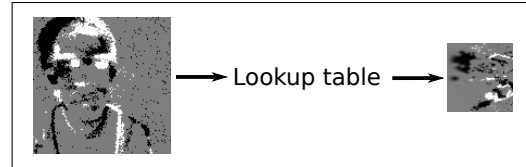
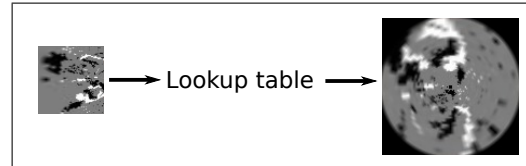


Figure 3. Picture of the iCub computing infrastructure: each computing node is connected to the network, the PC104 and i7 core are connected by a Gigabit Ethernet and communicate with the rest of the network by means of a WiFi connection.

to the FPGA board, this option is used to configure each vision sensor and optimally tune their behaviour. Any other module can read the events from the flux of data in the network as they travel as multicast datagram packets. Among them, the *frameConverter* module groups events into sequences of constant time frames for easy visualization of the input stimuli as traditional movies on a computer screen. Events are read from the network and only those whose timestamps belong to the current frame (delimited by initial timestamp  $t_i$  and  $t_f$ ) are represented in their respective location of the visual field. In this process we initially synchronize the timestamps of the frame with the counter clock of the FPGA board and then increment them of the elapse of time between two successive frames.



(a) Transformation from the Cartesian coordinate space to the logpolar coordinate space.



(b) Transformation from the logpolar to the Cartesian coordinate space.

Figure 4. Logpolar mapping of events. Each event is represented as a dot (white for transitions from dark to light and black for the opposite transition).

### 3. An example: visual attention

As an example application of the whole system to a specific problem of computer vision, in this section we will describe the implementation of visual attention on the newly developed visual system.

The visual attention framework is designed using the logpolar mapping of the visual input channel. The logpolar mapping is a model of the topological transformation of the primate visual pathways from the retina to the visual cortex. In the human eye, the distribution of photoreceptors has a high density centre, the fovea, and progressively degrades towards the periphery. This arrangement preserves high resolution in the fovea, needed for high detail analysis of visual input, and at the same time optimizes data compression with respect to the width of the visual field, where the low-resolution activity in the periphery is used to redirect the gaze at a new fixation point. Finally, it is shown that the use of logpolar images increases the size range of objects that can be tracked using a simple translation mode [3].

The generation of the logpolar events is realized as software task in the GAEP. Each address event consists of three fields:  $x$ ,  $y$  Cartesian coordinates and polarity of the contrast change. After extracting the coordinates of each address event, the transformation into the logpolar representation is retrieved from look up tables, whose parameters are computed from [1]. It is handled as simple procedure call in a C-based AER application framework. The resulting mapping is shown in Fig. 4(a). The look-up table implementation on the GAEP of the logpolar transform enables a flexible, fast and low-power calculation of logpolar events that does not interfere with the high temporal resolution of the sensors' events.

In order to have a better comprehensive visualization of

this transformation, the logpolar representation can be projected back in the cartesian space, as shown in Fig. 4(b). This transform illustrates the retinal organization with the high concentration of photo-receptor cells at the center (fovea), that decreases towards the periphery. As this representation is done for visual purpose only, its computation is let to an external module to avoid unnecessary use of the GAEP.

The logpolar events are then sent to the PC104, where they are collected by the *axGrabber* for further processing. Specifically, the *frameConverter* module collects the events and creates packets synchronized with the CPU time scheduling, creating artificial frames that, differently from standard frames, preserve the timing information of events belonging to the same frame. While developing modules for event-based artificial vision, the representation of events as frames is a crucial step toward the innovative goal of integrating traditional visual perception with event-driven perception. Processing events that have temporal and spatial information brings evident advantages in both the computation required for processing and the discrimination of stimuli with respect to time. The first step towards this goal is the integration of event-based attention modules with the available iCub attentive system, needed to direct attention and gaze towards salient regions of the visual input, as shown in Fig. 5. The visual attention system is shaped on a model of stimulus-driven primates spatial attention [15], enriched with object-based components. In the spatial-based approach the location to attend is determined after the computation is carried out on the complete set of sensory cues. On the other hand, in the object-based approach the processing is performed only in the subpart of the sensory space where the object is localized even before it is recognized [25].

In biological vision, neurons at the early stages of processing (e.g. the lateral geniculate nucleus, the primary visual cortex, etc.) respond to simple visual attributes such as intensity, contrast, orientation, motion and colour opponency. At a higher level, neurons respond also to corners [26] and edges that improve the richness of extracted features. Combining together these features, the model identifies views of object [12, 33] or proto-objects [28, 16, 29]. Motivated by this evidence, we designed the computing infrastructure for the emulation of the early visual stages of processing. Centre-surround receptive fields and feature map normalization [15] are obtained by convolution with difference of Gaussians. For instance, the response of center-on and center-off neurons is computed by convolving the luminance channel of logpolar images with a difference of Gaussian. At higher level, edges and orientations are extracted and fed as input to the proto-object extractor.

Event-based motion, orientation and contrast represent

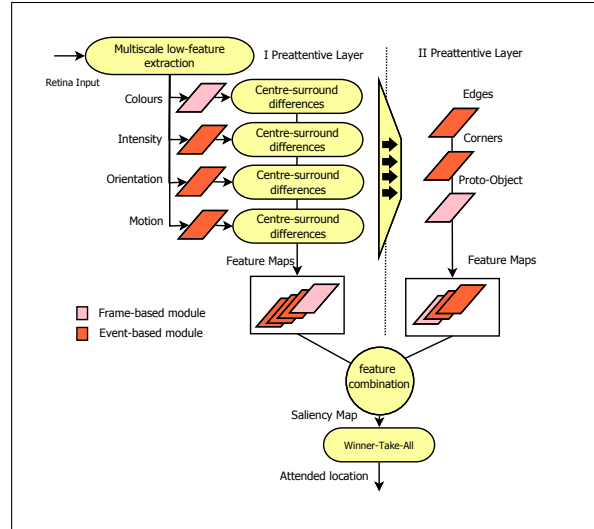


Figure 5. Representation of different layers of processing of the visual attention implementation.

additional feature maps that will be added later into the computation of the saliency map.

These feature maps are then combined together by weighting each of them depending on the current task, implementing a simple form of top-down attentional bias. From this process we are able to extract the saliency map and apply a winner-take-all competitive stage [15] which sequentially selects salient regions of the visual field, in order of decreasing saliency. This information is used to re-direct the gaze of the robot towards interesting regions of the scene. The saccade command is generated by a dedicated software module, the *gazeArbiter*, that smoothly changes its state between a set of possible oculomotor commands and finally asks the head controller to recalculate head and torso configuration in space to reach the new fixation point. The *gazeArbiter* contains also additional oculomotor commands as required to control vergence or high priority saccades. The former is used to control the angle of vergence that brings an object at zero disparity in the fovea of both cameras [21], the latter can be used when a persistent train of events in the periphery attracts the attention as a sudden and very salient event.

The combination between event-based and frame-based colour feature maps augments the skills of the humanoid robot; in addition, through event-based and logpolar vision we are able to require a smaller amount of computation power. This allows the system to evolve towards autonomy, in the way processing can be restricted to the mobile infrastructure, and asks only a minor part of the process to be allocated on external computing nodes.

## 4. Conclusions

We presented a hardware-software infrastructure for developing vision optimized for robotics which uses biomorphic concepts. In particular, we used the humanoid platform iCub by modifying the cameras and replacing electronics and certain interfaces. At the hardware level, we integrated the DVS cameras, a custom processor (GAEP) and FPGA, using the asynchronous AER communication protocol as interface. At the software level, we developed the basic processing modules required for the development, evaluation and debugging of asynchronous artificial vision algorithms.

As a first step, we integrated the asynchronous vision system with the robot's standard frame-based color vision system, and developed a logpolar attentive module that makes use of feature maps computed using both traditional and novel event-driven vision sensors. We followed a modular approach, to allow progressive instantiation of additional event-based modules, implemented for example on embedded hardware such as the dedicated GAEP (e.g. for logpolar mapping of events and feature extraction) or on standard PC systems. This modular approach will allow us to interchange and combine frame-based vision sensors with event-based processing systems; the latter has a much smaller computational cost.

Future work will explore the implementation of dedicated custom logpolar asynchronous vision sensors, based on recent developments [30, 7, 27], for encoding spatio-temporal contrast or intensity levels using a non-uniform retinotopic arrangement of pixels. This new endeavor, complemented by the recent developments on color-based asynchronous vision sensors [4], will lead to a new set of powerful vision sensors for event-based frame-less machine vision.

We plan to integrate custom AER chips for post-processing of sensory events, by means of the serial connection mentioned in Sec. 2.2. Specifically, in the context of the implementation of visual attention we will integrate the Selective Attention Chip (SAC) [2]. The SAC is an asynchronous neuromorphic chip that receives the calculated saliency map as AER input and implements a winner-take-all competitive stage together with a self-inhibition mechanism known as inhibition of return (IOR) [15] to eventually produce the targets for subsequent saccades.

The implementation of the asynchronous visual system on an Open Source robotic platform such as the iCub, capable of meaningful interaction with the real world in real-time, is the key to validate the advantages of the neuromorphic approach and make them available to a wide community.

## 5. Acknowledgments

## References

- [1] R. Alan and P. Li. On the computation of the discrete logpolar transform, 2007. 132
- [2] C. Bartolozzi and G. Indiveri. Selective attention in multi-chip address-event systems. *Sensors*, 9(7):5076–5098, 2009. 134
- [3] A. Bernardino and J. Santos-Victor. Binocular visual tracking: Integration of perception and control, 1999. 132
- [4] R. Berner and T. Delbruck. Event-based color change pixel in standard cmos. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, 30 2010. 134
- [5] K. Boahen. Neuromorphic microchips. *Scientific American*, pages 56–63, May 2005. 129
- [6] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. Douglas, and T. Delbruck. A pencil balancing robot using a pair of aer dynamic vision sensors. In *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on*, pages 781–784, May 2009. 130
- [7] J. Costas-Santos, T. Serrano-Gotarredona, R. Serrano-Gotarredona, and B. Linares-Barranco. A spatial contrast retina with on-chip calibration for neuromorphic spike-based aer vision systems. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 54(7):1444–1458, 2007. 134
- [8] T. Delbruck, R. Berner, P. Lichtsteiner, and C. Dualibe. 32-bit configurable bias current generator with sub-off-current capability. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 1647–1650, 2010. 131
- [9] T. Delbruck, B. Linares-Barranco, E. Culurciello, and C. Posch. Activity-driven, event-based vision sensors. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 2426–2429, 2010. 130
- [10] D. Fasnacht, A. Whatley, and G. Indiveri. A serial communication infrastructure for multi-chip address event system. In *International Symposium on Circuits and Systems, ISCAS 2008*, pages 648–651. IEEE, May 2008. 130
- [11] P. Fitzpatrick, G. Metta, and L. Natale. Towards long-lived robot genes. *Robotics and Autonomous Systems*, 56(1):29–4, 2008. 131
- [12] G. Kreiman, C. Koch, and I. Fried. Category-specific visual responses of single neurons in the human medial temporal lobe. *Nature Neuroscience*, 3:946–953, 2000. 133
- [13] G. Gritsch, N. Donath, B. Kohn, and M. Litzenberger. Night-time vehicle classification with an embedded, vision system. In *Intelligent Transportation Systems, 2009. ITSC '09. 12th International IEEE Conference on*, pages 1–6, 2009. 130
- [14] M. Hofstätter, P. Schön, and C. Posch. A sparc-compatible general purpose address-event processor with 20-bit 10ns-resolution asynchronous sensor data interface in 0.18 $\mu$ m cmos. In *International Symposium on Circuits and Systems, ISCAS 2010*, pages 4229–4232. IEEE, May 2010. 130
- [15] L. Itti and C. Koch. Computational modeling of visual attention. *Nature Reviews Neuroscience*, 2(3):194–203, 2001. 133, 134

- [16] D. Kahneman and A. Treisman. *Varieties of attention*, chapter Changing views of attention and automaticity, pages 29–61. R. Parasuraman, & D. A. Davies (Eds.), 1984. 133
- [17] E. Kandel, J. Schwartz, and T. Jessell. *Principles of Neural Science*. Mc Graw Hill, 2000. 129
- [18] D. Kiper and M. Carandini. *Encyclopedia of Cognitive Science*, chapter The neural basis of pattern vision. Macmillan, 2002. 129
- [19] P. Lichtsteiner, C. Posch, and T. Delbruck. An 128x128 120dB 15 $\mu$ s-latency temporal contrast vision sensor. *IEEE J. Solid State Circuits*, 43(2):566–576, 2008. 129
- [20] M. Macé, A. Delorme, G. Richard, and M. Fabre-Thorpe. Spotting animals in natural scenes: efficiency of humans and monkeys at very low contrasts. *Animal Cognition*, 13:405–418, 2010. 10.1007/s10071-009-0290-4. 129
- [21] R. Manzotti, A. Gasteratos, G. Metta, and G. Sandini. Disparity estimation in log polar images and vergence control. *Computer Vision and Image Understanding*, 83:97–117, 2001. 133
- [22] C. Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–36, 1990. 129
- [23] A. Mortara. A pulsed communication/computation framework for analog VLSI perceptive systems. In T. Lande, editor, *Neuromorphic Systems Engineering*, pages 217–228. Kluwer Academic, Norwell, MA, 1998. 130
- [24] I. Ohzawa, G. Sclar, and F. R.D. Contrast gain control in the cat’s visual system. *Journal of Neurophysiology*, 54(3):651–667, 1985. 129
- [25] F. Orabona, G. Metta, and G. Sandini. Object-based visual attention: a model for a behaving robot. In *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, page 89, 2005. 133
- [26] A. Pasupathy and C. Connor. Responses to contour features in macaque area v4. *Journal Neurophysiol*, 82(5):2490–2502. 133
- [27] C. Posch, D. Matolin, and R. Wohlgenannt. A qvga 143db dynamic range asynchronous address-event pwm dynamic image sensor with lossless pixel-level video compression. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2010 IEEE International*, pages 400–401, 2010. 134
- [28] R. Rensink. Seeing, sensing, and scrutinizing. *Vision Research*, 40(10-12):1469–1487. 133
- [29] M. Riesenhuber and T. Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):1019–25, 1999. 133
- [30] P.-F. Ruedi, P. Heim, F. Kaess, E. Grenet, F. Heitger, P.-Y. Burgi, S. Gyger, and P. Nussbaum. A 128 times; 128 pixel 120-db dynamic-range vision-sensor chip for image contrast and orientation extraction. *Solid-State Circuits, IEEE Journal of*, 38(12):2325 – 2333, 2003. 134
- [31] G. Sandini, G. Metta, and D. Vernon. *50 Years of AI*, chapter The iCub Cognitive Humanoid Robot: An Open-System Research Platform for Enactive Cognition, pages 359–370. Springer-Verlag, 2007. 129
- [32] J. Tsotsos. Analyzing vision at the complexity level. *Behavioural and Brain Science*, 13(10-12):423–469. 131
- [33] D. Walther and C. Koch. Modeling attention to salient proto-objects. *Neural Networks*, 19:1395–1407, 2006. 133