# LABORATORY NOTES

## Saliency on a chip: a digital approach with an FPGA

Selective-visual-attention algorithms have been successfully implemented in analog VLSI circuits.[1] However, in addition to the usual issues of analog VLSI—such as the need to fine-tune a large number of biases—these implementations lack the spatial resolution and pre-processing capabilities to be truly useful for image-processing applications. Here we take an alternative approach and implement a neuro-mimetic algorithm for selective visual attention in digital hardware.

The overarching aim of this project is to demonstrate the feasibility of using programmable logic for aiding the development and acceleration of image and video processing applications in vision. Saliency was picked as a design driver for this purpose so that the design flow could be understood and added to the neuromorphic engineer's bag of tricks. The data-intensive and computationally challenging nature of the human visual attention system makes it an interesting algorithm for this study.

Itti, Koch, and Niebur[2] have suggested an influential model for saliency-based bottom-up visual attention and applied it successfully to a variety of visual tasks. The model attempts to represent visual attention in a computationally-efficient manner. The existing software implementation of this model, on a personal computer, runs at 30 frames per second (fps) at quarter-VGA (320×240) resolution.[3] Field-programmable gate arrays (FPGAs), on the other hand, offer an elegant solution to implementing the saliency computation in hardware, taking full advantage of the data parallelism available in the image processing operations.[4] The reprogrammable nature of the FPGAs provides for a quick, cheap platform for prototyping and debugging, and greatly simplifies development.

We wanted to be able to process a video stream at 30fps and at VGA resolution of 640×480 pixels (R, G, and B colors at 8bits/color/pixel), outputting the coordinates of the most salient pixel. Our current design exceeds that specification by processing composite video at 720×525 pixels and 30fps. The hardware was composed using modular elements that implement color-space conversion, Gaussian filtering, interpolation, decimation, and basic image arithmetic transformations, in addition to support for mapping image streams to and from an external memory. The computation of saliency is performed in a series of steps where intermediate 'maps' of the image are created by series of transformations.

The incoming video stream (see Figure 1) is first de-interlaced by a write-read operation through the external synchronous dynamic random-access memory (SDRAM). The de-interlaced image is then separated into its constituent components, which are post-processed to compute red-green and blue-yellow color opponencies, luminance, and orientation-filtered streams. Gaussian pyramids are generated for the incoming streams where each pyramid level is computed by successively low-pass filtering and subsampling the previous level. We have developed an architecture that computes the entire pyramid with a single filter kernel by time multiplexing the different scales/levels of the pyramid and buffering them into a single dual-ported memory using a specialized, mutually exclusive write addressing scheme.

The filtered data is then buffered in the off-chip SDRAM due to the on-chip memory capacity constraints of the FPGA. Feature maps are created next by calculating the difference of chosen pairs of center (finer) and surround (coarser) spatial scales. The latter is first scaled up by stretching it to the finer scale followed by point-wise subtraction of the two streams. Merging the different center-surround scales by decimating all the streams to a specific scale[4] and performing point-wise addition gives us 'conspicuity maps'. These are then normalized with the global maximum of the previous image (as an approximation to the global maximum of the current image, for efficiency). The final saliency map is produced by merging the intensity, color, and orientation streams, and is processed to compute the most salient point.

The Berkeley development board[5]—with its video support, decent-sized Xilinx FPGA and generous amount of SDRAM, see Figure 2—was used as the platform for developing the saliency algorithm. A key contribution of this research was the design of a decentralized memory controller for mapping streams to an external memory and a modular method for plugging in new streams. An area- and memory-efficient mechanism for computing Gaussian pyramids was also demonstrated.

Much work concerning the implementation of the entire design stack needs to be done before more conclusive lessons can be learned, but we hope this project will eventually provide

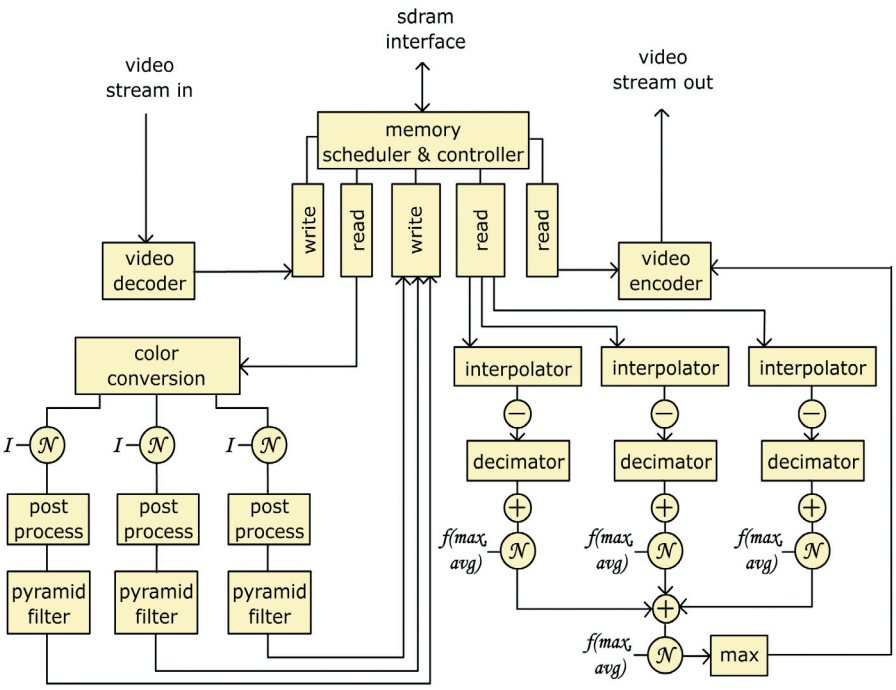


*Figure 1: Block diagram of the FPGA saliency implementation.*

***Kapre,** continued on p. 11*

***van Schaik et al.,*** *Continued from p. 10*

splitter has $N$ stages; the current at the $k^{th}$ stage is $I_{in}/2^k$.

We have used these bias generators in several generations of CMOS process technology (1.6$\mu$m, 0.8$\mu$m, and 0.35$\mu$m) with no striking differences in performance. Here we show a result from a bias generator with a 20-stage octave splitter built in a 0.35$\mu$m process using the design kit. Figure 2 shows the measure output currents of the octave splitter biased with a single generated master bias current of 10 $\mu$A. It is amazingly ideal over 20 octaves (six decades) spanning strong to weak inversion. A current of 10pA is reliably generated from a master current of 10$\mu$A. For more details of the bias generator, the design kit, and measurements, readers are referred to References 1 and 4.

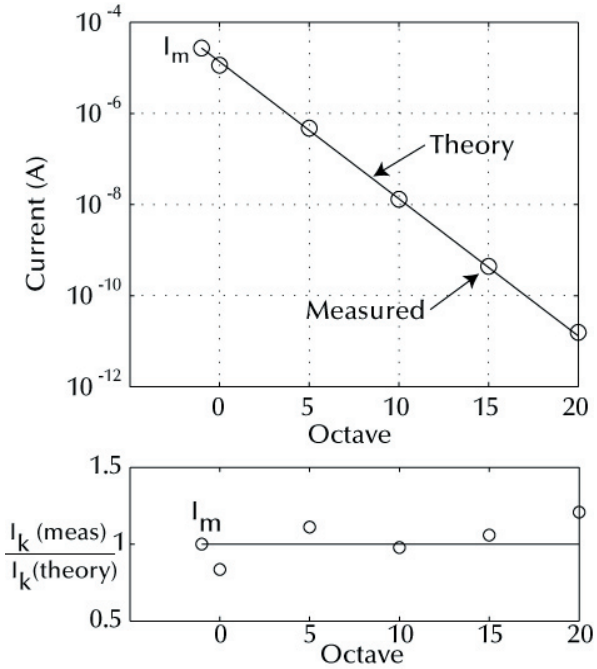


*Figure 2: Behavior of the octave splitter.*

**André van Schaik and Tobi Delbrück***
Univ. of Sydney, Australia
E-mail:
andre@ee.usyd.edu.au
*Inst. of Neuroinformatics, ETH/Univ. Zürich
Switzerland
E-mail:
tobi@ini.phys.ethz.ch

**References**
1. *www.ini.unizh.ch/~tobi/biasgen*
2. E. Vittoz and J. Fellrath, *CMOS analog integrated circuits based on weak inversion operation,* **IEEE J. of Solid-State Circuits SC-12,** pp. 224-231, 1977.
3. G. Bult and G. Geelen, *An inherently linear and compact MOST-only current division technique,* **IEEE J. of Solid-State Circuits 27,** pp. 1730-1735, 1992.
4. T. Delbrück and A. van Schaik, *Bias Current Generators with wide dynamic range,* **IEEE ISCAS,** 2004.

## Calendar

**IEEE International Conference on Electronics, Circuits, and Systems—ICECS**
13-15 December 2004
Tel-Aviv, Israel
http://www.ee.bgu.ac.il/~icecs04/

**Neural Information Processing Systems—NIPS**
13-18 December 2004
Vancouver, Canada
http://www.nips.cc/

**IEEE International Symposium on Circuits and Systems—ISCAS**
23-26 May 2005
Kobe, Japan
http://www.iscas05.org/

**International Joint Conference on Neural Networks—IJCNN**
31 July – 4 August 2005
Montreal, Canada
http://faculty.uwb.edu/ijcnn05/

***Kapre,*** *Continued from p. 9*

a reusable image-processing core for other saliency-based applications that might benefit from accelerated saliency computation. The long-term aim is to put a design flow in place that reuses off-the-shelf components, with saliency being the first of a series of basic neuromorphic image-processing operators that can be reused. However, far more needs to be done to tackle the issues of design complexity and development time before widespread use of programmable logic becomes a reality in this potentially-rich application area.

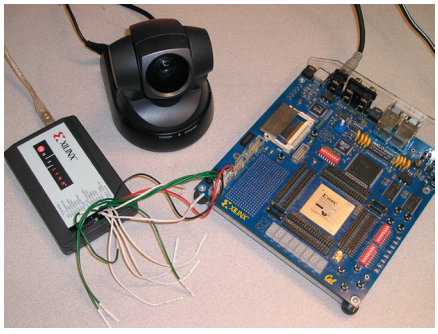

*Figure 2: Photo of the Berkeley development board.*

**Nachiket Kapre, Dirk Walther, Christof Koch, and André DeHon***
California Institute of Technology
E-mail: {nachiket, walther, koch}@caltech.edu
*E-mail: andre@acm.org

**References**
1. G. Indiveri, *A Neuromorphic VLSI device for implementing 2D selective attention systems,* **IEEE Trans. on Neural Networks,** November, 2001.
2. L. Itti, C. Koch and E. Niebur, *A model of saliency-based visual attention for rapid scene analysis,* **IEEE Trans. on Pattern Analysis and Machine Intelligence,** 1998.
3. L. Itti, *The iLab Neuromorphic Vision C++ Toolkit: Free tools for the next generation of vision algorithms,* **The Neuromorphic Engineer,** 2004.
4. André DeHon. *The Density Advantage of Configurable Computing,* **IEEE Computer**, April 2000.
5. CALINX Development Board, http://calinx.eecs.berkeley.edu.

*If have a toolkit or other goodies to share, contact the Editor at* ***sunny@sunnybains.com***