# Neuromorphic Visual Saliency Implementation Using Stochastic Computation

Chetan Singh Thakur[1], Jamal Lottier Molin[1], Tao Xiong[1], Jie Zhang[3], Ernst Niebur[2], Ralph Etienne-Cummings[1]
[1]Dept. of Electrical and Computer Engineering, [2]Dept. of Neuroscience, Johns Hopkins University, Baltimore, MD, USA
[3]Dept. of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, MA, USA
cthakur2@jhu.edu

*Abstract*—**Visual saliency models are difficult to implement in hardware for real time applications due to their computational complexity. The conventional digital implementation is not optimal because of the requirement of a large number of convolution operations for filtering on several feature channels across multiple image pyramids** [1], [2]. **Here, we propose an alternative approach to implement a neuromorphic visual saliency algorithm** [3] **in digital hardware using stochastic computation, which can achieve very low power and small area. We show the real time implementation of important building blocks of the system and compare the overall system with its software implementation. Our implementation will be useful for facilitating high-fidelity selective rendering in computer graphics applications using the output of the saliency model, and for communications, where the non-salient parts of an image can be compressed more heavily than the salient parts. Our implementation will find several applications as a frontend co-processor for information triaging, compression and analysis in computer vision tasks. Our proposed SC-based convolution circuit could be a potential building block for implanting deep convolutional neural networks (CNN) on hardware.**

## I. Introduction

Biological nervous systems have evolved over millions of years to efficiently perform tasks such as locomotion, visual processing and audio processing, which traditional engineering approaches struggle to solve. The field of neuromorphic engineering aims to emulate nature to build systems that match the performance of biological systems in such challenging tasks [4][5]. An overwhelming amount of sensory information is transmitted from the retina to the brain – estimated to be up to 100Mbps at the optic nerve [6]. This is an order of magnitude higher than the data rate of blu-ray; however organisms are able to efficiently process this information to parse complex scenes in real time. This, and much other evidence, suggests that the human visual system processes only parts of an image in detail, with only limited processing of areas outside of the focus of attention.

Visual Saliency is the distinct subjective perceptual quality which makes some objects of the environment stand out from their neighbours and immediately grab the observer's attention. We have previously developed a novel biologically plausible model of object based visual saliency [3]. This model mimics human search performance for images featuring pop out and predicts human eye fixations significantly better than chance. The model uses border ownership cells, found in monkey cortex [7], to provide the perceptual organization of a scene. The global contour information of figures is integrated into tentative proto-objects using grouping cells whose function is based on Gestalt principles [8]. Image saliency is computed from grouping cell activity. The model works as follows: (i) An input image is decomposed into various feature channels such as color, intensity, and orientation using appropriate filters. (ii) A center surround mechanism and normalization operator in each channel awards 'high activity' to unique, conspicuous features and 'low activity' to common features. (iii) Modality specific activation differences are removed by normalizing the results of each channel, which are then linearly summed to form the saliency map as shown in Fig. 1.

Typically, the implementation of the visual saliency pathway is computationally very demanding. The conventional digital implementation requires a large number of convolution operations for filtering on several feature channels across multiple image pyramids [1], [2]. The visual saliency pathway contains 9 feature channels (for different opponencies such as color and intensity, orientation, texture, motion), each feature channel downsamples an input image into 10 different images, and each image is further processed with 4 different orientations of edge detector filters, requiring large number of convolution operations. For these reasons, the model is difficult to be implemented as a software solution for use in real time low power applications. The complexity of such models calls for unconventional computation methods that can scale to a nano-integrated circuits regime that provide high-speed and compact implementation.

In order to solve the implementation problem, we propose to use a stochastic computation (SC) framework that uses simple and small circuits to build massively parallel circuits. The SC method uses bit-streams at minimal hardware cost [9]. Being small, they have very low power requirements, allowing them to be integrated at the edge of sensory systems or as part of a fog computation network. For example, the multiplication of two N-bit stochastic bit streams $x1$ and $x2$ to form the arithmetic product $x1 \times x2$ can be done in N-clock cycles using a single AND gate. Besides low area and power, SC has the advantage of high error tolerance (bit flips have little impact on signal probability) and support for massive, low-level parallelism. The SC theory has been successfully employed in various applications such as implementation of neural networks, image processing tasks [10] and even modelling probabilistic biological networks [11]. In this paper, we propose an SC-based framework for the hardware implementation of our proto-object based saliency model.

This work has been supported by the NIH grant R01EY027544

978-1-4673-6853-7/17/$31.00 ©2017 IEEE

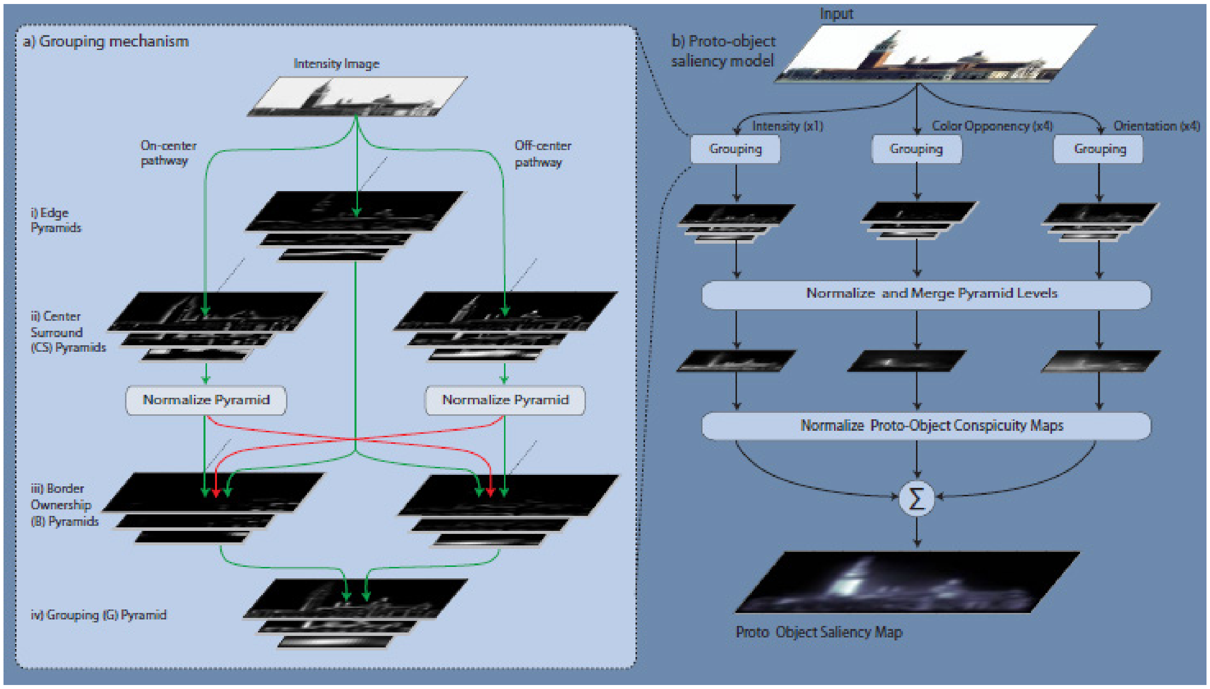

**Fig. 1: Proto-object saliency model** [3]**.** An input scene is decomposed into various feature types (e.g. intensity and color opponency). Within each feature map, tentative proto-objects are formed through the grouping mechanism [3]. The grouping mechanism (*left panel*) uses edge cells to extract local contour information. The local edge information is then combined with the global estimate of object location to calculate the response of border-ownership selective (*B*) cells. The output of the *B* cells is then combined using the Gestalt principles of continuity and proximity to activate grouping (*G*) cells. The activity of the *G* cells provides an estimate of the location of proto-objects within the visual field. Scale invariance is achieved by incorporating *G* cells with varying receptive field size into the model.

## II. SC IMPLEMENTATION OF BUILDING BLOCKS

In our visual saliency model, important blocks are the edge detection filters, center surround (CS) filters, border ownership and grouping filters, and normalization operations (Fig. 2). In total, the model has 25 different types of filters including 4 orientations, which will be applied to 9 feature channels each with a pyramid of 10 images. We have used an FSM-based technique to implement the normalization operation presented in [12]. In the original model, the receptive field of the filters is large, for example, the center surround and grouping filters use matrix sizes of 15×17 and 13×13. We have optimized receptive fields of all type of filters and changed them to a fixed size of 7×7 elements. Some examples are shown in Fig. 2. This enables us to use the same circuit architecture for all filters and eliminates the necessity of long stochastic spike sequences for accurate computation. In the filtering operation, filter coefficients are convolved with an image (i.e. 2-D convolution), which requires large amount of matrix multiplication and summation operations. This computationally expensive operation inhibits the real time implementation of many image processing tasks. In Fig. 3, we show how the convolution operation can be achieved using SC. In SC, all the variables are represented using spike train, thus we need to convert the image pixels using a pre-processing circuit, as shown in Fig. 3A.

Alaghi et al. have shown how the correlation can be exploited in the SC design [13]. We have utilized this important principle in our SC design to make it area efficient. This leads us to use a single Random Number Generator (RNG) for generating spike trains for each pixel in the image (Fig. 3A). An RNG can be easily implemented using a Linear Feedback Shift Register [14] on an FPGA. We have used the

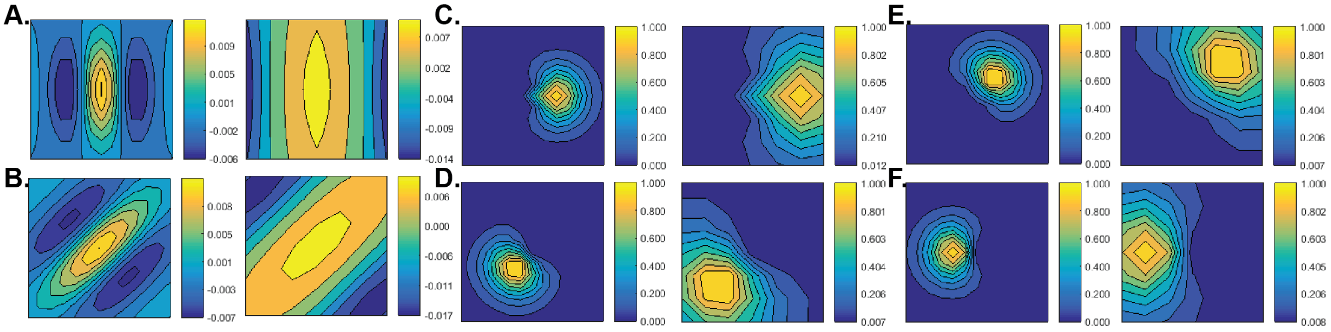

**Fig. 2: Examples of original filters (*left* panels) and their modified SC counterpart (*right* panels) in the model.** (A, B-*left*) original CS filter with 90˚, 45˚ orientation, (A, B-*right*) modified CS filter with 90˚, 45˚ orientation. Original group1 (C,E-*left*) & group2 (D, F-*left*) filters with 45˚ and 0˚ orientation. Modified group1 (C, E-*right*) & group2 (D, F-*right*) filters with 45˚ and 0˚ orientation.

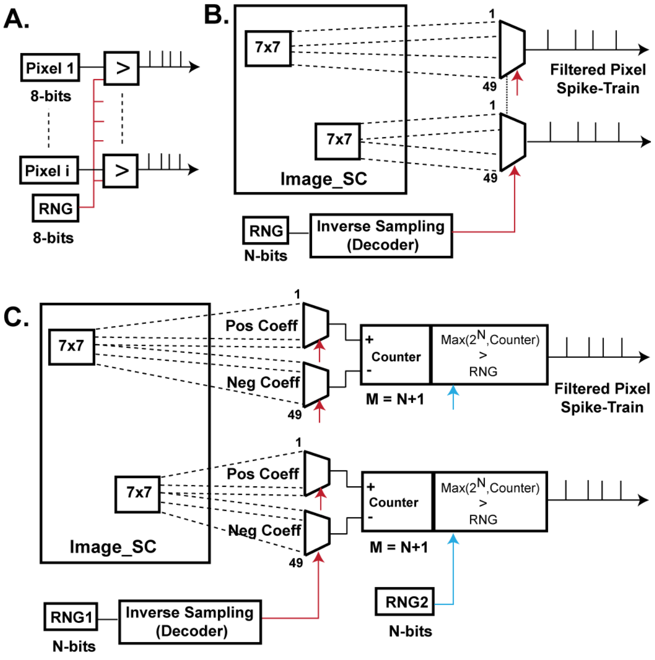

**Fig. 3: SC Implementation of Convolution Operations.** (**A**) Schematic showing conversion of pixels into spike train using only one RNG across all the pixels. Implementation of 2D convolution using SC with a single RNG with: (**B**) all positive coefficients, and (**C**) both positive and negative coefficients.

same correlation principle to implement convolution operations (Fig. 3B, C), where we have used a single RNG for all convolution operations in the image, thus effectively using a single multiplexer (mux) operation per pixel. We use the inverse transform sampling approach [15] to generate the select line for the mux. The sum of all coefficients of the filters is normalized to one. We create a cumulative sum of the filter coefficients and at each time, the RNG generates a random number that is checked for where it falls in the cumulative sum, and accordingly a select line of the multiplexer is activated. We need only one decoder to implement the inverse transform sampling for all the pixels. There are two types of filters – one with all its coefficients positive, and a second with some (generally half) coefficients negative. For the latter case, we need a separate mux logic corresponding to negative coefficients, and a counter that counts up if spikes from the positive mux are received, and counts down otherwise.

As the pixel value cannot be negative, this type of filter also needs a rectification operation, which can be easily accommodated in the counter itself by resetting the counter value to half of the maximum (Fig. 3C). We have implemented the edge detection filters using Robert's operator [13] using SC, which requires only an XOR gate, but has a very small receptive field. Our final saliency map was not significantly affected by the small receptive field of this filter (Fig. 4), because of multiple scale pyramid of the image (Fig. 1) and a low level operation. However, if needed, an edge detection filter with a larger receptive field of 7×7 can be implemented using the circuit shown in Fig. 3.

## III. RESULTS

First, we approximated all the filters in the visual saliency model with lower receptive field and implemented them using SC. We achieved good approximations for all filters, and Fig. 4 shows the results for two of the filters (group2, orientation 90˚ and 0˚) on an image (Fig. 4A). For SC implementation, the image was first converted into SC spike trains using the circuit described in Fig. 3A that uses the same RNG across all the pixels, and then the convolutional circuit (Fig. 3B) was applied. The resulting image is shown in Fig. 4D and 4G (orientation 90˚ and 0˚ resp) with mean square error of 0.65% and 0.72% respectively. Fig. 4(B and E) and 4(C and F) show images obtained by the MATLAB convolutional operation using the original receptive field size and a modified receptive field, respectively. As shown in Fig. 4, the performance of the visual saliency model is not degraded while porting the filter operation on the SC hardware.

We further extended our SC implementation by implementing all the filters using the circuits shown in Fig. 3B and 3C. Fig. 5 shows the results of the implementation of the visual saliency model on a set of images (Fig. 5A) using software simulations using: the traditional approach (Fig. 5B), and our new approach where filters are replaced by SC (Fig. 5C). Table 1 compares the qualitative performance of the software and SC-based implementations for the complete dataset. We have used the ImgSal v1.0 dataset with 60 images dedicated as "small salient regions" category [16]. The goal here is to show that the original model's ability to predict human eye fixation is not compromised by using SC-implemented modified filters. We used the Kullback-Leibler (KL) divergence method to validate the saliency model against predicting human eye fixations as done in [3]. KL divergence is used for computing how much one distribution of probabilities diverges from another distribution. Table 1 compares the qualitative performance of the software and SC-based implementations.

**Table 1**

| KLD of original model | KLD of modified model using SC | KLD b/w original and modified |
|---|---|---|
| 1.4309 | 1.2366 | 0.1724 |

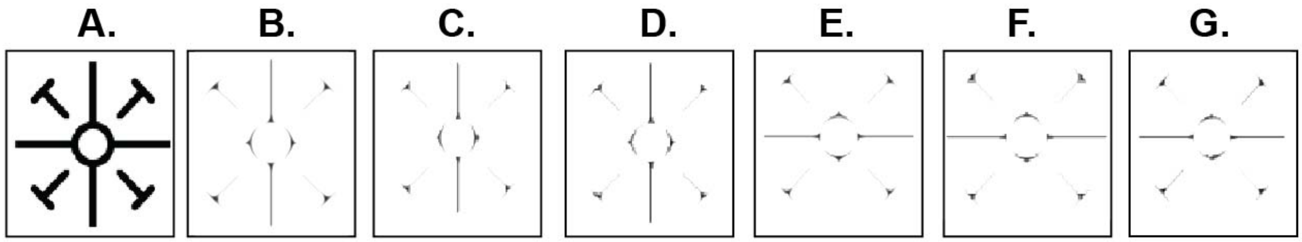

**Fig. 4: Comparison of a filter operation on an image.** (**A**) Original Image. Software implementation of group2 filter with orientation (**B**) 90˚, and (**E**) 0˚ using a receptive field of 13×13. Software implementation of group2 filter with orientation (**C**) 90˚, and (**F**) 0˚ in using modified receptive field of 7×7. SC hardware implementation of group2 filter with orientation (**D**) 90˚, and (**G**) 0˚ with a modified receptive field using the circuit shown in Fig. 3.

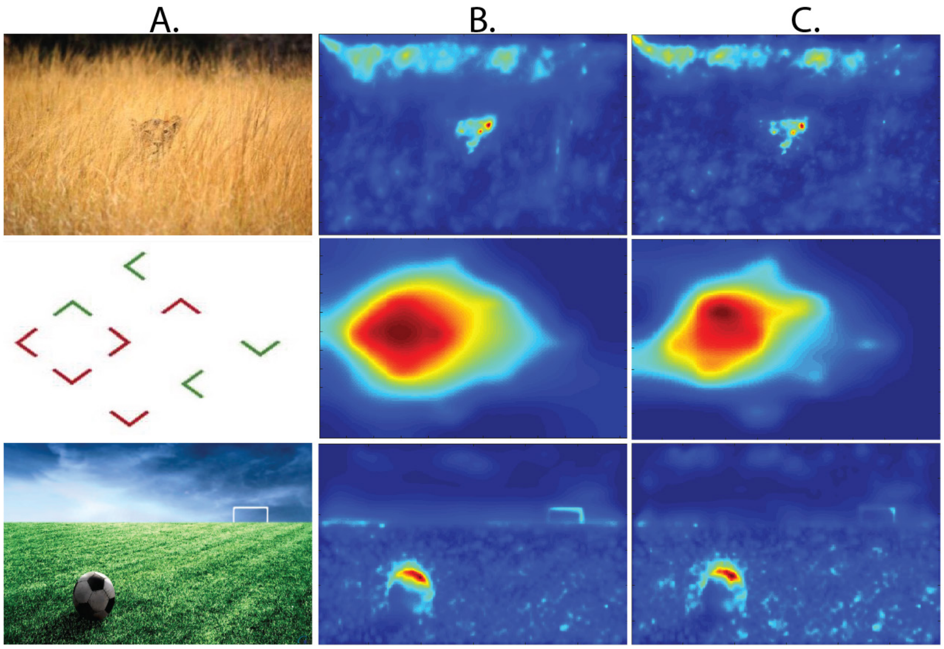

**Fig. 5. : Comparison of the visual saliency algorithm. (A)** Original images, **(B)** Implementation using software simulations with the original model [3], and **(C)** Implementation by replacing all filters with the SC framework**.**

## IV. Conclusions

In this paper, we have presented a novel architecture to implement computationally complex visual saliency models using SC. Our works shows the SC hardware implementation of convolution filters using SC with simple logic gates. Most importantly, we have used a single RNG for the entire image, based on the correlation analysis presented by Alaghi et al. [13]. Our proposed parallel architecture implementation using SC will enable implementation of deep convolutional neural networks (CNN) on hardware, which are difficult to build using traditional architecture due to their enormous complexity. The circuit shown in Fig. 3C which contains a counter implementing the ReLU (rectified linear unit) nonlinearity may serve as the building block of the CNN. Future work will involve complete integration of the system with all the major building blocks shown in the paper. Our real-time implementation will find various applications where automatic triaging, thumb nailing, highlighting effects, enhanced rendering, auto-cropping and scene labelling is needed. The approach and resultant "information", i.e. *not* "data", will significantly reduce the computational processing in cameras and virtual reality headsets by focussing only on the salient features of an image, saving considerable computation power in mobile computation devices.

## References

[1] B. Kaushik, R. Saini, A. Saini, S. Singh, and A. S. Mandal, "An FPGA implementation of image signature based visual-saliency detection," *18th International Symposium on VLSI Design and Test, VDAT 2014*, pp. 1–6, 2014.

[2] F. Barranco, J. Diaz, B. Pino, and E. Ros, "Real-Time Visual Saliency Architecture for FPGA With Top-Down Attention Modulation," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 3, pp. 1726–1735, 2014.

[3] A. F. Russell, S. Mihalaş, R. von der Heydt, E. Niebur, and R. Etienne-Cummings, "A model of proto-object based saliency," *Vision Research*, vol. 94, pp. 1–15, 2014.

[4] C. S. Thakur, T. J. Hamilton, J. Tapson, A. van Schaik, and R. F. Lyon, "FPGA implementation of the CAR Model of the cochlea," in *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2014, pp. 1853–1856.

[5] C. S. Thakur, T. J. Hamilton, R. Wang, J. Tapson, and A. van Schaik, "A neuromorphic hardware framework based on population coding," in *2015 International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1–8.

[6] K. Koch, J. McLean, M. Berry, P. Sterling, V. Balasubramanian, and M. Freed, "Efficiency of Information Transmission by Retinal Ganglion Cells," *Current Biology*, vol. 14, pp. 1523–1530, 2004.

[7] H. Zhou, H. S. Friedman, and R. Von Der Heydt, "Coding of border ownership in monkey visual cortex.," *The Journal of neuroscience : the official journal of the Society for Neuroscience*, vol. 20, no. 17, pp. 6594–6611, 2000.

[8] K. Koffka, "Principles of Gestalt psychology," *Principles of gestalt psychology*, pp. 1–14, 1935.

[9] B. R. Gaines, "Stochastic Computing Systems," *Advances in information systems science. Springer US*, 1969.

[10] A. Alaghi, C. Li, and J. P. Hayes, "Stochastic circuits for real-time image-processing applications," in *Proceedings of the 50th Annual Design Automation Conference on - DAC '13*, 2013, p. 1.

[11] C. S. Thakur, S. Afshar, R. M. Wang, T. J. Hamilton, J. Tapson, and A. van Schaik, "Bayesian Estimation and Inference using Stochastic Hardware," *Frontiers in Neuroscience*, vol. 10, no. March, pp. 1–28, 2015.

[12] P. Li, D. J. Lilja, W. Qian, K. Bazargan, and M. D. Riedel, "Computation on stochastic bit streams digital image processing case studies," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 3, pp. 449–462, 2014.

[13] A. Alaghi and J. P. Hayes, "Exploiting correlation in stochastic circuit design," *2013 IEEE 31st International Conference on Computer Design, ICCD 2013*, no. c, pp. 39–46, 2013.

[14] P. Alfke, "Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators," *TechNotes*, vol. 1996, pp. 1–6, 1996.

[15] L. Devroye, "Non-Uniform Random Variate Generation," 1986.

[16] J. Li, M. D. Levine, X. An, X. Xu, and H. He, "Visual saliency based on scale-space analysis in the frequency domain," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 4, pp. 996–1010, 2013.