Available online at www.sciencedirect.com
SCIENCE DIRECT®
Real-Time Imaging 9 (2003) 189–196
www.elsevier.com/locate/rti

Real-time visual attention on a massively parallel SIMD architecture

Nabil Ouerhani*, Heinz Hügli

Institute of Microtechnology, University of Neuchâtel, Rue A.-L. Breguet 2, CH-2000 Neuchâtel, Switzerland

Abstract

Visual attention is the ability to rapidly detect the visually salient parts of a given scene on which higher level vision tasks, such as object recognition, can focus. Found in biological vision, this mechanism represents a fundamental tool for computer vision. This paper reports the first real-time implementation of the complete visual attention mechanism on a compact and low-power architecture. Specifically, the saliency-based model of visual attention was implemented on a highly parallel single instruction, multiple data architecture called ProtoEye. Conceived for general purpose low-level image processing, ProtoEye consists of a 2D array of mixed analog–digital processing elements. To reach real-time, the operations required for visual attention computation were optimally distributed on the analog and digital parts. The currently available prototype runs at a frequency of 14 images/s and operates on 64×64 gray level images. Extensive testing and run-time analysis of the system stress the strengths of the architecture.

1. Introduction

Visual attention is the ability to rapidly detect visually salient parts of a given scene. This mechanism is useful in computer vision because it permits a rapid selection of a subset of the available sensory information. Once detected, the salient parts are the specific image locations on which higher level computer vision tasks can focus.

The computational modeling of visual attention has been a key issue in artificial vision during the last two decades [1–4]. First reported in 1985 [5], the saliency-based model of visual attention is largely accepted today [6]. The model has been used in various computer vision applications and gave rise to numerous software implementations in recent works [7–10].

Due to its complexity, the reported model of visual attention needs, for a real-time operation, higher computation resources than available in conventional processors. Until now, it was not achieved on a compact system. Some previous works reported hardware models of visual attention implemented on fully analog very large-scale integration chips [11,12]. The authors considered, however, simplified versions of the saliency-based model of visual attention and implemented only small parts of it. In both works, emphasis has been put only on the last stage of the attention model, namely, the winner-take-all (WTA) network.

A complete real-time implementation of the saliency-based model of visual attention has been reported recently in [13]. However, the implementation has been carried out on a large system consisting of a 16-CPU cluster, named Beowulf. Involving 10 interconnected personal computers, the system might raise problems related to portability, power consumption and price.

This paper reports the first real-time implementation of the complete saliency-based model of visual attention on a compact system, consisting of a low-power, one-board, highly parallel single instruction, multiple data (SIMD) architecture, called ProtoEye (Fig. 1) [14]. ProtoEye is a general purpose image processing board which consists of a 64×64 array of mixed analog–digital processing elements (PEs). There is a PE for each single pixel of an image provided by a CMOS camera. The complete architecture is controlled by a general purpose microcontroller running at a frequency of 4 MHz, yielding an effective performance of over 8 Giga operations per second.

As ProtoEye was not specifically tailored for our application, the operations required for visual attention computation had to be optimally distributed on the analog and digital parts of ProtoEye. The digital part

*Corresponding author.
E-mail addresses: nabil.ouerhani@unine.ch (N. Ouerhani), heinz.hugli@unine.ch (H. Hügli).

1077-2014/03/$ - see front matter © 2003 Elsevier Ltd. All rights reserved.
doi:10.1016/S1077-2014(03)00036-6

contains, among other components, a 4-bit ALU and six registers. Despite reduced resources, it offers a high programming flexibility. This part of ProtoEye performs the logical and arithmetic image operations appearing in the attention model. The analog part consists mainly of an analog diffusion network. It is used to efficiently perform image filtering, which represents the most time-consuming task in the model of visual attention.

The remainder of this paper is organized as follows. Section 2 presents the saliency-based model of visual attention. The architecture of ProtoEye is reported in Section 3. Section 4 describes the implementation of the visual attention model on ProtoEye and reports the performance analysis of the system. Finally, the conclusions are stated in Section 5.

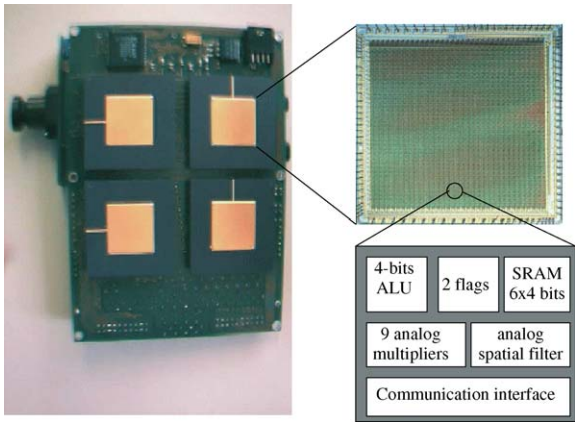


Fig. 1. ProtoEye platform: This vision platform is composed essentially of four ProtoEye chips, a CMOS camera, a microcontroller and a video output.

2. Saliency-based model of visual attention

The considered model of visual attention transforms the input image into a map expressing the intensity of salience at each location: the saliency map. The original version of the saliency-based model of visual attention presented in [9] deals with static color images, but a general model that operates on n image features can be considered. This model is composed of four main steps (see Fig. 2(a)):

(1) First, a number (n) of features are extracted from the scene by computing the so-called feature maps (color, intensity, orientations, ...).
(2) In a second step, each feature map is transformed in a conspicuity map that highlights the parts of the image that strongly differ, according to a specific feature, from their surroundings. This is performed by a multiscale center-surround mechanism. Multiscale *difference-of-Gaussians* ($\mathscr{D}o\mathscr{G}$) filters, which

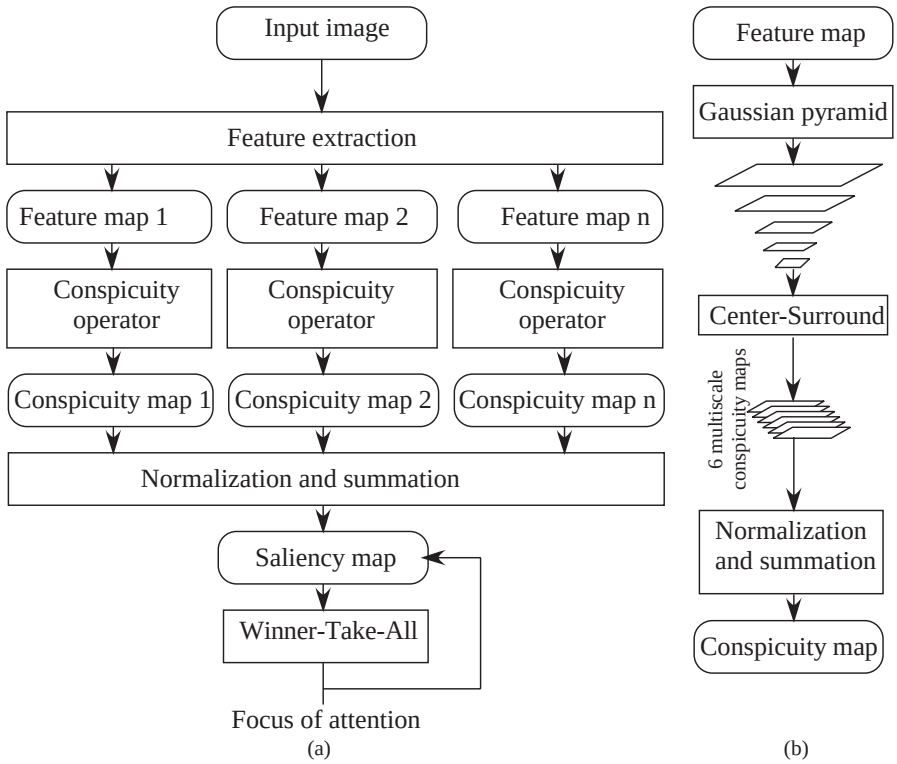


Fig. 2. Saliency-based model of visual attention. (a) represents the four main steps of the visual attention model. Feature extraction, conspicuity computation (for each feature), saliency-map computation by integrating all conspicuity maps and finally the detection of spots of attention by means of a WTA network. (b) illustrates, with more details, the conspicuity operator, which computes six multiscale intermediate conspicuity maps. Then, it normalizes and integrates them into the final conspicuity map.

can be implemented using Gaussian pyramids, are suitable means to implement the conspicuity operator. The multiscale concept allows the conspicuity operator to detect objects of different sizes. Typically, the combination of six conspicuity maps, computed at different scales, allows the detection of objects of "reasonable" sizes in natural scenes [9] (see Fig. 2(b)).

(3) In the third stage of the attention model, conspicuity maps are integrated into a *saliency map*, which expresses the intensity of salience at each location. This is performed by a normalization and weighted summation process.

(4) Finally, the most visually salient locations are detected by applying, iteratively, a WTA network on the saliency map.

3. ProtoEye: SIMD machine for image processing

The complete vision system is conceived for general purpose image processing and is composed of a CMOS imager (352×288 pixels), a video output, a general purpose microcontroller and four ProtoEye chips (Fig. 1). A 64×64 pixel subimage is grabbed from the image provided by the camera, and transferred to the ProtoEye architecture by means of a DMA interface. The same DMA interface is used to transfer the final results from ProtoEye to the external memory, which is interfaced to the video output. It is noteworthy that the data transfer between the DMA and the ProtoEye (and vice versa) is achieved in a serial manner.

The ProtoEye architecture is a SIMD machine composed of a microcontroller and a 64×64 array of identical PEs. The microcontroller consists of a 4 MHz clocked RISC processor implemented on an FPGA.

A PE, which is associate to a single pixel and is connected to its eight neighbors, is composed of a digital part and an anlog one (Fig. 3). The digital part of a PE is organized around an internal 4-bit D-bus (D[3:0]). It contains a 4-bit ALU, which has as input the D-bus and the accumulator. The ALU set of operations includes all logical functions, addition, subtraction, shifts of the accumulator content and comparison. The flag F1 masks conditional operations. The six registers are used to keep temporary results within the PE. In digital mode, transfers between neighboring PEs can be performed by shifting the accumulator content.

The analog part of each PE (shaded area in Fig. 3) is connected to the digital part through A/D and D/A converters. Its essential component is the analog spatial filter, which is based on a diffusion network, made of pseudoconductances connecting the PEs [15]. The input of the spatial filter is the content of the register RAM5, converted by the D/A converter. Its output is a low-pass filtered version of the input image, which cut-off frequency is programmable.

Note that the 64×64 array of PEs is implemented on four chips.

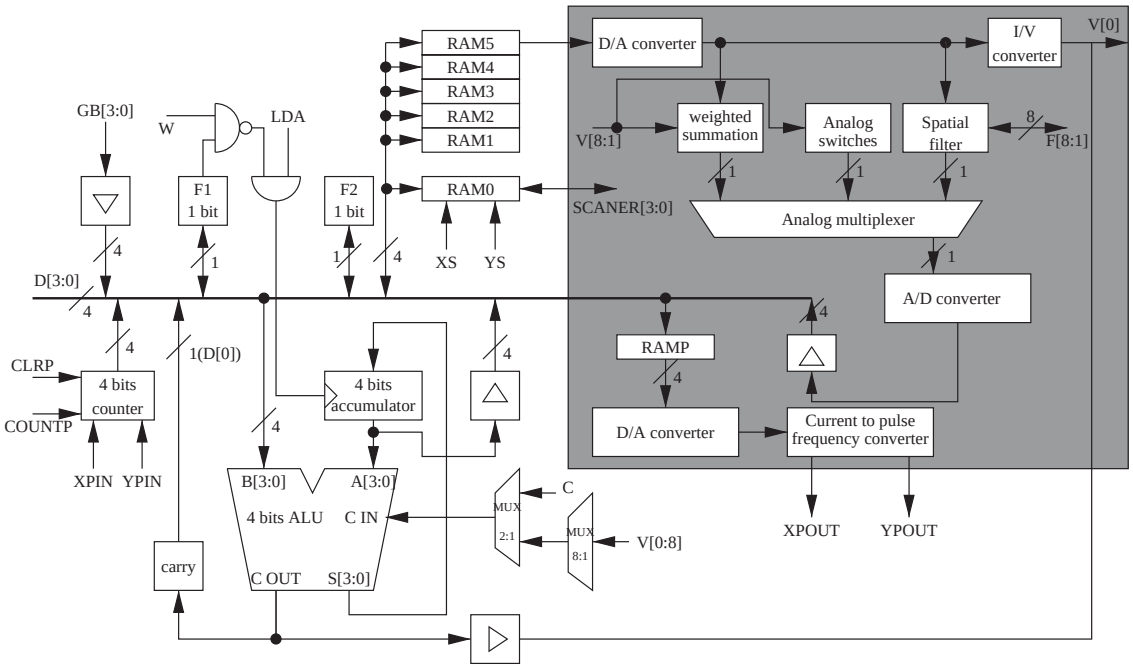


Fig. 3. ProtoEye: Architecture of a single PE. A PE is composed of a digital part (non-shaded area) and an analog part (shaded area). The digital part consists of a 4-bit ALU, an accumulator, six registers (RAM0 ... RAM5) and two 1-bit flags (F1 and F2). The main component of the analog part is the analog spatial filter which is based on a diffusion network.

4. Implementation issues and performance analysis

This section reports the implementation of the described model of visual attention on the reported SIMD architecture ProtoEye. In addition, the performance of the system is deeply analyzed. Finally, we give some perspectives of our work.

4.1. Implementation issues

We report a version of our implementation of visual attention on ProtoEye that considers a single image feature, which is intensity. From this feature we compute six multiscale conspicuity maps, which are then combined into the final conspicuity map. Since we consider a single image feature, the final conspicuity map corresponds directly to the saliency map. One or more spots of attention are then derived from the saliency map. In the following, we report the main implementation issues related to the four steps of the visual attention model:

(1) *Feature extraction*: Since we consider a single image feature—intensity—this step is straightforward.

(2) *Conspicuity operator*: The computation of the conspicuity map is performed in two steps: The computation of the six multiscale conspicuity maps, using the center-surround transformation, and their integration into the final conspicuity map. We will describe each step separately.

(a) *Center-surround transformation*: We implement this transformation by means of a multiscale difference-of-exponential ($\mathscr{D}o\mathscr{E}xp$) filter. Practically, a nine-level exponential pyramid $\mathscr{P}$ is built by progressively low-pass filtering the intensity image I using an exponential filter H. Formally, this pyramid is defined according to the following equations:

$$\begin{aligned} &\mathscr{P}(0) = I, \\ &\mathscr{P}(i+1) = \mathscr{P}(i) ** H, \end{aligned} \tag{1}$$

where $(**)$ refers to the spatial 2D convolution operator.

Six multiscale conspicuity maps are then computed from the exponential pyramid as absolute differences between fine scales (center) and coarse scales (surround), according to the following set of equations:

$$\begin{aligned} &C2\_5 = |\mathscr{P}(2) - \mathscr{P}(5)|, \quad C2\_6 = |\mathscr{P}(2) - \mathscr{P}(6)|, \\ &C3\_6 = |\mathscr{P}(3) - \mathscr{P}(6)|, \quad C3\_7 = |\mathscr{P}(3) - \mathscr{P}(7)|, \\ &C4\_7 = |\mathscr{P}(4) - \mathscr{P}(7)|, \quad C4\_8 = |\mathscr{P}(4) - \mathscr{P}(8)|. \end{aligned} \tag{2}$$

The $\mathscr{D}o\mathscr{E}xp$ filter is more suitable for ProtoEye than the $\mathscr{D}o\mathscr{G}$ one used in the original model. The analog diffusion network, which simulates exponential filtering, efficiently computes the exponential pyramid in a negligible time. Furthermore, this modification does not affect the results of the conspicuity transformation, compared with the original model, since the effect of both filters ($\mathscr{D}o\mathscr{E}xp$ and $\mathscr{D}o\mathscr{G}$) is very similar [16].

The number (six) and the size (4 bits) of the internal registers represents an additional constraint to deal with in our implementation. Six registers are not enough to store the nine levels of the exponential pyramid. To overcome this constraint, we keep only two levels of the pyramid, at once, in the registers: a center level (levels 2, 3 and 4 successively) in RAM2 and a surround level (levels 5, 6, 7 and 8 successively) in RAM3. Once a conspicuity map is computed, the corresponding pyramid levels are replaced by the two levels needed to compute the next conspicuity map (see Fig. 4).

Note that the first center level ($\mathscr{P}(2)$) and the first surround one ($\mathscr{P}(5)$) of the exponential pyramid are computed directly using two exponential filters with different diffusion lengths H_2 and H_5, where $H_1 = H$ is the original exponential filter and $H_{i+1} = H_i ** H$ (see Fig. 4).

Due to the same constraint, each computed conspicuity map is transferred to external memory (*To memory( )* on Fig. 4), after the normalization of its gray-level values to the range 0–15 (*Norm( )* on Fig. 4).

(b) *Computation of the final conspicuity map*: The final conspicuity map is the mean of the six normalized multiscale conspicuity maps. Practically, the normalized maps stored in the external memory are restored into two internal registers (RAM4 and RAM5) for the final merge which mainly consists of a double-precision summation and a division by 8.

(3) *Computation of the saliency map*: Since we consider a single image feature—intensity—the final conspicuity map related to intensity corresponds directly to the saliency map.

(4) *Detection of the spots of attention*: In order to detect the most salient locations of the image, a maximum network, based on a large ($\mathscr{D}o\mathscr{E}xp$) filter, is applied to the saliency map. This again is performed by the analog diffusion network. One or several spots are detected, depending on the diffusion length.

So far, the final result of the developed visual attention system is a binary image which displays the most salient image locations. This image is transferred to the external memory, which permits its use in further tasks.

Fig. 5 illustrates a saliency map and the spots of attention computed from a real gray-level image.

4.2. Performance analysis

To analyze the performance of the system, we measure the computation time of the complete process as well as the computation time of each step of the visual attention model. These measurements are represented in Table 1.

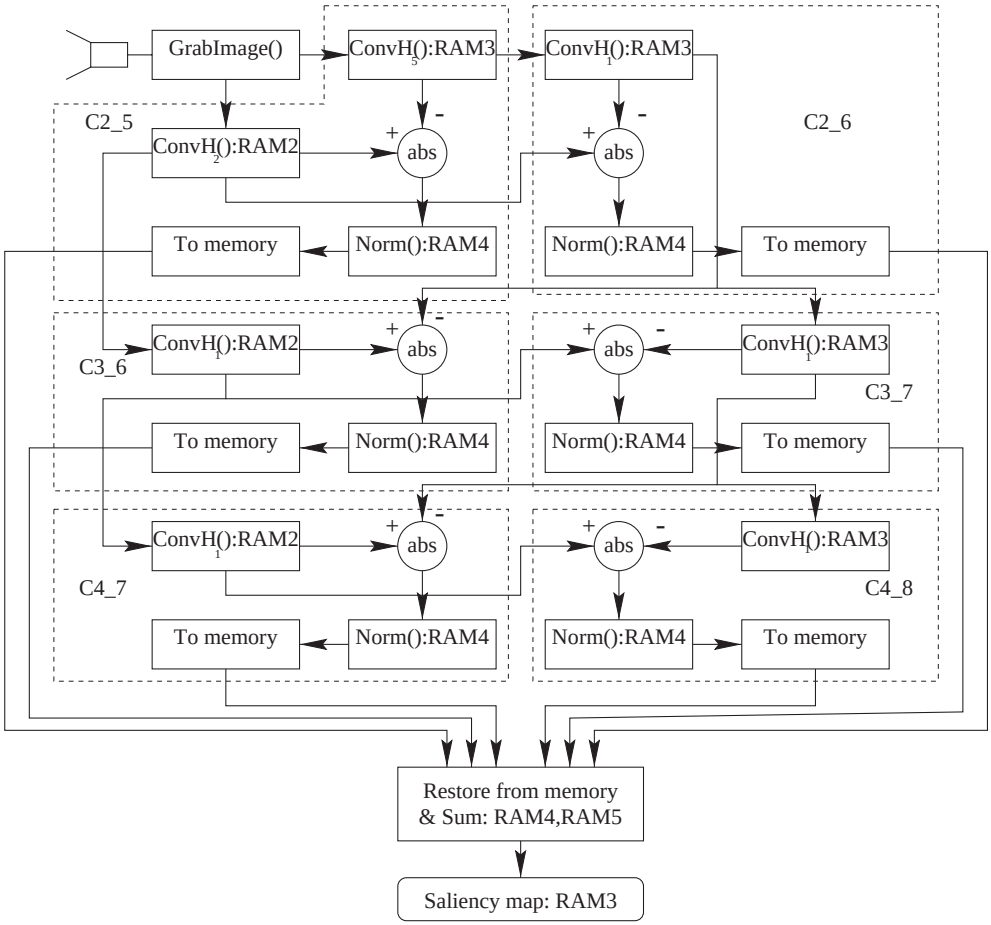
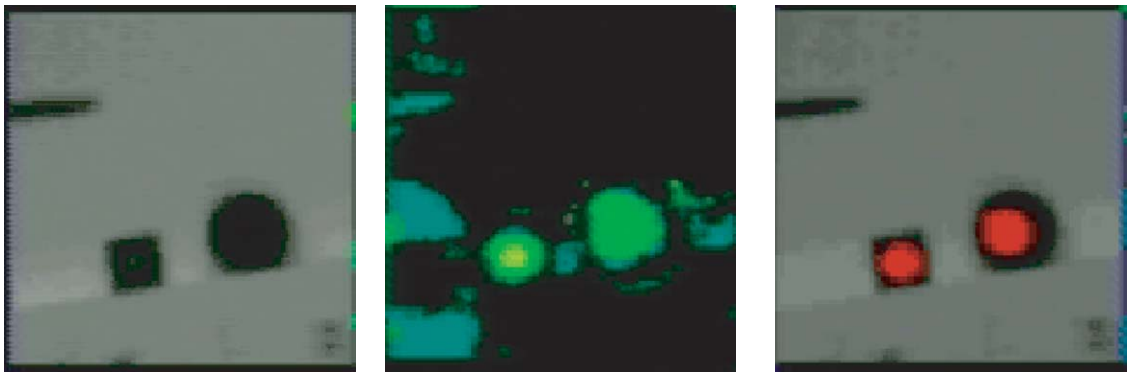


Fig. 4. Resources attribution to the different steps of the visual attention model. From a grabbed gray-level image, an exponential pyramid is built by progressively low-pass filtering the input image, using the function $ConvH_i()$, where $H_1 = H$ is an exponential filter and $H_{i+1} = H_i{*}{*}H$. The absolute difference between different levels of the pyramid gives the six multiscale conspicuity maps C2_5, C2_6, C3_6, C3_7, C4_7, and C4_8, which are progressively normalized and stored into the external memory. These maps are then restored to internal registers (RAM4 and RAM5) and a double-precision summation of all maps is computed. This sum, divided by 8, corresponds to the saliency map.



Original image Saliency map Spots of attention

Fig. 5. Detecting the most salient locations in a gray-level image. First, we compute a saliency map from the intensity feature. Then, we apply a maximum network on the saliency map, in order to detect the most salient locations or spots of attention. These spots are marked up with red stains.

The complete computation of a saliency map takes 71.126 ms, leading to a computation frequency of 14 images/s.

Let us consider now the individual contribution of each function to the complete computation time. According to Fig. 6, the image grab, which consists in

Table 1
Computation time of the complete process and of the single operations

Functions	Execution time of operations (ms)						Total (ms)
	Conv( )	*absDiff( )*,	*Sum( )*	*Norm( )*	*toMem( )*	*restore( )*	
GrabImage( )							30.75
C2_5	0.78		0.004	2.118	1.776	1.78	6.459
C2_6	0.695		0.004	2.118	1.776	1.78	6.373
C3_6	0.086		0.004	2.118	1.776	1.78	5.764
C3_7	1.391		0.004	2.118	1.776	1.78	7.069
C4_7	0.173		0.004	2.118	1.776	1.78	5.851
C4_8	1.4		0.004	2.118	1.776	1.78	7.069
Final map			0.015		1.776		1.791
All							71.126

Conv( ) is the smoothing function, which is used to compute the exponential pyramid $\mathscr{P}(i)$. *absDiff( )* and *Sum( )* are the two arithmetic operations needed to compute the six conspicuity maps and the final map, respectively. *Norm( )* is the normalization function. *toMem( )* and *restore( )* are responsible for the image transfers between the PEs and the external memory and vice versa.

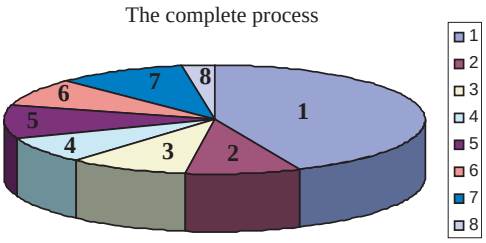


Fig. 6. Distribution of the computation time over the functions which constitute the complete process of visual attention. (1) *GrabImage( )*, (2) ... (7): C2_5 ... C4_8, and (8) final map.

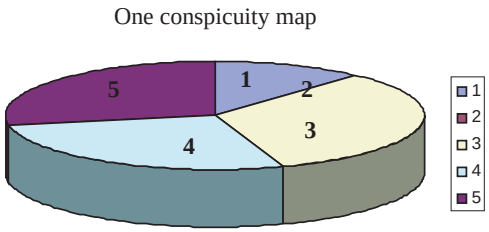


Fig. 7. Distribution of the computation time over the operations which are used to compute a single conspicuity map. (1) *Conv( )*, (2) *absDiff( )*, (3) *Norm( )*, (4) *toMemory( )*, and (5) *restore( )*.

acquiring the image by the imager and transferring the data to ProtoEye, constitutes over 43% of the entire computation time.

The computation of the final saliency map, which consists in a double-precision summation of the conspicuity maps and a division, represents less than 3% of the processing time. The time required for the computation of the six conspicuity maps exceeds 54% of the entire computation time.

Due to its large contribution to the computation time of the entire process, the computation time of the conspicuity maps is closely analyzed. Fig. 7 illustrates the distribution of the computation time of a single conspicuity map over the required operations. It is noteworthy that the operations, which require an image transfer between the processing elements and the external memory (*Norm( )*, *toMemory( )*, *restore( )*), hold the major part of the computation time because, as mentioned above, the data transfer from/to the PEs is achieved in a serial manner. *Norm( )* requires this transfer in order to compute the global maximum of the maps, which is used in their normalization. The two operations *toMemory( )* and *restore( )* require over 55% of the computation time of a conspicuity map. As mentioned above, these two functions allow the storage of individual conspicuity maps in external memory for lack of enough internal registers.

To summarize, the computation time of the entire process is distributed as follows:

- The image grab and its transfer to ProtoEye represents 43% of the cycle time.
- Of the remaining cycle time used for computation, the image transfer needed to store and restore temporary results holds 80% of the time.
- Of the same remaining cycle time used for computation, real processing which takes place in ProtoEye represents less than 20% of the time.

4.3. Perspectives

We discuss now how the presented architecture can be further improved to cope with higher frequencies, more scene features and larger images. Considering the above performance analysis, it appears that the image grab and transfer time makes more than 40% of the cycle. This part is composed of a large delay for image acquisition and a single transfer cycle. The first can be eliminated by image buffering and the second can be counted with the other many image transfers required for the conspicuity computation. Under this hypothesis, the processing performance reaches a frequency of about $f_0 = 20$ Hz.

We consider then the extension from 1 to n features and a possible speed-up of the architecture operating

frequency by a factor of x. The realistic hypothesis that the system scales according to a linear rule with x and inverse rule with n leads to the following formula for the new image frequency f:

$$f = f_0 \frac{x}{n}. \quad (3)$$

Given that the current circuits are running at a frequency of 4 MHz and that it is not unrealistic to consider an operation at 40 MHz, we count on a possible speed-up of $x = 10$ giving enough room for increasing n by addition of new features or/and increasing f by operating the system at a higher frequency.

Considering now the system extension towards larger images, it is useful to consider from above that each cycle consists approximately for 20% in a parallel processing part and for 80% in a part devoted to a data transfer which has serial character. The duration of the first part is thus independent of the image size while the second scales according to a quadratic rule with a 1D image scale s. It appears therefore that the system activity becomes increasingly dominated by data transfers when the image size is increased. A means to overcome this obstacle is the introduction of some fast and parallel data transfer procedure, like for example the simultaneous access to several image lines. Assuming a parallelism of degree P, the formula for the new cycle time T of an s scaled system with respect to the original cycle time $T_0 = 50$ ms is given by

$$T = T_0\left(0.2 + 0.8\frac{s^2}{P}\right). \quad (4)$$

For maintaining the original performance of the above-described system with a larger image of 256×256 ($s = 4$) for example, it appears that a fast data transfer is required and the formula tells that a degree of parallelism of $P = s^2 = 16$ is required at least.

It appears finally that the presented architecture can cope with higher frequencies and more scene features, mainly by boosting the speed of the processor. Coping with larger image sizes requires the rather simple replication of the elementary PEs, but in addition, a parallel data transfer mechanism must be developed.

5. Conclusion

This paper reports the first real-time implementation of the saliency-based model of visual attention on a compact system consisting of a low-power, one-board, highly parallel SIMD architecture. Conceived for general purpose low-level image processing, the fully programmable SIMD machine consists of an array of mixed digital–analog processing elements that offer high-performance functionalities for implementing the various functions appearing in the model of visual attention. The current prototype processes 64×64 images at a rate of 14 images/s, which allows the use of visual attention in real-time applications related to computer vision. Extensive performance analysis confirms the strengths of the architecture, but also shows its high performance potential. Future designs, where a 10 times increase in computing performance seems straightforward, will allow the integration of further image features into the attention model and faster image rates. Due to the parallel architecture, the extension to larger images is also possible at the cost of simple hardware replication, provided that the serial image transfer is upgraded to a parallel form.

Acknowledgements

This work was partially supported by the CSEM-IMT Common Research Program. The authors deeply appreciate the fruitful collaboration with Pierre-Yves Burgi and Pierre-François Ruedi regarding ProtoEye.

References

[1] Julesz B, Bergen J. Textons, the fundamental elements in preattentive vision and perception of textures. Bell System Technical Journal 1983;62(6):1619–45.
[2] Ahmed S. Visit: an efficient computational model of human visual attention. PhD Thesis, University of Illinois at Urbana-Champaign, 1991.
[3] Milanese R. Detecting salient regions in an image: from biological evidence to computer implementation. PhD Thesis, Dept. of Computer Science, University of Geneva, Switzerland, Dec. 1993.
[4] Tsotsos JK, Culhane SM, Wai WYK, Lai YH, Davis N, Nuflo F. Modelling visual attention via selective tuning. Artificial Intelligence 1995;78(1–2):507–45.
[5] Koch Ch, Ullman S. Shifts in selective visual attention: towards the underlying neural circuity. Human Neurobiology 1985;4:219–27.
[6] Itti L, Koch Ch. Computational modeling of visual attention. Nature Reviews Neuroscience 2001;2(3):94–203.
[7] Ouerhani N, Bracamonte J, Hugli H, Ansorge M, Pellandini F. Adaptive color image compression based on visual attention. In: Ardizzone E, Di Gesu V, editors. Proceedings of International Conference on Image Analysis and Processing (ICIAP 2001), Palermo, Italy, Sep. 2001. Silver Spring MD: IEEE Computer Society Press, 2002. p. 416–21.
[8] Ouerhani N, Archip N, Hugli H, Erard PJ. A color image segmentation method based on seeded region growing and visual attention. International Journal of Image Processing and Communication 2002;8(1):3–11.
[9] Itti L, Koch Ch, Neibur E. A model of saliency-based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence 1998;20(11):1254–9.
[10] Ouerhani N, Hugli H. Computing visual attention from scene depth. In: Sanfeliu A, Villanueva J.J., Vanrelli M, Alquezar R, Huang T, Serra J, editors. Proceedings of International Conference on Pattern Recognition (ICPR 2000), Barcelona, Spain, vol. 1, Silver Spring MD: IEEE Computer Society Press, 2000. p. 375–8.

[11] Brajovic V, Kanade T. Computational sensor for visual tracking with attention. IEEE Journal of Solid State Circuits 1998;33(8): 1199–207.

[12] Indiveri G. Modeling selective attention using a neuromorphic VLSI device. Neural Computation 2000;12(12):2857–80.

[13] Itti L. Real time high-performance attention focusing in outdoors color video streams. In: Rogowitz B, Pappas T.N, editors. Proceedings of SPIE Human Vision and Electronic Imaging IV San Jose, CA, 2002. p. 235–43.

[14] Ruedi P-F, Marchal PR, Arreguit X. A mixed digital–analog and simd chip tailored for image perception. Proceedings of International Conference on Image Processing 96, Lausanne, vol. 2, 1996. p. 1011–4.

[15] Vittoz EA, Arreguit X. Linear networks based on transistors. Electronic Letters 1993;29:297–9.

[16] Deriche R. Fast algorithms for low-level vision. IEEE Transactions on Pattern Analysis and Machine Intelligence 1990;PAMI-12(1):78–87.