International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:2, No:9, 2008

Pulsed Multi-Layered Image Filtering: A VLSI Implementation

Christian Mayr, Holger Eisenreich, Stephan Henker, and René Schüffny

Abstract**—**Image convolution similar to the receptive fields found in mammalian visual pathways has long been used in conventional image processing in the form of Gabor masks. However, no VLSI implementation of parallel, multi-layered pulsed processing has been brought forward which would emulate this property. We present a technical realization of such a pulsed image processing scheme. The discussed IC also serves as a general testbed for VLSI-based pulsed information processing, which is of interest especially with regard to the robustness of representing an analog signal in the phase or duration of a pulsed, quasi-digital signal, as well as the possibility of direct digital manipulation of such an analog signal. The network connectivity and processing properties are reconfigurable so as to allow adaptation to various processing tasks.

Keywords**—**Neural image processing, pulse computation application, pulsed Gabor convolution, VLSI pulse routing.

I. Introduction

THE resurgence in interest concerning pulse-coded neural nets has led to various attempts to integrate pulsed computation schemes in mixed signal VLSI. Most of these technical implementations of neural circuits aim to copy some of the image segmentation/analysis/decomposition properties exhibited by mammalian visual pathways. VLSI-based spiking neurons so far have mainly exploited the synchronization patterns of locally coupled spiking neurons using various synapse adaptation rules [1,2,3]. However, this type of network pattern is relegated to simple image analysis. More complicated neurons and synapses, as well as more complex connection matrices are needed to achieve high-level, non-trivial image processing functions [4,5,6]. In Section II.A, a neural microcircuit capable of extracting the uncorrelated pulses from two pulse streams [7] and its modification for a hardware implementation are briefly described. This microcircuit is based on several information processing schemes postulated from biological evidence. Also, a digital pulse distribution router has been developed, employing an Address-Event-Representation (AER) and router engine which can route pulse events to arbitrary locations on an IC consisting of 128*128 neural circuits, mimicking synapses (section II.B). Pulses can be routed to more than one location, so multiple connections of biological synapses can be emulated. The IC has been enabled for the 3D integration technology described in [8], so that pulse events can also be routed vertically across an IC stack. The data input to the system comes from pulsing pixel cells, described in section II.C, which use a simple and area efficient integrator circuit to convert CMOS pixel currents to pulse frequencies. In section II.D, the system implementation of the complete neural router IC is detailed. Finally, section III elaborates on its application to image filtering tasks up to the complexity of Gabor masks and the relevant processing cascade.

II. Building Blocks of Pulsed Neural IC

A. The Microcircuit and its Implementation

Recurrent, stereotyped neural microcircuits occurring in biological neural networks have been described in [6]. One important aspect of these microcircuits is their individual simplicity, contrasting with the complex processing functions a network of these circuits is capable of. In [2,6], it is postulated that extracting correlations between inputs is one of the major neural processing functions, which has been achieved in [1]. To explore the complex processing possible through networks of microcircuits, and to take advantage of rapid processing inherent in feed-forward excitatory neural structures, [1] has been expanded to the following microcircuit consisting of simple leaky Integrate-and-Fire (IAF) neurons connected by two types of adaptive synapses (Fig.1), which has been introduced in [7].

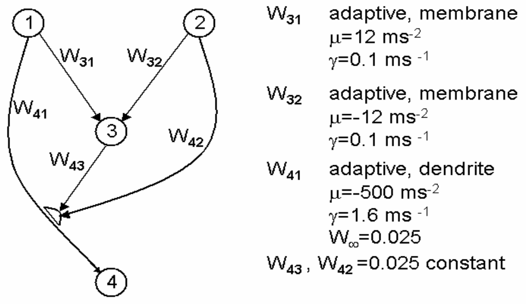


Fig. 1 The neural microcircuit

Manuscript received August 29, 2006. This work was supported by the BMBF (German Federal Ministry of Research) project “Vision IC”. Additionally, part of the research presented herein has encompassed preliminary work for the FACETS project, so we would also thank the European Union for the financial support in the framework of the Information Society Technologies program, Biologically Inspired Information Systems branch, project FACETS (Nr. 15879).

All authors are with the Circuits and Systems Laboratory, Department of Electrical Engineering, Dresden University of Technology, Dresden, Germany. (corresponding author is Christian Mayr, phone: +49-351-463-34943; fax: +49-351-463-37260; e-mail: mayr@iee.et.tu-dresden.de).

The dynamics of the IAF neurons are given by multiplying the pulses running along the synapses with the respective synapse weights and integrating them on the neuron membrane. Once the membrane potential reaches a firing threshold θ of one, the neuron emits an output pulse, immediately resets the integrator and is open for new inputs (no refractoriness period, i.e. temporal blocking of the membrane integrator). All synapse weights are excitatory, i.e. restricted to positive values.

The adaptation rule of the first two synapses (W_{31} and W_{32}), here called a membrane adaptation, is given in equation (1) (Indices are shown for the synapse connecting neurons 1 and 3, expressed by W_{31}):

$$\tfrac{d}{dt} W_{31} = -\gamma \cdot W_{31} + \mu \cdot (a_3 - \tfrac{\theta}{2}) \cdot \chi(X_1) \tag{1}$$

It is a basic Hebbian learning rule [5, section 13.5.1] intended to synchronize pulses with almost constant phase relationships [1]. with γ as decay term, μ as learning rate, a_3 denoting the membrane accumulator state, θ the positive/negative learning threshold, and χ being the indicator function of neuron 1 output X_1, one if neuron 1 exhibits a pulse, zero otherwise. The decay term 'forgets' the learned weight if it is not reinforced by pulse activity, while the second term acts as correlator between the accumulator state of neuron 3 and the pulses exhibited by neuron 1. The accumulator of neuron 3 has to be high (i.e. close to its firing threshold) when neuron 1 fires, for the weight W_{31} to increase, which reflects the Hebbian aim of increasing a synaptic weight if the presynaptic neuron takes part in firing the postsynaptic one.

In this particular application, (1) is employed to extract correlated pulses from the output pulse streams of neurons 1 and 2. To illustrate the correlation function of neurons 1 through 3 governed by (1), let's assume neuron 3 has just emitted a pulse and has a membrane potential a_3 close to zero. If neuron 2 emits a pulse next, the corresponding weight W_{32} is increased ($\mu<0$ and $a_3-\theta/2<0$) and a_3 pushed above θ/2. If neuron 1 emits a pulse next, its corresponding weight is also increased ($\mu>0$ and $a_3-\theta/2>0$) and neuron 3 is pushed above the firing threshold. Only this particular phase relationship (i.e. a pulse of neuron 2 followed by a pulse of neuron 1) results in neuron 3 emitting pulses, thus neuron 3 emits only pulses correlated between neurons 1 and 2.

The second adaptation rule acting on synapse W_{41}, the dendrite adaptation, is given as:

$$\tfrac{d}{dt} W_{41} = -\gamma \cdot (W_{41} - W_{\infty}) - \mu \cdot (X_3 \cdot W_{43} + X_2 \cdot W_{42} - I_{\theta}) \cdot W_{41} \cdot \chi(X_1) \tag{2}$$

which works in such a way that, with no pulses present, the first term draws W_{41} asymptotically to W_{∞}, letting pulses from neuron 1 pass to neuron 4. If, however, the second term is added through a pulse event $\chi(X_1)$, with $I_\theta=0.02<X_3\bullet W_{43}$, the weight W_{41} is decreased. This means, that if a pulse is detected further up the dendritic tree (W_{43} or W_{42}), this pulse blocks any that would be transmitted by W_{41} to neuron 4. Compared to the membrane adaptation the dendritic adaptation acts very fast, as is evident by the differing adaptation parameters μ given in Fig. 1. This type of adaptation operates on single pulses, producing a quasi-digital gating function [5, section 19.3.2]. W_{42} is entered into the circuit to precharge the dendritic adaptation, also mitigating the delay inherent in propagating the correlated pulse across neuron 3.

The net effect of theses rules and synapses is a one-way pulse subtraction, i.e. if there are uncorrelated pulses from neuron 1, these are transmitted across the neural microcircuit, uncorrelated (super numerous) pulses from neuron 2 are ignored. For a more detailed discussion of the neural microcircuit, please see [7].

One of the adaptation rules, the membrane adaptation, has been implemented in analog hardware for a different application, exhibiting its veracity compared to the simulation [1]. However, for the IC implementation of the microcircuit, due to size and design time constraints, a pulsed pseudo-digital representation of the microcircuit has been carried out, exhibiting the same behavior. This microcircuit is part of a neural processing unit (NPU, Fig. 2), with additional functions for pulse weighting and a digitally realized neuron, both carried out by the same digital accumulator, which either transmits the weighted signal (synapse function) or a single pulse if the accumulator reaches a certain threshold (neuron function). The weighting can be governed by external configuration signals, i.e. further adaptation rules carried out in digital computation in an external FPGA.

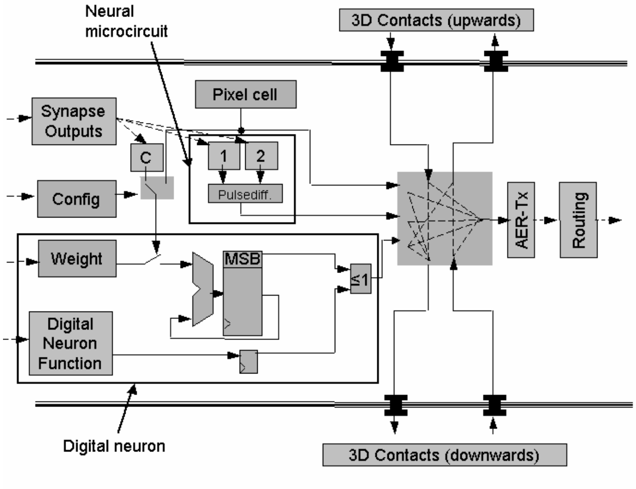


Fig. 2 Block diagram of the Neural Processing Unit with neural microcircuit, digital neuron, and 3D connectivity

The NPU receives the pulses from the router, enters them to the appropriate inputs of the neural circuits, and the output pulses of the NPU are then either transmitted to the bottom or top 3D contacts [8] or to the AER on the same IC. Also, the 3D-contacts can be directly linked, transmitting pulses vertically across the IC stack, and/or the pulses from either top or bottom 3D contact can be encoded by the AER for lateral transmission via the router on the current IC. Pulsed processing, e.g. based on neurons, is especially well suited to mixed signal transmission across the 3D contacts, since their resistance and leakage current vary across wide ranges [8,9], so direct transmission of analog currents or voltages cannot be realized. PWM or phase signals such as those resulting from presenting e.g. a photo current to the input of an Integrate-and-Fire neuron are better suited for transmission across the 3D contacts. Also, compensation for faulty 3D contacts is generally more easily accomplished if only pseudo-digital (i.e. two-level) signals are considered

B. Pulse Encoding and Distribution

In the following, a brief overview of the AER developed in [3] is given. The AER was originally intended as a method for analyzing the pulse outputs of a locally coupled Hebbian adaptive neural net [1], and has been adapted for the use in this pulse router. It is collision free, using an arbiter to detect overlapping pulses. The pulse encoding and compression is locally oriented, i.e. pulses are transmitted most economically if coincidental ones are located in close neighborhood to each other. Fig. 3 details the system as well as the encoding scheme. The AER control clocks the corresponding AER circuitry at 120-200 MHz (task dependent, user selectable), making this the average number of pulses that the AER can transmit. This gives an average timing precision of 5-8.3 ns, which is accurate enough to analyze the wave activity expected in [1] with neurons operating at 10 kHz.

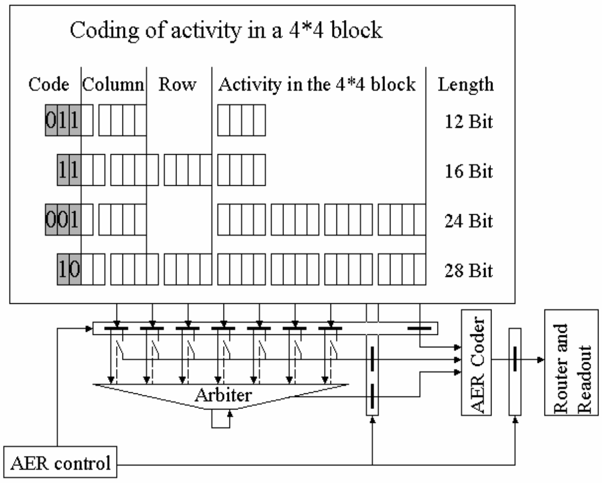


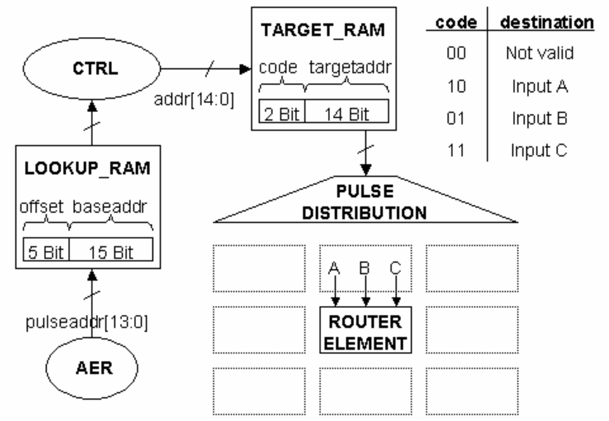
Fig. 3 Coding protocol and arbitrated pulse capture of AER

The arbiter selects among overlapping pulses the order of transmission, and the AER coder encodes the pulses selected by the arbiter according to the coding scheme detailed in Fig. 3. The coding scheme encodes the activity in a 128*128 network of neural circuits, with 5 Bit each encoding the column and row of a 4*4 block, and the remaining Bits either coding a single pulse with 2 Bit each for the column and row in the block (12 and 16 Bit code word) or the complete activity of the 4*4 block (24 and 28 Bit code word), with each bit representing a spike of the corresponding neuron in the current timeslot. Also, if the activity occurs in the same row as the last transmitted code word, only the column is transmitted in the new code word (12 and 24 Bit), otherwise column and row of the block showing activity are transmitted. Following the AER coder, the router and readout are situated, which give the ability to analyze or retransmit the pulses occurring across the network. These components are both implemented as a FIFO memory to buffer short burst activities, but will loose pulses if the network stays above mean activity levels for too long. This is done as a compromise between pulse loss and pulse timing accuracy, since especially in the pulse redistribution case, the pulse timing may be critical to information processing. If pulses enter a full FIFO, and have to ripple completely through it, they will probably only contain obsolete information, especially if that information is inherent in, e.g., phase information [5,6]. Economic usage of the localized nature of the AER coding can be ensured in the router IC if closely correlated pulse processing tasks are situated next to each other.

The pulse distribution is done in the form of a RAM lookup table, i.e. the pulses received from the NPUs, once they have passed the AER encoding and FIFO memory, are decoded into RAM addresses of the lookup table, and the corresponding target NPUs are identified. So-called target-on-target information is also extracted from the RAM, i.e. which input of the NPU the pulse is meant for (see Fig. 2). The pulse is then redistributed to the targets by a 1 from 128 decoder network operating at the same frequency as the AER, which gives the same 120 to 200 million pulses that the router can redistribute across the network, aligning the rates of pulses captured and routed.

Fig. 4 takes a closer look at the organization of the router RAM, with the pulses entering from the AER being assigned a base address and offset in the lookup RAM according to their location in the network, i.e. the lookup RAM is directly linked to each individual NPU. The base address gives the place in the target RAM where the target addresses for this pulse are stored, and the offset controls how many of the following places in the target RAM also contain targets for this pulse. This means, that a single pulse can be distributed to up to 32 targets (corresponding to an offset of 5 Bit). The control CTRL selects the target RAM content according to base address and offset. The content of the target RAM is divided into the target address (2*7 Bit, the 128*128 address of the target NPU), and the two Bit code identifying the input of the NPU the pulse is routed to, with A being the positive input of the neural microcircuit (‘1’ in Fig. 2), B the negative input (‘2’ in Fig. 2), and C is the input to the accumulator/digital neuron. The target RAM, in contrast to the lookup RAM, is therefore dynamically proportioned in accordance with the routing/processing task at hand and its connectivity. This way, the RAM realizes virtual dendrites linking arbitrary NPUs across the IC.

Also, pulses can be fed to the router from sources off-chip at the full router data rate (equivalent to the AER data rate), so pulsed processing functions completely independent of the pulsing pixel cells can be realized.



code	destination
00	Not valid
10	Input A
01	Input B
11	Input C

Fig. 4 Organization of router RAM as directly linked lookup RAM and routing-dependently proportioned target RAM

This type of routing is constant, as long as the router RAM is left unchanged, the network activity is distributed in the same way across the board, regardless of the activity on the corresponding virtual dendrites. However, the RAM can be reconfigured very rapidly, so by monitoring the output of the AER and enacting certain learning rules for network connectivity e.g. governed by pulse rates, dynamic routing could also be realized via an external, supervisory FPGA board.

C. Pulsing Pixel Cell

Also, a pixel cell converting pixel current to a pulsed output signal has been implemented and measured. Fig. 5 shows the circuit of the pulsing pixel cell. The circuit is very simple, made up of an integrator, threshold and reset. The integrator consists of the photo diode D1, with the diffusion capacity acting as the integration capacitor, whose charge is decreased through the (negative) photo current. The resulting output voltage is fed to a differential pair M1/M2 (with M5 providing the biasing tail current and M3/M4 the necessary current mirror).

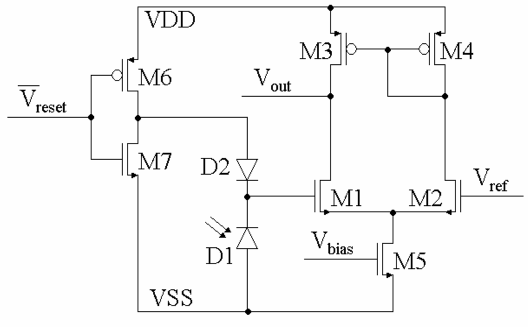


Fig. 5 Circuit of the pulsing pixel cell

The differential pair switches if the integrator voltage falls below V_{ref}, and the resulting V_{out} is fed to a digital buffer (not shown), which in turn activates the reset inverter M6/M7. This low-active reset inverter ordinarily keeps the anode of D2 at ground potential, so no current can flow through this diode. With $\overline{V}_{reset}$ low, however, the anode of D2 is pulled to VDD, resulting in a current flow through D2 which charges the integration capacitor of D1 to VDD minus the forward voltage of D2, thus resetting D1. Fig. 6 details measurement data taken from a VLSI implementation of the pulsing pixel cell in a 130 nm CMOS technology.

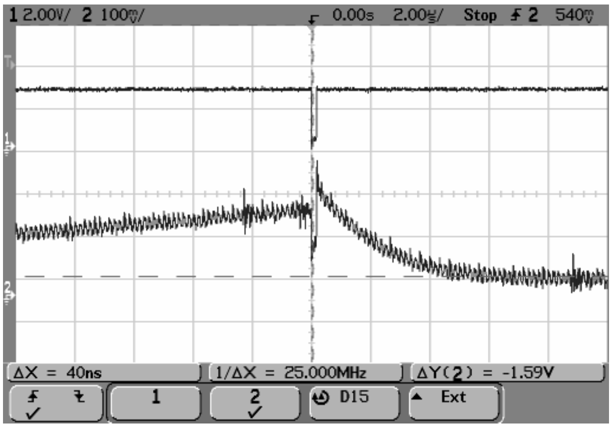


Fig. 6 Digital output (upper curve) and integrator voltage (lower curve) of the pulsing pixel cell for a single reset event

The upper curve shows a single output pulse (negative logic) of the pixel cell, the lower curve displays the voltage across the integrating diode capacity. As can be seen prior to the reset event the integrator voltage rises linearly due to the constant photo current. After the reset has been applied, the voltage reaches its resting potential within approximately 4 us. Please note, due to circuit constraints the integrator voltage has been buffered by an inverting OTA for measurement off-chip, so the integrator voltage curve in figure 6 shows positive charging, while in the actual circuit of figure 5 the charging is negative, as explained above.

When tested at illumination levels comparable to room lighting (about 72 lux), the pulse frequency is approximately 800 Hz, for very dark illumination of 9 lux, the frequency is 140 Hz. Bright daylight (200-300 lux) would put the frequency at 2-4 kHz. The measured power draw of the pulsing pixel cell is 50nA at 2.2V, independent of pixel frequency, since this constitutes the bias current of the differential pair, the (frequency-dependent) power draw by the reset inverter is negligible.

D. System Implementation

The router IC has been implemented in a 130 nm Infineon technology, realized as full custom digital design with auto generated RAM blocks and mixed-signal insets. The NPUs have been implemented from high-level descriptions via a place and route tool, with the analog pulsing pixel cells as hand-layout inserted in placeholder spaces. The IC is currently undergoing production. Fig. 8 depicts the floor plan of the router IC, divided into the various building blocks. The floor plan is not to scale, for a size orientation, see the dimensions indicated at the borders. The router array shows the column and row decoders transmitting a pulse from the router to a pulse processing element. Also demonstrated is the block structure of the AER coder. The configuration of the NPUs, i.e. what kind of processing is performed on the pulses, is done via a JTAG interface contained in the block CHIP_INTERFACE. Simulated power consumption at a mean pulse distribution rate of $160*10^6$ pulses/s is about 2 W for the router and periphery, additional power demand for the NPUs is 0.7 W.

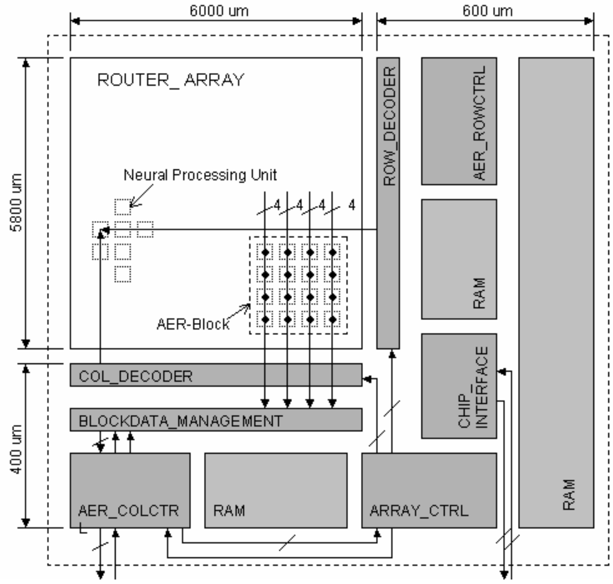


Fig. 8 Floor plan of the router IC

III. Pulsed Image Convolution

On a general note, the results presented in Figs. 9 and 10 are based on system-level Mentor ModelSim simulations of VHDL/AHDL code of the IC. Major components of the IC, however, have been implemented and verified previously, like the Adress-Event-Representation employed for pulse communication [3], the adapting synapses [1], or the pulsing pixel cell (detailed above). Where applicable, the VHDL/AHDL code has been augmented by measurement results to enhance its veracity.

A. Simple Gabor Edge Filtering

A group of the neural microcircuits can be used to construct a simple Gabor convolution mask approximation [4] by stacking them according to Fig. 9 (top left), where gray scale values lighter than the mean denote positive 1 (+) inputs of the neural microcircuits, while gray scale values darker than mean denote negative 2 (-) inputs. To achieve the different coefficients inherent in a Gabor convolution mask, the mask is discretized, and the discretized levels are converted to a corresponding multiple access of neural microcircuits to the same pixel, thus achieving a weighting function of the input image in accordance with the Gabor coefficients. The respective input neurons 1 (+) and 2 (-) of the neural microcircuit are then fed with pulsed representations of the grayscale image by the pulsing pixel cells, and their outputs are summed. Fig. 9 shows the (simulated) summed output of such a Gabor mask if a rotating edge is presented to it. The pulsed greyscale images are obtained by supplying an AHDL representation of the pulsing pixel cell presented earlier with input current linearly based on pixel brightness (greyscale value).

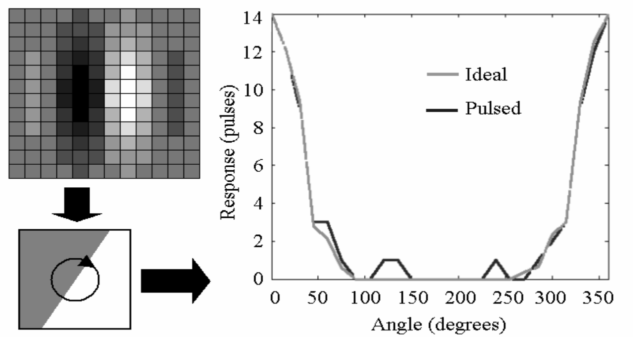


Fig. 9 Representation of discretized Gabor mask and response to a rotating edge

The deviation between ideal and pulsed Gabor response e.g. between 100° and 150° is caused by the one-way subtraction, i.e. the outer edges of the Gabor mask produce spurious results which would normally be suppressed by the large negative response of the central Gabor mask. This one-sidedness of the NPU subtraction obstructs the use of such a one-stage Gabor filter as a true spatial frequency filter, since such a filter must match the positive as well as the negative portions of a spatial grayscale wave pattern.

B. Pulsed Processing for Gabor Decomposition

By setting up a processing pyramid, i.e. several ordered processing steps, of these microcircuits and introducing pathways to the neurons in the extended neighborhood, more complex image filtering functions can be accomplished, in particular, true wavelet (i.e. localized spatial frequency) decomposition of an image as postulated from biological evidence [4] can be realized as a pulsed image computation.

A first step would be the suppression of errors such as the one evident in Fig. 9. This can be done by correcting the mask response with an adjustment signal delivered by the exactly opposite mask. Consider for example, a 1D convolution with mask (1 –2 1), which could be realized with a pair of NPUs in the form (+ -- +), where both negative inputs (-) access the same pixel. If an input pattern of (3 2 1) were presented to the ideal mask, the response would of course be 0. Using the neural microcircuits, however, the first NPU would deliver 3-2=1, whereas the result for the second NPU is 1-2=0, because of the one-sidedness of its subtraction (section 2.1), so the summed response of the NPU mask would be 1. If we introduce a second, negative mask (- ++ -), its result would be 1 as well, so we get the correct result of 0 by subtracting the result for the negative mask (- ++ -) from the positive mask response. This procedure will not alter the response for a perfect fit to the positive mask, since the negative mask responds with 0 in such a case. This correction is not perfect, an input pattern of (2 2 1) will result in –1 for the ideal mask, whereas even the corrected signal (pos-neg) for the NPUs is 0. The second step would be computing the absolute value Gabor response R_{abs} from the corrected mask response $R_+ - R_-$ obtained from the summed responses R_+ and R_- to the positive and negative Gabor masks, respectively. This absolute value is of course defined as:

$$\left|R_+ - R_-\right| = \begin{cases} R_+ - R_- & for\ R_+ \geq R_- \\ R_- - R_+ & for\ R_+ < R_- \end{cases} \tag{3}$$

If we input R_+ and R_- once in every direction to a neural microcircuit and sum the results, the one-sided subtraction results in the same computation:

$$\left(R_+ - R_-\right) + \left(R_- - R_+\right) = \begin{cases} R_+ - R_- + 0 & for\ R_+ \geq R_- \\ 0 + R_- - R_+ & for\ R_+ < R_- \end{cases} \tag{4}$$

Fig. 10 illustrates the accuracy of this pulsed convolution method as compared to conventional image convolution. The vertical stripes evident in Fig. 10c are caused by the non-ideal correction described above.

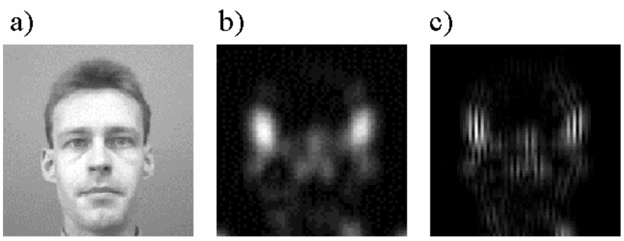


Fig. 10 Comparison of original image (a), amplitude response of an ideal Gabor image filter (b), and pulsed realization using the NPUs and router (c)

IV. Conclusion

We have presented a VLSI implementation of a pulsed image processing scheme. This processing starts out with pulse-stream representations of grayscale images, generated by a pulsing CMOS pixel sensor, and employs pulse processing based on biological evidence, contained in a Neural Processing Unit. The neural microcircuit is based on two information processing principles postulated from biological evidence, Hebbian pulse correlation [5, section 13.5.1] and dendritic pulse gating [5, section 19.3.2]. While performing a simple pulse correlation/decorrelation individually, suitably shaped networks of these microcircuits are capable of realizing complex image filtering tasks such as Gabor wavelet decomposition as pulse-based computation. These networks employ some of the principles postulated from biological evidence, e.g. the building of complex masks through several simpler interim steps in a layered configuration, and the use of directly opposing masks at the same image location (similar to On/Off centers [4,5,7]) to cover the whole spectrum of convolution mask responses to a given image.

References

[1] J. Schreiter, U. Ramacher, A. Heittmann, D. Matolin, and R. Schüffny, “Cellular pulse coupled neural network with adaptive weights for image segmentation and its VLSI implementation,” in *Proc. IS&T/SPIE 16th International Symposium on Electronic Imaging: Science and Technology*, San Jose (CA), USA, 2004, 5298, pp. 290-296.

[2] Y. Ota and B.M. Wilamowski, “CMOS architecture of synchronous pulse-coupled neural network and its application to image processing,” in *Proc. of the 26th Annual Conference of the IEEE Industrial Electronics Society (IECON'00), Nagoya, Japan*, 2000, pp. 1213-1218.

[3] Bernd Richter, “Development of a readout circuit for pulse-coupled artificial neural networks,” Diploma thesis, University of Technology Dresden, Endowed Chair for Neural Circuits and Parallel VLSI-Systems, Dresden, Germany, 2003.

[4] D.H. Hubel and T.N. Wiesel, “Receptive fields and functional architecture of monkey striate cortex,” *Journal of Physiology*, vol. 195, no. 1, pp. 215-243, March 1968.

[5] C. Koch, *Biophysics of computation – information processing in single neurons*, Oxford University Press, Oxford, 1999.

[6] W. Maass, T. Natschläger, and H. Markram, “Real-time computing without stable states: a new framework for neural computation based on perturbations,” *Neural Computation*, vol. 14, no. 11, pp. 2531-2560, Nov. 2002.

[7] A. Heittmann and U. Ramacher, “An architecture for feature detection utilizing dynamic synapses,” in *Proc. of the 47th IEEE International Midwest Symposium on Circuits and Systems*, Hiroshima, Japan, 2004, pp. II-373 - II-376.

[8] P. Benkart, A. Heitmann, H. Huebner, U. Ramacher,, A. Kaiser, A. Munding, M. Bschorr, H.-J. Pfleiderer, E. Kohn, “3D chip stack technology using through-chip interconnects,” *IEEE Journal of Design & Test of Computers*, vol. 22, no. 6, pp. 512-518, Nov. 2005.

[9] A. Munding, Department of Electron Devices and Circuits, University of Ulm, Ulm, Germany, private communication, July 2005.