Programmable 2D Image Filter for AER Vision Processing

Teresa Serrano-Gotarredona[1], Andreas G. Andreou[2], and Bernabé Linares-Barranco[1]

[1]Instituto de Microelectrónica de Sevilla (IMSE), Centro Nacional de Microelectrónica (CNM),
Ed. CICA, Av. Reina Mercedes s/n 41012 Sevilla, SPAIN. Phone: 34-5-4239923,
Fax: 34-5-4231832, E-mail: terese@imse.cnm.es

[2]Dept. of Electrical and Computer Engineering, The Johns Hopkins University,
Baltimore, MD 21218, USA

Abstract

A VLSI architecture is proposed for the realization of real-time 2D image filtering in an Address-Event-Representation (AER) vision system. The architecture is capable of implementing any convolutional kernel *F(x,y)* as long as it is decomposable into *x*-axis and *y*-axis components, i.e. *F(x,y)=H(x)V(y)*, for some rotated coordinate system {x,y}, and if this product can be approximated safely by a signed minimum operation. The proposed architecture is intended to be used in a complete vision system, known as the Boundary-Contour-System and Feature-Contour-System (BCS-FCS) Vision Model.

I. Introduction

Human beings have the capability of recognizing objects, figures, and shapes even if they appear embedded within noise, are partially occluded or look distorted. To achieve this, the human vision processing system is structured into a number of massively interconnected neural layers with feedforward and feedback connections among them. Neurons communicate by means of electrical streams of pulses. Each neuron broadcasts its output to a large number of other neurons, which can be inside the same or at different layers, and the way this is done is through physical connections called *synapses* [1]. One big problem encountered by engineers when it comes to implement bio-inspired (vision) processing systems is to overcome the massive interconnections. The Address Even Representation (AER) [2]-[5] approach is a possible solution. Fig. 1 shows a schematic figure outlining the essence behind AER. Suppose we have an "emitter" chip containing a large number of neurons or cells *D1, D2, D3, ...* whose activity changes in time with a "relatively slow" time constant. For example, if *Chip* 1 is a retina chip and each neuron's activity represents the illumination sensed by a pixel, the time constant with which this activity changes can be equivalent to *Frame-Rate* (i.e., 25-30 changes per second or a time constant of about 30-40*ms*)[1]. The purpose of an AER based communication scheme is to be able to reproduce the time evolution of each neuron's activity inside a second or "receiver" chip, using a fast digital bus with a small number of pins. In the "emitter" chip the activity of each pixel has to be transformed into a pulse stream signal such that pulse width is minimum and the spacing between pulses is reasonably high to time multiplex the activity of a relatively large number of neurons. Every time a neuron produces a pulse its address or code should be written on the bus. For the case more than one pulses are produced simultaneously by several neurons, a classical arbitration tree can be introduced [2]-[4], or one based in Winner-Takes-All (WTA) row-wise competitions [6], or simply by making no neuron accessing the bus in case of a "collision" [7]. Whatever method is used the result will be the presence of a sequence of addresses or codes on the digital bus that one or more receiver chips can read. Each receiver chip must contain a decoding circuitry so that a pulse reaches the neuron (or neurons) specified by the address read on the bus. If each neuron integrates the sequence of pulses properly, the original activity of the neurons in the emitter chip will be reproduced. AER allows easily to add more complicated processing. For example, input images can be translated or rotated by remapping the addresses while they travel from one chip to the next. By properly programming an EEPROM as a look-up table any address remapping can be implemented, by simply inserting the EEPROM between the two chips. Furthermore, many EEPROMs can be connected in parallel each performing, for example, a rotation at a specific angle, and each delivering the remapped addresses to a set of specialized processing chips. In the architecture proposed in this paper, we implement a synaptically weighted projection field for each address read on the bus. This can be done by either having a hard-wired kernel in the filtering chip [8], or by implementing a programmable one, as proposed in this paper.

1. In this paper we consider "*Real-Time*" a processing that is performed at frame rate (30-40*ms*) or faster.

II. The Programmable Filter

The programmable filter described in this paper is intended to be used in a vision model system, known as the Boundary-Contour-System (BCS) and Feature-Contour-System (FCS) [9]. Such vision model consists of a set of layers computing convolutions. The convolutional kernels used in most of these layers $F(x, y)$ are decomposable into x- and y-axis components, $F(x, y) = H(x)\,V(y)$, for some rotated coordinate system $\{x, y\}$. Using AER allows us to implement a filtering chip only for the coordinate system $\{x, y\}$ for which $F(x, y)$ is decomposable. To do the filtering for another coordinate system $\{x', y'\}$, rotated with respect to $\{x, y\}$ an arbitrary angle α, we can use the same chip but providing addresses which have been rotated $-\alpha$ previously.

In the filtering chip, the convolutional kernel is implemented as follows. Let us call $\mathcal{P}_o(p, q)$ the sequence of pulses the AER bus receives for address (p,q). Every time a pulse for address (p, q) is received, pulses are sent to all pixels in its vicinity. This way, a lossy integrator at pixel (x, y) of the receiver chip will integrate the sequence of pulses

0-7803-5471-0/99/$10.00©1999 IEEE

$$\mathcal{P}_1(x,y) = \sum_{p-L}^{p+L} \sum_{q-L}^{q+L} F(x-p, y-q)\, \mathcal{P}_o(p,q) \qquad (1)$$

which are all pulses coming in from its vicinity, weighted by the convolutional kernel $F(x,y)$. The weighting is performed by modulating the width of each incoming pulse. Thus, every time a pulse is received for pixel (p,q), a pulse of width $F(x-p, y-q)$ is sent to pixel (x,y) in its vicinity ($x \in \{p-L, \ldots, p+L\}$, $y \in \{q-L, \ldots, q+L\}$).

Pulse width modulation is done as follows. When a pulse for coordinate (p,q) is received, all columns x in the vicinity of column p receive a pulse of width $|H(x-p)|$, and all rows y in the vicinity of rows q receive a pulse of width $|V(y-q)|$. The values of $H(\cdot)$ and $V(\cdot)$ are stored in a small on-chip RAM. The integrator at coordinate (x,y) receives a pulse of width equal to the minimum of $|H(x-p)|$ and $|V(y-q)|$. Consequently, the convolutional kernel the system will implement is an approximation to $F(x,y) = H(x)\,V(y)$, which is

$$F_m(x,y) = sign[H(x)]\, sign[V(y)]\, min(|H(x)|, |V(y)|) \quad (2)$$

the signed minimum of the vertical and horizontal components.

III. Circuit Description

The address bus provides the coordinates (x_0, y_0) of the neuron (or pixel) around which the convolutional kernel should be applied. Pulses will be applied to all rows with *y-coordinate* in the interval $[y_0 - L, y_0 + L]$, and all columns with *x-coordinate* in the interval $[x_0 - L, x_0 + L]$, where $2L+1$ is the width considered for the kernel. Pulses will be modulated in width according to function $|V(y)|$ for the rows, and function $|H(x)|$ for the columns. At each pixel there is an AND gate which provides a pulse of width equal to the minimum of $|V(y)|$ and $|H(x)|$. This pulse will generate a fixed magnitude current pulse of the same width which will be integrated on a capacitor. Each pixel contains two integrators. One of them, called the "positive integrator", integrates the pulses of width $min(|H(x)|, |V(y)|)$ when $\text{sgn}(H(x))\,\text{sgn}(V(y)) > 0$; while the other, called the "negative integrator", integrates the pulses when $\text{sgn}(H(x))\,\text{sgn}(V(y)) < 0$. The values of $V(n)$ and $H(n)$ ($n = -L, \ldots 0 \ldots, L$) are stored digitally on chip in a small RAM.

Fig. 2 shows the block diagram of the system. In *RAM X* and *RAM Y* digital words of $n+1$ bits are stored ($Rx_{-L}, \ldots, Rx_l, \ldots, Rx_L$ and $Ry_{-L}, \ldots, Ry_s, \ldots, Ry_L$). The first bit Sx_l (or Sy_s) indicates the sign of the function $H(x)$ (or $V(y)$). The following n bits indicate the absolute value $|H(x)|$ (or $|V(y)|$). These n bits linearly control the length of the pulse triggered by monostables Mx_l (or My_s). The monostables achieve this by charging with a constant current a programmable capacitor controlled by the n bits in Rx_l or Ry_s. The pulses generated by the monostables are sent through lines Tx_l (or Ty_s) and are triggered whenever an external pulse arrives to the system. When an external pulse arrives, the input decoders activate lines x_i and y_j corresponding to the address of the arriving pulse. The selection cells controlled by x_i (cells $Cx_{i-l,l}$ in Fig. 2, $l = -L, \ldots 0 \ldots L$) connect the pulse in line Tx_l to line Px^+_{i-l} if the sign bit Sx_l is '1'. If the sign bit Sx_l is '0' line Tx_l is connected to the negative line Px^-_{i-l}. This way, pulses Tx_l (or Ty_s) are sent through lines Px^+_{i-l} or Px^-_{i-l} (Py^+_{j-s} or Py^-_{j-s}) depending on the sign of the weight stored in Rx_l (or Ry_s).

Each neuron c_{ij} has two integrators. The positive integrator accumulates charge when pulses are simultaneously arriving through horizontal and vertical lines of the same sign. That is, it integrates a pulse when lines Px^+_i and Py^+_j (or lines Px^-_i and Py^-_j) are simultaneously high, or equivalently it performs the operation $(Px^+_i \cap Py^+_j) \cup (Px^-_i \cap Py^-_j)$. Similarly, the negative integrator accumulates charge when pulses arriving through horizontal and vertical lines of opposite sign Px^+_i and Py^-_j (or Px^-_i and Py^+_j) are simultaneously high, that is, it performs the operation $(Px^+_i \cap Py^-_j) \cup (Px^-_i \cap Py^+_j)$.

Fig. 3(a) depicts the schematic of the selection cell $Cx_{i-l,l}$ (or $Cy_{i-s,s}$) used to select the neighborhood of cells where the monostable pulses have to be sent. It consists of two NAND gates controlling the PMOS switches MP^+ and MP^-, and two NMOS pull down transistors MN^+ and MN^-. Each selection cell $Cx_{i-l,l}$ has two control signals (the decoder output x_i and the sign bit Sx_l from *RAM X*), one input signal (the monostable output Tx_l) and two outputs (Px^+_{i-l} and Px^-_{i-l}). When a pulse arrives with address (x_i, y_j), it activates the decoders output x_i and y_j, respectively. The decoder output x_i controls all the selection cells $Cx_{i-l,l}$ with $l \in [-L, \ldots, L]$. When x_i is high, if the sign bit Sx_l is '1', the selection cell $Cx_{i-l,l}$ connects the monostable output line Tx_l to the positive line Px^+_{i-l}. If the sign bit Sx_l is '0', line Tx_l is connected to the negative line Px^-_{i-l}. The same is valid for the Y coordinate selection cells.

Each synaptic cell c_{ij} has two integrators, the positive and the negative. Fig. 3(b) shows the circuit diagram for the positive integrator. The negative is identical, except for labelling. The integrator is based on the Capacitor-Diode integrator concept for subthreshold MOS operation [4]. This integrator has some interesting properties with respect to a conventional linear RC-integrator:

- Steady state current is proportional to pulse stream frequency
- Steady state current is proportional to pulse width
- Steady state current ripple is independent of current level
- Steady state is reached, for a given precision, after a constant number of pulses

In Fig. 3(b), the two AND and the NOR gates provide a pulse of width equal to the minimum of the pulse width coming in horizontally and vertically. This pulse turns ON current source M_w providing a current pulse of amplitude I_w (controlled by bias voltage V_w). Since transistors M^+_1 and M^+_2 are biased in subthreshold, the integrator input and output currents, I^+_{in} and I^+_{ij}, are related by [4]

$$Q_T \frac{dI^+_{ij}}{dt} = I^+_{ij}\left(I^+_{in} - \frac{I^+_{ij}}{A}\right) \qquad (3)$$

where $A = \exp((V_{dd} - V_A)/\nu_T)$, $Q_T = C\nu_T/\kappa$, ν_T is thermal voltage and κ is a characteristic subthreshold dimensionless technology parameter whose value may range from 0.60 to 0.98 [10]. When a train of pulses of width T_h and frequency $1/T$ ($T \gg T_h$) is applied to this integrator, the steady state output current and ripple are[4]

$$I^+_{ij}(\infty) \approx A I_w \frac{T_h}{T} \qquad \frac{\Delta I^+_{ij}(\infty)}{I^+_{ij}(\infty)} \approx \frac{T_h}{\tau} \qquad (4)$$

where $\tau = Q_T/I_w \gg T_h$. Equation (4) expresses that the relative resolution in the integrator output is constant,

independent of the signal level, and that each integrator outputs a current which is proportional to the frequency $1/T$ and width T_h of the input pulses. If the AER input image pixel intensity is linearly encoded with the frequency of the arriving pulses, and the convolutional kernel is encoded as the pulses width, the output current of the integrators would be the input image filtered by the convolutional kernel.

Fig. 4(a) shows an Hspice transient simulation for one of the integrator cells in Fig. 3(b). Transistor sizes are $W = 12\mu m$ and $L = 12\mu m$, integrating capacitor is $C = 0.1pF$, pulse amplitude is $I_w = 13.5nA$, pulse width is $T_h = 100ns$, frequency of pulse stream is $1/T = 80KHz$, $V_{dd} = 5V$, and voltage V_A was set to $4.67V_{+/-}$ (which yields a current gain from transistor $M_1^{+/-}$ to $M_2^{+/-}$ of around 2000). Similar simulations were performed by sweeping the frequency of the input pulse stream $1/T$ and the width of the pulses T_h. The results are shown in Fig. 5. Fig. 5(a) shows the steady-state current level as a function of frequency, while maintaining $T_h = 10ns$. Fig. 5(b) shows the steady-state current level as a function of pulse width, while maintaining the frequency constant at $4KHz$.

Sometimes, in 2D image filtering processing, a rectification operation has to be performed. This is the case, for instance, when doing orientation extraction with Gabor-like kernel filters. The output of the filter is rectified for each pixel [9]. Because of this, the chip scan-out circuitry, which brings out of the chip the state of a c_{ij} cell, has been designed to be able to add a rectification operation. The random access scanning circuitry can read the rectified output current of any cell selected by the *Random Scan Bus* of Fig. 2. The output decoder X (see Fig. 2) selects a column i through pin Scx_i. When a column is not selected, the output currents I_{ij}^+ and I_{ij}^- of all c_{ij} cells in that column flow to a line of constant voltage V_{REF} (see Fig. 3(b)). If column i is selected, currents I_{ij}^+ and I_{ij}^- of all c_{ij} cells in these columns flow to lines I_j^+ and I_j^-, respectively, of the scan out cell $Scan_j$ shown in Fig. 6. Each scan out cell $Scan_j$ receives two input currents I_j^+, I_j^- and provides an output current $|I_j^+ - I_j^-|$. Current I_j^- is mirrored through a PMOS current mirror and subtracted from current I_j^+. The PMOS current mirror has an active input clamped to a voltage V_{REF}. This maintains a constant voltage V_{REF} at output nodes $I_{ij}^{+/-}$ of cells c_{ij} when they are selected, thus speeding up the read out of currents. Current $I_j^+ - I_j^-$ enters the current comparator composed of transistors M_1, M_2 and $OPAMP_1$ [11], whose input node (and output I_{ij}^+ of all selected c_{ij} cells) is clamped to voltage V_{REF}. If current $I_j^- - I_j^+$ is positive transistor M_2 will sink this current. Transistor M_4 shares its gate with M_2 and its source is connected to a voltage reference of value V_{REF}, thus transistor M_4 mirrors the current passing through M_2,

$$[I_j^- - I_j^+]^+ = \begin{cases} I_j^- - I_j^+ & if \quad I_j^- - I_j^+ > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

If current $I_j^+ - I_j^-$ is positive transistor M_1 sources this current, which is mirrored by transistor $M3$ because its source is clamped to V_{REF} by the current comparator composed by transistors $M5, M6$ and $OPAMP2$. Therefore, the current through $M3$ and $M5$ is,

$$[I_j^+ - I_j^-]^+ = \begin{cases} I_j^+ - I_j^- & if \quad I_j^+ - I_j^- > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

This current is again reflected by the PMOS transistor pair M_5, M_7. At the output node, the currents through transistors M_4 and M_7 are added together to get the rectified current $I_o = |I_j^+ - I_j^-|$. Since transistors $M1 - M7$ operate in weak inversion, increasing the source voltage of transistors $M4$ and $M7$ with respect to V_{REF} will make the current mirrors $M2, M4$ and $M5, M7$ to have a gain higher than one (actually the gain will be exponentially controlled by this voltage difference). This allows to have a current gain such that the output current is of the order of hundreds of $\mu Amps$ or even some $mili - Amps$, making it possible to drive this current directly off-chip at high speeds. Fig. 4(b) shows an Hspice simulation of the DC characteristic of a scancell. In this simulation, current I_j^+ was set to $80nA$ and current I_j^- was swept from $0nA$ to $160nA$. Two traces are shown in Fig. 4(b). The dotted line shows the current $[I_j^- - I_j^+]^+$ flowing through transistor M_4. The solid line corresponds to current $[I_j^+ - I_j^-]^+$ flowing through transistor M_7.

Using the analytical solution of eq. (3) and the Hspice simulation results, behavioral system level simulations were performed on the arquitecture of Fig. 2 for a 128×128 array, doing a Gabor filtering for vertical lines extraction. Fig. 7(b) shows the output obtain for the input image in Fig. 7(a).

IV. Conclusions

An architecture that implements a programmable 2D image filter has been presented. The architecture allows to implement any 2D filter $F(p,q)$ decomposable into x-axis and y-axis components $F(p, q) = H(p)V(q)$ such that the product can be approximated by a signed minimum. Positive and negative values of $H(p)$ and $V(q)$ can be programmed. The architecture requires an AER input. This allows to rotate the 2D convolution kernel any angle. Circuit simulation results of critical components were given. System-level behavioral simulations of a 128×128 array have been included which validate the proposed approach.

V. References

[1] Gordon M. Shepherd, *The Synaptic Organization of the Brain*, Oxford University Press, 3rd Edition, 1990.

[2] J. Lazzaro, J. Wawrzynek, M. Mahowald, M. Sivilotti, and D. Gillespie, "Silicon Auditory Processors as Computer Peripherals," *IEEE Transactions on Neural Networks*, May, 1993.

[3] M. Mahowald, *An Analog VLSI System for Stereoscopic Vision*, Kluwer Academic Publishers, 1994.

[4] Kwabena Boahen, "Retinomorphic Vision Systems," *Microneuro'96: Fifth Int. Conf. on Neural Networks and Fuzzy Systems*, Laousanne, Switzerland, February 1996.

[5] A. G. Andreou, R. C. Meitzler, K. Strohbehn, and K. A. Boahen, "Analog VLSI Neuromorphic Image Acquisition and Pre-Processing Systems," *Neural Networks*, vol. 8, No. 7/8, pp. 1323-1347, 1995.

[6] Z. Kalayjian, J. Waskiewicz, D. Yochelson, and A. G. Andreou, "Asynchronous Sampling of 2D Arrays using Winner-Takes-All Arbitration," *Proceedings of the 1996 IEEE Int. Symp. on Circuits and Systems* (ISCAS'96), Atlanta, vol. 3, pp. 393-396, 1996.

[7] A. Mortara, E. A. Vittoz, and P. Venier, "A Communication Scheme for Analog VLSI Perceptive Systems," *IEEE Journal of Solid-State Circuits*, vol. 30, No. 6, pp. 660-669, June 1995.

[8] P. Venier, A. Mortara, X. Arreguit, and E. A. Vittoz, "An Integrated Cortical Layer for Orientation Enhancement," *IEEE Journal of Solid-State Circuits*, vol. 32, No. 2, pp. 177-186, February 1997.

[9] S. Grossberg, E. Mingolla, and J. Williamson, "Synthetic Aperture Radar Processing by a Multiple Scale Neural System for Boundary and Surface Representation," *Neural Networks*, vol. 8, No. 7/8, pp. 1005-1028, 1995.

[10] A. G. Andreou and K. A. Boahen, "Translinear Circuits in Subthreshold MOS," *Analog Integrated Circuits and Signal Processing*, vol. 9, pp. 141-166, 1996.

[11] A. Rodríguez-Vázquez, R. Domínguez-Castro, F. Medeiro, and M. Delgado-Restituto, "High Resolution CMOS Current Comparators: Design and Applications to Current-Mode Function Generation," *Analog Integrated Circuits and Signal Processing*, vol. 7, pp. 149-165, 1995.

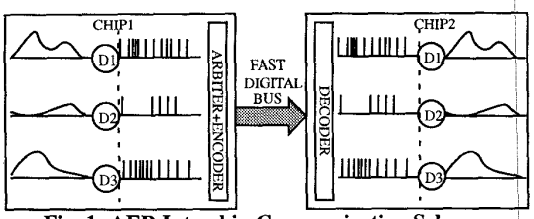

Fig. 1: AER Interchip Communication Scheme

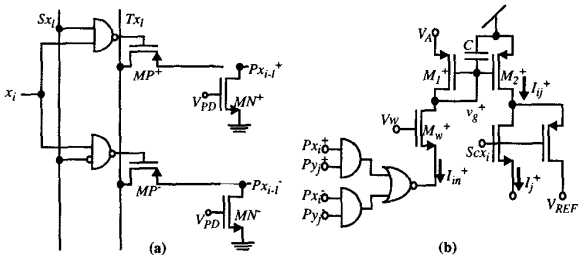

Fig. 3: Schematic of (a) neighborhood selection cell and (b) one half of core cell diode-capacitor integrator

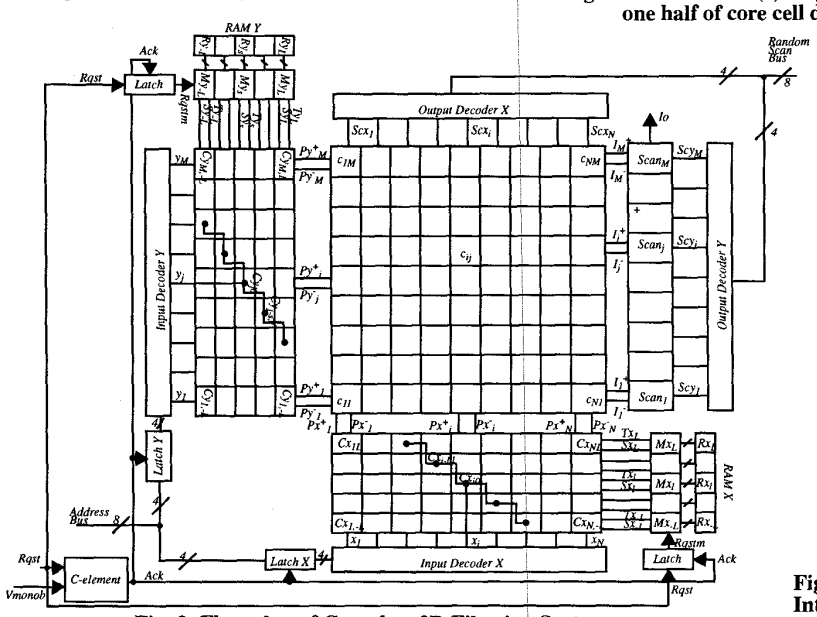

Fig. 2: Floorplan of Complete 2D Filtering System

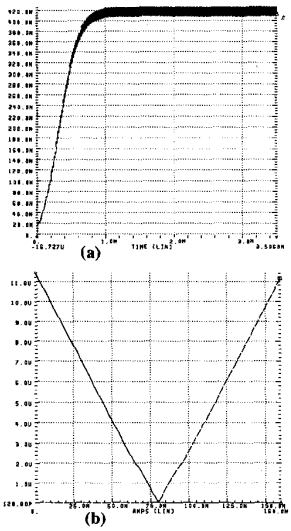

Fig. 4: Hspice Simulation of (a) Integrator Cell Transient and (b) DC characteristics of Scan-Out Cell

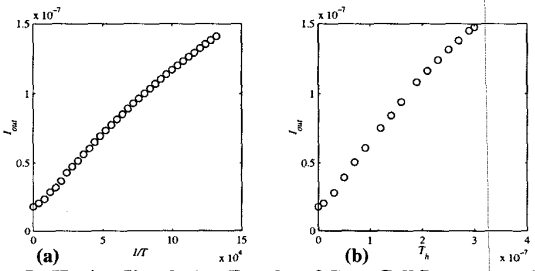

Fig. 5 : Hspice Simulation Results of Core Cell Integrator, (a) Steady-State Current vs. frequency of Input Pulse Stream for T_h=10*ns*, (b) Steady-State Current vs. T_h , for 1/*T*=4*KHz*

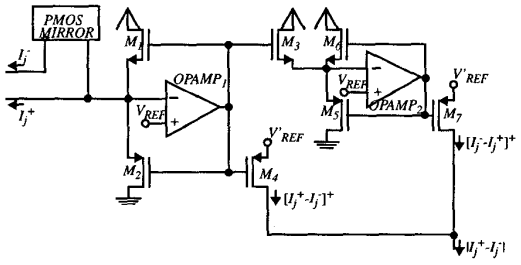

Fig. 6: Schematic of a cell to scan out the absolute value of the difference of two currents

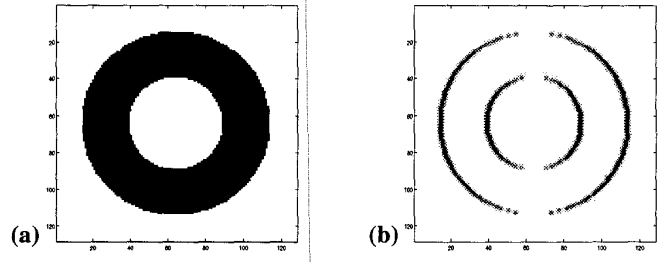

Fig. 7: System-Level Behavioral Simulations of a 128×128 Array. (a) Input Image, (b) Output Image of Pseudo-Gabor Filter extracting Vertical Edges