

# P1-13: A MATRIX Transform Imager and Architecture

Paul Hasler and Abhishek Bandyopadhyay

Georgia Institute of Technology, Atlanta, GA 30332, phasler@ece.gatech.edu, (404) 894-2944

## ABSTRACT

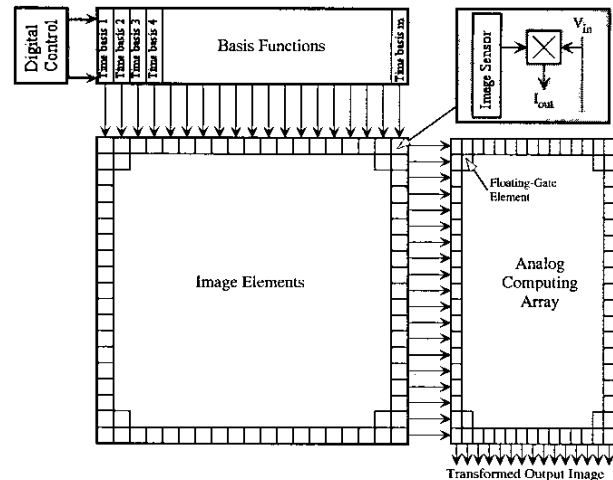
We introduce our Transform Imager Technology and Architecture. This approach allows for Retina and higher-level bio-inspired computation in a programmable architecture that still possesses similar high-fill factor pixels of APS imagers. This imager is capable of programmable matrix operations on the image, where we can represent the image as either a full matrix or using block matrix operations. The resulting data-flow architecture directly allows computation of spatial transforms, motion computations, and stereo computations. The core imager performs computation at the pixel plane, but still holds to a fill factor greater than 40 percent. Each pixel is composed of a photodiode sensor element and a multiplier.

We introduce our Transform Imager Technology and Architecture. This approach allows for Retina and higher-level bio-inspired computation in a programmable architecture that still possesses similar high-fill factor pixels of APS imagers. Figure 1 shows the block diagram of our Transform imagers. If the incoming voltages represent functions in time, particularly transform bases like sine and cosine, then we are performing computations analogous to matrix image transforms. The output is a continuous stream of each row of the transformed image, repeated over a desired fundamental frequency. This approach is enabled by floating-gate circuits [2], in storing arbitrary analog waveforms for image transforms, in programming waveforms to account for average device mismatch, and in computing additional matrix-vector computations.

In this paper, we present our approach in four sections. In Section 1, we briefly describe the similarities of this transform imager to other photodiode based imagers. In Section 2, we describe our transform imager concepts. This imager is capable of programmable matrix operations on the image, where we can represent the image as either a full matrix or using block matrix operations. In Section 3, we present the resulting data-flow architecture for image processing built around this transform imager.

## 1. TRANSFORM IMAGERS AS A MIXTURE OF CMOS IMAGERS AND RETINAS

Imagers based on photodiodes tend to be characterized as either bioinspired, such as retina models or other focal-plane processing, or as closer to APS image sensors. Transform imagers borrow both from focal-plane imagers like retinas as well as standard CMOS and random-access imagers to



**Fig. 1.** Top view of our matrix transform imager. The image output rate will be the same as the time to scan a given image; then the resulting image transfer rate is significantly reduced if information is refined. Approach allows arbitrary separable matrix image transforms; these transforms are programmable because we use floating-gate circuits. Voltage inputs from various time bases are broadcast along columns, and output currents are summed along lines on each row. Each pixel processor multiplies the incoming input with the measured image sensor result, and outputs a current of this result. Time basis could be oscillators, pattern generating circuits, or arrays or stored analog values (i.e. floating-gate storage). With can compute block image transforms with smaller time bases, digital control, and smaller block matrices for block image transforms. Finally, we can get multiple parallel results, since all of the matrix transforms could operate on the same image flow.

create this unique architecture.

The first widespread research in silicon imagers that enabled further computations started with the silicon Retina, a biologically inspired neural systems that originated from Carver Mead's group [3, 4, 7]. These processors modeled the edge enhancement properties in the early retina processing based upon photodiodes and phototransistors that naturally occur in a silicon CMOS process. Future designs improved so to be usable in systems at high density levels [6, 7, 8] and for high performance [5]. Typically, because of the large pixel size and research costs, these circuits are rarely built for large image processing, but are typically focused at machine vision tasks where the required pixel count is one to two orders of magnitude less. From these retina chips, several higher level processing ICs have been built to investigate stereo processing [17, 18], com-

munication architectures for action potentials [9], attention computations, and motion [10, 11, 12, 13, 14, 15, 16]. Typically, because of the large pixel size and research costs, these circuits rarely compute on large images, but are typically focused at machine vision tasks where the required pixel count is one to two orders of magnitude less.

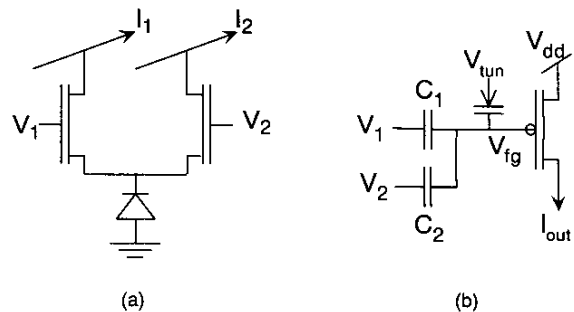
CMOS imagers took a related route to the silicon retina models. These approaches, typically credited as started by Fossum, et. al, [19, 20, 21, 22, 23, 24, 25, 26, 27] worked with photodiode based arrays with minimal circuitry in the pixel, resulting in large imaging arrays, and therefore a technology viable for digital cameras and is a platform for more sophisticated computations. To characterize the spatial efficiency of a pixel, the concept of fill-factor, which equals the ratio of image sensor area over the pixel area, is defined; the larger the fill-factor, the larger the resulting imager that can be built. Typical CMOS imagers have fill factors from 50% to 30%; typical focal plane imagers have fill factors around 1% to 4%.

Our transform imager cell performs computation at the pixel plane, but still holds to a fill factor greater than 40%, and allows for retina and advanced biological-type processing. Therefore, we have the best of both worlds in a single architecture. This imager is capable of programmable matrix operations on the image, where we can represent the image as either a full matrix or using block matrix operations. The pixel for the Transform imager has a fill factor greater than 40%, which allows for retina and advanced biological-type processing, in a programable architecture that still preserves the overall high fill-factor of APS imager for the pixels.

This approach is enabled by floating-gate circuits in three ways. First, we can store arbitrary analog waveforms enabling arbitrary matrix image transforms or block image transforms. Second, we can program these waveforms to account for average device mismatch along a column, thereby getting significantly higher image transform quality. Third, we can use floating-gate circuits to compute additional matrix-vector computations. A floating-gate matrix-vector computation block is also known as an Analog Computing Array (ACA) [1]. This system using the output image stream will compute a transposed matrix transform; therefore we can directly compute arbitrary separable matrix transforms. Some examples of this processing include image filtering, computing spatial derivatives, and 2D spatial transforms, like 2D DCT and 2D DST. In particular, one could compute *clean* derivatives by incorporating smoothing into a derivative kernel.

## 2. TRANSFORM IMAGER AND PIXEL ELEMENT

This section will describe the first block of this architecture, the basic Transform imager. We will discuss the basic processor structure of the computation (multiplication) of the sensor signal in each Pixel. This approach could in-

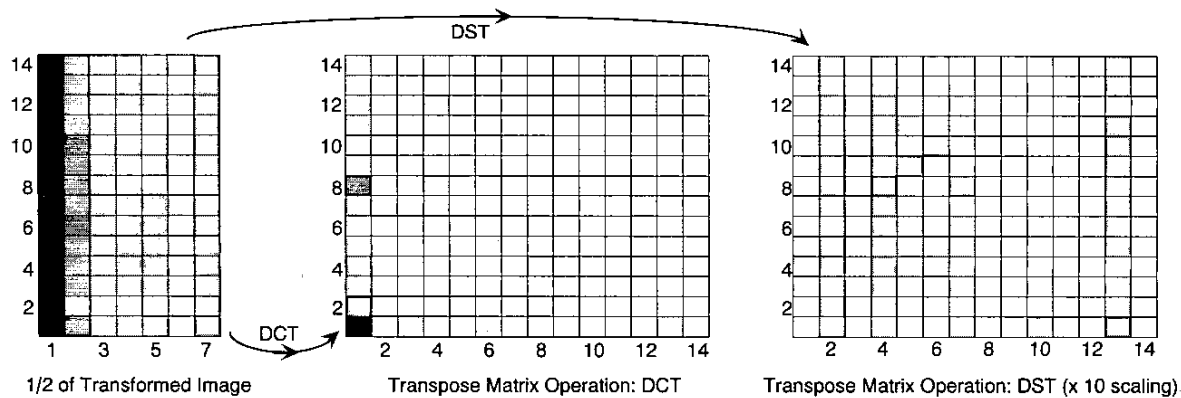


**Fig. 2.** Key circuit elements for the Transform imager technology. (a) **Pixel element:** To multiply the transduced photodiode current by incoming basis functions, we use a differential pair to modulate a fraction of the sensor current through the transistors. For sufficiently small differential input voltages, we get a linear multiplication, as illustrated in the resulting experimental data. The simplicity of the pixel circuit results in fill factors competitive with APS imagers. (b) **Floating-gate transistor:** This circuit can store a current based upon the charge at the floating-gate node. Therefore we use this element to store the basis functions for the Transform imagers. This circuit can also be used as a transistor, and when operating with subthreshold currents, this transistor computes a product of the input voltage with the stored current. Therefore, we use this element in the matrix-vector multiplication memory arrays.

clude more advanced image sensors elements/circuits with a corresponding modification to the resulting fill factor. We present experimental data from a small  $14 \times 14$  image block, requiring roughly  $150\mu\text{m} \times 200\mu\text{m}$  for the array in  $0.5\mu\text{m}$ . We will present results from a signal pixel, the resulting computation, and effect of mismatch and offsets through this circuit. We will discuss the overall computation using a  $14 \times 14$  pixel array in the context of DST and DCT transforms. The results can be extended to arbitrary matrix transforms.

### 2.1. Basic Pixel Element

Each Pixel is composed of a photodiode sensor element and an analog multiplier. Figure 2a shows that the circuit element for this multiplication is an nFET differential pair. For the differential pair operating with subthreshold bias currents (which should always be the case due to the low-level image sensor currents), we can express the differential output current when the circuit is in its linear range as the product of the sensor current and the applied differential voltage. Each pixel could be directly read out by this technique, since a column scan is equivalent to multiplication by a digital value moving by one position for each step ( $\tanh(x) \approx 1$  or  $-1$  for large  $x$  magnitudes). A single pixel could include more advanced image sensors elements / circuits with a corresponding modification to the resulting fill factor. Offsets in differential pairs are important for most analog design problems, and is no exception for this imager. We present elsewhere that this imager concept is insensitive to offsets in the differential pair if the signal swing is held in the device's linear range, even for small transistor sizes [2]. In applications where very high performance (and therefore



**Fig. 3.** Experimental data from a 14 x 14 test imager. We present one half of the output image after transforming the image (uniform illumination) using sine waves. The output image is symmetric; therefore we have output only the first half. Sampled at integer points. **DCT Transform:** Result of an additional Cosine transform on initial sine-transformed imager data. We nearly get the ideal impulse function at (0,0) position, as predicted by taking a 2D Cosine transform of an image of uniform illumination. **DST Transform:** Result of an additional Sine transform on initial sine-transformed imager data. The plot of this Sine transform multiplied by a factor of 10 in comparison with the Cosine transform; without the scaling factor (x 10), the image would look nearly white. We get nearly zero matrix as we would expect for an input image of uniform illumination.

nearly zero offsets) is required, one could use floating-gate tuning techniques for differential pairs [27], with the accompanying decreases in fill factor. Our measurements show that a single pixel element shows little change from dc to 100Hz for typical fluorescent lights. This frequency response will be dependent upon the incoming light levels; we finally see a corner frequency at 30Hz for 4 orders of magnitude lower light intensity from average room light. From these measurements, we expect sufficient bandwidth for a 1024 x 1024 imager performing full matrix operations at 60Hz image rate.

## 2.2. Imager Data

This Transform imager can compute arbitrary separable matrix transforms. We perform separable matrix transforms as

$$\mathbf{Y} = \mathbf{A}^T \mathbf{P} \mathbf{B} \quad (1)$$

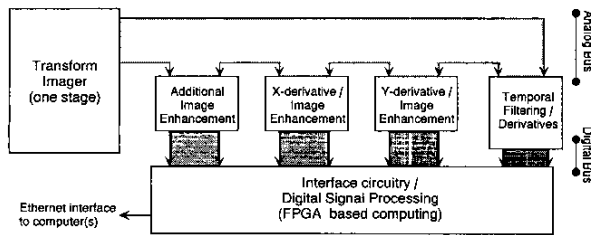
where  $\mathbf{P}$  is the row and column array of pixels,  $\mathbf{Y}$  is the computed output image array,  $\mathbf{A}$  is a transform matrix corresponding to the transform on the image plane by the basis functions, and  $\mathbf{B}$  is a transform matrix corresponding to the floating-gate enabled transform after the image plane. The values of  $\mathbf{A}$  are stored in an analog floating-gate array typically on the imager IC, and applied to the pixel column inputs. Further, if the input waveforms are continuous, then the result is a continuous waveform, resulting in added computational options. For example, the choice of output signal sampling will result in different discrete-time inspired computations with an identical setup. For this paper we will concentrate on computing a DST / DCT like transforms of the image as a representative of possible matrix computations. To characterize this imager, we will compute these transforms for uniform illumination; the ideal DST would be all zeros, and the ideal DCT would be an impulse at po-

sition (1,1).

We present experimental data from a small 14x14 image block, requiring roughly  $150\mu\text{m} \times 200\mu\text{m}$  for the array in  $0.5\mu\text{m}$ . Figure 3 shows the results of DST/DCT type transforms on a uniformly illuminated image. We input sine waves of integer frequencies and obtained the first image result by sampling at  $\pi/2$  phase of the primary harmonic (This transform is symmetric, so we show only the first half of the output waveform). From the resulting waveforms (not sampled waveforms), we computed the second matrix transform using DST coefficients and then for DCT coefficients. We see some distortion in the transformed images, which correlates well to harmonic distortion from the differential pairs. Since the input patterns are fixed, the effect of harmonic distortion is fixed and appears as additional spatial (smoothing) filter. In practice we can account for this additional linear spatial filter, by modifying the matrix transform coefficients to account for this filter. In the same process, we can scale to a  $128 \times 128$  imager with matrix processing for  $16 \times 16$  block transforms in an area of  $4\text{mm}^2$ .

## 3. IMAGE PROCESSING ARCHITECTURE BASED ON THE TRANSFORM IMAGER

The transform imager architecture is both modular and programmable making it ideal for image data-flow computations. This architecture's scalability makes it feasible to compute large-scale, digital-camera resolution images. Furthermore, the image processing architecture computes on the image plane allowing for data reduction that is compatible with machine vision and biological modeling. The sensor can be used to subsample the incoming data if desired, or if the resulting system can handle the data rate, can be passed on so that easier refinement could occur further up the processing chain, either by additional analog circuitry,



**Fig. 4.** Block diagram of a motion (optical-flow) computation system. This system computes the two spatial derivatives ( $x$  and  $y$ ), a temporal derivative, and a filtered version of the original image. Computing gradients in  $x$ ,  $y$ , and time are necessary for optical flow computation; we can make fairly smooth derivative operations in the spatial coordinates by combining a smoothing filter with the derivative kernel over a moderate window size ( $16 \times 16$  or greater). The time derivative is computed by subtracting two (or more) successive frames, which requires a sample and hold for each image frame.

or additional digital processing.

This architecture is modular because the output data flow is a sequence of columns from an image; this image is either from a set of sensors or the output of some amount of signal processing. We can have multiple image processing steps, where each intermediate result can be acquired by the controlling digital system for higher levels of processing. Further, the outputs are continuous waveforms, allowing for time-domain filters to be used to obtain spatial responses and image interpolation.

In the following subsections, we will address several key features of this architecture, as well as design issues associated with this architecture. We will describe some basic transforms, and interfacing issues in the following three subsections. In the last two subsections, we will apply these concepts to two practical case studies.

### 3.1. Additional Matrix Image Transforms

Because the data flows through each processor a column or row at a time, we can perform matrix operations (by image matrix) on an image by a sequence of matrix-vector computations on the image flow. We can perform matrix-vector computations using our existing Analog Computing Array (ACA) technology based upon floating-gate circuits [1], which is related to dense implementations of neural networks. Further, any of these matrix transforms also transposes the image; therefore, in two matrix-vector operations, we can compute any separable matrix transforms. For example, we could perform 2D DCTs, image smoothing, edge enhancement, and differentiation. In particular, one could compute clean derivatives by incorporating smoothing into a derivative kernel. Finally, we can get multiple parallel results, since all of the matrix transforms could operate on the same image flow. One could envision using this approach for real-time image compression based upon DCT transforms and/or Vector Quantization,

### 3.2. Temporal filtering

One interesting question with this flow model is how to perform temporal filtering. We could either build the filters directly into the pixel, which would result in much larger pixels and greatly increase the system cost for a given resolution, or we could store a delayed version of the transformed image. This approach requires a temporary storage array for currents or voltages for each delay; therefore one would be practically limited by the number of delays in building temporal filters. Our approach is to build a set of current sample-and-hold elements into an array that could be used for temporal filters; one can build dynamic current sources that can store their current at reasonable accuracy for seconds, particularly with on-chip compensation of leakage through MOSFET switches. Temporal filters should be used sparingly, or after spatial compression, due to the number of required sample-and-hold elements.

### 3.3. Interfacing between blocks

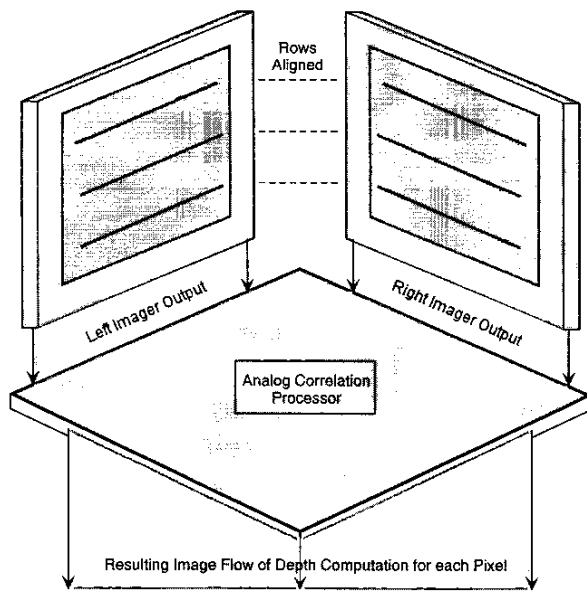
In practice, one must also consider the interface between computation blocks. For a  $1024 \times 1024$  imager computing at a 60Hz image rate requires a parallel data rate (1024 signals) of 120kHz. If two blocks are adjacent on the same IC, then this data rate is trivial to accommodate. If these signals are being passed between chips, requires over 100M analog samples per second, which is a more challenging specification. This rate is similar to simply reading out pixels from any standard CMOS array. Each pixel could be directly read out in a transform imager, since a column scan is equivalent to multiplication by a digital value moving by one position for each step. In general, this issue is significant when interfacing to a digital system, since multiple "images" could be transmitted to the controlling digital system.

### 3.4. Case I: Preprocessing for Optical Flow

Figure 4 shows the block diagram using these imagers and additional matrix computations as a front-end processor to compute optical flow. One must compute separate spatial (for  $x$  and  $y$  directions) and temporal differencing blocks. The spatial differencing block are either matrix transforms with a derivative kernel, or continuous-time derivative circuits. The temporal differencing block requires a set of current sample-and-hold elements are built into an array that could be used for computing temporal derivatives of this image. One could imagine using these outputs that compute optical flow that could be used by digital processing to compute global displacement estimation, target tracking, and time-to-contact computations,

### 3.5. Case II: Depth from Stereo

Figure 5 shows a block diagram using these transform imagers to compute depth from two imagers (Stereo image processing). Since the two imagers have pixels aligned



**Fig. 5.** Picture using two transform imagers to compute stereo computation. Because each row is aligned with another row on the second imager along the same axis, finding depth for each pixel is a row by row operation, which maps directly into the transform imager's output. The inputs to the stereo processing could be outputs from two imagers, or any symmetric processing starting from two imagers. The resulting computation is similar to Mahowald's one-dimensional stereo imager design [18], although a variety of algorithms can be computed using this architecture.

along one axis, computing depth from stereo only requires computing depth from a row by row basis. In our architecture, a row by row basis is instantaneously computing this function based upon successive outputs from the two transform imagers. The resulting computation is similar to Mahowald's one-dimensional stereo imager design [18], although a variety of algorithms can be computed using this architecture. Further, one could also imagine similar algorithms to compute three-dimensional motion enabled by two or multiple imagers.

#### 4. CONCLUSION

We introduced our Transform Imager Technology and Architecture. This approach allows for Retina and higher-level bio-inspired computation in a programmable architecture that still possesses similar high-fill factor pixels of APS imagers. This imager is capable of programmable matrix operations on the image, where we can represent the image as either a full matrix or using block matrix operations. The core imager performs computation at the pixel plane, but still holds to a fill factor greater than 40 percent. Each pixel is composed of a photodiode sensor element and a multiplier. Further, the resulting data-flow architecture directly allows computation of spatial transforms, motion computations, and stereo computations, in a straightforward on-chip or multi-chip architecture.

#### 5. ACKNOWLEDGEMENTS

This work was partially supported by grants National Science Foundation ( CISE-1068549, ECS (CAREER): 0093915, ECS-9988905).

#### 6. REFERENCES

- [1] M. Kucic, P. Hasler, J. Dugger, and D. Anderson, "Programmable and Adaptive Analog Filters using Arrays of Floating-Gate Circuits," *IEEE Advanced Research in VLSI*, Salt Lake City, UT, March 2001.
- [2] Paul Hasler, Abhishek Bandyopadhyay, Paul Smith, "A Matrix transform imager allowing high-fill factor," *International Symposium on Circuits and Systems*, Phoenix, May 2002.
- [3] C. Mead, *Analog VLSI and neural systems*, Addison-Wesley, Reading, Massachusetts, 1989.
- [4] M. Mahowald and C. Mead, "The silicon retina," *Scientific American*, vol. 264, no. 5, 1991, pp. 76-82.
- [5] T. Delbruck, "An electronic photoreceptor sensitive to small changes in intensity," in D.S. Touretzky, *Advances in Neural Information Processing Systems 1*, Morgan Kaufman, 1988, pp. 720-727.
- [6] K.A. Boahen and A.G. Andreou, "A contrast sensitive silicon retina with reciprocal synapses," *Advances in Neural Information Processing Systems 4*, 1992, pp. 762-772.
- [7] A.G. Andreou and K.A. Boahen, "A 590,000 transistor 48,000 pixel, contrast sensitive, edge enhancing, CMOS imager-silicon retina," in *Advanced Research in VLSI*, pp. 225-240, 1995.
- [8] K. Boahen, "The retinomorph approach: pixel-parallel adaptive amplification, filtering, and quantization," *Analog Integrated Circuits and Signal Processing*, vol.13, no.1-2, May-June 1997, pp.53-68.
- [9] K. Boahen, "A Throughput-On-Demand Address-Event Transmitter for Neuromorphic Chips". *Advanced Research in VLSI*, Atlanta, GA, 1999, pp. 72-86.
- [10] J. Tanner and C. Mead, "An integrated analog optical motion sensor," . In R.W. Brodersen and H.S. Moscovitz, editor, *VLSI Signal Processing II*, pp. 59-87. IEEE, New York, 1988.
- [11] T. Delbruck, "Silicon Retina with correlation-based velocity-tuned pixels," *IEEE Transactions on Neural Networks*, vol. 4, no. 3, 1993, pp. 529-541.
- [12] T. Delbruck and C.A. Mead, "Time-derivative adaptive silicon photoreceptor array," *Proc. SPIE, Infrared sensors: Detectors, Electronics, and Signal Processing*, Vol. 1541, pp. 92-99, 1991.
- [13] R. Sarpeshkar, W. Bair, and C. Koch, "Visual motion computation in analog VLSI using pulses," . In S Hanson, J Cowan and C Giles, editor, *Advances in Neural Information Processing Systems 5*, Morgan Kaufman, San Mateo, CA, 1993, pp. 781-788.
- [14] C. M. Higgins and C. Koch, "A Modular Multi-Chip Neuromorphic Architecture for Real-Time Visual Motion Processing," *Analog Integrated Circuits and Signal Processing*, vol. 24, no. 3, pp 195-211, September 2000.
- [15] R.R. Harrison and C. Koch, "A robust analog VLSI Reichardt motion sensor," *Analog Integrated Circuits and Signal Processing*, 24, 2000, pp. 213-229.
- [16] R.R. Harrison and C. Koch, "An analog VLSI implementation of a visual interneuron: enhanced sensory processing through biophysical modeling," *International Journal of Neural Systems*, vol. 9, 1999, pp. 391-395.
- [17] M. Mahowald, *An Analog VLSI Stereoscopic Vision System*, Kluwer Academic Publishers, Boston, MA, 1994.
- [18] M. Mahowald, "Analog VLSI chip for stereocorrespondence," *IEEE Symposium on Circuits and Systems*, Vol. 6, 1994, pp. 347-350.
- [19] O. Yadid-Pecht, R. Ginosar, and Y.S.Diamond, "A random access photodiode array for intelligent image capture," *IEEE Transactions on Electron Devices*, 1991, p.1772-1780.
- [20] M. Kyomasu, "A new MOS Imager using photodiode as current source," *IEEE Journal of Solid-State Circuits*, 1991, p. 1116.
- [21] E.R. Fossum, "CMOS image sensors: electronic camera on a chip," *International Electron Devices Meeting*, 1995, pp. 17-25.
- [22] E.R. Fossum, "CMOS image sensors: electronic camera-on-a-chip," *IEEE Transactions on Electron Devices*, Vol. 44, No. 10, Oct. 1997, pp. 1689-1698.
- [23] O. Yadid-Pecht, O. and E.R. Fossum, "Wide intrascene dynamic range CMOS APS using dual sampling," *IEEE Transactions on Electron Devices*, Vol. 44, No. 10, Oct. 1997, pp. 1721-1723.
- [24] E.R. Fossum, "Digital camera system on a chip," *IEEE Micro*, Vol. 18, No. 3, May 1998, pp. 8 -15.

- [25] Cho Kwang-Bo, A. Krymski, E.R. Fossum, "A 1.2 V micropower CMOS active pixel image sensor for portable applications," *International Solid-State Circuits Conference*, 2000, pp. 114 -115.
- [26] R. Etienne-Cummings, Z.K. Kalayjian, C. Donghui, "A programmable focal-plane MIMD image processor chip," *IEEE Journal of Solid-State Circuits*, Vol. 36, No. 1, Jan. 2001, pp. 64 -73.
- [27] M. Cohen and G. Cauwenbergs, "Floating-gate adaptation for focal-plane online nonuniformity correction," *IEEE Transactions on Circuits and Systems*, vol. 48, no. 1, January 2001, pp. 83-89.
- [28] Paul Hasler and Jeff Dugger, "Correlation Learning Rule in Floating-Gate pFET Synapses," *IEEE Transactions on Circuits and Systems*, vol. 48, no. 1, January 2001.