



Using Analog VLSI Image Processors for Spatial Filtering and Edge Detection

Computational Sensors Corporation
211 N. Calle Cesar Chavez, Suite 203
Santa Barbara, CA 93103

Abstract

We show how analog VLSI image processors can be used to perform important image processing operations such as spatial filtering and edge detection. Examples are demonstrated, analyzed and comparisons made to conventional digital image processing techniques. Performance, power requirements and future technology trends are discussed.

Introduction

For the last three decades, image-processing systems have been steadily migrating towards using more and more digital technology and digital signal processing methods. In many systems that are operating today, the only analog devices remaining are the initial input optics. The performance bottleneck in most *state of the art* imaging systems is the speed of the digital signal processor(s). By using analog VLSI image processors for some common image processing operations, many performance bottlenecks can be eliminated resulting in dramatically increased throughput while at the same time reducing both cost and power demands.

Spatial Filtering

Consider the spatial filtering of an $m \times n$ image $I_{i,j}$ with $i = 0, 1, \dots, m-1$ and $j = 0, 1, \dots, n-1$. Spatial filtering requires the convolution of an input image $I_{i,j}$ with a spatial filter $S_{i,j}$. The filtered output image $I'_{i,j}$ is computed by:

$$I'_{i,j} = \sum_{k=0}^{m-1} \sum_{l=0}^{n-1} S_{i-k,j-l} I_{k,l} \quad (1)$$

Using the discrete Fourier transform (DFT) this may be computed as:

$$I' = (DFT)^{-1} (DFT(S)DFT(I)). \quad (2)$$

Assuming that both m and n are integral powers of two, each time I' is computed (assuming that $DFT(S)$ is pre-computed) requires approximately (using the well known FFT algorithm[1]) $5mn \log_2(mn)$ floating point operations. For images of size 1024×1024 this requires $\approx 5 \times 2^{20} \log_2(2^{20}) = 100 \times 2^{20} = 104,857,600$ floating point operations. On a processor that can achieve 10^9 floating-point operations per second (a one GigaFlop processor) this operation would take about 100 msec. In this case, the



frame rate would be limited to 10 frames each second. For images of size 2048×2048 the frame rate for a one GigaFlop processor is limited to about 2.2 frames each second.

Analog VLSI Filtering

Analog VLSI image processors[2] [3] , compute a class of convolution operations directly in hardware. Input images are directly represented by spatially distributed charges, and are interconnected through a resistive layer. The degree of spatial averaging (blurring) is completely determined by the charge diffusive character of the resistive layer.

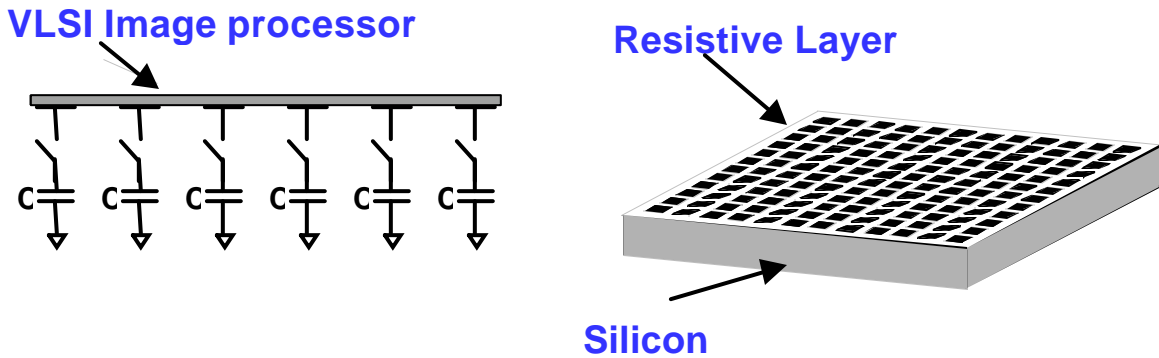


Figure 1 One-dimensional schematic of an analog VLSI image processor. The physical device is two-dimensional.

The time delay before the diffusion process is sampled determines the amount of spatial blurring that is achieved. In reference [3] the time delay for spatially averaging over 10 charge islands was about 0.08 milliseconds. For larger image arrays, the effective diffusion constant can be increased with the number of image pixels yielding fractional image blur times that are independent of image size. Together these properties permit size independent image-processing rates above 1,000 frames per second.

Typically, an isotropic analog VLSI image processor performs a Gaussian convolution with an input image. The impulse response to a point source image is shown in the figure below:

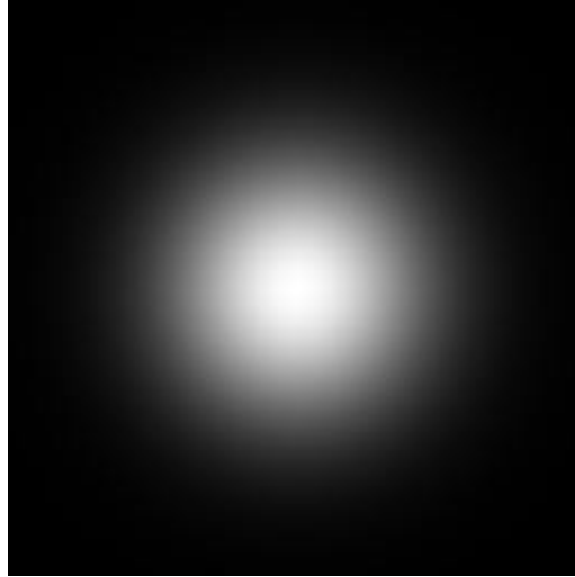


Figure 2 The impulse response function for an isotropic analog image processor is a **Gaussian blurring function**.

Performance Comparison

The following chart below compares convolution times for both a 1-and 10 GFlop digital processor with that of an analog VLSI image processor with a blur time of less than one millisecond[3] for several different image sizes.

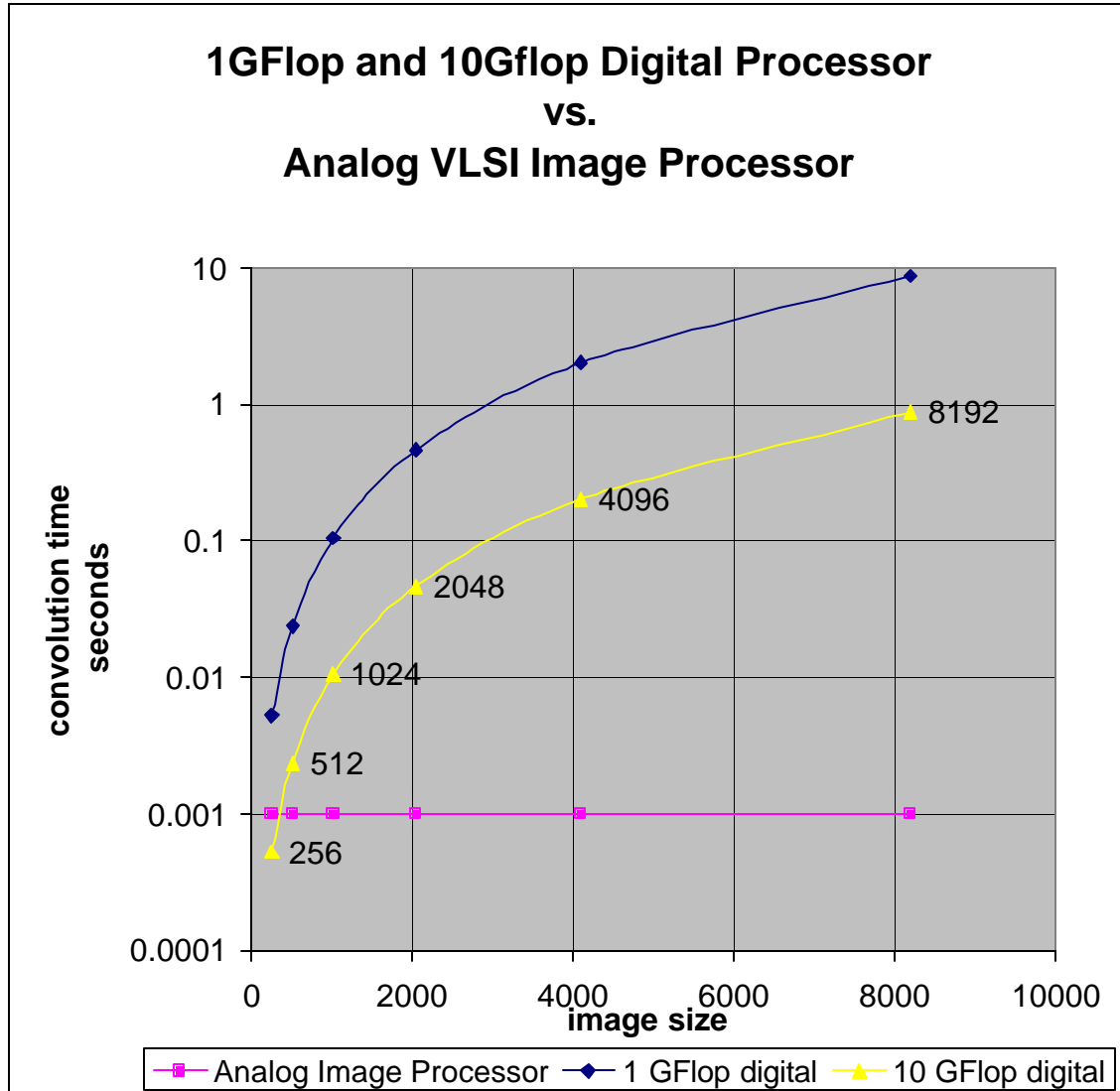


Figure 3 Convolution times for various image sizes comparing a 1GFlop and 10GFlop digital processors with an analog VLSI image processor.

We see from this chart that the time required to perform the image convolution using an analog VLSI image processor is independent of the image size. The convolution time for a digital signal processor grows proportional to (image size) x log(image size).

Edge Detection

Since edges consist of mainly high spatial frequencies, they can be detected using high pass frequency filters in the Fourier domain. As was shown in [3] the frequency space modulation transfer function (MTF) for an isotropic analog (one dimensional) image processor is given by:



$$MTF(k, t) = \exp\left\{-4 \frac{t}{RC} \sin(k/2)^2\right\} \quad (3)$$

By combining two such functions one can form a class of edge detectors:

$$edge(k) = MTF(k, t_1)(1 - MTF(k, t_2)) = MTF(k, t_1) - MTF(k, t_1 + t_2). \quad (4)$$

Output differencing of two analog VLSI image processors can perform this edge detection operation. The figure below shows the original image and the filtered (edge detected) output.



Figure 4: Original image and edge detected Image. In the analog VLSI image processor, the difference between two output images having different blur times performs edge detection.

Power Considerations

The approximate number of gates required for an analog VLSI image processor of size $m \times n$ is approximately $7mn$ transistors[3]. Today's (September, 2001) VLSI transistor densities will easily permit the manufacture of image arrays as large as 3000 x 2000 with a power consumption of approximately 3 Watts. To digitally filter an array of this size will require about 675 Million floating-point operations necessitating computational rates of 40 GFlops to sustain 60 frames per second throughput. A typical 1 GFlop digital signal processor consumes in excess of 2 Watts, therefore an 40 Gflop system will consume in excess of 80 Watts of power.

The power consumption of the 320 x 240 analog VLSI image processor described in is approximately 100mW. In the chart below we have projected the power requirements for both digital and analog VLSI image processing for different image sizes at 60 frames per

