High-Speed Architectures for Digital Image Processing

ANASTASIOS N. VENETSANOPOULOS, SENIOR MEMBER, IEEE, KICH M. TY, STUDENT MEMBER, IEEE, AND ALEXANDER C. P. LOUI, STUDENT MEMBER, IEEE

Abstract —This paper introduces the problem of and presents some state-of-the-art approaches for high-speed digital image processing. An architecture based on distributed arithmetic, which eliminates the use of multipliers, is described. A minimum-cycle-time filter architecture, which features a high degree of parallelism and pipelining, is shown to have a throughput rate that is independent of the filter order. Furthermore, a new multiprocessing-element architecture is proposed. This leads to a filter structure which can be implemented using identical building blocks. A modular VLSI architecture based on the decomposition of the kernel matrix of a two-dimensional (2-D) transfer function is also presented. In this approach, a general 2-D transfer function is expanded in terms of low-order 2-D polynomials. Each one of these 2-D polynomials is then implemented by a VLSI chip using a bit-sliced technique. In addition, a class of nonlinear 2-D filters based on the extension of one-dimensional (1-D) quadratic digital filters is introduced. It is shown that with the use of matrix decomposition, these 2-D quadratic filters can be implemented using linear filters with some extra operations. Finally, comparisons are made among the different approaches in terms of cycle time, latency, hardware complexity, and modularity.

I. Introduction

THE NEED FOR high-speed digital image processing became evident with the increasing utilization of TV imaging to medical, geophysical, and industrial environments. Many of these applications involve acquisition, processing and display in fractions of a second. This paper will describe the problems and some approaches taken to achieve high-speed digital image processing. In particular, different high-speed architectures characterized by parallelism, pipelining, modularity, and reduction of cycle time will be presented and compared. Table I summarizes the importance of speed in a variety of image processing applications [1]–[5].

In this paper, high-speed image processing is defined as the processing of images in real-time or near real-time depending on the applications. The term "real-time image processing" is defined as "the processing of images at a speed, such that the data rate of the processed images is the same as that of the input images" [6]. If we consider an image of size $M \times N$ pixels and a TV scan rate of L images, the input data rate R is $M \times N \times L$ pixels/s. With a display size of 256×256 pixels, this implies a serial data stream at the rate of 1.97 Mpixels/s or one pixel every 508 ns. The corresponding values for a 512×512 pixel image are 7.86 Mpixels/s and 127 ns, respectively. By "near real-time," we usually mean that the image can be processed at a rate which is 10–100 times faster than those achieved with traditional sequential processors [7].

During recent years, a number of approaches have been considered to achieve high-speed image processing. These include array processors, VLSI architectures, and residue arithmetic. Array processors are well suited to perform typical image-processing functions. The use of array processors increases the total system throughput in two ways. First, the processing elements (memories, adders, etc.) are usually faster than those of a general-purpose processor. In addition, they run in parallel and/or may be pipelined, for further increasing the processing speed [8]. By making use of recent VLSI architectures, a set of VLSI building blocks (chips) which would meet the needs of a wide spectrum of algorithms can be designed. This approach has been considered in [9] and other recent contributions [10]. A programmable read only memories (PROM's) implementation of a two-dimensional (2-D) 5×5 matrix convolution filter using residue arithmetic was reported in [11]. This filter structure permits easy pipeline design and the inherent modular structure of residue arithmetic minimizes the design overhead. Special purpose architectures for residue arithmetic 2-D convolutions were also reported in [12].

A number of new architectures capable of achieving real-time or near real-time image processing will be presented in this paper. These architectures will be compared in terms of "cycle time" and "latency." Cycle time is defined as the time between two consecutive input samples, i.e., $1/R$. Latency is defined as the time interval separating the appearance of an input sample on the input port from the appearance of the corresponding output sample at the output port. In Section II, approaches based on distributed arithmetic [13] will be discussed. Recursive filters implemented by this method involve only memory fetches and additions. In particular, the hardware implementation of a second-order 2-D distributed filter will be presented. Section III describes a minimum-cycle-time filter architecture, which has a data throughput rate independent of the order of the filter [14]. Such an approach is different from other implementation schemes, in the sense that "a maximum number of arithmetic operations are performed in one clock cycle." In Section IV, a new

Manuscript received February 18, 1987.
The authors are with the Department of Electrical Engineering, University of Toronto, Toronto, Ontario, Canada M5S 1A4.
IEEE Log Number 8714911.

TABLE I
IMPORTANCE OF SPEED IN A VARIETY OF IMAGE-PROCESSING APPLICATIONS

Applications		Speed		
		Real-time	Near Real-time	Non Real-time
Biomedical	Angiocardiography	***	***	
	Blood-circulation	***	*	
	Radiology	***	***	*
	Tomography	***	***	*
Remote Sensing	Weather	*	***	***
	LANSAT	**	***	***
	Seismic Modelling	**	***	***
Computer-Aided-Manufacturing	Machine vision	***	**	
	Quality control	***	**	
	NDT	**	***	*
Commerical	Broadcast TV	***		
	Image transmission	*	***	
	Video conferencing	***		
Computer Graphics		**	***	
Reconnaissance		***	***	*

*** Critical importance
** Moderate importance
* Slight importance
Unimportance

multiprocessing-element architecture will be developed for the realization of 2-D recursive digital filters for high-speed image processing. The new architecture is developed by applying the "Divide and Conquer" [15] algorithm to the 2-D kernel matrix. Section V describes a modular VLSI architecture based on the decomposition of the kernel matrix of a 2-D transfer function. It has been shown that a general 2-D real rational transfer function can be expanded in term of low-order 2-D polynomials [16]. In this approach, each one of these 2-D polynomials is implemented in a modular manner by a VLSI chip using a bit-sliced technique [17], [18]. This provides a cost-effective way of implementing many costly and complex algorithms. So far we have considered only linear filter architectures. However, nonlinear filters such as median and homomorphic have been used extensively for noise reduction and image restoration. In Section VI, the implementation of a new class of nonlinear 2-D digital filters, based on matrix decompositions will be presented. These 2-D quadratic filters are an extension of one-dimensional (1-D) quadratic filters [19], [20], which can be represented by the second term of the discrete Volterra series. It is shown that these nonlinear filters can be easily implemented using linear filter architectures.

II. DISTRIBUTED ARITHMETIC ARCHITECTURE

The implementation of 1-D recursive filter structures using distributed arithmetic was first introduced by Peled and Liu [21]. This approach requires storing the finite number of possible outcomes of an arithmetic operation, as well as using them to obtain the next output sample through repeated additions and shifting operations. In this section, an extension of the distributed arithmetic approach to the implementation of high-speed 2-D recursive filters is described. Specifically, an implementation scheme for a second-order section is presented.

A 2-D recursive digital filter is described by the linear difference equation

$$y_{m,n} = \sum_{i=0}^{N_i} \sum_{j=0}^{N_j} a_{i,j} x_{m-i,n-j} - \sum_{k=0}^{N_k} \sum_{\substack{l=0 \\ k+l \neq 0}}^{N_l} b_{k,l} y_{m-k,n-l} \quad (1)$$

where $x_{m,n}$ and $y_{m,n}$ are the input and output image arrays, respectively, and $a_{i,j}$'s and $b_{k,l}$'s are the filter coefficients. For a second-order filter ($N_i = N_j = N_k = N_l = 2$), the direct-form realization requires 17 multiplications, 15 additions, and 1 subtraction for each output sample. Assuming all signals to be bounded by ± 1 and defining the input and output signals in two's complement code, B bits of accuracy including the sign bit, we have

$$x_{m-i,n-j} = \sum_{s=1}^{B-1} x^s_{m-i,n-j} 2^{-s} - x^0_{m-i,n-j} \quad (2a)$$

and

$$y_{m-k,n-l} = \sum_{s=1}^{B-1} y^s_{m-k,n-l} 2^{-s} - y^0_{m-k,n-l} \quad (2b)$$

where $x^s_{m-i,n-j}$ and $y^s_{m-k,n-l}$ are binary variables. Substituting (2) into (1), we rearrange the summations and define two functions

$$F^s_1(x^s_{m,n}, x^s_{m,n-1}, \cdots, x^s_{m-2,n-2}) = a_{00} x^s_{m,n} + a_{01} x^s_{m,n-1} + \cdots + a_{22} x^s_{m-2,n-2} \quad (3a)$$

and

$$F^s_2(y^s_{m,n-1}, y^s_{m,n-2}, \cdots, y^s_{m-2,n-2}) = b_{01} y^s_{m,n-1} + b_{01} y^s_{m,n-2} + \cdots + b_{22} y^s_{m-2,n-2}. \quad (3b)$$

It is possible to write (1), for a second-order filter, in terms of the two functions $F^s_1(\cdot)$ and $F^s_2(\cdot)$ as follows:

$$y_{m,n} = \sum_{s=1}^{B-1} F^s_1(\cdot) 2^{-s} - F^0_1(\cdot) - \sum_{s=1}^{B-1} F^s_2(\cdot) 2^{-s} + F^0_2(\cdot) \quad (4)$$

where $F^s_1(\cdot)$ and $F^s_2(\cdot)$ have a finite number of possible outcomes (2^9 and 2^8, respectively).

The distributed arithmetic realization of (4) consists mainly of four building blocks: 1) mask bit-shifters, 2) memories, 3) summers, and 4) subtractor. A schematic block diagram is shown in Fig. 1. All linear combinations of the nine coefficients $a_{i,j}$'s in (1) are stored in the $512 \times t$ memories. Similarly, all possible combinations of the eight $b_{k,l}$'s are stored in the $256 \times t$ memories.

Consider the implementation shown in Fig. 1 for an image of 512×512 pixels. The coefficients are quantized to 16 bits. All computations are done with $t = 16$ bits of precision and the input and output signals are represented by $B = 8$ bits. Each of the mask bit-shifters consists of two (512×1) shift registers and six single bit shift registers. These can be configured from TRW TDC1006J [22], (256

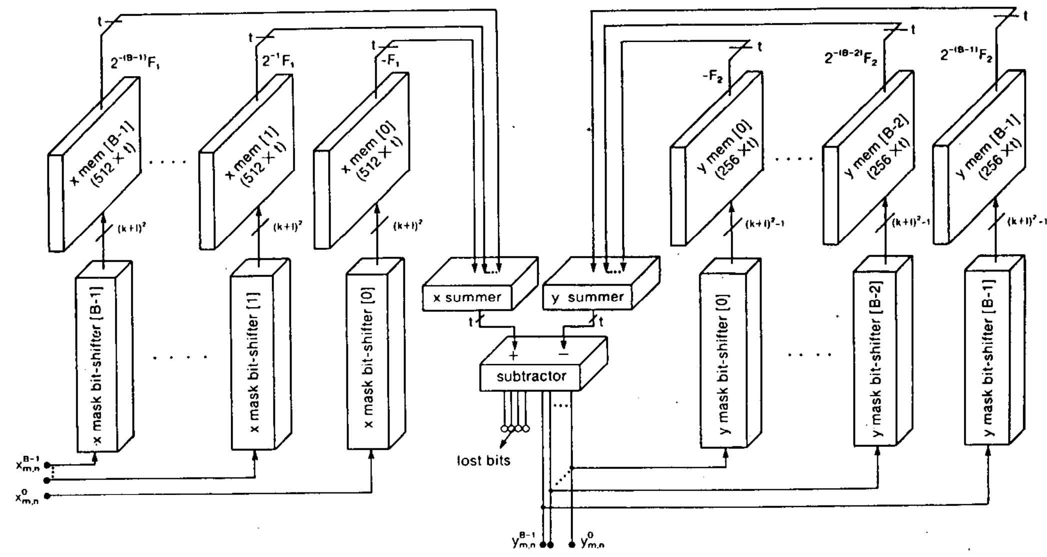


Fig. 1. Schematic block diagram of the distributed arithmetic implementation of a second-order section.

$\times 1$) shift registers and TTL SN74S174, hex D-type flip-flops. The memories can be constructed from AM27S191 (2K$\times$8) PROM's. The summers and subtractor can be configured from TTL 74S181's and TTL 74S182's.

The inherent parallelism of the architecture reduces the data rate to a memory fetch, $\log_2 B+1$ additions, and register's delays (due to recursive part). Using the components mentioned above, the time for a 16-bit addition is 19 ns and that of a memory access is 50 ns [23], [24]. In addition, the minimum set up time and maximum propagation delay of the shift registers are 0 ns and 30 ns, respectively [22]. This gives a maximum cycle time of $(4\times 19)+50+30=156$ ns or a minimum data rate of 6.41 Mpixels/s. This data rate is close to the real-time requirement for a 512×512 pixels image. Higher throughput rates can be attained if faster logic families, such as ECL, are used. With such logic families, the penalty for state-of-the-art performance is increased cost and power consumption.

When the order of the filter increases, the memory requirement of this structure increases as $2B(2^{(K+1)^2})\times t$, where K is the order of the filter. However, the memory address partition, which trades off memory with extra additions, can be used to reduce the amount of memory required. With this modification, the effective memory size decreases while the number of memory addresses remains the same. For example, $2^{16}\times t$ ($K=3$) bits of memory can be implemented using two $2^8\times t$ bits of memory plus one t bits adder. Hence, with an extra addition, the effective memory size is reduced considerably.

A prototype of the distributed arithmetic implementation of a 2-D recursive filter [13], which can process images of size up to 256×256, has been built using mainly TTL components. In this prototype, serial additions, rather than parallel additions, were used in order to reduce the hardware size. The prototype assumes the input and output samples are 8 bits in length, while all the intermediate computations are done in 16-bit precision. The X and Y mask bit shifters are constructed using TTL 74S174 hex D-type flip flops and MOS AM2856 dual 256×1 static shift registers. The memories are configured from 2716 2K$\times$8 EPROM's (Erasable Progammable Read Only Memories). With serial addition, an accumulator-type circuit is required. A photograph of the prototype is shown in Fig. 2. The distributed filter is linked to the host computer (VAX 11/780) via a 6809 microprocessor. The prototype cost approximately \$350 U.S. in 1984 dollars and operates at a speed of 350×10^3 pixels/s.

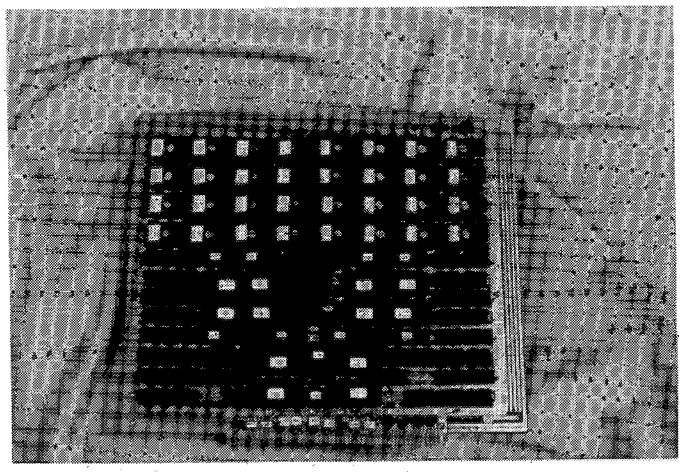


Fig. 2. Photograph of the 2-D second-order distributed filter.

III. Minimum-Cycle-Time (MCT) Filter Architecture

For 1-D filters, a configuration for which the critical path contains no more than one multiplication and one addition has been proposed [25]. The critical path of a digital filter is defined as the longest one among all possible paths from the output of a delay element to the input of the next one. An extension of this idea to 2-D recursive digital filters will be presented.

The digital filter described by (1) has the transfer function

$$H(z_1^{-1},z_2^{-1}) = \frac{N(z_1^{-1},z_2^{-1})}{D(z_1^{-1},z_2^{-1})} = \frac{\sum_{i=0}^{N_i}\sum_{j=0}^{N_j} a_{i,j} z_1^{-i} z_2^{-j}}{1+\sum_{\substack{k=0 \\ k+l\neq 0}}^{N_k}\sum_{l=0}^{N_l} b_{k,l} z_1^{-k} z_2^{-l}} \tag{5}$$

where z_1^{-1} and z_2^{-1} represent a unit column delay and a unit row delay, respectively.

For a 2-D filter, the optimal critical path should contain one more addition than that of the 1-D filter [14]. The extra addition is required for adding the intermediate results from the z_1^{-1} and z_2^{-1} blocks. In order to obtain the desired critical path, the original 2-D transfer function should be modified so that the new transfer function $\tilde{H}(z_1,z_2)$ has the form

$$\tilde{H}(z_1^{-1},z_2^{-1}) = H(z_1^{-1},z_2^{-1}) z_1^{-p} z_2^{-q} \tag{6}$$

with p and q being nonnegative integers. It is desirable to have the minimum possible latency. This can be achieved by setting $p=0$ and $q=1$. Thus, the output of the new filter $\tilde{y}_{m,n}$ is delayed by only one pixel compared with the original filter output $y_{m,n}$ (i.e., $\tilde{y}_{m,n} = y_{m,n-1}$). With the chosen values for p and q, the new equation describing the input and output relationship is

$$\tilde{Y}(z_1^{-1},z_2^{-1}) = z_2^{-1} X(z_1^{-1},z_2^{-1}) \sum_{i=0}^{N_i}\sum_{j=0}^{N_j} a_{i,j} z_1^{-i} z_2^{-j} - \tilde{Y}(z_1^{-1},z_2^{-1}) \sum_{\substack{k=0 \\ k+l\neq 0}}^{N_k}\sum_{l=0}^{N_j} b_{k,l} z_1^{-k} z_2^{-l}. \tag{7}$$

We now propose the new filter structure for the modified 2-D filter transfer function. Let

$$A = X(z_1^{-1},z_2^{-1}) \sum_{\substack{i=0 \\ i+j\neq 0}}^{N_i}\sum_{j=0}^{N_l} a_{i,j} z_1^{-i} z_2^{-j} \qquad B = -Y(z_1^{-1},z_2^{-1}) \sum_{l=2}^{N_k} b_{0,l} z_2^{-(l-1)}$$

$$C = a_{0,0} X(z_1^{-1},z_2^{-1}) \qquad D = A+B+C$$

$$E = -b_{0,1} Y(z_1^{-1},z_2^{-1}) \qquad F = z_2^{-1}(D+E)$$

$$G = -Y(z_1^{-1},z_2^{-1}) \sum_{k=1}^{N_k}\sum_{l=0}^{N_l} b_{k,l} z_1^{-k} z_2^{-l} \qquad I = F+G. \tag{8}$$

The relationships among all these signals of a second-order filter are shown in Fig. 3(a) and (b). It can be easily proven that $\tilde{Y}(z_1^{-1},z_2^{-1}) = I$. With the assumption that a multiplication takes at least twice the amount of time required for an addition, the critical path is the one shown in bold lines and contains only one multiplication and two additions. This critical path is independent of the order of the filter. The new filter has a very regular structure, with identical building blocks. This regularity property provides a simple hardware structure for the implementation of the filter.

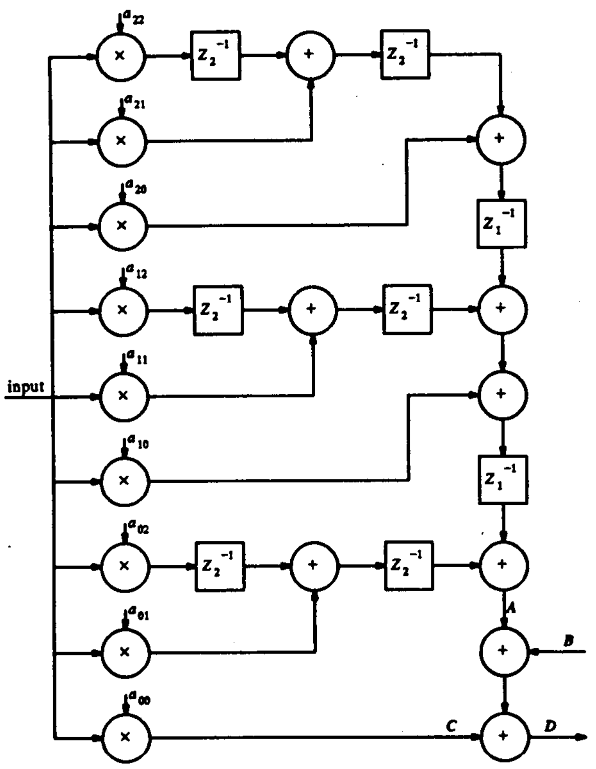
We now outline the hardware of a second-order 2-D recursive digital filter for the filtering of images of sizes up to 512×512 pixels in real-time with a TV scan rate of 30 frames/s. For a reasonable gray level resolution, the input and the final output signals are represented in 8 bits. All intermediate results have 16 bits of accuracy. Multipliers can be eliminated through the use of memories. The memories can be constructed from AM 27S20 (256×4) PROM's, which have a memory access time of 45 ns [26]. The adders are the same as those used in distributed arithmetic implementation.

The z_2^{-1} blocks can be configured from SN74S174 (hex D-type flip-flop), which has a maximum propagation delay of 17 ns and a minimum set up time of 5 ns [23]. The z_1^{-1} blocks, which consist of 16 parallel 512-bit shift registers, can be configured from TRW TDC 1006J. It requires two TDC 1006J chips for the implementation of one 512-bit shift register. Since the sum of the propagation delay and the set up time of SN 74S174 is shorter than that of TDC 1006J, it is desirable to drive the two different shift registers with two different clock signals, one being the delayed version of the other, so that the outputs of the two different shift registers will be available almost simultaneously. From the specifications of the shift registers, the clock signal driving the single bit shift register should be the one driving the 512-bit shift register delayed by 5 to 13 ns.

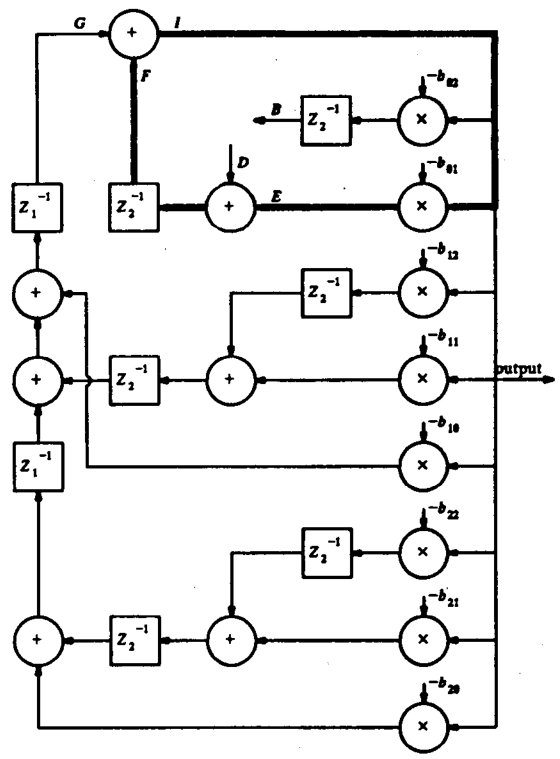
The cycle time of the new filter, built with the above components, consists of one memory access time, two addition times, and the sum of the propagation delay and set up time of the 512-bit shift register. The new filter can process images at a data throughput rate of one pixel every 113 ns (i.e., 2×19+45+30), which is less than the maximum allowable time 127 ns required for real-time processing.

IV. Multiprocessing-Element (MPE) Architecture

The "Divide and Conquer" algorithm [15] is frequently used to solve a complex problem through a recursive method. The motivation of applying this algorithm to image filtering is to provide a general implementation method by the use of simple identical processing elements.



(a)



(b)

Fig. 3. (a) Modified nonrecursive block of a second-order 2-D digital filter. (b) Modified recursive block of a second-order 2-D digital filter.

Consider a FIR filter

$$N(z^{-1}) = [a_0\ a_1\ \cdots\ a_M]\begin{bmatrix}1\\ z^{-1}\\ \vdots\\ z^{-N}\end{bmatrix}. \tag{9}$$

Equation (9) can be rewritten as

$$N(z^{-1}) = \left[a_0\ a_1\left[a_2\ a_3\left[\cdots\left[a_{2\left\lfloor\frac{M-1}{2}\right\rfloor}\ a_{2\left\lfloor\frac{M-1}{2}\right\rfloor+1}\ a_{2\left\lfloor\frac{M-1}{2}\right\rfloor+2}\right]\begin{bmatrix}1\\ z^{-1}\\ z^{-2}\end{bmatrix}\right]\cdots\right]\begin{bmatrix}1\\ z^{-1}\\ z^{-2}\end{bmatrix}\right]\begin{bmatrix}1\\ z^{-1}\\ z^{-2}\end{bmatrix} \tag{10}$$

where

$$a_{2\left\lfloor\frac{M-1}{2}\right\rfloor+2} = \begin{cases}0 & \text{if } M \text{ is odd}\\ a_M & \text{if } M \text{ is even}\end{cases}. \tag{11}$$

As we can see, (10) is a recursive form of $[p\ q\ r]\begin{bmatrix}1\\ z^{-1}\\ z^{-2}\end{bmatrix}$, where p, q, and r are the corresponding filter coefficients. If a processing element (PE) can perform the above operation, the use of multiprocessors can solve the original problem in (9). An all-pole 1-D filter can be implemented with a FIR filter in the feedback path.

The extension of this method to 2-D case will be presented as follows. A 2-D FIR filter can always be written as

$$N(z_1^{-1}, z_2^{-1}) = P_0(z_2^{-1}) + z_1^{-1}P_1(z_2^{-1}) + z_1^{-2}P_2(z_2^{-1}) + \cdots + z_1^{-N_i}P_{N_i}(z_2^{-1}) \tag{12}$$

where $P_m(z_2^{-1})$, $m = 0, 1, \cdots, N_i$, corresponds to the polynomial of z_2^{-1} associated with the factor z_1^{-i}. Thus, $N(z_1^{-1}, z_2^{-1})$ can also be written in the form of (10) as

$$N(z_1^{-1}, z_2^{-1}) = \left[P_0(z_2^{-1})\ P_1(z_2^{-1})\left[P_2(z_2^{-1})\ P_3(z_2^{-1})\left[\cdots\left[P_{2\left\lfloor\frac{M-1}{2}\right\rfloor}(z_2^{-1})\ P_{2\left\lfloor\frac{M-1}{2}\right\rfloor+1}(z_2^{-1})\ P_{2\left\lfloor\frac{M-1}{2}\right\rfloor+2}(z_2^{-1})\right]\right.\right.\right.$$
$$\times\begin{bmatrix}1\\ z_1^{-1}\\ z_1^{-2}\end{bmatrix}\cdots\left.\left.\left.\right]\begin{bmatrix}1\\ z_1^{-1}\\ z_1^{-2}\end{bmatrix}\right]\begin{bmatrix}1\\ z_1^{-1}\\ z_1^{-2}\end{bmatrix}. \tag{13}$$

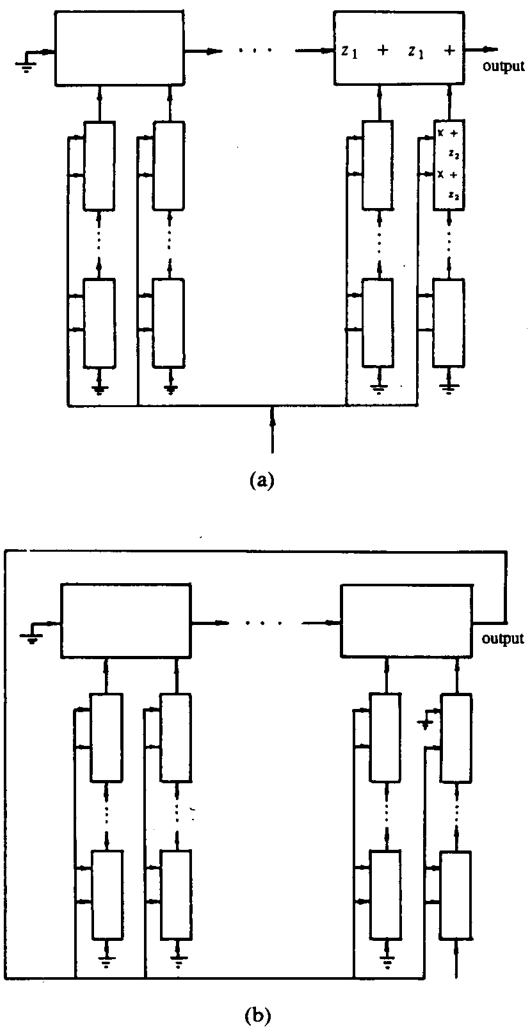


Fig. 4. MPE architecture. (a) Nonrecursive block. (b) Recursive block.

Equation (13) suggests that the 2-D computation can also be evaluated recursively. Similarly, the denominator $D(z_1^{-1}, z_2^{-1})$ of the filter transfer function $H(z_1^{-1}, z_2^{-1})$ can be also written as

$$D(z_1^{-1}, z_2^{-1}) = 1 + D_1(z_1^{-1}, z_2^{-1}) \tag{14}$$

with

$$D_1(z_1^{-1}, z_2^{-1}) = Q_0(z_2^{-1}) + z_1^{-1}Q_1(z_2^{-1}) + z_1^{-2}Q_2(z_2^{-1}) + \cdots + z_1^{-N_k}Q_{N_k}(z_2^{-1}) \tag{15}$$

and $Q_0(z_2^{-1})$ is a polynomial with no constant term. In order to reduce the critical path length, the original transfer function must be modified. Fig. 4(a) and (b) show the MPE architecture for the nonrecursive and recursive blocks, respectively. From Fig. 4(b), the modified transfer function $\tilde{H}(z_1^{-1}, z_2^{-1})$ is

$$\tilde{H}(z_1^{-1}, z_2^{-1}) = H(z_1^{-1}, z_2^{-1}) z_2^{-2\left\lceil \frac{N_l+1}{2} \right\rceil} \tag{16}$$

and the critical path length is one multiplication and four additions. The total latency is $2\left\lceil \dfrac{N_l+1}{2} \right\rceil$ cycle times + 2 addition times + shift register propagation delay. As illustrated, besides having a regular structure with identical PE's, the MPE architecture provides a high throughput rate suitable for high-speed digital image processing.

V. VLSI Architectures by Decomposition

A general 2-D rational transfer function of the form in (5) can be expressed by the terms of the form $(z_1^{-1} - a_{1i})$ and $(z_2^{-1} - a_{2j})$ only, where a_{1i}, a_{2j} are constants and $i = 1, 2, \cdots, \max(N_i, N_k)$, $j = 1, 2, \cdots, \max(N_j, N_l)$ [16]. The numerator polynomial $N(z_1^{-1}, z_2^{-1})$ can be written in the form

$$N(z_1^{-1}, z_2^{-1}) = \boldsymbol{Z}_1^T \boldsymbol{A} \boldsymbol{Z}_2 \tag{17}$$

where

$$\boldsymbol{Z}_1 = \begin{bmatrix} 1 \\ z_1^{-1} \\ \vdots \\ z_1^{-N_i} \end{bmatrix}, \quad \boldsymbol{Z}_2 = \begin{bmatrix} 1 \\ z_2^{-1} \\ \vdots \\ z_2^{-N_j} \end{bmatrix},$$

$$\boldsymbol{A} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N_j} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N_j} \\ \vdots & \vdots & & \vdots \\ a_{N_i,0} & a_{N_i,1} & \cdots & a_{N_i,N_j} \end{bmatrix}. \tag{18}$$

The polynomial in (17) can be written equivalently as

$$N(z_1^{-1}, z_2^{-1}) = \boldsymbol{Z}_1^T \boldsymbol{R} \boldsymbol{S} \boldsymbol{Z}_2 \tag{19}$$

by writing the matrix $\boldsymbol{Q}$ as a product of two other matrices $\boldsymbol{R}, \boldsymbol{S}$. The matrix $\boldsymbol{R}$ might be chosen arbitrarily as a nonsingular $(N_i+1) \times (N_i+1)$ matrix. In this case, the matrix $\boldsymbol{S}$ is determined by

$$\boldsymbol{S} = \boldsymbol{R}^{-1} \boldsymbol{Q} \tag{20}$$

and has the dimension $(N_i+1) \times (N_j+1)$. $\boldsymbol{Z}_1^T \boldsymbol{R}$ is a $1 \times N_i$ matrix, which is a function of the first variable z_1^{-1} and $\boldsymbol{S}\boldsymbol{Z}_2$ is a $N_i \times 1$ matrix, which is a function of the other variable z_2^{-1}. Thus, $N(z_1^{-1}, z_2^{-1})$ can be expressed as a sum of products of first-order terms, each one of which is a function of only one of the two variables. The denominator polynomial can also be decomposed in the same manner. In order to eliminate the complex first-order terms in the decomposition as a result of complex conjugate pairs and to save on hardware, a building block implementing a second-order 2-D FIR filter was chosen [17].

In this section, the VLSI implementation of a second-order first quadrant 2-D digital filter is considered. The filter, referred to as the "Slice," has the following specifications:

(i) it implements either all-pole or all-zero transfer function in 4-bit slices; the wordlength is indefinitely expandable in an expansion chain, the only penalty being the speed of operation,

(ii) it operates on 512×512 pixel images,

(iii) it is easily cascadable in a pipeline to implement more complex functions, with the same throughput as a single Slice, and

(iv) it uses 512×8 bits of external scratchpad memory and a host controller to program the filter coefficients, present input data, and store the results.

The filter was implemented in 5-μm NMOS VLSI technology (available at the University of Toronto) on two chips: the "Datapath" and the "Sequencer."

The Datapath contains all the circuitry to carry out the arithmetic functions and the registers to hold the data and filter coefficients. Since a second-order filter requires an array support of eight values, there are nine (one for the input) 4-bit data registers and nine 4-bit coefficient registers. Multiplications and additions are carried out by the Arithmetic Logic Unit (ALU). The ALU uses a 4-bit adder with internal carry-look-ahead and external fast carry, for word-length expansion. The output of the adder is fed to an accumulator which is internal to the ALU. In addition to data input/output (I/O) for the Slice, the Datapath has I/O port for the scratchpad RAM. As mentioned earlier, the Slice is switchable between all-zero and all-pole filtering, and selection is carried out by setting a control signal on the Datapath. Within the Datapath, the regularity of the filter structure enabled this section to be broken into eight basic blocks. Of these, five were further divided into subblocks, the lowest hierarchical level in the Datapath. It was at this point that the actual circuit design was started. Once the subblocks had been designed, they were simply connected together to form the basic blocks, and these were interconnected to form the Datapath. Due to this design technique, there is very little random logic in the Datapath. This is essential as the implementation of random logic in VLSI is extremely inefficient and time consuming.

The Sequencer is responsible for interfacing with the host and other Slices in a cascade or expansion chain, Datapath coefficient programming, scratchpad RAM management, data blanking control, global processing control, multiply-bit bus control, and Datapath ALU control. The Sequencer was broken into two blocks, a two-output, 18-bit counter, and a large PLA (programmable-logic-array). The PLA generates all the control signals required by the filter. Each of these two blocks were further broken down into subblocks. The circuit design started at this point.

In the Slice, column by column recursion [17] was chosen for its smaller buffer requirements ($2N+2$ output values must be buffered for an $N \times N$ pixel image) and simpler recursion algorithm. Fixed two's complement arithmetic was used because of the simplicity of addition and subtraction, ease of detecting overflows, and correct final output (if the final output is within the output dynamic range) even if intermediate results have overflowed.

IV. Two-Dimensional Quadratic Digital Filter Architecture

In recent years, nonlinear filters have been used extensively in image processing. It is well known that linear filtering techniques are simplier to implement but have the disadvantage that they blur the edges. They also do not perform well in the presence of signal-dependent noise [27]. Examples of nonlinear filters include homomorphic filters for restoration of an image which is subject to multiplicative degradation, and median filters for removing impulse noise. In this section, we propose a new type of 2-D nonlinear filter based on a second-order characteristic. More importantly, it is shown that these filters can be implemented using linear 2-D FIR filters. One application of these nonlinear filters is for texture discrimination. It is shown that the coarseness of the texture is proportional to the spread of the autocorrelation function and, hence, the second-order moment [28].

One way to describe the input and output relationship of a 1-D nonlinear system and whose design requires only a limited amount of knowledge of higher order statistics is to use a discrete Volterra series representation [29]. In particular, the first and second terms of this series describe the linear and quadratic parts of the nonlinear system. The 1-D quadratic digital filter [19] is defined as

$$y_n = \sum_{i=0}^{N} \sum_{j=0}^{N} h_{i,j} x_{n-i} x_{n-j} \tag{21}$$

where x_n and y_n are the input and output sequences and $h_{i,j}$ is the quadratic kernel which can be assumed to be a symmetric function of its indices without loss of generality. The direct implementation of (21) requires a large number of computation. It has been shown that (21) can be implemented using 1-D FIR filters and some extra multipliers [20], [30]. In particular, using the singular-value decomposition on the quadratic kernel $h_{i,j}$ (21) can be rewritten as follows:

$$y_n = \sum_{i=1}^{k} \lambda_i \left[\sum_{j=0}^{N} r_{i,j} x_{n-j} \right]^2, \qquad k \leqslant q \tag{22}$$

where q is the rank of kernel matrix $\{h_{i,j}\}$, and λ_i's are the eigenvalues resulting from the singular value decomposition. Hence, the implementation of second-order nonlinear filters is equivalent to the implementation of 1-D linear filters with some extra operations.

Based on the 1-D quadratic filter of (21), we define a 2-D quadratic digital filter as follows:

$$y_{m,n} = \sum_{i=0}^{N_i} \sum_{j=0}^{N_j} \sum_{k=0}^{N_k} \sum_{l=0}^{N_l} h_{i,j,k,l} x_{m-i,n-j} x_{m-k,n-l} \tag{23}$$

where $x_{m,n}$ and $y_{m,n}$ are the input and output image pixels and $h_{i,j,k,l}$ is the 2-D quadratic kernel. We can assume without loss of generality that $N_i = N_k$ and $N_j = N_l$. The 2-D quadratic kernel can be represented by the following "ordered-pair" matrix:

$$H = \{h_{i,j,k,l}\} = \begin{bmatrix} h_{00,00} & h_{00,01} & \cdots & h_{00,N_iN_j} \\ h_{01,00} & h_{01,01} & \cdots & h_{01,N_iN_j} \\ \vdots & \vdots & & \vdots \\ h_{N_iN_j,00} & h_{N_iN_j,01} & \cdots & h_{N_iN_j,N_iN_j} \end{bmatrix}. \tag{24}$$

It should be obvious that the "ordered-pair" kernel matrix H is a symmetric matrix due to the permutations of the product terms $x_{m-i,n-j}$ and $x_{m-k,n-l}$. Hence, (23) can be rewritten as follows:

$$y_{m,n} = \sum_{i=0}^{N_i} \sum_{j=0}^{N_j} \sum_{k=i}^{N_k} \sum_{l=j}^{N_l} h'_{i,j,k,l} x_{m-i,n-j} x_{m-k,n-l} \tag{25}$$

where

$$h'_{i,j,k,l} = \begin{cases} h_{i,j,k,l} & i=k,\ j=1 \\ 2h_{i,j,k,l} & \text{otherwise} \end{cases}. \tag{26}$$

Equation (25) can be realized as follows:

$$y_{m,n} = \sum_{i=0}^{N_i} \sum_{j=0}^{N_j} w_{i,j,m,n} x_{m-i,n-j} \tag{27}$$

where

$$w_{i,j,m,n} = \sum_{k=i}^{N_k} \sum_{l=j}^{N_l} h'_{i,j,k,l} x_{m-k,n-l}. \tag{28}$$

From (27) and (28), we observe that each output requires the generation of a new set of coefficients $w_{i,j,m,n}$, which will then be used to determine the final output by convolving with the input again. This implies a large number of computations is required especially when the order of the filter increases.

In order to reduce the amount of computations, a matrix decomposition [16] approach on the "ordered-pair" kernel matrix is proposed. In this approach, the "ordered-pair" matrix is viewed as a 2-D matrix. As in the 1-D case, using the singular-value decomposition, the quadratic filter of (23) can be rewritten as

$$y_{m,n} = \sum_{i=1}^{p} \lambda_i \left[v^i_{m,n}\right]^2, \qquad p \leqslant q \tag{29}$$

where

$$v^i_{m,n} = \sum_{k=0}^{N_k} \sum_{l=0}^{N_l} r^i_{k,l} x_{m-k,n-l} \tag{30}$$

q is the rank of the "ordered-pair" matrix of (24), $r^i_{k,l}$ is the element of the decomposed matrix, and λ_i's are the eigenvalues resulting from the singular-value decomposition. Hence, by choosing p smaller than q, the number of coefficients can be reduced. In other words, one can retain only the most significant stages of the decomposition depending on the accuracy required. In addition, with the decomposition, a modular structure as shown in Fig. 5 can be obtained. Therefore, 2-D quadratic filters can be implemented using 2-D FIR filter with only some extra multiplications. This is analogous to the implementation of 1-D quadratic filter described earlier. The 2-D FIR filters of Fig. 5 can be implemented using the various architectures described in previous sections. Alternatively, each of the 2-D FIR filters can be decomposed to further reduce the hardware complexity. Finally, it should be noticed that different types of matrix decompositions can be applied to the "ordered-pair" matrix of (24) to accomplish different objectives such as minimum number of coefficients, etc.

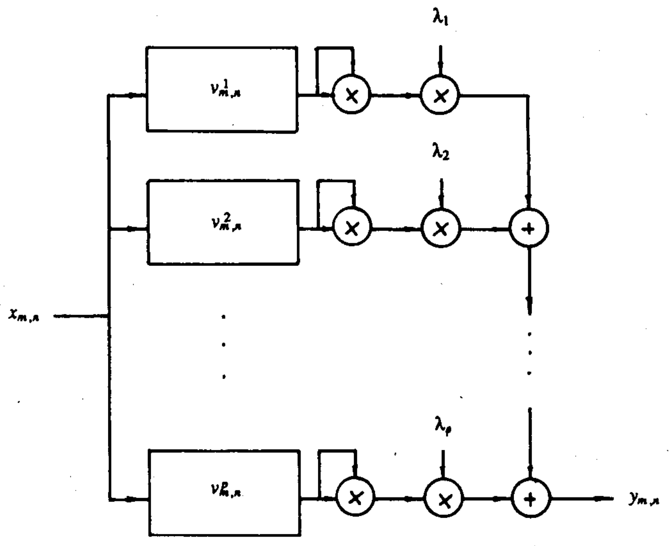


Fig. 5. Matrix decomposition realization of 2-D quadratic digital filter.

TABLE II
COMPARISONS AMONG VARIOUS FILTER ARCHITECTURES

Architectures	Cycle Time	Latency	Hardware Complexity	Modularity	Comments
Distributed Arithmetic Architecture	$\lceil 1+\log_2 B \rceil T_a$ $+T_{ma}+T_s+T_{pd}$	Same as cycle time	Increases almost linearly with total number of coefficients	No	Excellent for low order filters
Minimum-Cycle-Time Architecture	$T_{ma}+2T_a+T_s$ $+T_{pd}$	System clock period $+T_a$ $+T_s+T_{pd}$	Increases linearly with total number of coefficients	No	Cycle time and latency independent of filter order
Multiple-Processing-Element Architecture	T_m+4T_a $+T_s+T_{pd}$	$\lceil (N_l+1)/2 \rceil T_a$ $+2T_a+T_s+T_{pd}$	Increases linearly with total number of coefficients	Yes	Modular high-speed architecture for high order filters
VLSI Architecture by Decomposition	$9T_{ma}+9T_a$ + control delay (for single chip set)	Same as cycle time	Increases linearly with total number of coefficients	Yes	Highly modular architecture for general 2-D filters

T_a - Addition Time
T_{ma} - Memory Access Time
T_m - Multiplication Time
T_s - Shift Register Set Up Time
T_{pd} - Shift Register Propagation Delay Time

VII. CONCLUSIONS

This paper has presented some state-of-the-art architectures for high-speed image processing. High throughput rate is achieved through a high degree of parallelism and pipelining of arithmetic operations. Modular structures are shown to be very suitable for efficient VLSI implementation. Table II provides comparisons among the various architectures in terms of speed, latency, hardware complexity, and modularity. From the comparisons, the following comments are in order.

1) The distributed arithmetic architecture is very suitable for low-order filter. It offers a savings in both cost and power consumption when compared to the direct implementation.
2) The MCT filter architecture has the shortest cycle time among the four architectures. The cycle time and latency are also independent of the filter order.
3) The MPE architecture features a very short cycle time and is very modular. It is suitable for high-order filters.
4) The VLSI architecture by decomposition is very modular and flexible. High-order filters can be implemented by cascading of second-order 2-D filter chip sets.

The implementation of 2-D quadratic digital filters can be simplified by decomposing the "ordered-pair" kernel matrix. The resulting structure is composed of identical stages where each stage consists of a linear 2-D filter and two multipliers.

High-speed image processing and, in particular, real-time image processing is still in its infancy. More research is needed for this field to reach maturity. The advances in microelectronics technology will remain a major factor in the implementation of high-speed image processors.

References

[1] C. Cafforio and F. Rocca, "Tracking moving objects in television images," *Signal Processing*, pp. 133–140, 1979.
[2] R. J. Schalkoff and E. S. McVey, "Algorithm development for real-time automatic video tracking systems," in *Proc. 3rd Int. Computer Software and Application Conf.* (Chicago, IL), Nov. 1979, pp. 504–511.
[3] S. Tsuji, M. Osada, and M. Yachida, "Tracking and segmentation of moving objects in dynamic line images," *IEEE Trans. Pat. Anal. Machine Intel.*, vol. PAMI-2, pp. 516–522, 1980.
[4] A. L. Gilbert, M. K. Giles, G. M. Flachs, R. B. Rogers, and Y. Hsun, "A real-time video tracking system," *IEEE Trans. Pat. Anal. Machine Intel.*, vol. PAMI-2, pp. 47–56, 1980.
[5] M. Onoe, K. Preston, Jr., and A. Rosenfeld, eds., *Real-Time/Medical Image Processing.* New York: Plenum Press, 1980.
[6] A. N. Venetsanopoulos and V. Cappellini, "Real-time image processing," *Multidimensional Systems: Techniques and Applications*, S. G. Tzafestas, Ed. New York: Marcel Dekker, 1986, pp. 345–399.
[7] J. R. Adams, E. C. Driscoll, and C. Reader, "Image processing systems," in *Digital Image Processing Techniques*, M. P. Ekstrom, Ed. New York: Academic Press, 1984.
[8] G. J. Wolfe, "Use of array processors in image processing," *SPIE*, vol. 301, pp. 43–47, 1981.
[9] P. Narendra, "VLSI architecture for real-time image processing," in *IEEE Proc. Int. Conf. Computers* (San Francisco), Feb. 23–26, 1981, pp. 303–306.
[10] H. T. Kung, "Special-purpose devices for signal and image processing: An opportunity in very large scale integration (VLSI)," *SPIE*, vol. 241, pp. 76–84, 1980.
[11] C. H. Huang, "High-speed two-dimensional filtering using residue arithmetic," *SPIE*, vol. 241, pp. 215–218, 1980.
[12] R. Krishnan, G. A. Jullien, and W. C. Miller, "Complex digital signal processing using quadratic residue number," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-34, no. 1, pp. 166–177, Feb. 1986.
[13] H. Jaggernauth, A. C. P. Loui, and A. N. Venetsanopoulos, "Real-time image processing by distributed arithmetic implementation of two-dimensional digital filters," *IEEE Trans. ASSP*, vol. ASSP-33, no. 6, pp. 1546–1555, Dec. 1985.
[14] K. M. Ty and A. N. Venetsanopoulos, "Two-dimensional digital filters with minimum cycle time," in *Proc. ICASSP* (Tampa, FL), vol. 4, pp. 1527–1530, Mar. 26–29, 1985.
[15] D. E. Knuth, *The Art of Computer Programming*, 1973.
[16] A. N. Venetsanopoulos and B. G. Mertzios, "A decomposition theorem and its implications to the design and realization of two-dimensional filters," *IEEE Trans. ASSP*, vol. ASSP-33, no. 6, pp. 1562–1575, Dec. 1985.
[17] S. Lee and A. N. Venetsanopoulos, "A two-dimensional digital filter chip set for modular two-dimensional filter implementation," in *Proc. ICASSP '85* (Tampa), Mar. 26–29, 1985, vol. 3, pp. 1005–1008.
[18] B. G. Mertzios and A. N. Venetsanopoulos, "VLSI implementation of two-dimensional digital filters via two-dimensional filter chips," *IEEE Trans. CAS*, vol. CAS-33, pp. 239–249, Feb. 1986.
[19] B. Picinbono, "Quadratic filters," in *Proc. IEEE ICASSP*, (Paris, France), May 1982, pp. 298–301.
[20] A. C. P. Loui, A. N. Venetsanopoulos and C. L. Nikias, "Modular implementation of quadratic digital filter," in *29th Midwest Symp. on Circuits and Systems* (Lincoln, NE), Aug. 11–12, 1986.
[21] A. Peled and B. Liu, "A new hardware realization of digital filters," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-22, pp. 456–462, Dec. 1974.
[22] *TRW Data Book*, 1984.
[23] *The TTL Data Book for Design Engineers*, Texas Instruments, Inc., 2nd Ed., 1981.
[24] *Signetics Bipolar and MOS Memory Data Manual*, Signetics Corp., 1978.
[25] D. Dubois and W. Steenaart, "High speed stored product recursive digital filters," *IEEE Trans. Circuits Syst.*, vol. CAS-29, pp. 390–393, June 1982.
[26] *Advanced Micro Devices Condensed Catalog*, Advanced Micro Devices, Inc., 1981.
[27] I. Pitas and A. N. Venetsanopoulos, "Nonlinear mean filters in image processing," *IEEE Trans. Acoustic, Speech, Signal Process.*, vol. ASSP-34, pp. 573–584, June 1986.
[28] W. K. Pratt, *Digital Image Processing.* New York: Wiley & Sons, 1978.
[29] E. Biglieri, "Theory of Volterra processors and some applications," in *Proc. IEEE ICASSP*, (Paris, France), May 1982, pp. 294–297.
[30] H. H. Chiang, C. L. Nikias, and A. N. Venetsanopoulos, "Efficient implementation of quadratic digital filters," *IEEE Trans. on ASSP*, in press.

Anastasios N. Venetsanopoulos (S'66–M'69–SM'79) received the B.S. degree in electrical and mechanical engineering from the National Technical University of Athens, Greece (1965), and the M.S. (1966), the M. Phil. (1968), and the Ph.D. (1969) degrees in electrical engineering, all from Yale University. He joined the University of Toronto, Canada, in September 1968, where he is now Professor and Chairman of the Communication Group, Department of Electrical Engineering. He held visiting posts at NTU, the Federal University of Rio de Janeiro, and the University of Florence. He was on research leave at the University of Grenoble, the Imperial College of Science and Technology, and was an Adjunct Professor of Concordia University (1981–84).

He has been lecturer of numerous short courses to industry and continuing education programs, contributor to eight books and over 200 papers in digital communications, digital filters, and image processing, and consultant to several organizations. He served as the Assistant Editor (1979–80) and Editor (1981–83) of the *Canadian Electrical Engineering Journal*, the President of the Canadian Society for Electrical Engineering and Vice-President of the Engineering Institute of Canada (1983–86). He was Fulbright Scholar, an A. F. Schmitt Scholar, and recipient of the J. Vakis Award. He is a member of the New York Academy of Sciences, Sigma Xi, and the Technical Chamber of Greece, a Fellow of the Engineering Institute of Canada, and is a registered Professional Engineer in Ontario and Greece.

Dr. Venetsanopoulos was Program Chairman of the International Communications Conference ICC'78 and for ICC'86. He has served as Chairman of the Central Canada Council of IEEE, is presently Associate Editor for Digital Signal Processing of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, and will be the Guest Editor of a special issue of the same Transactions on Digital Image Processing (November 1987).

Kich M. Ty (S'85) received the B.A. Sc. (with honors) and M.A. Sc. degrees in electrical engineering from the University of Toronto, Ontario, Canada, in 1982 and 1985, respectively. He is now working towards the Ph.D. degree at the same university.

He was awarded the J. E. Reid Memorial Prize (communications) from the University of Toronto in 1982, postgraduate fellowships from the same university in 1982, 1983, and 1985, the Connaught Scholarship from the same university in 1984, and the Mary H. Beatty Fellowship in 1986.

His interests include digital signal processing, image processing and analysis, digital video, high-speed processing architecture, and digital communications.

Alexander C. P. Loui (S'82) received the B.A. Sc. and M.A. Sc. degrees in electrical engineering from the University of Toronto, Canada, in 1983 and 1986, respectively. He is now working towards the Ph.D. degree in electrical engineering, also at the same university.

In the summer of 1982, he worked at the Atomic Energy of Canada Ltd., as a Research Assistant. During the period of 1983–87, he has been a Teaching Assistant with the Department of Electrical Engineering, University of Toronto, Canada. From 1983 to 1986, he was awarded the Postgraduate Scholarship from the Natural Sciences and Engineering Research Council of Canada. His research interests include digital image processing, image-processing architectures, and knowledge-based image-processing/analysis systems.