Neural, Parallel & Scientific Computations 15 (2007) 455-478

A Generalized Neural Network for Edge Detection and Its Hardware Realization

Srimanta Pal
Electronics and Communication Sciences Unit, Indian Statistical Institute, 203 B T Road, Calcutta 700108, India.
e-mail: srimanta@isical.ac.in

Abstract
This paper describes a set of orthogonal polynomial, and based on these polynomials, a set of nine basis operators for edge detection has been designed. Also these operators can be used to define many well-known edge detection gradient operators such as Robert, Sobel, Prewitt, Laplace, Isotropic, LoG, etc. which are the linear combination of these basis operators. This paper proposes two very general purpose multi-layered neural networks -- one can compute point, line, and edge detectors (such as Robert, Sobel, Prewitt, etc.); and other can compute all the nine polynomial basis operators. Also it presents two possible hardware realizations -- one is a generalized parallel pipeline VLSI for the computation of these polynomial basis operators and other is an analog VLSI for the design of possible future neuro-computer.
Keywords - Neural networks, orthogonal polynomial, edge detection, VLSI.

1. INTRODUCTION

In computer vision (Marr, 1982) and image processing system (Andrews & Hunt,1977; Bezdek et al., 1999; Jain, 1989), edge (Abdou & Pratt, 1979; Davis, 1975; Hueckel, 1971; Shin et al., 2001; Vliet & Young, 1989; Tadrous, 1995), line (Hueckel, 1973; Mansouri et al., 1987; Nevatia, 1977; Smith, 1987), and point detection (Ando, 2000) are important low level image segmentation procedures. The degree of success of any computer vision system is highly dependent on the performance of its low level segmentation operations. The enhancement/thresholding edge detection methods (Ando, 2000; Brown et al., 1998; Chanda et al. 1998; Haddon, 1988; Heath et al., 1998; O'Gorman, 1978) became very popular because of their simplicity and very low time complexity. In these methods, the gradient values of different orders are computed by convolving the various gradient operators such as Robert (Roberts, 1965; Rosenfeld, 1981), Sobel (Sobel, 1970), Prewitt (Prewitt, 1970), Laplace (Berzins, 1984; Vliet & Young, 1989), etc. The LoG (Marr, 1980; Torre & Poggio, 1986) operator has been proposed as an approximation of the optimal edge operator using the zero crossing (Mehrotra & Zhan, 1996) in the directionless Laplacian. Haralick (Haralick, 1984) has proposed a method for edge detection using the zero crossings in the second directional derivatives. The more effective edge detection method is proposed by Canny (Canny, 1986) based on maximization of the amplitude responses of the gradient operators. Also it is difficult to implement because the exact shape of the optimal smoothing filter is unknown (Fleck, 1992). It cannot detect the high frequency information (Kang & Wang, 2007). The measure of confidence level is also used to reduce the ambiguity obtained from edge information (such as gradient estimation, nonmaxima suppression, hysteresis thresholding) in edge detection (Meer & Georgescu, 2001). The edge detector needs a manual adjustment of threshold to produce a reasonable output. Automatic edge thresholding methods are developed by Fuzzy logic (Liang & Looney, 2003; Kim et al., 2004) or statistical approach Rakesh et al., 2004). Also the edge detection can be viewed as an optimization problem in which the object function depends on different edge parameters like edge directions, edge intensity, edge map, edge direction map, non-maxima suppression criteria, etc. (Kang & Wang, 2007).

Received March 7, 2006; In final form October 8, 2007

The low level segmentation procedures are (Frank et al., 1985; Huang, 1989; Kanopoulos et al., 1988; Kittler & Duff, 1985; Milgram & Pierre, 1990; Ruetz & Brodersen, 1987) being implemented on a single VLSI chip in order to provide a powerful back-end processor for application specific computer vision systems. The main bottleneck in this regard is that among the existing, numerous edge, line and point detection methods none is generalized. In the sense that a particular method may be suitable to a particular class of images but not to all kinds of images. Hence, instead of providing an all purpose back-end processor for low level segmentation only application specific back-end processors are being offered.

Initially, the edge detection problem has been viewed as analogous to surface fitting vis-a-vis orthogonal transformation problem. Because different signal/image restoration schemes have been proposed based on different orthogonal transformations. The possible set of orthogonal transformations is very large (Ahmed & Rao, 1975; Harmuth, 1972). Among them well known are Fourier, Haar, Walsh, Slant, Cosine, and Karhunen-Loeve transform, some of widely used hybrid transforms are Walsh-Hadamard, Hadamard-Haar and Hadamard-Fourier. A set of orthogonal polynomials has been used for this purpose. The completeness criterion of this class of polynomials has been established. Its measures of independence has also been estimated. The design of the polynomial based edge detectors (Bhattacharyya & Ganesan, 1997; Pal, 1991) has been discussed. Also we observed that the widely known edge detection operators such as Robert, Sobel, Prewitt, Laplacian, etc. are the linear combinations of these orthogonal polynomial based edge detectors.

In this paper, a neural network is designed based on the above observation. Also its hardware realization is described. This hardware consists of pipelined array of registers and adders with a simple and modular structure that is easily amenable to VLSI implementation. Because of the important advantage of these basis operators which can represent the most common gradient operators being used in low level image processing for detection of lines, edges and points, it is possible to implement all known gradient edge detectors in a single circuit with small size, low power requirement and quick turn-around time in order to obtain a fairly high performance. This is because of the fact that convolutions with the orthogonal polynomial based operators can be realized very simply by using only shift registers and adders, and not requiring any more complex hardware units. Another concept of hardware realization of the neural network is presented.

2. MODELS OF IMAGE PROCESSING

2.1. Classical Models

It is well known that the process of image formation (Gonzalez & Woods, 1999; Hall, 1979; Jain, 1989; Pratt, 1978) is generally described in terms of a linear system (Eqn. (1)).

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(u,v)h(x,y;u,v)dudv \tag{1}$$

The linear system is developed around an operator $h(x,y;u,v)$ which maps an object $f(u,v)$ into an image $g(x,y)$.

The function h is termed as the point-spread function (PSF) of the image formation system. It determines physically the radiant energy distribution in the image plane due to a point source of radiant energy located in the object plane.

The equation (1) is the model for continuous-continuous image formation system. There are other two models, they are discrete-continuous and discrete-discrete. From the computer application point of view discrete-discrete model is preferable. For discrete-discrete system, the equation (1) can be written as the equation (2).

$$g_{ij} = \sum_{p=1}^{N} \sum_{q=1}^{N} f_{p,q} h_{i,j;p,q} \quad \text{for } i,j = 1,2,\cdots,N. \tag{2}$$

Introducing lexicographic or stacking operators the image and object can be represented as vectors ***g*** and ***f*** respectively, and the PSF operator as [H]. Thus the equation (2) can be written in matrix form as equation (3)

$$\mathbf{g} = [\mathrm{H}]f. \tag{3}$$

If it is assumed that the object is sampled in a square array with N × N points as is the image, then [H] is an $N^2 \times N^2$ matrix.

When in an imaging system the PSF is separable space-invariant (SSI), the separable assumption refers to the rectangular co-ordinate separability of the impulse response resulting as in equation (4).

$$[\mathrm{H}] = [\mathrm{A}] \otimes [\mathrm{B}] \tag{4}$$

where [A] and [B] are two matrices of size N × N and ⊗ is the direct or kronecker product of matrices. Considering rectangular separability and using unstacked matrix notation the equation (3) can be written as

$$[\mathrm{G}] = [\mathrm{A}]\,[\mathrm{F}]\,[\mathrm{B}]^t. \tag{5}$$

In equation (5), [A] blurs the columns of the object [F] and [B] blurs the rows of [F].

With the above mentioned image formation concept, image processing technique can be considered as a reverse process of deblurring.

2.2. The Polynomial Model

As we know, different signal/image restoration schemes have been proposed based on different orthogonal transformations such as Fourier, Haar, etc. Here, we are considering a set of orthogonal polynomials (Pal, 1991; Bhattacharyya & Ganesan, 1997)

$$P_0(x,\mu,n), P_1(x,\mu,n), \cdots \text{ of degree } 0,1,\cdots \tag{6}$$

where, x = a sample, n = sample size, $\mu = \text{mean of } x = E(x) = \frac{1}{n} \sum_{1<x<n} x$, $P_0(x,\mu,n) = 1$ and $P_1(x,\mu,n) = x - \mu$ for this signal/image model.

The generating formula for the above set of polynomials (6) is as follows

$$P_{i+1}(x,\mu,n) = (x-\mu)P_i(x,\mu,n) - b_i(n)P_{i-1}(x,\mu,n) \quad \text{for } i \geq 1 \tag{7}$$

$$\text{where, } b_i(n) = \frac{\langle P_i, P_i \rangle}{\langle P_{i-1}, P_{i-1} \rangle}. \quad \text{[see Appendix I]} \tag{8}$$

In case of digital signal/images, the range of values of the spatial coordinate x may be considered to be
$x = i$, $i = 1,2, \ldots, n$. If $x = i$ for $i = 1, 2, \ldots, n$ we obtain b_is using (7) as

$$b_i(n) = \frac{i^2(n^2 - i^2)}{4(4i^2 - 1)} \quad \text{[see Appendix II} \tag{9}$$

Hence, for signal or image processing applications we get the following set of orthogonal polynomials as shown in Eqn. (10).

$$\left.\begin{aligned} &P_0(x,\mu,n)=1\\ &P_1(x,\mu,n)=x-\mu,\ \text{where, } \mu=\frac{n+1}{2}\\ &P_2(x,\mu,n)=(x-\mu)^2-\frac{n^2-1}{12}\\ &P_3(x,\mu,n)=(x-\mu)^3-\frac{(x-\mu)(3n^2-7)}{20}\\ &\quad\cdots \end{aligned}\right\} \tag{10}$$

The approximation $\underline{g}(x)$ of any piecewise continuous function $g(x)$ by the set of orthogonal polynomials
$P_0, P_1, \ldots$, in Eqn. (10) can be obtained as follows

$$g(x) \approx \underline{g}(x) = \sum_{1\le i\le t} \beta_{i-1}P_{i-1}(x,\mu,n) \tag{11}$$

where, β_is are expansion coefficients and can be obtained easily as follows.

Let $\beta_i' = \sum_{1\le x\le n} g(x)P_i(x,\mu,n)$ then $\beta_j = \beta_j' \Big/ \left(\sum_{1\le x\le n} P_j^2(x,\mu,n)\right)$.

Equation (11) is an approximation of $g(x)$ by using only a specific set of t orthogonal polynomials $\{P_0, P_1, \ldots, P_{t-1}\}$.

Also in case of signal/image processing environment, any function $g(x)$ can be completely approximated by these t orthogonal polynomials $\{P_0, P_1, \ldots, P_{t-1}\}$. So the Bessel's inequality (Courant & Hilbert, 1975) is as follows.

$$\sum_{x=1}^{n} [g(x) - \sum_{1\le i\le t} \beta_{i-1}P_{i-1}(x,\mu,n)]^2 > 0 \text{ or, } \sum_{x=1}^{n} g^2(x) \le \sum_{x=1}^{t}\left(\beta_{i-1}' \Big/ \sqrt{\sum_{1\le x\le n} P_{i-1}^2}\right)^2.$$

In order to approximate $g(x)$ in the highest order of accuracy, the value of t (the highest degree of polynomial P_{t-1} used in the approximation) will be such that the Bessel's inequality will become an equality. Hence, the completeness relation may be stated as follows.

$$\sum_{x=1}^{n} g^2(x) = \sum_{x=1}^{t} Z_{i-1}^2 \quad \text{where, } Z_i^2 = \beta_i'^2 \Big/ \sum_{1\le x\le n} P_i^2 \tag{12}$$

According to the classical theory of approximation the value of t can be obtained by measuring the linear independence of the set $\{P_0, P_1, \ldots, P_{t-1}\}$ of orthogonal polynomials for the largest value of t.

Theorem 1. The set of orthogonal polynomials shown in (10), the highest degree $(t-1)$ of the polynomial used in the approximation must be equal to $n-1$ for linear independence.

Proof: For a set $\{P_0, P_1, \ldots, P_{t-1}\}$ of orthogonal polynomials, the Gram-Determinant $\Gamma(P_0,P_1,\cdots,P_{t-1})$ may be defined as follows.

$$\Gamma(P_0,P_1,\cdots,P_{t-1}) = \begin{vmatrix} \langle P_0,P_0\rangle & \langle P_0,P_1\rangle & \cdots & \langle P_0,P_{t-1}\rangle \\ \langle P_1,P_0\rangle & \langle P_1,P_1\rangle & \cdots & \langle P_1,P_{t-1}\rangle \\ \vdots & \vdots & \cdots & \vdots \\ \langle P_{t-1},P_0\rangle & \langle P_{t-1},P_1\rangle & \cdots & \langle P_{t-1},P_{t-1}\rangle \end{vmatrix} \tag{13}$$

It is well known (Courant & Hilbert, 1975) that the vanishing of Gram-Determinant $\Gamma(P_0,P_1,\cdots,P_{t-1})$ of the set $\{P_0, P_1, \ldots, P_{t-1}\}$, of orthogonal polynomials is equivalent to the vanishing of the measure of linear independence of the set of polynomials. Moreover, the vanishing of the Gram-Determinant is a necessary and sufficient condition for the linear dependence of a set of orthogonal polynomials (Courant & Hilbert, 1975).

Therefore, $\Gamma(P_0,P_1,\cdots,P_{t-1}) = \langle P_0,P_0\rangle\langle P_1,P_1\rangle\cdots\langle P_{t-1},P_{t-1}\rangle$ (14)

Since $b_i(n) = \dfrac{\langle P_i,P_i\rangle}{\langle P_{i-1},P_{i-1}\rangle}$. So for $i = 1$, $b_1(n) = \dfrac{\langle P_1,P_1\rangle}{\langle P_0,P_0\rangle}$.

Hence, $\langle P_1,P_1\rangle = nb_1$ [Since $\langle P_0,P_0\rangle = n$].

Similarly, $\langle P_2,P_2\rangle = nb_1b_2$ and $\langle P_i,P_i\rangle = nb_1b_2\cdots b_i$.

Now,

$$\Gamma(P_0,P_1,\cdots,P_{t-1}) = n\prod_{1\le i\le t-1}\langle P_i,P_i\rangle = n\prod_{1\le i\le t-1}\left(n\prod_{1\le k\le i}b_k\right) = n^t\prod_{1\le i\le t-1}\prod_{1\le k\le i}\frac{k^2(n^2-k^2)}{4(4k^2-1)} \text{ [Using (9)]}$$

Hence, in order to make $\Gamma(P_0,P_1,\cdots,P_{t-1})$ non-zero the maximum value (integral) of t must be equal to n.

Corollary 1: If $t > n$ then the set of polynomials used in the approximation will be linearly dependent.

Corollary 2: If $t < n$ then the completeness criterion (Eqn. (12)) of the approximation by the set of polynomials may not be satisfied.

Assuming the rectangular coordinate separability and using the discrete-discrete two dimensional model of image $g(x,y)$ formulation, we may obtain the following.

$$g_{kl} = \sum_{i=0}^{m}\sum_{j=0}^{n}\beta_{ij}P_i(x_k)P_j(x_l) \text{ or, } [g_{kl}] = \sum_{i=0}^{m}\sum_{j=0}^{n}\beta_{ij}\mathrm{P}_i\mathrm{P}_j^t \quad (15)$$

where P_i is a column vector of size $n \times 1$ consisting of values of the polynomial $\mathrm{P}_i(x)$ at $x = x_1$, x =

$x_2, \ldots, x = x_n$ respectively.

Let [M] = $[\mathrm{P}_0, \mathrm{P}_1, \ldots, \mathrm{P}_{m-1}]$ and [N] = $[\mathrm{P}_0, \mathrm{P}_1, \ldots, \mathrm{P}_{n-1}]$.

Hence, $[g_{kl}]$ = [M] $[\beta_{ij}]$ $[\mathrm{N}]^t$.

In general, the image formation is modeled as [G] = [M] [F] $[\mathrm{N}]^t$.

where [M] $\otimes$ [N] is the point spread function (PSF). [M] blurs the columns of the object [F] and [N] blurs the rows.

The image processing, in general, is termed as the reverse process of blurring i.e., deblurring, assuming [M] and [N] are unitary, we obtain

$$[\mathrm{F}] = [\mathrm{M}]^t\,[\mathrm{G}]\,[\mathrm{N}] \quad (16)$$

Note: In this case, $m = n$ and [M] is not unitary. As a result (16) may be written as

$$[F] = ([M]^t[M])^{-1}[M]^t[G][M]([M]^t[M])^{-1} \quad (17)$$

For $n = 2$ and $x_1 = 1$, $x_2 = 2$

$$[M] = [P_0 \quad P_1] = \begin{bmatrix} P_0(x_1) & P_1(x_1) \\ P_0(x_2) & P_1(x_2)\end{bmatrix} = \begin{bmatrix} 1 & -\frac{1}{2} \\ 1 & \frac{1}{2}\end{bmatrix} \cong \begin{bmatrix} 1 & -1 \\ 1 & 1\end{bmatrix} \text{ (Scaled)} \quad (18)$$

The four basis matrices (with rank 1) spanned by [M] are computed by Eqn. (19).

$$O_{ij}^2 = P_iP_j^t, \quad 0 \le i,j \le 1. \quad (19)$$

For example,

$$O_{01}^2 = P_0P_1^t = \begin{pmatrix} P_0(x_1) \\ P_0(x_2)\end{pmatrix}\begin{pmatrix} P_1(x_1) & P_1(x_2)\end{pmatrix} = \begin{pmatrix} 1 \\ 1\end{pmatrix}\begin{pmatrix} -\frac{1}{2} & \frac{1}{2}\end{pmatrix} = \frac{1}{2}\begin{bmatrix} -1 & 1 \\ -1 & 1\end{bmatrix}.$$

$$O_{11}^2 = P_1P_1^t = \begin{pmatrix}P_1(x_1)\\ P_1(x_2)\end{pmatrix}\begin{pmatrix}P_1(x_1) & P_1(x_2)\end{pmatrix} = \begin{pmatrix}-\frac{1}{2}\\ \frac{1}{2}\end{pmatrix}\begin{pmatrix}-\frac{1}{2} & \frac{1}{2}\end{pmatrix} = \frac{1}{4}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}$$

Similarly, for $n = 3$ and $x_1 = 1,\ x_2 = 2, x_3 = 3$

$$[M] = \begin{bmatrix}P_0 & P_1 & P_2\end{bmatrix} = \begin{bmatrix}P_0(x_1) & P_1(x_1) & P_2(x_1)\\ P_0(x_2) & P_1(x_2) & P_2(x_2)\\ P_0(x_3) & P_1(x_3) & P_2(x_3)\end{bmatrix} = \begin{bmatrix}1 & -1 & \frac{1}{3}\\ 1 & 0 & -\frac{2}{3}\\ 1 & 1 & \frac{1}{3}\end{bmatrix} \cong \begin{bmatrix}1 & -1 & 1\\ 1 & 0 & -2\\ 1 & 1 & 1\end{bmatrix}.$$

The nine rank 1 basis matrices span by [M] are computed as $O_{ij}^3 = P_iP_j^t,\quad 0 \le i, j \le 2$.

For example,

$$O_{01}^3 = P_0P_1^t = \begin{pmatrix}P_0(x_1)\\ P_0(x_2)\\ P_0(x_3)\end{pmatrix}\begin{pmatrix}P_1(x_1) & P_1(x_2) & P_1(x_3)\end{pmatrix} = \begin{pmatrix}1\\ 1\\ 1\end{pmatrix}\begin{pmatrix}-1 & 0 & 1\end{pmatrix} = \begin{bmatrix}-1 & 0 & 1\\ -1 & 0 & 1\\ -1 & 0 & 1\end{bmatrix}$$

$$O_{11}^3 = P_1P_1^t = \begin{pmatrix}P_1(x_1)\\ P_1(x_2)\\ P_1(x_3)\end{pmatrix}\begin{pmatrix}P_1(x_1) & P_1(x_2) & P_1(x_3)\end{pmatrix} = \begin{pmatrix}-1\\ 0\\ 1\end{pmatrix}\begin{pmatrix}-1 & 0 & 1\end{pmatrix} = \begin{bmatrix}1 & 0 & -1\\ 0 & 0 & 0\\ -1 & 0 & 1\end{bmatrix}$$

The basis matrices O_{ij}^2 s and O_{ij}^3 s are shown in Table 1(a) and (b).

In terms of these basis matrices the image formation can be represented as follows using equations (15) and (19).

$$[g] = \sum_{i=0}^{n}\sum_{j=0}^{n} \beta_{ij}O_{ij}^n \tag{20}$$

Table 1(a):Polynomial Operators: 2 × 2 operator and four basis operators.

1	-1
1	1

[M]

1	1
1	1

O_{00}^2

-1	1
-1	1

O_{01}^2

-1	-1
1	1

O_{10}^2

1	-1
-1	1

O_{11}^2

Table 1(b):Polynomial Operators: 3 × 3 operator and nine basis operators.

1	-1	1
1	0	-2
1	1	1

[M]

1	1	1
1	1	1
1	1	1

O_{00}^3

-1	0	1
-1	0	1
-1	0	1

O_{01}^3

1	-2	1
1	-2	1
1	-2	1

O_{02}^3

-1	-1	-1
0	0	0
1	1	1

O_{10}^3

1	0	-1
0	0	0
-1	0	1

O_{11}^3

-1	2	-1
0	0	0
1	-2	1

O_{12}^3

1	1	1
-2	-2	-2
1	1	1

O_{20}^3

-1	0	1
2	0	-2
-1	0	1

O_{21}^3

1	-2	1
-2	4	-2
1	-2	1

O_{22}^3

3. POLYNOMIAL OPERATORS

We call the polynomial basis matrices, namely O_{ij}^2 s and O_{ij}^3 s, as polynomial operators and discuss here their usage in line, edge and point detection.

Enhancement/threshold type edge, line or point detection may be expressed by the general convolution relation

$$e(i,j) = \mathrm{W}[g_{ij}] \tag{21}$$

where, g_{ij} is the discrete gray level intensity value at the (i,j)th pixel of the input image; $e(i,j)$ is the convolution output for the processed image and W is the convolution mask defined over some neighborhood of (i,j). The mask W is chosen so as to detect a given property in an image.

If a set of 3×3 matrices is used for the mask W, the convolution output in the neighborhood of a pixel (i,j) is shown in Eqn. (22).

$$e = W_{11}g_{i-1,j-1} + W_{12}g_{i-1,j} + W_{13}g_{i-1,j+1} + W_{21}g_{i,j-1} + W_{22}g_{i,j} + W_{23}g_{i,j+1} + W_{31}g_{i+1,j-1} + W_{32}g_{i+1,j} + W_{33}g_{i+1,j+1} \tag{22}$$

Here, W_{kr} (for k, r = 1, 2, 3) are the elements of the mask W. If the elements W_{kr} are limited to only 0, ±1, ±2 or ±4, it has been observed that the convolution can be realized very simply by using only shift registers and adders, and not requiring more complex hardware units. An important usage of this property will be discussed in Section 5.

It is well known that various types of differentiation operations when performed on an image, give high values at points lying on edges and low values at points lying in the smooth regions.

In the neighborhood of a pixel (i,j) of an image, if 3×3 polynomial matrix [M] is used to transform the image matrix $[g_{kl}]$ into the coefficient matrix $[\beta'_{ij}]$ following the matrix operation $[M]^t[g_{kl}][M]$, the elements of the coefficient matrix vis-a-vis the convolution output of the nine 3×3 basis matrices are written as equations (23) - (31).

$$1:\ \beta'_{00} = O^3_{00}[g_{ij}] = g_{i-1,j-1} + g_{i-1,j} + g_{i-1,j+1} + g_{i,j-1} + g_{i,j} + g_{i,j+1} + g_{i+1,j-1} + g_{i+1,j} + g_{i+1,j+1} \tag{23}$$

$$2:\ \beta'_{01} = O^3_{01}[g_{ij}] = (g_{i-1,j+1} + g_{i,j+1} + g_{i+1,j+1}) - (g_{i-1,j-1} + g_{i,j-1} + g_{i+1,j-1}) \tag{24}$$

$$3:\ \beta'_{02} = O^3_{02}[g_{ij}] = (g_{i-1,j-1} + g_{i,j-1} + g_{i+1,j-1}) - 2(g_{i-1,j} + g_{i,j} + g_{i+1,j}) + (g_{i-1,j+1} + g_{i,j+1} + g_{i+1,j+1}) \tag{25}$$

$$4:\ \beta'_{10} = O^3_{10}[g_{ij}] = (g_{i+1,j-1} + g_{i+1,j} + g_{i+1,j+1}) - (g_{i-1,j-1} + g_{i-1,j} + g_{i-1,j+1}) \tag{26}$$

$$5:\ \beta'_{11} = O^3_{11}[g_{ij}] = (g_{i+1,j+1} - g_{i+1,j-1}) - (g_{i-1,j+1} - g_{i-1,j-1}) \tag{27}$$

$$6:\ \beta'_{12} = O^3_{12}[g_{ij}] = (g_{i+1,j-1} - g_{i-1,j-1}) - 2(g_{i+1,j} - g_{i-1,j}) + (g_{i+1,j+1} - g_{i-1,j+1}) \tag{28}$$

$$7:\ \beta'_{20} = O^3_{20}[g_{ij}] = (g_{i-1,j-1} + g_{i-1,j} + g_{i-1,j+1}) - 2(g_{i,j-1} + g_{i,j} + g_{i,j+1}) + (g_{i+1,j-1} + g_{i+1,j} + g_{i+1,j+1}) \tag{29}$$

$$8:\ \beta'_{21} = O^3_{21}[g_{ij}] = (g_{i-1,j+1} - g_{i-1,j-1}) - 2(g_{i,j+1} - g_{i,j-1}) + (g_{i+1,j+1} - g_{i+1,j-1}) \tag{30}$$

$$9:\ \beta'_{22} = O^3_{22}[g_{ij}] = (g_{i-1,j-1} - 2g_{i-1,j} + g_{i-1,j+1}) - 2(g_{i,j-1} - 2g_{i,j} + g_{i,j+1}) + (g_{i+1,j-1} - 2g_{i+1,j} + g_{i+1,j+1}) \tag{31}$$

The operations corresponding to $\beta'_{01} + \beta'_{02}$ and $\beta'_{10} + \beta'_{20}$ are the operations sensitive to vertical and horizontal edges, respectively. These operations are equivalent to blurring the image in the vertical (horizontal) direction and then taking a second or first difference in horizontal (vertical) direction. The operations corresponding to O_{02}^3 and O_{20}^3 are respectively, sensitive to vertical and horizontal lines. Laplacian operators are obtained by combining the operations O_{02}^3, O_{20}^3 and O_{22}^3 (Table 2), and this operation is sensitive to points. O_{00}^3 is the local averaging operator. Other operators which are being used as edge detectors, namely, Roberts (Davis, 1975; Rosenfeld & Kak, 1982), Sobel (Davis, 1975; Rosenfeld & Kak, 1982), Prewitt (Davis, 1975; Prewitt, 1970; Rosenfeld & Kak, 1982), and Frei and Chen (Gonzalez & Woods, 1999). Among these only Frei and Chen masks span the nine dimensional space of 3 × 3 basis matrices. But convolution with Frei and Chen masks is difficult to realize in hardware because of the occurrence of the matrix element $\sqrt{2}$. Whereas use of these polynomial operators for convolution with the image matrix can be realized very easily, as shown in Section 5 by using only registers, shift registers and add/subtract units.

Moreover, the widely used line/edge operators can be expressed in terms of these polynomial operators as shown in Table 2.

Table 2: Some well-known operators and their relations with polynomial operators.

Operators			
Standard	Polynomial	Standard	Polynomial
Roberts			
$W_1 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$	$\frac{1}{2}(O_{01}^2 - O_{10}^2)$	$W_2 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$-\frac{1}{2}(O_{01}^2 + O_{10}^2)$
Sobel			
$W_1 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}(O_{21}^3 - 4O_{01}^3)$	$W_2 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$	$\frac{1}{3}(4O_{10}^3 - O_{12}^3)$
$W_3 = \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$	$(O_{01}^3 + O_{10}^3)$	$W_4 = \begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$	$(O_{01}^3 - O_{10}^3)$
Prewitt			
$W_1 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$	O_{01}^3	$W_2 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$	O_{10}^3
Point		*Average*	
$W_1 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	$O_{22}^3 - (O_{02}^3 + O_{20}^3)$	$W_1 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	O_{00}^3
Line			
$W_1 = \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$	$-O_{20}^3$	$W_2 = \begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}$	$\frac{1}{2}(O_{22}^3 - 3O_{11}^3)$
$W_3 = \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$	$-O_{02}^3$	$W_4 = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$	$\frac{1}{2}(O_{22}^3 + 3O_{11}^3)$

Table 2: Some well-known operators and their relations with polynomial operators. (Cont.)

Operators			
Standard	Polynomial	Standard	Polynomial
Laplace			
$W_1 = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & -1 \end{bmatrix}$	O_{22}^3	$W_2 = \begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$	$-(O_{02}^3 + O_{20}^3)$
$W_3 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix}$	$\frac{1}{3}(O_{02}^3 + O_{20}^3 - 2O_{22}^3)$		
Isotropic			
$W_1 = \begin{bmatrix} -1 & 0 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ -1 & 0 & 1 \end{bmatrix}$	$\frac{2+\sqrt{2}}{3}O_{01}^3 - \frac{\sqrt{2}-1}{3}O_{21}^3$ $\cong 1.1381O_{01}^3 - 0.1381O_{21}^3$	$W_2 = \begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$	$\frac{2+\sqrt{2}}{3}O_{10}^3 - \frac{\sqrt{2}-1}{3}O_{12}^3$ $\cong 1.1381O_{10}^3 - 0.1381O_{12}^3$
$3\times 3\ \mathrm{LoG}_{\sigma=0.35}$ $\mathrm{LoG}_{\sigma=0.35} =$ $\begin{bmatrix} -1 & -30 & -1 \\ -30 & 124 & -30 \\ -1 & -30 & -1 \end{bmatrix}$	$-10.7(O_{02}^3 + O_{20}^3)$ $+20.3O_{22}^3$	$3\times 3\ \mathrm{LoG}_{\sigma=0.26}$ $\mathrm{LoG}_{\sigma=0.26} =$ $\begin{bmatrix} 0 & -30 & -1 \\ -30 & 124 & -30 \\ -1 & -30 & -1 \end{bmatrix}$	$-0.33(O_{02}^3 + O_{20}^3)$ $+0.67O_{22}^3$

The gradient amplitude values of the noise free ideal edges under Roberts, Sobel and Prewitt operators have been calculated as follows. The edge amplitude has been computed using the expressions shown in equations (32) and (33).

Amplitude response utilizing *r.m.s.* (root mean square) point nonlinearity

$$A(j,k) = \sqrt{(e_1(j,k))^2 + (e_2(j,k))^2 + \cdots} \tag{32}$$

Amplitude response utilizing magnitude point nonlinearity

$$A(j,k) = |e_1(j,k)| + |e_2(j,k)| + \cdots \tag{33}$$

where $e_i(j,k) = W_i[g_{jk}]$, W_i is the ith operator in the set of operators, viz. Roberts, Sobel or Prewitt, etc.

The binary edge $E_{opr}(j,k)$ is computed by the following rule

$$E_{opr}(j,k) = \begin{cases} 1 & A(j,k) \geq T_{opr} \\ 0 & \text{otherwise} \end{cases} \tag{34}$$

where, T_{opr} is the threshold value of the edge amplitude for an operator.

Edge orientation $\theta(j,k)$ with respect to the horizontal axis can be computed as

$$\textit{Roberts}:\ \theta(j,k) = \frac{\pi}{4} + \tan^{-1}\left[\frac{e_2(j,k)}{e_1(j,k)}\right] \tag{35}$$

$$\textit{Sobel} \text{ and } \textit{Prewitt}:\ \theta(j,k) = \tan^{-1}\left[\frac{e_2(j,k)}{e_1(j,k)}\right] \tag{36}$$

4. NEURAL NETWORK MODEL

The neural network (Haykin, 2001) architecture of widely used edge detection operators such as Robert, Sobel, Prewitt, etc. is a multi-layered network. There exists some connections between the consecutive layers but no connection within a layer (Fig. 1).

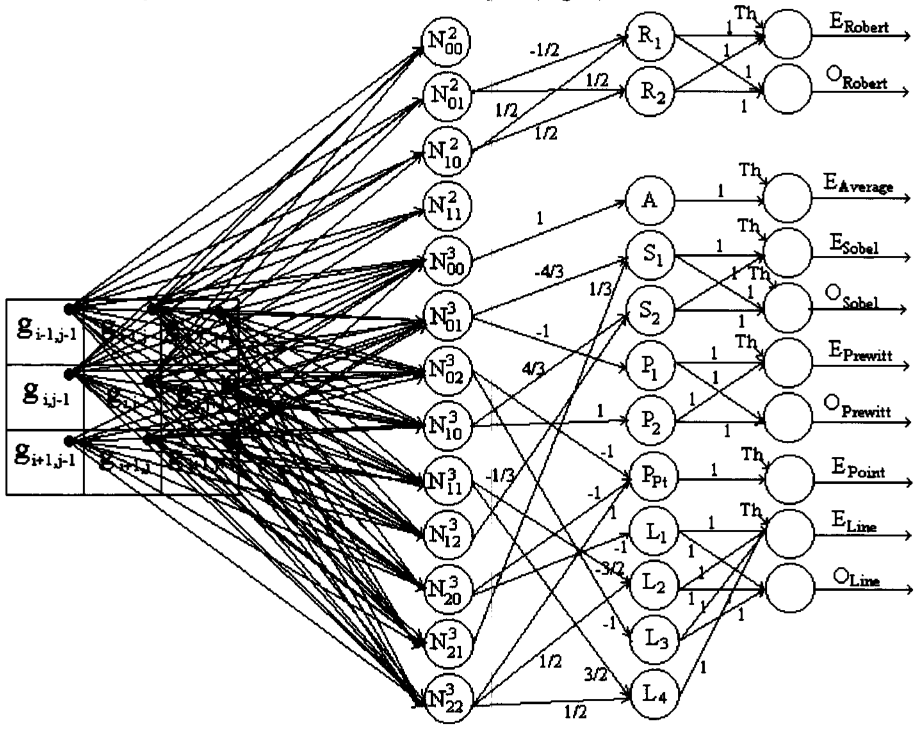

Figure 1: A generalized neural network architecture for operator based well-known edge detectors.

The first layer of this network is the input layer, it accepts a gray level image for edge detection. In figure 1, the connections are shown only to process the (i,j)th picture element of the input. The first layer contains 9 neurons, $N_{i+u,j+v}$ where $u,v = -1, 0, 1$. Since the processing of (i,j)th element depends on either 2×2 or 3×3 neighboring elements. Therefore, the input of the neuron $N_{i+u,j+v}$ is $g_{i+u,j+v}$ for $u,v = -1, 0, 1$.

The second layer of this network contains 13 neurons in which 4 neurons, N^2_{ij} for $i,j = 0,1$ are used for 2×2 polynomial operators and 9 neurons, N^3_{ij} for $i, j = 0,1,2$ are used for 3×3 polynomial operators. So the connection between the first and the second layer is as follows. For the computation of convolution by 2×2 polynomial operators: each neuron $N_{i+u,j+v}$, for $u,v = -1,0$ in the input layer are connected with N^2_{pq} for $p,\ q = 0,\ 1$ in the second layer. Similarly, for the computation of convolution by 3×3 polynomial operators: each neuron $N_{i+u,j+v}$ for $u,v = -1, 0, 1$ in the input layer are connected with N^2_{pq} for $p,\ q = 0,\ 1,\ 2$ in the second layer. The connection weight between first and second layers are as follows. The weight between $N_{i+u-2,j+v-2}$ and

N^2_{pq} is $O^2_{pq}(u,v)$ for p, q = 0,1 and u,v = 1, 2. Similarly, the weight between $N_{i+u-2,j+v-2}$ and N^3_{pq} is $O^3_{pq}(u,v)$ for p, q = 0,1,2 and u, v = 1, 2, 3 . Therefore, the output of a neuron in the second layer is computed as follows. The output of N^2_{pq} is $\sum_{1\le u,v\le 2} O^2_{pq}(u,v)\times g_{i+u-2,j+v-2}$ for p, q = 0, 1. Similarly, the output of N^3_{pq} is $\sum_{1\le u,v\le 3} O^3_{pq}(u,v)\times g_{i+u-2,j+v-2}$ for p, q = 0, 1, 2.

The third layer of this network is designed based on Table 2 where the well-known operators such as Roberts, Sobel, Prewitt, etc. are the linear combinations of polynomial operators. In this layer, the number of neurons depends upon the consideration of well-known operators. In Fig. 1, Roberts operator require 2 neurons, R_1 and R_2, Average operator require one neuron A, Sobel operators require 2 neurons S1, S2, Prewitt operators require 2 neurons P_1, P_2, Point operator require one neuron P_{Pt}, Line operators require 4 neurons L_1, L_2, L_3, L_4. Similarly other operators in Table 2 can also be implemented in this layer. Neurons R_1 and R_2 are both connected with N^2_{01} and N^2_{10}. The connection weights between (N^2_{01},R1), (N^2_{10},R_1), (N^2_{01},R_2) and (N^2_{10},R_2) are $-\frac{1}{2}$, $\frac{1}{2}$, $\frac{1}{2}$ and $\frac{1}{2}$ respectively. The output of R_1 is

$$-\frac{1}{2}\sum_{1\le u,v\le 2} O^2_{01}(u,v)\times g_{i+u-2,j+v-2}+\frac{1}{2}\sum_{1\le u,v\le 2} O^2_{10}(u,v)\times g_{i+u-2,j+v-2}$$

and the output of R_2 is

$$\frac{1}{2}\sum_{1\le u,v\le 2} O^2_{01}(u,v)\times g_{i+u-2,j+v-2}+\frac{1}{2}\sum_{1\le u,v\le 2} O^2_{10}(u,v)\times g_{i+u-2,j+v-2}\,.$$

Other connections and connection weights between the neurons of second and third layer are shown in the Fig. 1. Similarly, the output of other neurons in the third layer can be easily computed as in R_1 and R_2.

The fourth layer of this network is designed based on the consideration of the structure of the third layer. The objective of this layer is to calculate edge magnitude and orientation (if necessary). An activation is applied on the edge magnitude for binary edge. Here the activation is basically a step function in Eqn. (34).

The VLSI realization of Fig. 1 is not simple. An alternate representation of the network in Fig. 1 is shown in Fig. 2. Its hardware realization is described in Section 5.

The first two layers of the neural network in Fig. 1 can be realized by a detailed neural network in Fig. 2 using two input neurons where these two input neurons are easily realized by simple hardware like adders, subtractors and shift registers.

In figure 2, the nine input nodes are $N^1_1, N^1_2, N^1_3, N^1_4, N^1_5, N^1_6, N^1_7, N^1_8$ and N^1_9. The nodes N^2_1, N^2_4, N^2_7, N^3_2, N^3_5, N^3_8, N^4_1, N^4_2, N^4_4, N^5_5, N^5_7, N^5_9, N^6_1, N^6_2, N^6_3 are adders of two input neurons. Also N^2_3, N^2_6, N^2_9, N^3_1, N^3_4, N^3_7, N^4_3, N^4_5 are the neurons equivalent to shift registers. The weight between the nodes N_1 and N_2 is represented by $w(N_1,N_2)$. Here $w(N^1_i,N^2_i)=1$, $w(N^1_i,N^2_{i+1})=-1$, $w(N^1_{i+1},N^2_i)=1$, $w(N^1_{i+1},N^2_{i+1})=1$, $w(N^1_{i+2},N^2_{i+2})=2$, $w(N^1_{i+2},N^3_{i+1})=1$, $w(N^2_i,N^3_{i+1})=1$, $w(N^2_i,N^3_{i+2})=1$, $w(N^2_{i+1},N^3_i)=2$ and $w(N^2_{i+2},N^3_{i+2})=-1$ for i = 1, 4, 7. Similarly, we can set the weights for other connections.

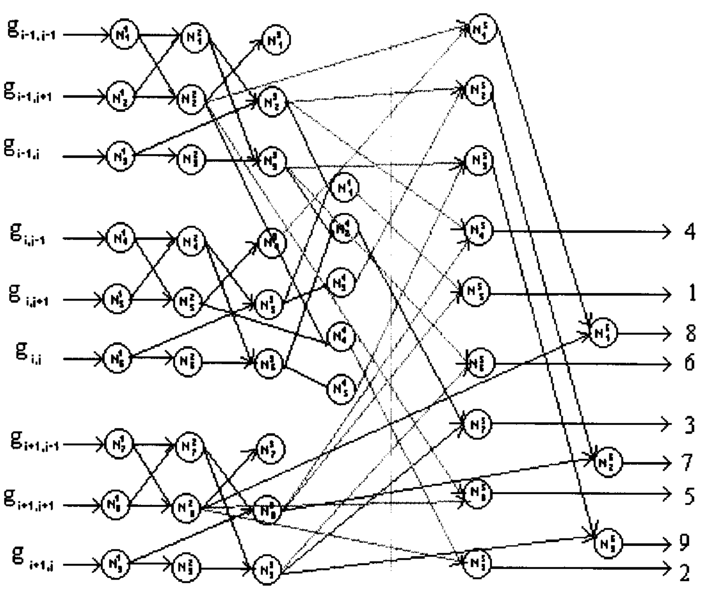

Figure 2: A neural network for the computation of basis operators (equivalent to the combination of First and second layers of Fig. 1).

5. HARDWARE REALIZATION

One of the usages of this polynomial based operators is to implement edge, line and point detection on the same hardware circuit. The design of this circuit consists of pipelined array of registers and adders with a simple and modular structure that is easily amenable to VLSI implementation.

5.1. Parallel Pipeline VLSI Implementation

The hardware architecture becomes very simple and regular because of the fact that the polynomial operators contain mostly 0, ±1 and ±2 elements (Table 1). From a close scrutiny of the polynomial operators, it is also observed that one single pipeline can generate convolution with the operators O^3_{00}, O^3_{10} and O^3_{20} (Figure 3(a)); another with O^3_{02}, O^3_{12} and O^3_{22} (Figure 3(b)) while a third can provide convolutions with O^3_{01}, O^3_{11} and O^3_{21} (Figure 3(c)). This hardware handles a set of nine operators.

The architecture of this image processing chip is shown in Figure 3. There are two major blocks -- (1) one for generation of convolution output and (2) the other for detecting edge/line magnitude and direction as well as isolated points. The structure is fully pipelined and synchronous. It requires the control of a two phase non-overlapping clock pulse CK1 and CK2. There is an initial computational delay of five cycles while the pipeline is being filled. Afterwards the convolution output are produced at every clock cycle. The chip accepts 8 *bit* words representing the gray level intensity of pixels in a raster scan sequence, i.e. from left to right in a line and subsequent lines in a top-down fashion. A off-chip line memory supplies data in a sequence such that three data values $g_{i,j-1}, g_{i,j}$ and $g_{i,j+1}$ are available simultaneously at the windows for computing convolutions centering around the *(i,j)*th pixel.

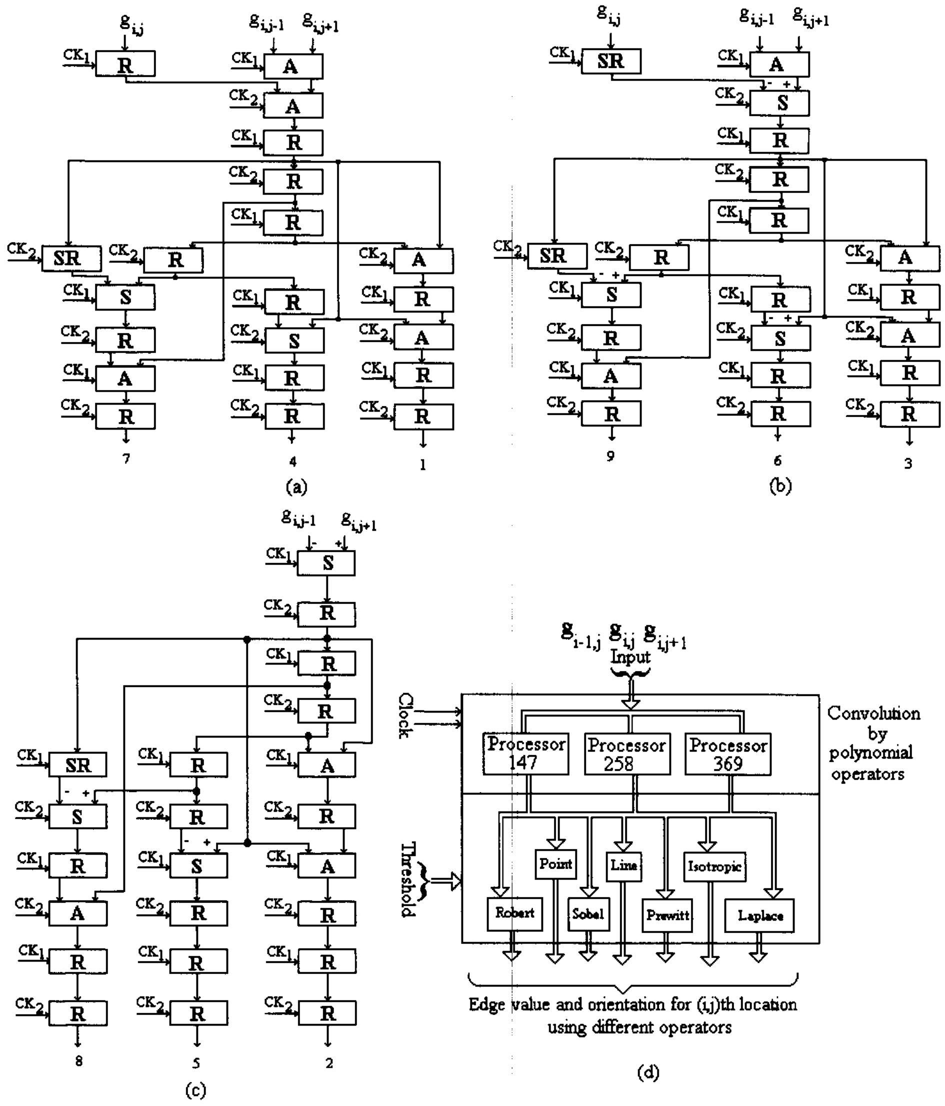

Figure 3: Hardware realization of basis operator for edge detection: Symbol A for adder, S for subtractor, R for register, SR for shift register, CK1 for clock pulse and CK2 for clock pulse with reverse in phase of CK1: (a) processor 147 : $g_{i,j-1}, g_{i,j}, g_{i,j+1}$ but moves according to the clock pulses and produces output $\beta'_{00}, \beta'_{10}, \beta'_{20}$, (b) processor 369 : input $g_{i,j-1}, g_{i,j}, g_{i,j+1}$ but moves according to the clock pulses and produces output $\beta'_{02}, \beta'_{12}, \beta'_{22}$, (c) processor 258: input $g_{i,j-1}, g_{i,j+1}$ but moves according to the clock pulses and produces output $\beta'_{01}, \beta'_{11}, \beta'_{21}$, and (d) combined edge detection processor: It accepts $g_{i,j-1}, g_{i,j}, g_{i,j+1}$ as input and in the first phase produces convolved output by the polynomial operators and then in the second phase it produces different edge magnitude in 0 or 1 form with direction if exists in the operator also it has two clock pulse line as well as threshold inputs.

The polynomial convolution processor comprises three subprocessors such as processor 147, processor 258, and processor 369 for handling the three pipelines. The functional block diagram of the subprocessor (processor 147) for convolution with the operators O_{00}^3, O_{10}^3 and O_{20}^3 is shown in Figure 3(a). The other two subprocessors, processor 258 and processor 369 have similar architecture as shown in Figure 3(c) and Figure 3(b) respectively. Each subprocessor consists of 12 *bit* adders/subtractors, simple registers and shift registers. As all the coefficients in polynomial operators are related by powers of 2, any multiplication can be accomplished by a shift in the data bit position. Taking advantage of common subexpressions, the number of adders and registers have been reduced to 7 and 14, respectively, in the combined pipeline for O_{00}^3, O_{10}^3, O_{20}^3 (Figure 3(a)) from the corresponding number of at least 11-12 adders and 18 registers for separate parallel pipelines.

Let us illustrate the operations of processor 147 using some example.

Example

Let us consider a sample input image in Table 3, where $g_{i,j}$ is the gray value or intensity at *(i,j)*th location, to explain the operations of processor 147.

Table 3: A sample image segment

$g_{1,1}$	$g_{1,2}$	$g_{1,3}$	...
$g_{2,1}$	$g_{2,2}$	$g_{2,3}$	...
$g_{3,1}$	$g_{3,2}$	$g_{3,3}$	...
$g_{4,1}$	$g_{4,2}$	$g_{4,3}$	...
$g_{5,1}$	$g_{5,2}$	$g_{5,3}$	...
$g_{6,1}$	$g_{6,2}$	$g_{6,3}$	...
$g_{7,1}$	$g_{7,2}$	$g_{7,3}$	...
$g_{8,1}$	$g_{8,2}$	$g_{8,3}$	...
...	...	...	...

In Figure 3(a), a two phase non-overlapping clock pulse CK1 and CK2 controls the circuit operations. This circuit have 10 levels with 21 components in which 7 adder like components for addition/subtraction, 13 registers for storage and 1 shift register for division/multiplication by 2. In this circuit, first level contains 2 components, each of the second to fifth level contains one component and each of the sixth to tenth level contains 3 components. Level 1, 3, 5, 7, 9 are simultaneously activated by clock pulse CK1 and similarly level 2, 4, 6, 8, 10 are activated by clock pulse CK2. Also each level between 6 to 10 contains 3 components. Activity of each components with clock pulse of the circuit in Figure 3(a) are shown in Table 4.

Initial input vector is ($g_{1,1}$, $g_{1,2}$, $g_{1,3}$). After clock pulse CK1, at level 1, the circuit component R accepts $g_{1,2}$ and component A adds to ($g_{1,1}$+ $g_{1,3}$). Similarly, level 3, 5, 7 and 9 describe their outputs as in serial number 1 of Table 4. Also the input vector changes to next ($g_{2,1}$, $g_{2,2}$, $g_{2,3}$) (say). At the same time, the outputs of the components in levels 2, 4, 6, 8 and 10 remain the same. After the next clock pulse CK2, at level 2, the circuit component A adds to ($g_{1,1}$+$g_{1,2}$+$g_{1,3}$) and similarly others outputs at level 4, 6, 8 and 10 are described in serial number 2 of Table 4. Also the outputs of the components in levels 1, 3, 5, 7 and 9 remain the same. Now the next clock pulse is CK1 and the results of each component are described in serial number 3 of Table 4. This process is continued as described in Table 4. The pipeline is filled after 5 full cycle (CK1+CK2) then each full cycle produces the respective convolution output as the operators O^3_{00}, O^3_{10} and O^3_{20}.

Table 4: Activity of each components of processor 147.

Sl	Ck	1	2	3
1	1	$g_{1,2}, g_{1,1}+g_{1,3}$	0	0
		4	5	
		0	0	
		6	7	8
		0,0,0	0,0,0	0,0,0
		9	10	
		0,0,0	0,0,0	
2	2	1	2	3
		$g_{1,2}, g_{1,1}+g_{1,3}$	$g_{1,1}+g_{1,2}+g_{1,3}$	0
		4	5	
		0	0	
		6	7	8
		0,0,0	0,0,0	0,0,0
		9	10	
		0,0,0	0,0,0	
3	1	1	2	3
		$g_{2,2}, g_{2,1}+g_{2,3}$	$g_{1,1}+g_{1,2}+g_{1,3}$	$g_{1,1}+g_{1,2}+g_{1,3}$
		4	5	
		0	0	
		6	7	8
		0,0,0	0,0,0	0,0,0
		9	10	
		0,0,0	0,0,0	
4	2	1	2	3
		$g_{2,2}, g_{2,1}+g_{2,3}$	$g_{2,1}+g_{2,2}+$g2,3	g1,1+g1,2+g1,3
		4	5	
		g1,1+g1,2+g1,3	0	
		6	7	8
		0, 2(g1,1+g1,2+g1,3), g1,1+g1,2+g1,3	0, 0, 0	0, g1,1+g1,2+g1,3, g1,1+g1,2+g1,3
		9	10	
		0,0,0	0,0,0	
5	1	1	2	3
		g3,2,g3,1+g3,3	g2,1+g2,2+g2,3	g2,1+g2,2+g2,3
		4	5	
		g1,1+g1,2+g1,3	g1,1+g1,2+g1,3	
		6	7	8
		0, $2(g_{1,1}+g_{1,2}+g_{1,3})$, $g_{1,1}+g_{1,2}+g_{1,3}$	$-2(g_{1,1}+g_{1,2}+g_{1,3})$, 0, $g_{1,1}+g_{1,2}+g_{1,3}$	0, $g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$
		9	10	
		$g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$	0, 0, 0	

Table 4: Activity of each components of processor 147 (contd.).

Sl	Ck	1	2	3
6	2	1	2	3
		$g_{3,2}, g_{3,1}+g_{3,3}$	$g_{3,1}+g_{3,2}+g_{3,3}$	$g_{2,1}+g_{2,2}+g_{2,3}$
		4	5	
		$g_{2,1}+g_{2,2}+g_{2,3}$	$g_{1,1}+g_{1,2}+g_{1,3}$	
		6	7	8
		$g_{1,1}+g_{1,2}+g_{1,3}$, $2(g_{2,1}+g_{2,2}+g_{2,3})$, $g_{1,1}+g_{1,2}+g_{1,3}+g_{2,1}+g_{2,2}+g_{2,3}$	$-2(g_{1,1}+g_{1,2}+g_{1,3})$, 0, $g_{1,1}+g_{1,2}+g_{1,3}$	$-2(g_{1,1}+g_{1,2}+g_{1,3})$, $g_{2,1}+g_{2,2}+g_{2,3}$, $g_{1,1}+g_{1,2}+g_{1,3}+$ $g_{2,1}+g_{2,2}+g_{2,3}$
		9	10	
		$g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$	$g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$	
7	1	1	2	3
		$g_{4,2}, g_{4,1}+g_{4,3}$	$g_{3,1}+g_{3,2}+g_{3,3}$	$g_{3,1}+g_{3,2}+g_{3,3}$
		4	5	
		$g_{2,1}+g_{2,2}+g_{2,3}$	$g_{2,1}+g_{2,2}+g_{2,3}$	
		6	7	8
		$g_{1,1}+g_{1,2}+g_{1,3}$, $2(g_{2,1}+g_{2,2}+g_{2,3})$, $g_{1,1}+g_{1,2}+g_{1,3}$ $+g_{2,1}+g_{2,2}+g_{2,3}$	$g_{1,1}+g_{1,2}+g_{1,3}$ $-2(g_{2,1}+g_{2,2}+g_{2,3})$, $g_{1,1}+g_{1,2}+g_{1,3}$ $g_{1,1}+g_{1,2}+g_{1,3}+g_{2,1}+g_{2,2}+g_{2,3}$	$-2(g_{1,1}+g_{1,2}+g_{1,3})$, $g_{2,1}+g_{2,2}+$g2,3, g1,1+g1,2+g1,3 +g2,1+g2,2+g2,3
		9	10	
		g2,1+g2,2+g2,3 -2(g1,1+g1,2+g1,3), g2,1+g2,2+g2,3, g1,1+g1,2+g1,3 +g2,1+g2,2+g2,3	g1,1+g1,2+g1,3, g1,1+g1,2+g1,3, g1,1+g1,2+g1,3	
8	2	1	2	3
		g4,2,g4,1+g4,3	g4,1+g4,2+g4,3	$g_{3,1}+g_{3,2}+g_{3,3}$
		4	5	
		$g_{3,1}+g_{3,2}+g_{3,3}$	$g_{2,1}+g_{2,2}+g_{2,3}$	
		6	7	8
		$g_{2,1}+g_{2,2}+g_{2,3}$, $2(g_{3,1}+g_{3,2}+g_{3,3})$, $g_{2,1}+g_{2,2}+g_{2,3}$ $+g_{3,1}+g_{3,2}+g_{3,3}$	$(g_{1,1}+g_{1,2}+g_{1,3})-$ $2(g_{2,1}+g_{2,2}+g_{2,3})$, $g_{1,1}+g_{1,2}+g_{1,3}$, $g_{1,1}+g_{1,2}+g_{1,3}$ $+(g_{2,1}+g_{2,2}+g_{2,3})$	$g_{1,1}+g_{1,2}+g_{1,3}$ $-2(g_{2,1}+g_{2,2}+g_{2,3})$, $(g_{3,1}+g_{3,2}+g_{3,3})$ $-(g_{1,1}+g_{1,2}+g_{1,3})$, $\sum_{1\leq i,j\leq 3} g_{i,j}$
		9	10	
		$g_{2,1}+g_{2,2}+g_{2,3}$ $-2(g_{1,1}+g_{1,2}+g_{1,3})$, $g_{2,1}+g_{2,2}+g_{2,3}$, $g_{1,1}+g_{1,2}+g_{1,3}+g_{2,1}+g_{2,2}+g_{2,3}$	$g_{2,1}+g_{2,2}+g_{2,3}$, $-2(g_{1,1}+g_{1,2}+g_{1,3})$, $g_{2,1}+g_{2,2}+g_{2,3}$, $g_{1,1}+g_{1,2}+g_{1,3}+g_{2,1}+g_{2,2}+g_{2,3}$	

Orientation

The hardware for realizing edge amplitude and orientation, which is not shown in Fig. 4 in detail, is also accomplished by a fully pipelined architecture of adders/comparators, simple registers and shift registers.

Table 4: Activity of each components of processor 147 (contd.).

Sl	Ck	1	2	3
9	1	$g_{5,2}, g_{5,1}+g_{5,3}$	$g_{4,1}+g_{4,2}+g_{4,3}$	$g_{4,1}+g_{4,2}+g_{4,3}$
		4	5	
		$g_{3,1}+g_{3,2}+g_{3,3}$	$g_{3,1}+g_{3,2}+g_{3,3}$	
		6	7	8
		$g_{2,1}+g_{2,2}+g_{2,3}$, $2(g_{3,1}+g_{3,2}+g_{3,3})$, $g_{2,1}+g_{2,2}+g_{2,3}+g_{3,1}+g_{3,2}+g_{3,3}$	$(g_{2,1}+g_{2,2}+g_{2,3})-$ $2(g_{3,1}+g_{3,2}+g_{3,3})$, $g_{2,1}+g_{2,2}+g_{2,3}$, $g_{2,1}+g_{2,2}+g_{2,3}+(g_{3,1}+g_{3,2}+g_{3,3})$	$(g_{1,1}+g_{1,2}+g_{1,3})$ $-2(g_{2,1}+g_{2,2}+g_{2,3})$, $(g_{3,1}+g_{3,2}+g_{3,3})$ $-(g_{1,1}+g_{1,2}+g_{1,3})$, $\sum_{1\le i,j\le 3} g_{i,j}$
		9	10	
		$g_{3,1}+g_{3,2}+g_{3,3}+g_{1,1}+g_{1,2}+g_{1,3})$ $-2(g_{2,1}+g_{2,2}+g_{2,3})$, $g_{3,1}+g_{3,2}+g_{3,3}-(g_{1,1}+g_{1,2}+g_{1,3})$, $\sum_{1\le i,j\le 3} g_{i,j}$	$(g_{2,1}+g_{2,2}+g_{2,3})$ $-2(g_{1,1}+g_{1,2}+g_{1,3})$, $g_{2,1}+g_{2,2}+g_{2,3}$, $g_{1,1}+g_{1,2}+g_{1,3}+g_{2,1}+g_{2,2}+g_{2,3}$	
10	2	1	2	3
		$g_{5,2}, g_{5,1}+g_{5,3}$	$g_{5,1}+g_{5,2}+g_{5,3}$	$g_{4,1}+g_{4,2}+g_{4,3}$
		4	5	
		$g_{4,1}+g_{4,2}+g_{4,3}$	$g_{3,1}+g_{3,2}+g_{3,3}$	
		6	7	8
		$g_{3,1}+g_{3,2}+g_{3,3}$, $2(g_{4,1}+g_{4,2}+g_{4,3})$, $g_{3,1}+g_{3,2}+g_{3,3}+$ $(g_{4,1}+g_{4,2}+g_{4,3})$	$(g_{2,1}+g_{2,2}+g_{2,3})-$ $2(g_{3,1}+g_{3,2}+g_{3,3})$, $g_{2,1}+g_{2,2}+g_{2,3}$, $(g_{2,1}+g_{2,2}+g_{2,3})+$ $+(g_{3,1}+g_{3,2}+g_{3,3})$	$(g_{2,1}+g_{2,2}+g_{2,3})$ $-2(g_{3,1}+g_{3,2}+g_{3,3})$, $(g_{4,1}+g_{4,2}+g_{4,3})-$ $(g_{2,1}+g_{2,2}+g_{2,3})$, $\sum_{2\le i\le 4}\sum_{1\le j\le 3} g_{i,j}$
		9	10	
		$g_{3,1}+g_{3,2}+g_{3,3}+$ $(g_{1,1}+g_{1,2}+g_{1,3})-$ $2(g_{2,1}+g_{2,2}+g_{2,3})$, $(g_{3,1}+g_{3,2}+g_{3,3})-(g_{1,1}+g_{1,2}+g_{1,3})$, $\sum_{1\le i,j\le 3} g_{i,j}$	$(g_{3,1}+g_{3,2}+g_{3,3})+$ $(g_{1,1}+g_{1,2}+g_{1,3})-2(g_{2,1}+g_{2,2}+g_{2,3})$, $(g_{3,1}+g_{3,2}+g_{3,3})-(g_{1,1}+g_{1,2}+g_{1,3}$, $\sum_{1\le i,j\le 3} g_{i,j}$	
11	1	1	2	3
		$g_{6,2}, g_{6,1}+g_{6,3}$	$g_{5,1}+g_{5,2}+g_{5,3}$	$g_{5,1}+g_{5,2}+g_{5,3}$
		4	5	
		$g_{4,1}+g_{4,2}+g_{4,3}$	$g_{4,1}+g_{4,2}+g_{4,3}$	
		6	7	8
		$g_{3,1}+g_{3,2}+g_{3,3}$, $2(g_{4,1}+g_{4,2}+g_{4,3})$, $g_{3,1}+g_{3,2}+g_{3,3}+$ $(g_{3,1}+g_{3,2}+g_{3,3})$	$(g_{3,1}+g_{3,2}+g_{3,3})-$ $2(g_{4,1}+g_{4,2}+g_{4,3})$, $g_{3,1}+g_{3,2}+g_{3,3}$, $(g_{3,1}+g_{3,2}+g_{3,3})$ $+(g_{4,1}+g_{4,2}+g_{4,3})$	$(g_{2,1}+g_{2,2}+g_{2,3})$ $-2(g_{3,1}+g_{3,2}+g_{3,3})$, $(g_{4,1}+g_{4,2}+g_{4,3})-$ $(g_{2,1}+g_{2,2}+g_{2,3})$, $\sum_{2\le i\le 4}\sum_{1\le j\le 3} g_{i,j}$
		9	10	
		$g_{4,1}+g_{4,2}+g_{4,3}$ $(g_{2,1}+g_{2,2}+g_{2,3})-$ $2(g_{3,1}+g_{3,2}+g_{3,3})$, $(g_{4,1}+g_{4,2}+g_{4,3})-(g_{2,1}+g_{2,2}+g_{2,3})$, $\sum_{2\le i\le 4}\sum_{1\le j\le 3} g_{i,j}$	$(g_{3,1}+g_{3,2}+g_{3,3})+$ $(g_{1,1}+g_{1,2}+g_{1,3})-2(g_{2,1}+g_{2,2}+g_{2,3})$, $(g_{3,1}+g_{3,2}+g_{3,3})-(g_{1,1}+g_{1,2}+g_{1,3})$, $\sum_{1\le i,j\le 3} g_{i,j}$	

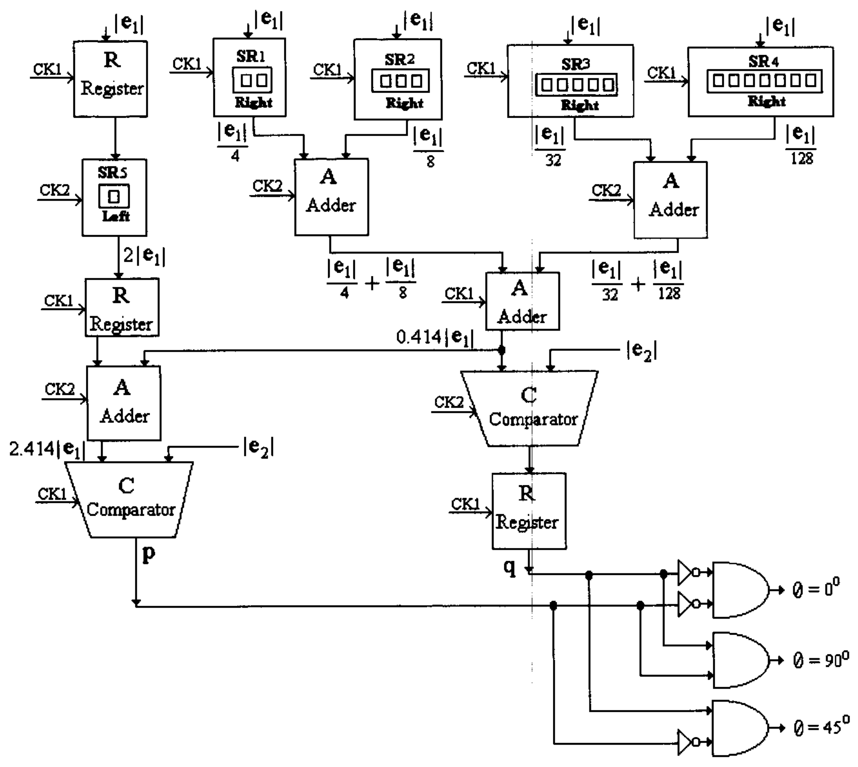

Figure 4: Circuit for edge orientation determination: Notations are used as R for register, SR for shift registers, A for adder, C for comparator, θ for orientation.

Computation of edge orientation is done by a simple procedure of resolving the edge orientation to 45° using the Freeman chain coding. The signs of the angles (i.e., the quadrants) can be directly determined from the sign of the convolution output e_1 and e_2 by the operators W_1 and W_2. This is because, the edge direction can be computed as $\theta = \tan^{-1}\frac{e_1}{e_2}$. In case of Prewitt edge detectors, we use $W_1 = O_{01}^3$ and $W_2 = O_{10}^3$. In order to decide the actual direction of the edge in a quadrant, one has to make sure whether $22.5^\circ < \theta < 67.5^\circ$, i.e., orientation is 0° if $\theta < 22.5^\circ$, similarly orientation is 90° if $\theta > 67.5^\circ$ otherwise orientation is 45°. This can be decided, $e_1 \tan 22.5^\circ \le e_2 \le e_1 \tan 67.5^\circ$. Hardware realization of $e_1 \tan 22.5^\circ$ and $e_1 \tan 67.5^\circ$ can be accomplished by simple shifts of 2, 3, 5 and 7 bits, because

$$\tan 22.5^o = 0.414 \cong \frac{1}{4} + \frac{1}{8} + \frac{1}{32} + \frac{1}{128}$$

A simple Boolean logic can be developed for the determination of orientation as follows.

$$\text{Let } p = \begin{cases} 1 & \text{if } e_2 > 2.414e_1 \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad q = \begin{cases} 1 & \text{if } e_2 > 0.414e_1 \\ 0 & \text{otherwise} \end{cases}$$

Then we have the edge direction $\theta = \begin{cases} 0^o & \text{if } p=0, q=0 \\ 45^o & \text{if } p=0, q=1 \\ 90^o & \text{if } p=1, q=1 \end{cases}$

The circuit diagram for finding the edge orientation is shown in Figure 4. In this figure, shift registers are used for multiplication or division by a power (positive or negative) of 2 using a clock pulse. Subsequent results are shown in the Figure 4.

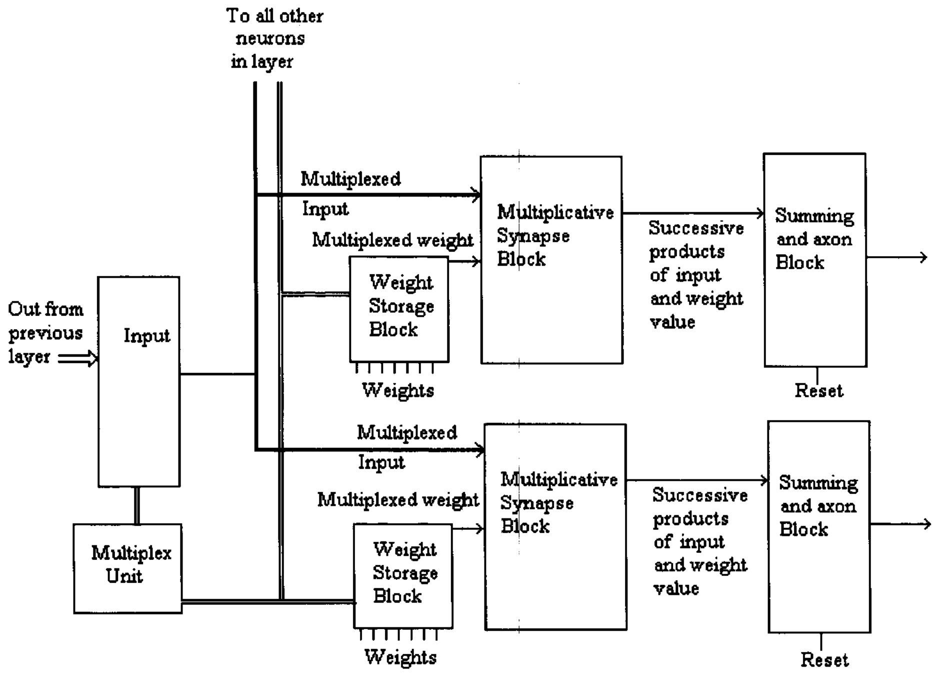

Figure 5: The functional decomposition of the task of a neuron.

Point/Line Detection

With the convolved output of the polynomial operators at our disposal, generation of operators for point or line detection is quite straightforward as discussed in Section 3. The division by 2 and multiplication by 3 are achieved by suitable shifts and adds as done in case of hardware design for direction processor. Similar procedures can be applied for computing convolutions with Sobel, Prewitt, Roberts or Laplacian operators in terms of the output of the polynomial convolution processor.

5.2. Analog VLSI Architecture

An analog VLSI architecture (Mead, 1989) for the neural network of Fig. 1 is presented in Fig. 5 and 6. The task of a neuron used in Fig. 1 is divided into different functional blocks such as the weight storage unit, multiplicative synapse, summing unit, axon (Fig. 5). In this neural network (Fig. 1), the connection weight is not updated during processing, that is, it maintains the initial weights.

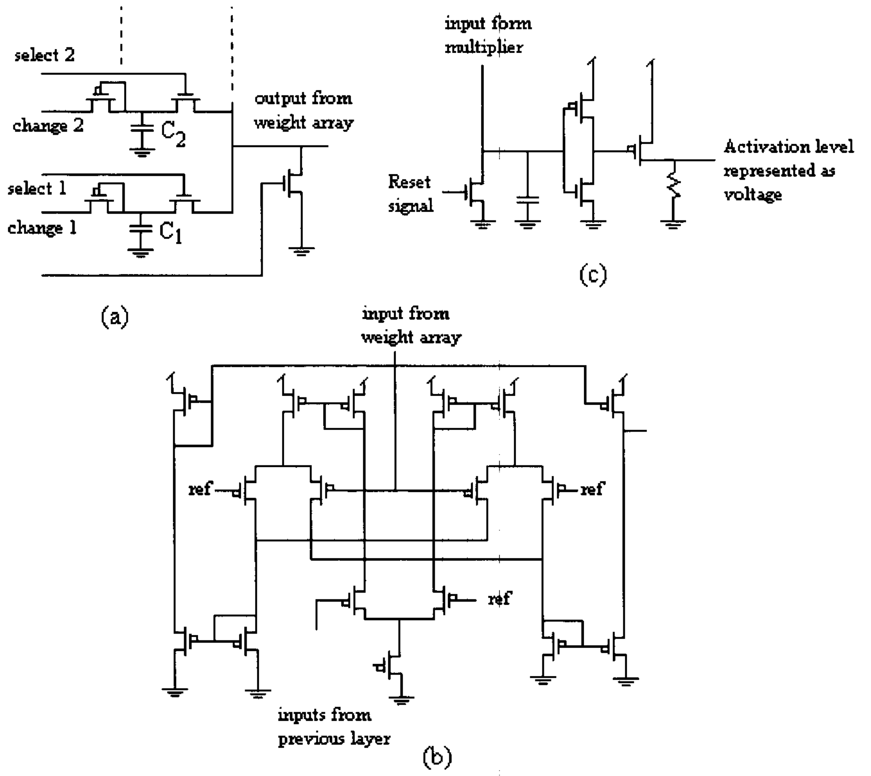

Figure 6: The hardware realization of (a) the weight storage block of a neuron using an array of capacitors, (b) the multiplicative block and (c) the summing blocks.

Working Principle of a Neuron

The working principle of a neuron described in Fig. 5 is based on the following assumptions.

Assumption: The neural network in Fig. 1 is a layer architecture and fully connected between the layers but no connection within a layer. If there is no connection between two neurons that means the connection weight between them is set to 0.

Initialize the weight storage unit block of each neuron using the given weights as well as reset the summing unit. A multiplex unit for a layer of the network (Fig. 1) helps to transmit an input from the previous layer of the network to each neuron of this layer. It also selects the corresponding weight from its weight storage unit block. The multiplicative synapse unit of each neuron of this layer computes the product of input and its weight. The product output of each neuron is stored into its summing unit. This multiplexing process continues till at finishes all inputs.

The weight storage block

The weights for a neuron are stored as voltages in an array of capacitors C_1, C_2, ... as shown in Fig 6(a). A significant capacitor is selected and its output as weight is realized by the common output line.

The multiplicative synapse block

The multiplex block, select an input and its corresponding weight that is (input, weight) pair for a neuron. Each of the (input, weight) pairs is applied in turn to the multiplicative synapse block of the neuron and produces an output as the product of the input and the weight of the pair. This task is performed by using the standard *Gilbert multiplier* (Mead, 1989) circuit (Fig 6(b)).

The summing and axon block

This summing circuit is very simple, it is only a capacitor. The axon is also a simple amplifier circuit (Fig. 6(c)). The input of the axon circuit is taken from the capacitor and the resulting output of it provides the activation level of the neuron.

6. CONCLUSION

This paper describes a set of basis operators for edge detection. These are designed based on a set of orthogonal polynomials for which the completeness criterion has been established and the measure of independence during approximation estimated. It has also been shown that other well-known edge detection gradient operators like Robert, Sobel, Prewitt, etc. can be expressed in terms of these polynomial based operators as a linear combination of them. A neural network is designed from these basis edge detection operators and that can be viewed as a class of well-known edge detection operators. Also a possible hardware realization has been described using simple hardware components like *registers, shift registers, adders, subtractors, comparators* and *simple AND/NOT* gates. Analog VLSI architecture is also proposed for future neuro-computer.

ACKNOWLEDGMENTS

The author gratefully acknowledges Prof. N. R. Pal and the reviewers for their valuable comments that led to considerable improvements of this paper.

APPENDIX I

Derivation of Eqn. (8)

Since P_i belongs to a set of orthogonal polynomials for all $1 \le i \le n$ then

$$\langle P_i, P_i \rangle = \sum_{1\le x\le n} P_i^2 \text{ and } \langle P_i, P_j \rangle = \sum_{1\le x\le n} P_i P_j = 0,\ i \ne j. \qquad (37)$$

Now, b_i's can be computed as follows.

$$\text{Let } i = 1, b_1(n) = \frac{\langle P_1, P_1 \rangle}{\langle P_0, P_0 \rangle} = \frac{1}{n}\sum (x-\mu)^2 = \text{var}(x) \qquad (38)$$

$$\text{then } P_2(x,\mu,n) = (x-\mu)P_1(x,\mu,n) - b_1(n)P_0(x,\mu,n) \qquad (39)$$

$$\text{Let } i = 2 \text{ then, } P_3(x,\mu,n) = (x-\mu)P_2(x,\mu,n) - b_2(n)P_1(x,\mu,n) \qquad (40)$$

Both sides of Eqn (40) is multiplied by $P_1(x,\mu,n)$ and taking $\sum$ over x then

$\sum P_1P_3 = \sum (x-\mu)P_1P_2 - b_2\sum P_1^2$. That is, $b_2\langle P_1, P_1 \rangle = \sum xP_1P_2$

Again, $\sum P_2^2 = \sum (x-\mu)P_1P_2 - b_1\sum P_0P_2$ [by Eqn. (39)]

Therefore, $\sum xP_1P_2 = \sum P_2^2 = \langle P_2, P_2\rangle$. Hence, $b_2 = \dfrac{\langle P_2, P_2\rangle}{\langle P_1, P_1\rangle}$.

Similarly, for $i = 3, P_4(x,\mu,n) = (x-\mu)P_3(x,\mu,n) - b_3(n)P_2(x,\mu,n)$ (41)

Both sides of Eqn. (41) multiplied by $P_2(x,\mu,n)$ and taking $\sum$ over x then

$\sum P_2P_4 = \sum(x-\mu)P_2P_3 - b_3\sum P_2^2$. Therefore, $b_3\langle P_2, P_2\rangle = \sum xP_2P_3$.

Again $\sum P_3^2 = \sum(x-\mu)P_2P_3 - b_2\sum P_1P_3$, that is, $\sum xP_2P_3 = \sum P_3^2 = \langle P_3, P_3\rangle$.

Hence, $b_3 = \dfrac{\langle P_3, P_3\rangle}{\langle P_2, P_2\rangle}$ (42)

Let us assume that the relation in Eqn. (8) is true for $i = k < n$.

$P_{k+1}(x,\mu,n) = (x-\mu)P_k(x,\mu,n) - b_k(n)P_{k-1}(x,\mu,n)$ (43)

$P_k(x,\mu,n) = (x-\mu)P_{k-1}(x,\mu,n) - b_{k-1}(n)P_{k-2}(x,\mu,n)$ and $b_k(n) = \dfrac{\langle P_k, P_k\rangle}{\langle P_{k-1}, P_{k-1}\rangle}$.

Now it is true for $i = k+1$, that is, $P_{k+2}(x,\mu,n) = (x-\mu)P_{k+1}(x,\mu,n) - b_{k+1}(n)P_k(x,\mu,n)$ (44)

Now both sides of Eqn. (44) is multiplied by $P_k(x,\mu,n)$ and taking $\sum$ over x then

$$\sum P_{k+2}P_k = \sum(x-\mu)P_kP_{k+1} - b_{k+1}\sum P_k^2. \quad (45)$$

Therefore, $b_{k+1}\langle P_k, P_k\rangle = \sum xP_kP_{k+1}$.

Again $\sum P_{k+1}^2 = \sum(x-\mu)P_kP_{k+1} - b_k\sum P_{k-1}P_{k+1}$ and $\sum xP_kP_{k+1} = \sum P_{k+1}^2 = \langle P_{k+1}, P_{k+1}\rangle$

Therefore, $b_{k+1}(n) = \dfrac{\langle P_{k+1}, P_{k+1}\rangle}{\langle P_k, P_k\rangle}$. (46)

APPENDIX II

Derivation of Eqn. (9)

Since P_i 's are orthogonal polynomials then $\sum P_i(x,\mu,n) = 0$ for $i > 1$.

Now, $P_1(x,\mu,n) = x-\mu$, where, $\mu = E(x) = \dfrac{n+1}{2} \Rightarrow \sum P_1(x,\mu,n) = 0$

Again, $P_2(x,\mu,n) = (x-\mu)P_1(x,\mu,n) - b_1(n)P_0(x,\mu,n)$ and $\sum P_2(x,\mu,n) = 0$.

Therefore, $b_1(n) = \frac{1}{n}\sum(x-\mu)^2 = \frac{1}{n}\sum_{1\le i\le n} i^2 - \left(\dfrac{n+1}{2}\right)^2 = \dfrac{n^2-1}{12} = \dfrac{1^2(n^2-1^2)}{4(4\times 1^2-1)}$ (47)

$$b_2(n) = \frac{\langle P_2, P_2\rangle}{\langle P_1, P_1\rangle} = \frac{\sum P_2^2}{\sum P_1^2} = \frac{\sum[(x-\mu)P_1 - b_1(n)P_0]^2}{\sum P_1^2} = \frac{1}{nb_1}\sum\left[(x-\mu)^2 - b_1\right]^2$$

$$= \frac{1}{nb_1}\left(\sum x^4 + 4nb_1\mu^2 + n(\mu^2+b_1)^2 - 4\mu\sum x^3 + 2(\mu^2-b_1)\sum x^2\right)$$

Now, $\sum_{1\le x\le n} x^4 = \dfrac{n^5}{5} + \dfrac{n^4}{2} + \dfrac{n^3}{3} - \dfrac{n}{30}$, $4nb_1\mu^2 = \dfrac{n^5}{12} + \dfrac{n^4}{6} - \dfrac{n^2}{6} - \dfrac{n}{12}$,

$n(\mu^2+b_1)^2 = \dfrac{n^5}{9} + \dfrac{n^4}{4} + \dfrac{13n^3}{36} + \dfrac{n^2}{6} + \dfrac{n}{36}$, $4\mu\sum x^3 = -\dfrac{n^5}{2} - \dfrac{3n^4}{2} - \dfrac{3n^3}{2} - \dfrac{n^2}{2}$, and

$2(\mu^2-b_1)\sum x^2 = \dfrac{n^5}{9} + \dfrac{n^4}{2} + \dfrac{7n^3}{9} + \dfrac{n^2}{2} + \dfrac{n}{9}$. That is, $nb_1b_2 = \dfrac{n^5}{180} - \dfrac{n^3}{36} + \dfrac{n}{45}$.

Therefore, $b_2(n) = \dfrac{n^2-4}{15} = \dfrac{2^2(n^2-2^2)}{4(4\times 2^2-1)}$. (48)

Since equation (9) is true for $i = 1, 2$. So by mathematical induction we can prove that the equation (9) is also true for any $i = 1, 2, \ldots, n$.

REFERENCES

1. Abdou, I. & and Pratt, W. K. (1979). Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors. *Proc. IEEE*, v.67, 753-763.
2. Ahmed, N. & Rao, K. (1975). Orthogonal Transforms for Digital Signal Processing. *Springer Verlag*.
3. Ando, S. (2000). Image Field Categorization and Edge/Corner Detection from Gradient Covariance. *IEEE Tr. PAMI*, v.22, 179-190.
4. Andrews, H. C. & Hunt, B.R.(1977). Digital Image Restoration. *Prentice Hall*, N.J.
5. Berzins, V.(1984). Accuracy of Laplacian Edge Detectors. *Comp. Vis., Graph. and Image Proc.*, v. 27, 195-210.
6. Bezdek, J. C., Keller J., Krisnapuram R. & Pal N. R.(1999). Fuzzy Models and Algorithms for Pattern Recognition and Image Processing. *Kluwer Academic Publication*, Boston.
7. Bhattacharyya, P. & Ganesan, L. (1997). An Orthogonal Polynomials Based Framework for Edge Etection in 2-D Monochrome Images. *Patt. Recog. Lett.*, v.18, 319-335.
8. Brown, M., Blackwell, K., Khalak, H., Barbour, G. & Vogl, T.(1998). Multi-Scale Edge Detection and Feature Binding: An Integrated Approach. *Patt. Recog.*, v. 31, 1479-1490.
9. Chanda, B., Kundu, M. & Padmaja, Y. (1998). A Multi-Scale Morphologic Edge Detector. *Patt. Recog.*, v. 31, 1469-1478.
10. Courant, R. & Hilbert, D. (1975). Methods of Mathematical Physics, Volume I. *Wiley Eastern*, New Delhi.
11. Canny, J. (1986). A Computational Approach to Edge Detection. *IEEE Tr. on PAMI*, v. 8, 679-698.
12. Davis, L. S. (1975). A Survey of Edge Detection Techniques. *Comp. Grap. and Image Proc.*, v. 4, 248-270.
13. Fleck, M.M. (1992). Some Defects in Finite-Difference Edge Finders. *IEEE Tr. on PAMI*, v. 14, 337-345.
14. Frank, G.A., Smith, C.U. & Cuadrado, J.L. (1985). Architecture Design and Assessment System for Software/Hardware Codesign. *Proc. 22nd Design Automation Conf.* 417-424.
15. Gonzalez, R.C. & Woods, R.E. (1999). Digital Image Processing. *Addison-Wesley*.
16. Haddon, J.F. (1988). Generalised Threshold Selection for Edge Detection. *Pattern Recog.*, v. 21, 195-203.
17. Hall, E.L. (1979). Computer Image Processing and Recognition. *Academic Press*, New York.
18. Haralick, R.M. (1984). Digital Step Edges from Zero Crossing of Second Directional Derivatives. *IEEE Tr. PAMI*, v. 6, 58-68.
19. Harmuth, H. (1972). Transmission of Information by Orthogonal Functions. 2nd Edn, New York.
20. Haykin, S. (2001). Neural Networks A Comprehensive Foundation. *Macmillan College Publishing Co.*, New York.
21. Heath, M., Sarkar, S., Sanocki, T. & Bowyer, K. (1998). Comparison of Edge Detectors: A Methodology and Initial Study. *Computer Vision and Image Understanding*, v. 69, 38-54.
22. Huang, K.S. (1989). Binary Image Algebra and Optical Cellular Logic Processor Design. *Computer Vision, Graphics, Image Processing*, v. 45, 295-345.
23. Hueckel, M.F. (1973). A Local Visual Operator which Recognizes Edges and Lines. *JACM*, v. 20, 634-647.
24. Hueckel, M.F. (1971). An Operator which Locates Edges in Digitized Pictures. *JACM*, v. 18, 113-125.
25. Jain, A.K. (1989). Fundamentals of Digital Image Processing. *Prentice Hall*, New Jersey.
26. Kang, C.C. & Wang, W.J. (2007). A Novel Edge Detection Method Based on the Maximizing Objective Function, *Patt. Recog.*, v. 40, 609-618.
27. Kanopoulos, N., Vasanthavada, N. & Baker, R.L. (1988). Design of an Image Edge Detection Filter Using the Sobel Operator. *IEEE Tr. Solid State Circuits*, v. 23, 358-367.

28. Kim D.S., Lee W.H. & Kweon I.S. (2004). Automatic Edge Detection Using 3 × 3 Ideal Binary Pixel Patterns and Fuzzy-Based Edge Thresholding, *Patt. Recog. Lett.* v. 25, 101-106.
29. Kittler, J. & Duff, M.J.B. (1985). Image Processing System Architectures. *John Wiely and Sons Inc.*
30. Liang L.R. & Looney C.G. (2003). Competitive Fuzzy Edge Detection, *Appl. Soft Comput. J.* v. 3, 123-137.
31. Mansouri, A.R., Malowang, A.S. & Levine, M.D. (1987). Line Detection in Digital Pictures: A Hypothesis Prediction/Verification Paradigm. *Computer Vision, Graphics and Image Processing*, v. 40, 95-114.
32. Marr, D. (1982). VISION A Computational Investigation into the Human Representation and Processing of Visual Information. *W.H. Freeman and Company*, U.S.A.
33. Marr, D. & Hildreth, E. (1980). Theory of Edge Detection. *Proc. Royal Society of Lond.*, v. 207B, 187-217.
34. Mehrotra, R. & Zhan, S. (1996). A computational Approach to Zero-Crossing-Based Two-Dimensional Edge Detection. *Graph. Models Image Process.*, v. 58, 1-17.
35. Meer, P. & Georgescu, B. (2001). Edge Detection with Embedded Confidence. *IEEE Tr. PAMI* v. 23, 1351-1365.
36. Mead, C. (1989). Analog VLSI and Neural Systems. *Addison-Wesley.*
37. Milgram, M. & Pierre, T.D.S. (1990). Boundary Detection and Skeletonization with Massively Parallel Architecture. *IEEE Tr. PAMI*, v. 12, 74-78.
38. Nevatia, R. (1977). Evaluation of a Simplified Hueckel Edge-Line Detector. *Computer Graphics & Image Processing* v.6, 582-588.
39. O'Gorman, F. (1978). Edge Detection Using Walsh Functions. *Artificial Intell.*, v. 10, 215-223.
40. Pal, S. (1991). Some Low Level Image Segmentation Methods, Algorithms and their Analysis. *Ph.D. Thesis, Department of Computer Science and Engineering, Indian Institute of Technology, Kharagpur.*
41. Pratt, W.K. (1978). Digital Image Processing. *John Wiley & Sons*, New York, 1978.
42. Prewitt, J.M.S. (1970). Object Enhancement and Extraction. *Picture Processing and Psychopictorics* (Lipkin B S and Rosenfeld A, Eds), Academic Press.
43. Roberts, L.G. (1965). Machine Perception of Three-Dimensional Solids. *Optical and Electro-Optical Information Processing*, Tippet, J.T., ed., MIT Press.
44. Rakesh R.R.,Chaudhuri P. & Murthy C. A. (2004). Thresholding in Edge Detection: A Statistical Approach, *IEEE Tr. Image Process.*, v. 13, 927-936.
45. Rosenfeld, A. & Kak, A.C. (1982). Digital Picture Processing. 2nd ed., *Academic Press*, New York.
46. Rosenfeld, A. (1981). The Max Roberts Operator is a Hueckel-type of Edge Detector. *IEEE Tr. PAMI* v. 3, 101-103.
47. Ruetz, P.A. & Brodersen, R.W. (1987). Architectures and Design Techniques for Real-Time Image Processing IC's. *IEEE Tr. Solid-State Circuits*, V. 22, pp. 233-250.
48. Shin, M.C., Goldgof, D. B., Bowyer, K.W. & Nikiforou S. (2001). Comparison of Edge Detection Algorithms using a Structure from Motion Task. *IEEE Tr. SMC B*, v. 31, 589-601.
49. Smith, R.W. (1987). Computer Processing of Line Images: A Survey. *Pattern Recognition*, v. 20, 7-15.
50. Sobel, I. E. (1970). Camera Models and Machine Perception. *Ph.D. Thesis, Stanford University*, 277-284.
51. Tadrous, P. (1995). A Simple and Sensitive Method for Directional Edge Detection in Noisy Images. *Patt. Recog.*, 28, 1575-1586.
52. Torre, V. & Poggio, T A. (1986). On Edge Detection. *IEEE Tr. PAMI*, v. 8, 147-163.
53. Vitulano, S., Ruberto, C.D. & Nappi, M. (1998). Edge Detection: Local and Global operators. *Int. J. of PR and AI*, v. 12, 677-694.
54. Vliet, L.J.V. & Young I T. (1989). A Nonlinear Laplace Operator as Edge Detector in Noisy Images. *Computer Vision, Graphics and Image Processing*, v. 45, 167-195.