International Journal of Electronics
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tetn20

A self-correction Hopfield neural network for computing the bit-level transform image coding

PO-RONG CHANG

Published online: 10 Nov 2010.

To cite this article: PO-RONG CHANG (1997) A self-correction Hopfield neural network for computing the bit-level transform image coding, International Journal of Electronics, 83:2, 215-234, DOI: 10.1080/002072197135544

To link to this article: http://dx.doi.org/10.1080/002072197135544

INT. J. ELECTRONICS, 1997, VOL. 83, NO. 2, 215–233

A self-correction Hopfield neural network for computing the bit-level transform image coding

PO-RONG CHANG†

A Hopfield-type neural network approach is presented, which leads to an analogue circuit for implementing the bit-level transform image. Unlike the conventional digital approach to image coding, the analogue coding system would operate at a much higher speed and it requires less hardware than a digital system. To utilize the concept of neural net, the computation of a two-dimensional DCT-based transform coding should be reformulated as minimizing a quadratic nonlinear programming problem subject to the corresponding 2s complement binary variables of two-dimensional DCT coefficients. A Hopfield-type neural net with a number of graded-response neurons designed to perform the quadractic nonlinear programming would lead to such a solution, in a time determined by RC time constants, not by algorithmic time complexity. Nevertheless, the existance of local minima in the energy function of the original Hopfield model implies that in general the correct globally optimal solution is not guaranteed. To tackle this difficulty, a network with an additional self-correction circuitry is developed to eliminate these local minima and yields the correct digital representations of 2-D DCT coefficients. A fourth-order Runge–Kutta simulation is conducted to verify the performance of the proposed analogue circuit. Experiments show that the circuit is quite robust and independent of parameter variations, and the computation time of an 8×8 DCT is estimated as 128 ns for $RC = 10^{-9}$.

1. Introduction

The goal of transform image coding is to reduce the bit-rate to minimize communication channel capacity or digital storage memory requirements while maintaining the necessary fidelity of data. The discrete cosine transform (DCT) has been widely recognized as the most effective among various transform coding methods for image and video signal compression. However, it is computationally intensive and is very costly to implement using discrete components. Many investigators have explored ways and means of developing high-speed architectures for real-time image data coding (Sun *et al.* 1987, Liou and Bellisio 1987). Up to now, all image coding techniques, without exception, have been implemented by digital systems using digital multipliers, adders, shifters and memories. As an alternative to the digital approach, an analogue approach based on a Hopfield-type neural network (Hopfield 1984, Tank and Hopfield 1986) is presented.

Neural network models have received more and more attention in many fields where high computation rates are required. Hopfield and Tank showed that the neural optimization network can perform some signal-processing tasks, such as the signal decomposition/decision problem. Culhane *et al.* (1989) applied their concepts to discrete Hartley and Fourier transforms. Chua and Lin (1988) and Chang *et al.* (1991) proposed an analogue approach based on Hopfield neural network to

Received 21 November 1996; accepted 16 December 1996.

†Department of Communication Engineering, National Chiao-Tung University, Hsin-Chu, Taiwan, Republic of China. Fax: + 886 35 710116; e-mail:prchang@cc.nctu.edu.tw.

implement the two-dimensional (2-D) DCT transform. Chang *et al.* (1991) demonstrated that the computation time for the 2-D DCT transform is within the *RC* time constants of the neural analogue circuit. Owing to the nature of the energy function of the Hopfield neural network, the solution of this network is highly dependent on its initial state. The energy function may decrease to settle down at one of the equilibrium points called 'spurious states' or local minima that does not correspond to the exact digital representations of the 2-D DCT coefficients. Moreover, Lee and Sheu (1988, 1989) investigated the conditions for determining the local minima and detailed analysis on the equilibrium properties of Hopfield networks. Simulated annealing (Kirkpatrick *et al.* 1983) is one heuristic technique to help escape the local minima by perturbing the energy function with the annealing temperature and artificial noise. It is proven that the solution obtained by the simulated annealing is independent of the initial state and is very close to the global minimum. As the network should settle down at each temperature and the temperature decrement is very small, an extraordinary long time is required in the computation and it does not meet the real-time requirement. A different approach to eliminating the local minima has been proposed by Lee and Sheu (1989). Their method is based on adding an additional self-correction circuitry to the original Hopfield network, and it yields the global minimum in real time. Here we use Lee and Sheu's concept in our design.

In this paper a neural-based optimization formulation is proposed to solve the two-dimensional (2-D) discrete cosine transform in real time. It is known that the direct computation of a 2-D DCT of size $L \times L$ is to perform the triple matrix product of an input image matrix and two orthonormal base matrices. After proper arrangements, the triple matrix product can be reformulated as minimizing a large-scale quadratic nonlinear programming problem subject to $L \times L$ DCT coefficient variables. However, a decomposition technique is applied to divide the large-scale optimization problem into $L \times L$ smaller-scale subproblems, each of which depends on its corresponding 2-D DCT coefficient variable only and then can be easily solved. To achieve the digital video applications, each 2-D DCT coefficient variable should be considered in the 2s complement binary representation. Therefore, each subproblem has been changed to be a new optimization problem subject to a number of binary variables of the corresponding 2-D DCT coefficient. Indeed, the new optimization problem is also a quadratic programming with minimization which occurs on the corners of the binary hypercube space. This is identical to the energy function involved in the Hopfield neural model (Hopfield 1984, Tank and Hopfield 1986). They showed that a neural net has associated with it an 'energy function' which the net always seeks to minimize. As the energy function of Hopfield model has many local minima, the network output is usually the closest local minimum to the initial state which may be not identical to the desired DCT coefficient. A self-correction circuitry for the neural-based DCT transform coder is developed in § 5 to improve the probability of finding the correct global minimum solution. With the extra self-correction logic, the energy function decreases until the net reaches a steady-state solution (global minimum point which is the desired 2-D DCT coefficient. Experimental results shown in § 7 have been conducted to verify the performance of the self-correction neural network. It is seen that the architecture of the neural net designed to perform the 2-D DCT would, therefore, reach a solution in a time determined by *RC* time constants, not by algorithmic time complexity, and would be straightforward to fabricate.

2. An optimization formulation for the transform image coding

The Discrete cosine transform (DCT) is an orthogonal transform that consists of a set of basis vectors that are sampled cosine functions A normalized Lth-order DCT matrix $\mathbf{U}$ is defined by

$$u_{st} = \left(\frac{2}{L}\right)^{1/2} \cos\left[\frac{\pi(2s+1)t}{2L}\right] \tag{1}$$

for $0 \le s \le L-1$, $1 \le t \le L-1$ and $u_{st} = L^{-1/2}$ for $t = 0$. The two-dimensional (2-D) DCT of size $L \times L$ is defined as

$$\mathbf{Y} = \mathbf{U}^{\mathrm{T}}\mathbf{X}\mathbf{U} \tag{2}$$

where $\mathbf{U}^{\mathrm{T}}$ is the transpose of $\mathbf{U}$, and $\mathbf{X}$ is the given image data block of size $L \times L$ (typically 8×8 or 16×16).

Traditionally, the resultant matrix in the transform domain $\mathbf{Y}$ may be obtained by a direct implementation of (2) which is computationally intensive. By taking the advantage of the high-speed analogue implementation of the Hopfield-type neural network (Hopfield 1984, Tank and Hopfield 1986), the following formulations are required and would be described as follows.

From (2), we have

$$\begin{aligned}\mathbf{X} &= \mathbf{U}\mathbf{Y}\mathbf{U}^{\mathrm{T}} \\ &= \sum_{i=0}^{L-1}\sum_{j=0}^{L-1} y_{ij}\boldsymbol{u}_i\boldsymbol{u}_j^{\mathrm{T}}\end{aligned} \tag{3}$$

where y_{ij} denotes the (i,j) entry of $\mathbf{Y}$ and $\boldsymbol{u}_i$ is the ith column vector of $\mathbf{U}$.

Define the distance or norm between two matrices $\mathbf{A}$ and $\mathbf{B}$ to be

$$\mathrm{NORM}(\mathbf{A},\mathbf{B}) = \mathrm{tr}(\mathbf{A}^{\mathrm{T}}\mathbf{B}) \tag{4}$$

where $\mathrm{tr}(\mathbf{A})$ is equal to $\sum_{j=0}^{L-1} a_{ii}$.

Let $\Delta = \mathbf{X} - \mathbf{U}\mathbf{Y}\mathbf{U}^{\mathrm{T}}$ and $\|\Delta\|^2 = \mathrm{NORM}(\Delta,\Delta)$. Therefore, the coefficients y_{ij} in (3) minimize the distance function

$$\min_{\substack{y_{ij} \\ 0 \le i,j \le L-1}} \|\Delta\|^2 \; (= \|\mathbf{X} - \mathbf{U}\mathbf{Y}\mathbf{U}^{\mathrm{T}}\|^2) \tag{5}$$

In this way, given $\mathbf{X}$, the problem of computing $\mathbf{Y}$ by (3) has been changed into the problem of finding the minimum $\mathbf{Y} = [y_{ij}]$ of the function $\|\Delta\|^2$ in (5).

To reduce the complexity of performing the optimization problem in (5), $\|\Delta\|^2$ can be rewritten in the following form:

$$\|\Delta\|^2 = \sum_{i=0}^{L-1}\sum_{j=0}^{L-1}\|\mathbf{X} - y_{ij}\boldsymbol{u}_i\boldsymbol{u}_j^{\mathrm{T}}\|^2 - (L^2-1)\,\mathrm{tr}(\mathbf{X}^{\mathrm{T}}\mathbf{X}) \tag{6}$$

Observing (6), it should be noted that the second term of the right-hant side of (2) is constant, and the components involved in the summation of the first term are independent of each other. Therefore, the minimization problem (5) could be divided into L^2 subproblems as follows:

$$\min_{y_{ij}} \|\Delta_{ij}\|^2 \; (= \|\mathbf{X} - y_{ij}\boldsymbol{u}_i\boldsymbol{u}_j^{\mathrm{T}}\|^2), \quad 0 \le i,j \le L-1 \tag{7}$$

Indeed, (7) can be expanded and rearranged in the scalar form

$$\min_{y_{ij}} \|\Delta_{ij}\|^2 \left(= \sum_{s=0}^{L-1} \sum_{t=0}^{L-1} (x_{st} - y_{ij} u_{is} u_{tj})^2 \right) \tag{8}$$

This decomposition approach provides us with a technique to divide a large-scale optimization problem into a number of smaller-scale subproblems, each of which can be easily solved.

Owing to the requirement of many digital video applications, each y_{ij} is quantized into $\hat{y}_{ij}$ which can be represented by the 2s complement codes as follows:

$$\hat{y}_{ij} = -s_{ij}^{(m_{ij})} 2^{m_{ij}} + \sum_{p=-n_{ij}}^{m_{ij}-1} s_{ij}^{(p)} 2^p \tag{9}$$

where $s_{ij}^{(p)}$ is the pth bit of $\hat{y}_{ij}$ which has a value of either 0 or 1; $s_{ij}^{(m_{ij}-1)}$ is the most significant bit (MSB), $s_{ij}^{(-n_{ij})}$ is the least significant bit, and $s_{ij}^{(m_{ij})}$ is the sign bit.

By substituting (9) into (8), one may obtain the new minimization problem subject to the binary variables, $s_{ij}^{(p)} = 0$ or 1, $-n_{ij} \le p \le m_{ij}$; that is

$$\min_{\substack{s_{ij}^{(p)} \\ -n_{ij} \le p \le m_{ij}}} \|\Delta_{ij}\|^2 \tag{10}$$

In the following section a novel neural-based optimizer is proposed to solve the above minimization problem, to meet the real-time requirement of many digital video applications.

3. A neural-based optimization approach

Artificial neural networks contain a large number of identical computing elements or neurons with specific interconnection strengths between neuron pairs. The massively parallel processing power of neural network in solving difficult problems lies in the cooperation of highly interconnected computing elements. It is shown that the speed and solution quality obtained when using neural networks for solving specific problems in signal processing make specialized neural network implementations attractive. For instance, the Hopfield network can be used as an efficient technique for solving various combinatorial problems (Hopfield and Tank 1985) by the programming of synaptic weights stored as a conductance matrix.

The Hopfield model is a popular model of continuous interconnected N nodes. Each node is assigned a potential, $u_p(t)$, $p = 1, 2, \ldots, N$, as its state variable. Each node receives external input bias $I_p(t)$ and internal inputs from other nodes in the form of a weighted sum of firing rates $\sum_q T_{pq} g_q(\lambda_q u_q)$, where $g_q(\cdot)$ is a monotonically increasing sigmoidal bounded function converting potential to firing rate. The general structure of the networks is shown in Fig. 1. The equations of motion are

$$C\frac{\mathrm{d}u_p}{\mathrm{d}t} = -\frac{u_p}{R} + \sum_{q=1}^{N} T_{pq} v_q + I_p$$
$$v_p = g_p(\lambda_p u_p) \tag{11}$$

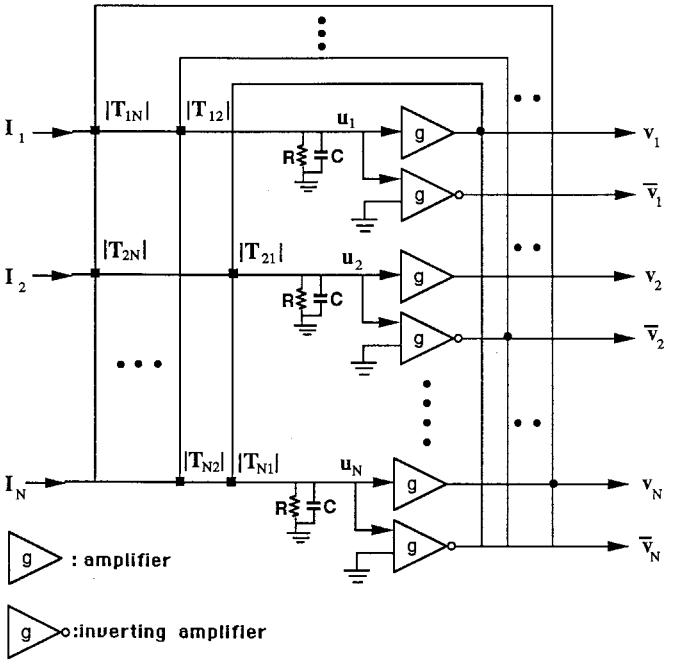


Figure 1. The circuit schematic of Hopfield model. Black squares at intersections represent resistive connections ($|T_{pq}|$). If $T_{pq} < 0$, the resistor is connected to the inverted amplifier.

where λ_p are the amplifier gains and $g_p(\lambda_p u_p)$ is typically identified as $\frac{1}{2}(1 + \tanh \lambda_p u_p)$).

Electrically, $T_{pq} v_q$ might be understood to represent the electrical current input to neuron p due to the present potential of neuron q. The quantity $|T_{pq}|$ represents the finite conductance between the output v_q and the body of neuron p. In other words, this connection is made with a resistor of value $R_{pq} = 1/|T_{pq}|$. If $T_{pq} > 0$ this resistor is conducted to the normal output of amplified q. If $T_{pq} < 0$ it is conducted to the inverted output of amplifier q. It would also be considered to represent the synapse efficacy. The term $-u_p/R$ is the current flow due to finite transmembrane resistance R, and it causes a decrease in u_p. I_p is any other (fixed) input bias current to neuron p. Thus, acording to (11), the change in u_p is due to the changing action of all the $T_{pq} v_q$ terms, balanced by the decrease due to $-u_p R$, with a bias set by I_p.

Hopfield and Tank have shown that in the case of symmetric connections ($T_{pq} = T_{qp}$), the equations of motion for this network of analogue processors always lead to a convergence to stable states, in which the output voltages of all amplifiers remain constant. In addition, when the diagonal elements (T_{pp}) are 0 and the amplifier gains λ_p are high, the stable states of a network composed of N neurons are the minima of the computational energy of Lyapunov function

$$E = -\frac{1}{2}\sum_{p=1}^{N}\sum_{q=1}^{N} T_{pq} v_p v_q - \sum_{p=1}^{N} I_p v_p \qquad (12)$$

The state space over which the analogue circuit operates is the N-dimensional hypercube defined by $v_p = 0$ or 1. However, it has been shown that in the high-gain limit networks with vanishing diagonal connections ($T_{pp} = 0$) have minima only at corners of this space (Tank and Hopfield 1986). Under these conditions the stable states of the network correspond to those locations in the discrete space consisting of the 2^N corners of this hypercube which minimize E.

To solve the minimization problem in (10) by the Hopfield-type neural network, the binary variables $s_{ij}^{(p)}$ should be assigned to their corresponding potential variables u_p with $N\ (= m_{ij} + n_{ij} + 1)$ neurons. Truly, the computational energy function E^{ij} of the proposed network for y_{ij} may be identified as $\|\Delta_{ij}\|^2$ in (10). However, with this simply energy function there is no guarantee that the values of $s_{ij}^{(p)}$ will be near enough to 0 or 1 to be identified as digital logic. As (10) contains diagonal elements of the **T**-matrix that are non-zero, the minimal points to the $\|\Delta_{ij}\|^2$ in (10) will not necessarily lie on the corners of the hypercube, and thus may not represent the exact 2s complement digital representation. One can eliminate this problem by adding one additional term to the function $\|\Delta_{ij}\|^2$. Its form can be chosen as

$$\Delta E^{ij} = \sum_{s=0}^{L-1}\sum_{t=0}^{L-1}\left\{\left[\sum_{p=m_{ij}}^{m_{ij}} s_{ij}^{(p)}(1 - s_{ij}^{(p)})2^{2p}\right](u_{is})^2(u_{tj})^2\right\} \tag{13}$$

The structure of this term was chosen to favour digital representations. Note that this term has the minimal value when, for each p, either $s_{ij}^{(p)} = 1$ or $s_{ij}^{(p)} = 0$. Although any set of (negative) coefficients will provide this bias towards a digital representation, the coefficients in (13) were chosen to cancel out the diagonal elements in (10). The elimination to diagonal connection strengths will generally lead to stable points only at corners of the hypercube. Thus the new total energy function E^{ij} for y_{ij} which contains the sum of the two terms in (10) and (13) has minimal value when the $\{s_{ij}^{(p)}\}$ is a digital representation close to the resultant y_{ij} in (3). After expanding and rearranging the energy function E^{ij}, we have

$$\begin{aligned} E^{ij} &= \|\Delta_{ij}\|^2 + \Delta E^{ij} \\ &= -\frac{1}{2}\sum_{p=m_{ij}}^{m_{ij}}\sum_{q=m_{ij}}^{m_{ij}} s_{ij}^{(p)} s_{ij}^{(q)} T_{pq}^{ij} \\ &\quad - \sum_{p=m_{ij}}^{m_{ij}} s_{ij}^{(p)} I_p^{ij} \end{aligned} \tag{14a}$$

where

$$T_{pq}^{ij} = \begin{cases} 0, & \text{for } p = q \\ -2\sum_{s=0}^{L-1}\sum_{t=0}^{L-1} 2^{p+q} u_{is}^2 u_{tj}^2, & \text{for } p \neq q, p \neq m_{ij}, q \neq m_{ij} \\ 2\sum_{s=0}^{L-1}\sum_{t=0}^{L-1} 2^{p+q} u_{is}^2 u_{tj}^2, & \text{for } p \neq q, p \neq m_{ij}, q = m_{ij} \\ 2\sum_{s=0}^{L-1}\sum_{t=0}^{L-1} 2^{p+q} u_{is}^2 u_{tj}^2, & \text{for } p \neq q, p = m_{ij}, q \neq m_{ij} \end{cases} \tag{14b}$$

$$I_p^{ij} = \begin{cases} \sum_{s=0}^{L-1}\sum_{t=0}^{L-1}(2^{p+1}u_{is}u_{tj}x_{st} - 2^{2p}u_{is}^2u_{tj}^2), & \text{for } p \neq m_{ij} \\ -\sum_{s=0}^{L-1}\sum_{t=0}^{L-1}(2^{p+1}u_{is}u_{tj}x_{st} + 2^{2p}u_{is}^2u_{tj}^2), & \text{for } p = m_{ij} \end{cases} \tag{14c}$$

As it can be shown that the term $(\sum_{s=0}^{L-1}\sum_{t=0}^{L-1}u_{is}^2u_{tj}^2)$ is identical to unity for $0 \leq i, j \leq L-1$, (14*b*) and (14*c*) become

$$T_{pq}^{ij} = \begin{cases} 0, & \text{for } p = q \\ -2^{p+q+1} & \text{for } p \neq q, p \neq m_{ij}, q \neq m_{ij} \\ 2^{p+q+1} & \text{for } p \neq q, p \neq m_{ij}, q = m_{ij} \\ 2^{p+q+1} & \text{for } p \neq q, p = m_{ij}, q \neq m_{ij} \end{cases} \tag{15a}$$

$$I_p^{ij} = \begin{cases} 2^{p+1}v_{s(ij)} + 2^{2p}V_R, & \text{for } p \neq m_{ij} \\ -2^{p+1}v_{s(ij)} + 2^{2p}V_R, & \text{for } p = m_{ij} \end{cases} \tag{15b}$$

where $v_{s(ij)}$ is the analogue input voltage and it equals $(\sum_{s=0}^{L-1}\sum_{t=0}^{L-1}u_{is}u_{tj}x_{xt})$, and V_R is the reference voltage and it equals 1 V.

Observing (15*a*), it may be found that the synapse weights T_{pq}^{ij} do not include the index (i,j) of their corresponding results y_{ij} directly. Meanwhile, the range of T_{pq}^{ij} depends on the (i,j)-related parameters m_{ij} and n_{ij} inherently. Moreover, the energy function for the simple Hopfield neural-based transform coding circuitry shown in Fig. 2 can be rewritten as

$$E = -\frac{1}{2}\sum_{p=-n}^{m}\sum_{q=-n}^{m}T_{pq}v_pv_q - \sum_{p=-n}^{m}(T_{pR}V_R + T_{ps}V_s)v_p \tag{16}$$

where v_p corresponds to $s_{ij}^{(p)}$, and T_{pR} is the conductance between the pth amplifier and the reference voltage V_R, and T_{ps} is the conductance between the pth amplifier input and analogue input voltage. Note that the index (i,j) is not included in (16) for the sake of simplicity. The values for T_{pR} and T_{ps} are given by

$$T_{pR} = 2^{2p} \tag{17a}$$

$$T_{ps} = \begin{cases} 2^{p+1}, & \text{for } -n \leq p \leq m-1 \\ -2^{p+1}, & \text{for } p = m \end{cases} \tag{17b}$$

The corresponding circuit dynamics of the simple Hopfield neural-based transform coding can be described as

$$C\frac{\mathrm{d}u_p}{\mathrm{d}t} = -\left(T_{pR} + T_{ps} + \sum_{q=-n}^{m}T_{pq}\right)u_p + \sum_{q=-n}^{m}T_{pq}v_q + T_{pR}V_R + T_{ps}V_s$$

$$v_p = g_p(\lambda_p u_p) \tag{18}$$

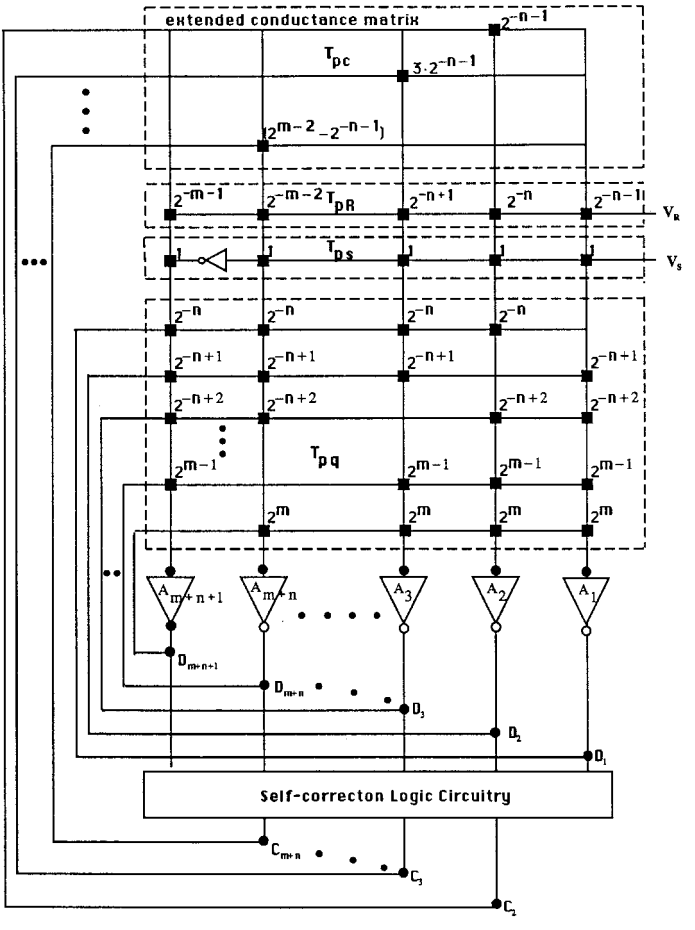


Figure 2. $A_n(m + n + 1)$-bit self-correction Hopfield neural network. $D_i = v_{i-n-1}$, $1 \le i \le m + n + 1$ and $C_l = f_{l-n-1}(V_{-n}, \ldots V_m)$, $2 \le l \le m + n$.

4. Local minima of Hopfield neural-based transform coding

Lee and Sheu (1988, 1989) showed that the energy function of a Hopfield network has many local minima and the resultant network output is the closest local minimum to the initial state. The existance of local minima in the energy function of the Hopfield network is not tolerable for a great variety of engineering optimization applications. In this section we discuss the properties of the local minima of neural-based transform image coding.

At a stable point, the term $(C\, \mathrm{d}u_p/\mathrm{d}t)$ of (18) becomes zero and every neuron input voltage is governed by

$$u_p \begin{cases} > 0 & \text{when } v_p = 1\,\mathrm{V} \\ < 0 & \text{when } v_p = 0 \end{cases} \tag{19}$$

From (18) and (19), the range of input voltage V_s to the pth amplifier for the corresponding stable state and a specific digital output can be calculated:

$$V_s\left\{\begin{array}{l} > -\left(\dfrac{T_{pR}}{T_{ps}}\right) V_R - \displaystyle\sum_{q=-n}^{m}\left(\dfrac{T_{pq}}{T_{ps}}\right) v_q, \quad \text{when } v_p = 1\,\text{V and} \quad -n \le p \le m-1 \\ < -\left(\dfrac{T_{mR}}{T_{ms}}\right) V_R - \displaystyle\sum_{q=-n}^{m-1}\left(\dfrac{T_{mq}}{T_{ms}}\right) v_q, \quad \text{when } v_m = 1\,\text{V} \end{array}\right\} \tag{20a}$$

$$V_s\left\{\begin{array}{l} < -\left(\dfrac{T_{pR}}{T_{ps}}\right) V_R - \displaystyle\sum_{q=-n}^{m}\left(\dfrac{T_{pq}}{T_{ps}}\right) v_q, \quad \text{when } v_p = 0 \quad \text{and} \quad -n \le p \le m-1 \\ > -\left(\dfrac{T_{mR}}{T_{ms}}\right) V_R - \displaystyle\sum_{q=-n}^{m-1}\left(\dfrac{T_{mq}}{T_{ms}}\right) v_q, \quad \text{when } v_m = 0 \end{array}\right\} \tag{20b}$$

Substitute (15 *a*), (17 *a*) and (17 *b*) into (20 *a*) and (20 *b*); this yields

$$V_s\left\{\begin{array}{ll} > 2^{p-1} + \displaystyle\sum_{q=-n}^{m} 2^q v_q - v_m 2^m, & \text{when } v_p = 1\,\text{V} \quad \text{and} \quad -n \le p \le m-1 \\ < -2^{m-1} + \displaystyle\sum_{q=-n}^{m-1} 2^q v_q, & \text{when } v_m = 1\,\text{V} \end{array}\right\} \tag{21a}$$

$$V_s\left\{\begin{array}{ll} < 2^{p-1} + \displaystyle\sum_{q=-n}^{m-1} 2^q v_q - v_m 2^m, & \text{when } v_p = 1\,\text{V} \quad \text{and} \quad -n \le p \le m-1 \\ > -2^{m-1} + \displaystyle\sum_{q=-n}^{m-1} 2^q v_q, & \text{when } v_m = 0 \end{array}\right\} \tag{21b}$$

During the transient period, the voltage v_p is changing in the direction that the energy function E decreases. When all amplifiers reach the stable condition in (19), a local minimum is reached and the searching process is terminated. The inequalities (21 *a*) and (21 *b*) are derived for the pth amplifier output voltage to be stable. For a stable output, the input voltage range is given by the logic-AND operation of the range decided by each amplifier. From (21 *a*) and (21 *b*), it is indicated that the lower limit and upper limit of V_s are determined by the first high-bit occurrence and low-bit occurrence of the digital code from the least significant bit (LSB), respectively. In other words, if a digital code has the first low bit at the pth bit, the next adjacent digital code has the first high bit at the pth bit. Thus, the upper limit and lower limit of input voltage between the two adjacent digital codes are decided by the pth amplifier.

To justify the behaviour of local minima, a characteristic parameter GAP_p proposed by (Lee and Sheu 1988, 1989) is used as an indicator of the existance of local minima. The parameter GAP_p is defined by the input voltage of the lower limit for $v_p = 1\,\text{V}$ and the upper limit for $v_p = 0$ decioded by the pth amplifier

$$\mathrm{GAP}_p = \left\{\begin{array}{ll} -\displaystyle\sum_{q=-n}^{m}\left(\dfrac{T_{pq}}{T_{ps}}\right)\left(v_q^u - v_q^l\right), & \text{for } -n \le p \le m-1 \\ -\displaystyle\sum_{q=-n}^{m-1}\left(\dfrac{T_{mq}}{T_{ms}}\right)\left(v_q^l - v_q^u\right), & \text{for } p = m \end{array}\right. \tag{22}$$

where v_p^u and v_p^l are the pth amplifier output voltages of the digital codes whose pth bits from the least significant bit (LSB) are logic 1 and logic 0, respectively. $\{v_q^u\}$ and $\{v_q^l\}$ are usually the adjacent digital codes and are given by

$$\left.\begin{array}{ll} v_q^u = v_q^l, & \text{if } q > p \\ v_q^u = 1\,\text{V and } v_q^l = 0, & \text{if } q = p \\ v_q^u = 0 \text{ and } v_q^l = 1\,\text{V}, & \text{if } q < p \end{array}\right\} \tag{23}$$

As GAP_p represents the overlapped range of the input voltage V_s to the pth amplifier, both adjacent digital codes can be stable and become the same converted output at a given input voltage. To guarantee that one of the adjacent digital codes is not a local minimum when the other code is the global minimum at a given input, the input voltage ranges for the codes should not be overlapped, i.e. $\text{GAP}_p \geq 0$ for every p. The only global minimum corresponds to each analogue input. Hence, the one-to-one correspondence between the digital output and analogue input should exist. Let us examine the existance of local mimina in neural-based transform coding. By substituting (23) into (22) we have

$$\text{GAP}_p = \begin{cases} -2^p + 2^{-n}, & \text{for } -n \leq p \leq m-1 \\ 2^m - 2^{-n}, & \text{for } p = m \end{cases} \tag{24}$$

Equation (24) shows that the indicator GAP_p is always negative except when $p = -n$ and $p = m$. Thus, there could exist more than two digital output codes (i.e. local minima) corresponding to a given analogue input for $-n < p < m$. In the next section a self-correction logic based on Lee and Sheu's concept is proposed to eliminate the overlapped input range.

5. Self-correction logic approach

The overlapped input voltage range between two digital codes can be eliminated by adding a correction logic circuitry at the amplifier (neuron) outputs as shown in Fig. 2. The values of conductances located in the main body of the neural network and the extended conductance network will be discussed and specified by (31)–(34). Here we give the overall structure to understand the function of correction logic circuitry. The correction logic monitors the Hopfield network outputs and generates the correcting information. The correction voltage outputs are fed back into the neuron inputs through the extended conductance network. Note that there is no feedback connection to the input of the first amplifier (i.e. A_1) because the digital code decided by the first amplifier does not produce a local minimum for any analogue input signal. Similarly, there is no connection to the last amplifier (i.e. A_{m+n+1}).

The correction logic circuitry and extended conductance network can be identified by using the following equation:

$$\left(T_{pR} + T_{ps} + T_{pc} + \sum_{\substack{q=-n \\ q\neq p}}^{m} T_{pq}\right) u_p = T_{pR} V_R + T_{ps} V_s + T_{pc} f_p(V_o) + \sum_{\substack{q=-n \\ q\neq p}}^{m} T_{pq} v_q \tag{25}$$

where V_o is the digital output voltage given as $\{-v_m 2^m + \sum_{q=-n}^{m-1} v_q 2^q\}$, $f_p(V_o)$ is the pth correction logic output, T_{pc} is a conductance in the extended conductance network to the pth amplifier, and $V_R = -1\,\text{V}$.

Using the procedure shown in the preceding section to derive the characteristic parameter, the indicator GAP_p can be calculated:

$$\mathrm{GAP}_p = -2^p + 2^{-n} - \left(\frac{T_{pc}}{2^{p+1}}\right)(f_p(V_o^l) - f_p(V_o^u)), \quad \text{for } -n < p < m \tag{26}$$

where V_o^l and V_o^u are the digital output voltages of the adjacent digital coded defined in (23), and they can be given by

$$V_o^l = -V_m^l 2^m + \sum_{q=-n}^{m-1} V_q^l 2^q \tag{27a}$$

$$V_o^u = -V_m^u 2^m + \sum_{q=-n}^{m-1} V_q^u 2^q \tag{27b}$$

Note that only the characteristic parameters GAP_p, $-n < p < m$, are considered in designing the correction logic circuitry.

The local minima are eliminated when the indicators GAP_p, $-n < p < m$, are set to be zero. Therefore, the extended conductance becomes

$$T_{pc} = \frac{2^{2p+1} - 2^{p+1-n}}{f_p(V_o^u) - f_p(V_o^l)} \tag{28}$$

As T_{pc} is always non-negative, the correction logic circuitry output can be selected as

$$f_p(V_o^u) = -f_p(V_o^l) > 0 \tag{29}$$

Thus, T_{pc} becomes

$$T_{pc} = \frac{2^{2p} - 2^{p-n}}{f_p(V_o^u)} \tag{30}$$

To characterize the correction logic circuitry specified by (29), the circuitry output can take a discrete value of -1, 0, or 1 V to be compatible with the amplifier output voltage and the reference voltage. Table 1 lists the relationship between the amplifier outputs $D_i = v_{i-n-1}$, for $1 \le i \le N$, and the correction logic outputs $C_l = f_{l-n-1}(V_o)$ for $2 \le l \le N-1$, where V_o is the digital output voltage and $N = m + n + 1 = 16$. The self-correction circuitry characterized by Table 1 can be implemented by the simple combinational logic or SRAM memory devices.

As the input voltage $\{u_p\}$ is determined by the ratios of the conductances, the scaling factor to realize absolute conductance values can be used as an integrated-circuit design parameter. For example, the conductances are reduced by a scaling factor $|T_{ps}|$ $(= 2^{p+1})$ and are given by

$$T_{pq} = \begin{cases} 0, & \text{for } p = q \\ -2^q, & \text{for } p \ne q, p \ne m, q \ne m \\ 2^q, & \text{for } p \ne q, p = m, q \ne m \\ 2^q, & \text{for } p \ne q, p \ne m, q = m \end{cases} \tag{31}$$

$$T_{pc} = 2^{p-1} - 2^{-n-1} \tag{32}$$

Amplifier output																Correction logic output													
D_{16}	D_{15}	D_{14}	D_{13}	D_{12}	D_{11}	D_{10}	D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	C_{15}	C_{14}	C_{13}	C_{12}	C_{11}	C_{10}	C_9	C_8	C_7	C_6	C_5	C_4	C_3	C_2
×	×	×	×	×	×	×	×	×	×	×	×	×	×	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	+1
×	×	×	×	×	×	×	×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1
×	×	×	×	×	×	×	×	×	×	×	×	×	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	+1	0
×	×	×	×	×	×	×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0
×	×	×	×	×	×	×	×	×	×	×	×	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	+1	0	0
×	×	×	×	×	×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0
×	×	×	×	×	×	×	×	×	×	×	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	+1	0	0	0
×	×	×	×	×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0
×	×	×	×	×	×	×	×	×	×	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	+1	0	0	0	0
×	×	×	×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0
×	×	×	×	×	×	×	×	×	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	+1	0	0	0	0	0
×	×	×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0
×	×	×	×	×	×	×	×	0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	+1	0	0	0	0	0	0
×	×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0
×	×	×	×	×	×	×	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	+1	0	0	0	0	0	0	0
×	×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0
×	×	×	×	×	×	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0	+1	0	0	0	0	0	0	0	0
×	×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0
×	×	×	×	×	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	+1	0	0	0	0	0	0	0	0	0
×	×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0
×	×	×	×	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0	+1	0	0	0	0	0	0	0	0	0	0
×	×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0
×	×	×	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	+1	0	0	0	0	0	0	0	0	0	0	0
×	×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0
×	×	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	+1	0	0	0	0	0	0	0	0	0	0	0	0
×	×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0
×	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	+1	0	0	0	0	0	0	0	0	0	0	0	0	0
×	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 1. The input–output relation for the self-correction logic when $N = 16$. × means 'don't care'.

$$T_{pR} = 2^{p-1} \tag{33}$$

$$T_{ps} = \begin{cases} 1, & \text{for } -n \le p \le m-1 \\ -1, & \text{for } p = m \end{cases} \tag{34}$$

6. System architecture for neural-based transform coding

Observing (15*b*), it is indicated that the bias is not fixed but depends on input analogue signal x_{st} and the index (i,j) of its corresponding result y_{ij}. From (31)–(34), one may find that the synapse weights (T_{pq}), extended conductance (T_{pc}), T_{pR} and T_{ps} do not include the index (i,j) directly. However, their ranges are determined by two (i,j)-related parameters m_{ij} and n_{ij}. Therefore, both synapse weights and bias should be programmable to capture the information from the data set. Chang *et al.* (1991) proposed a reconfigurable hybrid MOS neural circuit including electrically programming synapses and bias to implement the transform coding. As the realization of programmable synapses seems quite complicated, our paper presents a normalization technique to force their ranges to be within a fixed interval. After performing the normlaization, the bias would be the only remaining (i,j)-related term. In this paper a cost-effective design according to the normalization technique is developed to compute the transform coding.

The normalization procedure can be performed by letting the conductances, i.e. T_{pq}, T_{pR} and T_{ps} be reduced by a new scaling facter, $2^{p+m_{ij}+1}$. Thus, the normalized conductances become

$$T_{pq} = \begin{cases} 0, & \text{for } p = q \\ -2^{q-m_{ij}}, & \text{for } p \ne q, p \ne m_{ij}, q \ne m_{ij} \\ 2^{q-m_{ij}}, & \text{for } p \ne q, p = m_{ij}, q \ne m_{ij} \\ 2^{q-m_{ij}}, & \text{for } p \ne q, p \ne m_{ij}, q \ne m_{ij} \end{cases} \tag{35}$$

$$T_{pc} = 2^{p-m_{ij}-1} - 2^{-m_{ij}-n_{ij}-1} \tag{36}$$

$$T_{pR} = 2^{p-m_{ij}-1} \tag{37}$$

$$T_{ps} = \begin{cases} 2^{-m_{ij}}, & \text{for } -n_{ij} \le p \le m_{ij}-1 \\ -2^{-m_{ij}}, & \text{for } p = m_{ij} \end{cases} \tag{38}$$

It can be shown that the ranges of normalized terms $2^{q-m_{ij}}$ and $2^{p-m_{ij}-1}$ are within the intervals $\left[2^{-N+1}, 1\right]$ and $\left[2^{-N}, 2^{-1}\right]$ and $2^{-m_{ij}-n_{ij}-1}$ equals 2^{-N}, respectively, where $-n_{ij} \le p, q \le m_{ij}$, and $N\,(= m_{ij} + n_{ij} + 1)$ is the number of neurons. Therefore, T_{pq}, T_{pc} and T_{pR} are totally independent of the index (i,j) which corresponds to the resultant $\hat{y}_{ij}$ and depends on the normalized index (p,q), which corresponds to the size of neural network (or number of bits involved in $\hat{y}_{ij}$). Unfortunately, the conductance between the pth neuron and analogue input voltage, T_{ps}, becomes an (i,j)-related term which is in the term $2^{-m_{ij}}$ after performing normalization. To eliminate the dependence of index (i,j), several arrangements should be considered in the expression of bias I_p^{ij}:

$$I_p^{ij} = T_{ps} v_{s(ij)} + T_{pR} V_R = \hat{T}_{ps} \hat{v}_{s(ij)} + T_{pR} V_R \tag{39}$$

where

$$\hat{T}_{ps} = \begin{cases} 1, & \text{for } -n_{ij} \le p \le m_{ij} - 1 \\ -1, & \text{for } p = m_{ij} \end{cases} \tag{40}$$

$$\hat{v}_{s(ij)} = 2^{-m_{ij}} v_{s(ij)} = \sum_{s=0}^{L-1} \sum_{t=0}^{L-1} 2^{-m_{ij}} u_{is} u_{tj} x_{st} = \sum_{s=0}^{L-1} \sum_{t=0}^{L-1} u_{istj} x_{st} \tag{41}$$

$$u_{istj} = 2^{-m_{ij}} u_{is} u_{tj}, \quad 0 \le i, j, s, t \le L - 1 \tag{42}$$

Basically, the concept of the above arrangements is considered to combine the (i,j)-related terms $2^{-m_{ij}}$ and $u_{is}u_{tj}$ together. This new term is denoted by u_{istj}. As a result, those conductances located in the self-correction neural network are independent of the index (i,j). Based on the above discussion, a proposed system architecture and the function structures of normalized self-correction neural network and analogue MOS vector multiplier are illustrated in Figs 3, 4(*a*) and 4(*b*) respectively.

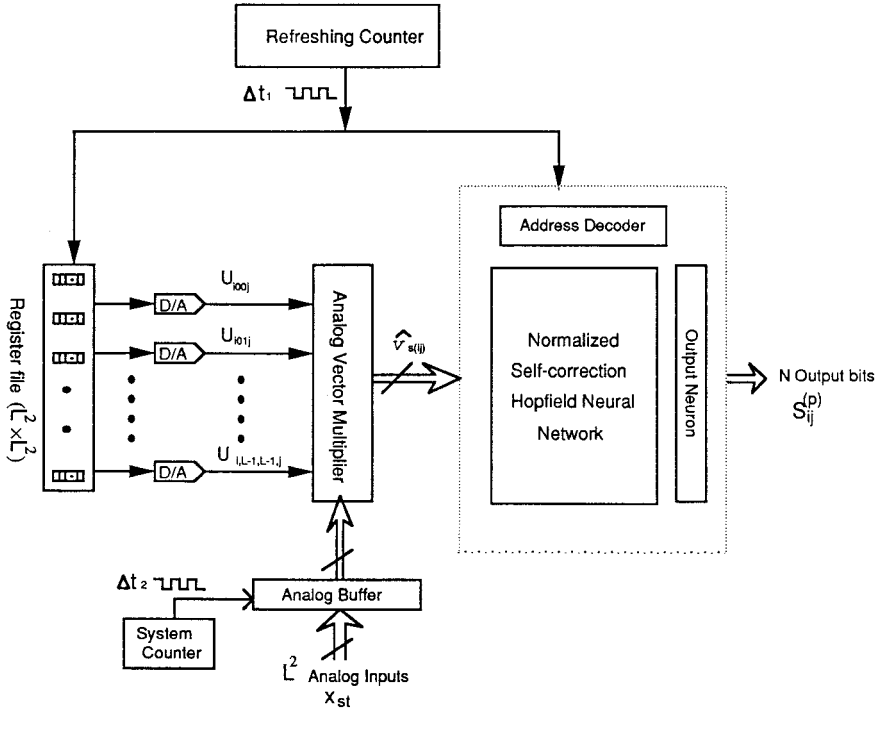
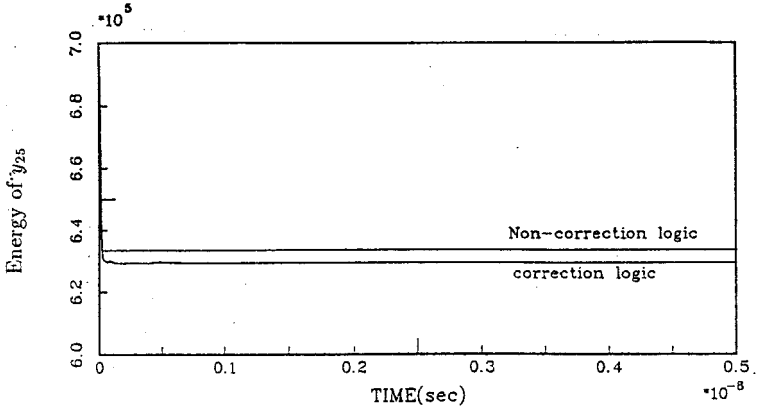
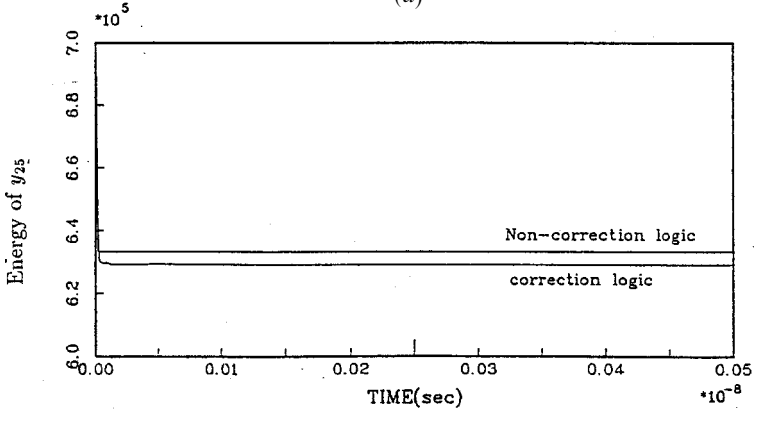


Figure 3. Architecture of image transform coding neural chip.



(*a*)



(*b*)

Figure 4. (*a*) The time evolution of the reduction of energy for y_{25} when $RC = 10^{-9}$ and $\lambda = 10^5$; (*b*) the time evolution of the reduction of energy for y_{25} when $RC = 10^{-10}$ and $\lambda = 10^5$.

The (i,j)-related parameters u_{istj} could be precomputed and are stored in the $(L^2 \times L^2)$ register file which is controlled by a refreshing counted with clock Δt_1. Note that Δt_1 is defined as the sum of $\Delta t_{\text{refresh}}$ (= time for refreshing the bias) and Δt_{neural} (= computation time for the neural network). Although computing a particular $\hat{y}_{ij}$, those parameters should be converted to analogue form and then pumped out from the register file to the analogue MOS vector multiplier. From (40), the analogue input to the network $\hat{v}_{s(i,j)}$ can be obtained by performing the analogue inner vector multiplication between two L^2-tuple vectors

$$\boldsymbol{x} = \begin{bmatrix} x_{00} & x_{10} & \dots & x_{L-1,L-1} \end{bmatrix}^{\mathrm{T}}$$

$$\boldsymbol{u}_{ij} = \begin{bmatrix} u_{i,0,0,j} & u_{i,1,0,j} & \dots & u_{i,L-1,L-1,j} \end{bmatrix}^{\mathrm{T}}$$

The time for updating the bias would be dominated by the computation time for the $L^2 \times L^2$ analogue inner vector multiplication Δt_{vm}. In other words, $\Delta t_{\text{refresh}} \simeq \Delta t_{vm}$.

As the L^2 input analogue signals x_{st} are required in computing the vector multiplication (with $u_{istj}, 0 \le s, t \le L-1$) involved in each $I_{ij}^p, 0 \le i, j \le L-1, x_{st}$ should stay in the analogue buffer until the $L^2\hat{y}_{ij}$ have been completed. This analogue buffer is controlled by a system counter with clock $\Delta t_2 = (L^2(\Delta t_1 + \Delta t_{\text{overhead}}))$, where $\Delta t_{\text{overhead}}$ is the time for input–output overhead. Usually, the analogue buffer if realized by analogue DRAM-style storage based on the floating-gate storage approach. The overall output of the analogue vector multiplier v_{out} is given by

$$v_{\text{out}} = \sum_{s=0}^{L-1} \sum_{t=0}^{L-1} c u_{istj} x_{st} \tag{43}$$

where c is the constant that depends on the characteristics of MOS implementation.

It is interesting to note that the constant c could be compensated for by absorbing the values into u_{istj}. For example, one may precompute the new u_{istj} as $c^{-1}u_{istj}$, where u_{istj} is the old one. The input–output compatibility of the overall MOS implementation is of particular interest because the relatively high output impedance node of the double inveter is connected to the almost infinite input impedance node of the MOSFET gates with almost no restriction on the fan-in/fan-out capability. More details about the analogue MOS vector multiplier are given by Mead (1989), Salam *et al.* (1989) and Khachab and Ismail (1987).

7. Illustrated examples

To examine the performance of the neural-based analogue circuit for computing the 2-D DCT transform coding, an often used 8×8 DCT will be considered in our simulation because it represents a good compromise between coding efficiency and hardware complexity. Because of its effectiveness, the CCITT H.261 recommended standard for $p \times$ 64 kbit/s $(p = 1, 2, \ldots, 30)$ visual telephony developed by CCITT, and the still-image compression standard developed by ISO JPEG all include the use of 8×8 DCT in their algorithms.

To obtain the size (N) of neural network required to compute its corresponding DCT coefficient, it is necessary to calculate their respective dynamic range and take into account the sign bit. To achieve this purpose, the range of each DCT coefficient can be determined by generating random integer pixel data values in the range 0 to 255 through the 2-D discrete cosine transform. For example, the range of y_{00} is from -1024 to 1023. Therefore, m_{00} is identified as 11; that is, 10 bits are for the magnitude of y_{00} and 1 bit is for the sign. As a result, the corresponding m_{ij} for each DCT coefficient y_{ij} is illustrated in Table 2. Another important parameter required in determining the size is n_{ij}, which depends on the required accuracy and the tolerable mismatch in the final representation of the reconstructed video samples. The analysis of the accuracy and mismatch involved in the finite length arithmetic DCT computation was discussed by Sun *et al.* (1987) and Liou and Bellisio (1987). Based on their results and the consideration of feasible hardware implementation, the number of bits (or size of neural network) involved in each DCT coefficient is set to be 16. Then n_{ij} would be equal to $(15 - m_{ij})$, for example $n_{00} = 5$. The above suggestion seems quite reasonable to improve the accuracy of a particular y_{ij} which has a small dynamic range.

We have simulated both the DCT-based neural analogue circuit without the correction logic of (18), and a circuit based on the proposed correction logic using

$i \backslash j$	0	1	2	3	4	5	6	7
0	11	9	9	9	9	9	9	9
1	9	9	9	9	9	9	9	9
2	9	9	9	9	9	9	9	9
3	9	9	9	9	9	9	9	9
4	9	9	9	9	9	9	9	9
5	9	9	9	9	9	9	9	9
6	9	9	9	9	9	9	9	9
7	9	9	9	9	9	9	9	9

Table 2. m_{ij} for y_{ij}.

the simultaneous differential equation solver (DVERK in the IMSL). This routine solves a set of nonlinear differential equations using the fifth-order Runge–Kutta method. It can be found that the convergence times for both neural networks is within the *RC* time constant. We used two different *RC* time constants, $RC = 10^{-10}$ $(R = 1\,\text{k}\Omega, C = 0{\cdot}01\,\text{pF})$ and $RC = 10^{-9}$ $(R = 1\,\text{k}\Omega, C = 0{\cdot}1\,\text{pF})$, and all amplifier gains λ_p are assigned to be 10^4 in our experiments, and ran simulations on a SUN workstation. The test input pixel data x_{st} are illustrated in Table 3 (*a*). Figs 4 (*a*) and (*b*) show an example of the time evolution of the reduction of energy performed by both networks with $N\ (= 16)$ neurons that represent y_{25} based on the 2s complement binary number representation for two different *RC* time constants. Moreover, both figures show that the curves for correction logic case reach the steady-state values which are less than the values for the non-correction logic case. In other words, the non-correction reaches a local minimum and yields the incorrect solution. The (2,5)-entry in Table 3 (*c*) shows the resulting DCT coefficient y_{25} obtained at the steady-state points on the correction logic curves of Figs 4 (*a*) and (*b*). It is shown that the result is almost independent of the *RC* time constants. However, each converegence time will be in proportion to its corresponding *RC* time constant. For example, the converge times for $RC = 10^{-10}$ and $RC = 10^{-9}$ are in proportion to the orders of timescale 10^{-10} s $(= 0{\cdot}1\,\text{ns})$ and 10^{-9} s $(= 1\,\text{ns})$, respectively. However, these two curves have almost the same time evolution. Starting from a very high energy state, the neural network reduces its energy spontaneously by changing its state so that the 2s complement binary variables $s_{ij}^{(p)}$ minimize the error energy function.

Considering the system architeture shown in Fig. 3, another important module is the analogue vector multiplier. For simplicity in estimating the computation time in computer simulation, we assume that the op-amp involved in the overall MOS vector implementation is ideal. The computation time for a 64×64 analogue MOS vector multiplier Δt_{vm} is estimated as 1 ns by performing the inner vector product on two 64-tuple vectors on SPICE-II. Thus, the refreshing clocl Δt_1 $(\simeq \Delta t_{\text{neural}} + \Delta t_{\text{vm}})$ is identified as 2 ns. Therefore, the computation time for computing all DCT coefficients would be estimated as 128 ns when $L = 8$ and $RC = 10^{-9}$. Computer simulation verifies the effectiveness of the proposed neural-based transform coder. Undoubtedly, this analogue coder can be practically implemented with ASICs using today's low-cost VLSI technology. The estimated computation time provides an index to evaluate and justify the computational efficiency and performance of the real VLSI implementation of our design.

	i \ j	0	1	2	3	4	5	6	7
(a)	0	159	87	86	105	172	105	103	55
	1	126	17	194	230	28	238	179	128
	2	253	105	48	181	103	94	255	26
	3	73	159	29	241	138	175	66	145
	4	120	74	220	201	97	5	87	70
	5	2	187	10	118	203	252	162	209
	6	160	56	97	19	53	221	14	78
	7	131	31	233	85	140	168	118	126
(b)	0	981·25	- 31·98	- 73·45	71·69	12·50	89·23	60·46	- 60·00
	1	34·81	39·82	- 10·03	15·74	41·30	135·48	- 97·65	- 26·24
	2	- 24·38	- 7·57	27·82	64·95	- 49·14	12·93	145·73	93·48
	3	- 40·17	- 40·78	- 24·64	- 15·38	- 3·28	- 136·40	- 18·50	- 5·26
	4	- 30·25	83·83	- 64·59	- 101·10	65·50	- 31·25	- 4·37	74·85
	5	- 72·70	26·73	42·29	134·43	38·60	- 117·16	- 43·07	- 64·55
	6	60·70	- 47·46	23·98	71·82	13·32	- 48·05	- 165·57	55·75
	7	- 86·75	172·09	56·20	7·28	19·01	128·77	4·92	137·72
(c)	0	981·25	- 31·97	- 73·45	71·69	12·50	89·23	60·46	- 60·00
	1	34·81	39·81	- 10·03	15·73	41·30	135·48	- 97·65	- 26·23
	2	- 24·38	- 7·58	27·83	64·95	- 49·14	12·92	145·73	93·47
	3	- 40·17	- 40·78	- 24·64	- 15·38	- 3·28	- 136·39	- 18·50	- 5·27
	4	- 30·25	83·83	- 64·59	- 101·10	65·50	- 31·25	- 4·36	74·86
	5	- 72·70	26·72	42·30	134·44	38·60	- 117·16	- 43·08	- 64·55
	6	60·70	- 47·47	23·98	71·83	13·32	- 48·05	- 165·58	55·75
	7	- 86·75	172·09	56·20	7·28	19·00	128·77	4·92	137·72

Table 3. (*a*) Input pixel data x_{st}; (*b*) corresponding DCT coefficients; (*c*) Neural-based DCT coefficients.

8. Conclusion

The computation of a 2-D DCT-based transform coding has been shown to solve a quadratic nonliner programming problem subject to the corresponding 2s complement binary variables of 2-D DCT coefficients. We have shown that the existance of local minima in neural-based transform coding will lead to an incorrect digital representation of the DCT coefficient. A self-correction logic based on Lee and Sheu's concept is developed to eliminate these local minima. Using this concept, a novel self-correction Hopfield-type neural analogue circuit designed to perform the DCT-based quadratic nonlinear programming could obtain the desired coefficients of an 8×8 DCT in 2s complement code within 128 ns with $RC = 10^{-9}$.

References

Chang, P. R., Hwang, K. S., and Gong, H. M., 1991, A high-speed neural analog circuit for computing the bit-level transform image coding. *IEEE Transactions on Consumer Electronics*, **37**, 337–342.

Chua, L. O., and Lin, T., 1988, A neural network approach to transform image coding. *International Journal of Circuit Theory and Applications*, **16**, 317–324.

Culhane, A. D., Peckerar, M. C., and Marrian, C. R. K., 1989, A neural net approach to discrete Hartley and Fourier transform. *IEEE Transactions on Circuits and Systems*, **36**, 695–702.

HOLLER, M., TAM, S., CASTRO, H., and BENSON, R., 1989, An electrically trainable artificial neural network (ETANN) with 10240 'Float gate' synapses. *International Joint Conference on Neural Networks*, Vol. 2, Washington DC, U.S.A., pp. 191–196.

HOPFIELD, J. J., 1984, Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Science, U.S.A.*, **81**, 3088–3092.

HOPFIELD, J., and TANK, D., 1985, Neural computations of decisions in optimization problems. *Biological Cybernetics*, **52**, 141–152.

KHACHAB, N., and ISMAIL, M., 1987, Novel continuous-time all-MOS four-quadrant multipliers. *Proceedings of the IEEE International Symposium on Circuits and Systems* (*ISCAS*), San Francisco, California, U.S.A., May, pp. 762–765.

KIRKPATRICK, S., GELATT, C. D., Jr., and VECCHI, M. P., 1983, Optimization by simulated annealing. *Science*, **220**, 650–671.

LEE, B. W., and SHEU, B. W., 1988, An investigation on local minima of Hopfield Network for optimization circuit. *IEEE International Conference on Neural Networks*, San Diego, California, U.S.A., Vol. 1, pp. 45–51; 1989, Design of a neural-based A/D converter using modified Hopfield network. *IEEE Journal of Solid-state Circuits*, **24**, 1129–1135; 1990, Parallel digital image restoration using adaptive VLSI neural chips. *Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers and Processors*, Cambridge, Massachusetts, U.S.A., 17–19 September.

LIOU, M. L., and BELLISIO, J. A., 1987, VLSI implementation of discrete cosine transform for visual communication. *Proceedings of the International Conference on Communication Technology*, Bejing, People's Republic of China (IEEE).

MEAD, C., 1989, *Analog VLSI and Neural Systems* (New York, U.S.A.: Addison-Wesley).

SALAM, F., KHACHAB, N., ISMAIL, M., and WANG, Y., 1989, An analog MOS implementation of the synaptic weights for feedback neural nets. *Proceedings of the IEEE International Symposium on Circuits and Systems* (*ISCAS*), San Diego, California, U.S.A. pp. 1223–1225.

SUN, M. T., WU, L., and LIOU, M. L., 1987, A concurrent architecture for VLSI implementation of discrete cosine transform. *IEEE Transactions on Circuits and Systems*, **34**, 992–994.

TANK, D. W., and HOPFIELD, J. J., 1986, Simple 'neural' optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit. *IEEE Transactions on Circuits and Systems*, **33**, 533–541.