Sequential and Parallel Image Restoration: Neural Network Implementations

Mário A. T. Figueiredo and José M. N. Leitão

Abstract—**Sequential and parallel image restoration algorithms and their implementations on neural networks are proposed in this paper. For images degraded by linear blur and contaminated by additive white Gaussian noise, *maximum a posteriori* (MAP) estimation and regularization theory lead to the same high dimension convex optimization problem. The commonly adopted strategy (in using neural networks for image restoration) is to map the objective function of the optimization problem into the energy of a predefined network, taking advantage of its energy minimization properties. Departing from this approach, we propose neural implementations of iterative minimization algorithms which are first proved to converge. The developed schemes are based on modified Hopfield networks of graded elements, with both sequential and parallel updating schedules. An algorithm supported on a fully standard Hopfield network (binary elements and zero autoconnections) is also considered. Robustness with respect to finite numerical precision is studied, and examples with real images are presented.**

I. Introduction

A. Image Restoration

IMAGE restoration is an inverse problem which aims at recovering an image from an observed degraded version of it. The degradation mechanism is closely related to the physical process involved. Typical examples are: finite resolution of sensor arrays, motion blur caused by sensor or object movements during acquisition, refraction, multipath, and out-of-focus blur. The acquired images are also contaminated by noise, the nature of which depends on the particular problem at hand.

Since the obtained images are not unique and/or do not depend continuously on the observed data, image restoration is in general an *ill-conditioned* or even *ill-posed* problem [1]–[4]. Bayesian estimation and the regularization approach are two of the classical formulations where *a priori* information/constraints are incorporated in order to deal with such difficulty. Independently of their conceptual background, both lead to very large dimension optimization problems [1]–[3], [5]–[7].

In this paper we will consider the observed image $\mathbf{y}$ as a linearly transformed (for example, blurred) and noisy version of the original image $\mathbf{x}$

$$\mathbf{y} = \mathbf{B}\mathbf{x} + \mathbf{n} \tag{1}$$

where $\mathbf{B}$ is a matrix modeling the linear observation, and $\mathbf{n}$ is a white Gaussian noise (WGN) vector with covariance matrix $\sigma^2\mathbf{I}$. In (1), images are represented by vectors obtained by the usual process of lexicographically ordering the image pixels. The exact dimensions of vectors $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{n}$, and of matrix $\mathbf{B}$, depend on the particular problem under study; if (1) resulted from the discretization of a continuous formulation of the observation model, then $\mathbf{y}$ is usually of slightly smaller dimension than $\mathbf{x}$ [1]. Vectors $\mathbf{x}$ and $\mathbf{y}$ can also be considered as having very different dimensions, as is typical in tomographic imaging, radio astronomy, electron microscopy, and other applications [8]–[10]. The ill-conditioned or ill-posed nature of the problem reveals itself in this vector formulation by the fact that matrix $\mathbf{B}$ may have very small singular values, its inversion (or pseudoinversion in the case of nonsquare $\mathbf{B}$) being then highly noise-sensitive.

Consider the original image $\mathbf{x}$ ($M \times N$ pixels) as a sample of a zero mean Gauss-Markov random field (ZMGMRF), with covariance matrix $\mathbf{A}$. The *maximum a posteriori* (MAP) estimation criterion corresponding to the above assumptions yields the following minimization problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E(\mathbf{x}) \quad \text{with} \quad E(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{C}\mathbf{x} - \mathbf{b}^T\mathbf{x}. \tag{2}$$

In (2), $\mathbf{C}$ is a $MN \times MN$ symmetric positive definite (PD) matrix, with strictly positive diagonal elements, [6], [11], and $\mathbf{b}$ is a MN-dimensional vector, defined, respectively, by

$$\mathbf{C} = \mathbf{A}^{-1} + \frac{1}{\sigma^2}\mathbf{B}^T\mathbf{B}, \quad \text{and} \quad \mathbf{b} = \frac{1}{\sigma^2}\mathbf{B}^T\mathbf{y}. \tag{3}$$

The minimization problem expressed in (2) is equivalent to the linear system of equations $\mathbf{C}\mathbf{x} = \mathbf{b}$, this emphasizing the fact that its solution involves the inversion of matrix $\mathbf{C}$. In fact, since $\mathbf{C}$ is a PD matrix, $E(\mathbf{x})$ is convex, and its minimizer can be found simply by solving $\nabla E(\mathbf{x}) = 0$, which is equivalent to $\mathbf{C}\mathbf{x} = \mathbf{b}$. Notice also that $\hat{\mathbf{x}} = \mathbf{C}^{-1}\mathbf{b}$, with $\mathbf{C}$ and $\mathbf{b}$ as given in (3), is the optimal Wiener estimate of the original image [1], [7], [12]. An equivalent optimization problem can be derived by using a regularization approach [3], [4], [6], [7].

B. Image Restoration Using Neural Networks

The Hopfield network is a neural structure which has been used to deal with optimization problems [13], [14]; it is briefly

Manuscript received January 28, 1993; revised August 5, 1993. The associate editor coordinating the review of this paper and approving it for publication was Dr. M. Ibrahim Sezan.

The authors are with the Centro de Análise e Processamento de Sinais—Complexo I, Departamento de Engenharia Electrotécnica e de Computadores, Instituto Superior Técnico, Lisboa, Portugal.

IEEE Log Number 9404456.

described in Appendix A, with emphasis put on its optimization properties. Mapping of the function to be minimized into the network's energy function (see Appendix A) to exploit its energy descent ability is the common strategy. In the field of image (and signal) restoration, recent publications have suggested Hopfield and Hopfield-like networks applied to the minimization problems involved [11], [15]–[18]. Neural structures have also been proposed in other areas of image processing and computer vision such as stereo matching [19], [20], edge detection [21], image segmentation [22], [23], and surface reconstruction [24]. Main research issues have been the representation scheme [25], i.e., how to represent the image gray levels on a binary network, and the condition of nonnegative autoconnections (see Appendix A) necessary to invoke Hopfield's convergence results. None of the mentioned work, however, reached a fully standard Hopfield network with binary elements, threshold-based updating rule, and zero (or at least nonnegative) autoconnections. In this paper, a new approach is followed: Instead of mapping the function to be minimized into the energy of a predefined network, neural implementations of iterative restoration schemes are studied.

Zhou, *et al.* [18] introduced image restoration based on a Hopfield network in which the image pixels are coded by the sum of a set of binary (0 or 1) elements. This type of representation scheme, proposed by Takeda, *et al.* [25] and referred to as *bit-density coding* (BDC), has the important advantage of being fault-tolerant, since many different configurations can represent the same solution [25]. However, if used as in [18], it leads to negative autoconnection weights, this meaning that energy descent is not guaranteed; an additional energy reduction check-step, which is an *ad hoc* and time consuming solution, has to be used.

Paik, *et al.* [16] suggested a modified Hopfield network of discrete valued neurons with nonzero autoconnections, with two updating schemes (one sequential and the other parallel) which were proved to converge. More recently, the same authors extended their work [11] by proposing another modified Hopfield network with a different updating rule, based on which several restoration algorithms are implemented and studied in terms of convergence properties.

In another direction, Abbiss, *et al.* [15] and Yeh, *et al.* [17] introduced Hopfield-like networks of binary elements with a modified updating rule using two threshold levels, instead of one. This modification guarantees energy descent, even with negative autoconnections (as is their case), at the cost of slightly more complex elements.

Still in [15], Abbiss, *et al.* suggest a network of graded elements. Their structure implements a restoration scheme with previously known convergence properties (the Gerchberg-Papoulis algorithm [26]). The issues of numerical precision and convergence acceleration are raised although not quantified.

C. Proposed Approach

As said before, the present work explores the following approach: Instead of mapping the function to be minimized into the energy of a predefined network, neural implementations of iterative restoration schemes are developed. The adopted algorithms are suitable for distributed implementation, the key feature being that the updating of each element depends only on local information [6]. This constriction rules out any algorithm in which a global parameter has to be determined, e.g., the step size of gradient descent methods which are best suited for other types of structures (see, for example, [27]).

It should be pointed out that the referred restoration criterion and image model are simplistic; all the necessary parameters (matrices $\mathbf{A}$ and $\mathbf{B}$, and the noise variance σ^2) are considered known, which is seldom true in real cases. However, iterative algorithms with simultaneous parameter estimation (e.g., the *expectation-maximization* (EM) algorithm [28]–[30]) usually involve a step in which an image estimate (given the current parameter estimates) is required; the neural algorithms proposed in this paper can thus be included as an intermediate step of a more sophisticated restoration scheme. Reconstruction techniques which aim at simultaneously detecting and preserving the discontinuities of the original image also include an intermediate step in which a quadratic problem similar to (2) has to be solved [31], [32]. Again, the schemes proposed in this paper can be imbedded into a more complex algorithm, as we have proposed in [31].

An outline of the paper is next presented:

- In Section II, we describe a general class of iterative methods along with convergence criteria. Neural implementations of two instances of this class are then introduced. The first is the well-known Gauss-Seidel algorithm; we show that, for the problem under study, convergence is guaranteed. The second is a modification of the Jacobi algorithm motivated by the observation that the original version cannot be guaranteed to converge. This last scheme leads to a network in which all elements update their state simultaneously. Both structures are Hopfield-type networks of graded elements.
- Section III considers a standard Hopfield neural network of binary elements with zero autoconnections. This network implements a unit-step version of the Gauss-Seidel algorithm. As in [11] and [18], *bit-density coding* (BDC) is used (i.e., each image pixel is represented by the sum, or average, of a set of binary elements) but in a fundamentally different manner: each subset representing a given pixel is interconnected in such a way that it automatically evolves towards the BDC representation. The resulting structure is a fully-standard Hopfield network with convergence guaranteed by the fact that it has zero autoconnections.
- A study of the errors associated with finite numerical precision is performed in Section IV. We show that these errors (with respect to what would be obtained by using infinite precision) depend not only on the numerical resolution used, but also, as expected, on the characteristics of matrix $\mathbf{C}$.
- Finally, Section V presents simulation results while Section VI contains some concluding remarks.

II. Image Restoration Algorithms: Implementation on Modified Hopfield Neural Networks

A. A General Class of Iterative Schemes

Under the assumptions of the previous section, image restoration involves the minimization of a huge dimension quadratic form

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E(\mathbf{x}) \quad \text{with } E(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{C}\mathbf{x} - \mathbf{b}^T\mathbf{x}. \quad (4)$$

which can be achieved by solving the linear system of equations $\mathbf{Cx} = \mathbf{b}$. The dimensionality of this system ($MN \times MN$, for a $M \times N$ pixels image) strongly demands iterative schemes, since any direct inversion is out of the question.

A known family of iterative schemes is obtained by splitting matrix $\mathbf{C}$ into $\mathbf{C} = \mathbf{G} - \mathbf{H}$ [33], [34]. This leads to the equivalent system $\mathbf{Gx} = \mathbf{Hx} + \mathbf{b}$ which (assuming that $\mathbf{G}^{-1}$ exists) suggests the iteration

$$\mathbf{Gx}(n+1) = \mathbf{Hx}(n) + \mathbf{b} \quad (5)$$

or equivalently

$$\begin{aligned}\mathbf{x}(n+1) &= \mathbf{G}^{-1}(\mathbf{Hx}(n) + \mathbf{b}) \\ &= \mathbf{x}(n) - \mathbf{G}^{-1}(\mathbf{Cx}(n) - \mathbf{b}) \end{aligned} \quad (6)$$

starting with some initial condition $\mathbf{x}(0)$ [33], [34]. Matrix $\mathbf{G}$ should be chosen so that it is easily invertible (e.g., diagonal or triangular) and to guarantee convergence towards the solution. Defining the error vector $\mathbf{e}(n) = \mathbf{x}(n) - \mathbf{C}^{-1}\mathbf{b}$, it follows that $\mathbf{e}(n) = (\mathbf{G}^{-1}\mathbf{H})^n\,\mathbf{e}(0)$; therefore, iteration (6) converges if and only if matrix $\mathbf{M} \equiv \mathbf{G}^{-1}\mathbf{H}$ is *convergent* [33], [34], i.e.

$$\lim_{n\to\infty} (\mathbf{G}^{-1}\mathbf{H})^n = \mathbf{0}. \quad (7)$$

A given matrix $\mathbf{M}$ is convergent if and only if $\rho(\mathbf{M}) < 1$, where $\rho(\mathbf{M})$ stands for the spectral radius of $\mathbf{M}$, i.e., its largest absolute eigenvalue [33], [34]. The rate of convergence of such algorithms is defined as

$$R = -\log \rho(\mathbf{G}^{-1}\mathbf{H}) \quad (8)$$

and verifies

$$\log \| \mathbf{e}(n) \|_2 \leq \log \| \mathbf{e}(0) \|_2 - nR \quad (9)$$

meaning that the logarithm of the error has an upper bound which decreases linearly with time slope R [34]. In (9), the notation $\| \cdot \|_2$ stands for the Euclidean vector norm.

The two networks that will be introduced in this section to implement instances of the iterative scheme (6) have a common structure, which is depicted in Fig. 1. Each element is characterized by its state $x_i(t)$ and computes its *total input* (TI) $u_i(t)$ which is a weighted (by the interconnection matrix) sum of the states of all other elements, plus a bias input I_i. When its turn arrives (this depending on the particular visiting schedule being used) the element updates its state as a function of its TI, its bias input, and (possibly) of its previous state

$$x_i(t+1) = f(u_i(t), I_i, x_i(t)).$$

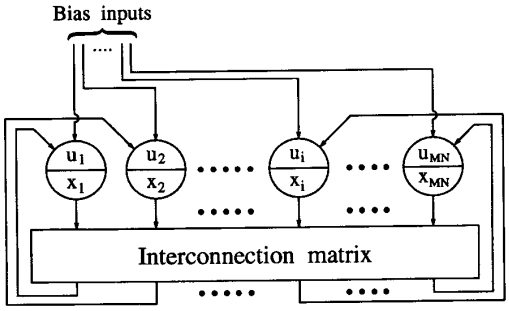


Fig. 1. General structure of the modified Hopfield network of graded elements.

B. Sequential Algorithm

Taking matrix $\mathbf{G}$ as the lower triangular part of $\mathbf{C}$ yields the Gauss-Seidel algorithm [33], [34]. Iteration (6), for this choice of $\mathbf{G}$, can be written explicitly as

$$x_i(n+1) = \frac{1}{C_{ii}}\left(b_i - \sum_{j=1}^{j=i-1} C_{ij}x_j(n+1) - \sum_{j=i+1}^{j=MN} C_{ij}x_j(n)\right) \quad (10)$$

for $i = 1, 2, \ldots, MN$ and $n = 1, 2, \ldots$, since when $x_i(n+1)$ is being computed, all new components of $\mathbf{x}(n+1)$, up to $x_{i-1}(n+1)$, are already known [34].

In the sequel, we will use a different time variable t which is incremented every time a component is updated. The Gauss-Seidel algorithm is implemented by the following neural structure and updating scheme [35]:

Network 1: Define a network of MN real valued elements $\{x_i \in R,\ i = 1, 2, \ldots, MN\}$, each assigned to one image pixel. Let $W_{ij} = -C_{ij}/C_{ii}$ be the interconnection strength between elements i and j, and $I_i = b_i/C_{ii}$ be the bias input to element i. Let a cyclic sequential visiting schedule to the elements be adopted, i.e., $i = 1, 2, 3, \ldots, MN, 1, 2, \ldots, MN, 1, \ldots$, and the network be initialized with any finite state. At time t, an element i, chosen according to the visiting schedule, updates its state according to

$$x_i(t+1) = x_i(t) + u_i(t) \quad \text{with} \quad u_i(t) = \sum_{j=1}^{MN} W_{ij}x_j(t) + I_i. \quad (11)$$

We stress the fact that the iteration variable t used in (11) is not the same as the variable n used in (10); one n step in (10) is equivalent to a full sweep over all the elements of the network (MN steps with t). To prove convergence, we invoke the known fact that the Gauss-Seidel algorithm converges if the system matrix is positive definite (PD) and has positive diagonal elements [33], [34]. As matrix $\mathbf{C}$ verifies this conditions, convergence is guaranteed.

To gain some further insight into the iterative process defined by (11), and to try to loosen the strict schedule imposed, we now look at it from the energy (function $E(\mathbf{x})$ is

usually called energy) descent point of view, using an approach that can be found in [33]. Each iteration of the form (11) specifies a new value $x_i(t+1)$ which is such that the ith equation of system $\mathbf{Cx} = \mathbf{b}$ is satisfied. But since system $\mathbf{Cx} = \mathbf{b}$ is equivalent to $\nabla E(\mathbf{x}) = 0$, its ith equation is equivalent to

$$\frac{\partial E(\mathbf{x})}{\partial x_i} \equiv \nabla_i E(\mathbf{x}) = 0.$$

This means that $x_i(t+1)$ is the value that yields the lower possible $E(\mathbf{x})$, under the constraint that only the ith coordinate of vector $\mathbf{x}$ can be changed. Two cases have to be considered. If, at the current position $\mathbf{x}(t)$, for at least one coordinate x_i, $\nabla_i E(\mathbf{x}) \neq 0$, one updated value $x_i(t+1)$ will be different from $x_i(t)$, and so $E(\mathbf{x}(t+1)) < E(\mathbf{x}(t))$. Suppose now that at the current position $\mathbf{x}(t)$, $\nabla_i E(\mathbf{x}(t)) = 0$ for all i; this is equivalent to stating that all equations of the system $\mathbf{Cx} = \mathbf{b}$ are satisfied, and so $\mathbf{x}(t)$ is its (unique) solution. In conclusion, from an energy reduction point of view, Network 1 proceeds by minimizing $E(\mathbf{x})$ successively with respect to each visited coordinate. These considerations also make clear that the visiting schedule has no influence on the convergence of the algorithm, as long as no coordinate is left out.

Observe that $W_{ii} = -1$ (for all i). Thus, the updated value $x_i(t+1)$ does not depend on $x_i(t)$, i.e., the adopted scheme naturally leads to a network which behaves like having zero autoconnection weights. This is made clear if (11) is rewritten as

$$x_i(t+1) = u_i(t) \text{ with } u_i(t) = \sum_{j=1}^{MN} W'_{ij} x_j(t) + I_i \quad (12)$$

in which the new weights W'_{ij} are defined as

$$W'_{ij} = \begin{cases} W_{ij} & \Leftarrow \quad i \neq j \\ 0 & \Leftarrow \quad i = j. \end{cases} \quad (13)$$

Next, we describe relations between the neural algorithm just presented and some previous work:

- Network 1 has some similarity with the one proposed by Paik, *et al.* [11], [16]. There are, however, some fundamental differences that should be noted. Unlike in the network of Paik, *et al.*, which uses unit-steps, the updating rule of Network 1 as given by (11) or (12) is optimal in the sense that the new value is the one that yields the maximum possible energy reduction, given that only one element is allowed to change. Thus, when compared to Network 1, the network of [16] progresses with much slower unit steps. Another difference is that instead of continuous-valued neurons, [16] has adopted discrete valued neurons. Our perspective is that this aspect (the discrete nature of the elements) should be included in the numerical analysis of the algorithm. In other words, a discrete valued version of an algorithm is nothing more than its implementation using finite numerical precision, the effect of which will be studied in Section IV. Furthermore, [16] considers bounded values. This restriction, however, does not need to be considered when convergence is being analyzed, since the Gauss-Seidel algorithm is a coordinatewise *contraction iteration* (i.e., it always moves closer to the solution) [33]. So, as long as the solution vector has all the components inside the bounds, this characteristic is irrelevant with respect to convergence of any contraction iteration.
- Under the *zero mean* Gauss-Markov *random field, linear blur*, and *additive white Gaussian noise* assumptions, Besag's *iterated conditional modes* (ICM) algorithm [5] is equivalent to the Gauss-Seidel scheme, as we observed in [6]. A relation between ICM and the iterative solution of a system of equations was already recognized in [5], although no use was made of it to study convergence.

C. Parallel Algorithm

The distributed structure of neural networks is fully exploited if, instead of updating just one element at each step, all elements change state simultaneously. To obtain a parallel algorithm, matrix $\mathbf{G}$ is taken as a diagonal matrix

$$\mathbf{G} = \text{diag}\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_{MN}\}. \quad (14)$$

Iterative scheme (6), with $\mathbf{G}$ as given by (14), is neurally implemented as next described [35]:

Network 2: Define a network of MN real valued elements $\{x_i \in R,\ i = 1, 2, \ldots, MN\}$, each assigned to one image pixel. Let $H_{ij} = -C_{ij}/\varepsilon_i$ be the interconnection weight between elements i and j, and $J_i = b_i/\varepsilon_i$ be the bias input to element i. At each iteration t, all elements update their states simultaneously, according to

$$x_i(t+1) = x_i(t) + u_i(t) \quad \text{with} \quad u_i(t) = \sum_{j=1}^{MN} H_{ij} x_j(t) + J_i. \quad (15)$$

Sufficient convergence conditions on the parameters ε_i are given by the following theorem (which is proved in Appendix B):

Theorem 1: Let $\mathbf{Cx} = \mathbf{b}$ be the system to be solved. Iteration (6), with $\mathbf{G}$ as given by (14), i.e., the iterative process defined by (15), converges if

$$\varepsilon_i > \frac{1}{2} \sum_{j=1}^{MN} |C_{ij}|, \quad \forall_{i=1,2,\ldots,MN}. \quad (16)$$

If matrix $\mathbf{C}$ verifies the diagonal dominance conditions

$$C_{ii} > \frac{1}{2} \sum_{j=1}^{MN} |C_{ij}| \quad (17)$$

or equivalently, since $C_{ii} > 0$

$$C_{ii} > \sum_{j=1,\ j \neq i}^{MN} |C_{ij}| \quad (18)$$

then the Jacobi algorithm (obtained with $\mathbf{G} = \text{diag}(\mathbf{C})$) converges and can be used [33], [34]. However, the diagonal dominance condition on $\mathbf{C}$ cannot be guaranteed *a priori*. This observation suggested replacing $\text{diag}(\mathbf{C})$ by $\mathbf{G}$ as given by (14).

To study the effect of parameters ε_i on the convergence rate, let us consider the simpler case $\varepsilon_i = \varepsilon,\ i = 1, 2, \ldots, MN$, that is $\mathbf{G} = \varepsilon\mathbf{I}$ (known as Richardson's method [33]) for which convergence condition (16) reduces to $\varepsilon > \rho(\mathbf{C})/2$ (see Appendix B). It can be shown (see [34]) that the best convergence rate is obtained with

$$\varepsilon = \varepsilon_{\text{opt}} \equiv \frac{\lambda_{\max}(\mathbf{C}) + \lambda_{\min}(\mathbf{C})}{2} = \frac{\rho(\mathbf{C})}{2}\left(\frac{\kappa(\mathbf{C}) + 1}{\kappa(\mathbf{C})}\right). \tag{19}$$

In (19), $\lambda_{\min}(\mathbf{C})$ and $\lambda_{\max}(\mathbf{C})$ stand respectively for the minimum and maximum eigenvalues of matrix $\mathbf{C}$; since $\mathbf{C}$ is positive definite (PD), $\lambda_{\min}(\mathbf{C})$ and $\lambda_{\max}(\mathbf{C})$ are positive and $\lambda_{\max}(\mathbf{C}) = \rho(\mathbf{C})$. Still in (19), $\kappa(\mathbf{C}) = \lambda_{\max}(\mathbf{C})/\lambda_{\min}(\mathbf{C})$ stands for the *condition number* of $\mathbf{C}$; this is the commonly used measure of the numerical difficulty of inverting a matrix [33], [34] (larger condition numbers correspond to more difficult matrices). By looking at (19) we can conclude that if $\mathbf{C}$ is very well conditioned, i.e., $\lambda_{\min}(C) \simeq \rho(\mathbf{C})$, it is advantageous to increase ε up to $\varepsilon_{\text{opt}} \simeq \rho(\mathbf{C})$. If not, i.e., $\lambda_{\min}(C) \ll \rho(\mathbf{C})$, then $\varepsilon_{\text{opt}} \simeq \rho(\mathbf{C})/2$ and the convergence rate cannot be much increased.

In [16], a different parallel algorithm intended to solve the same problem was presented. Its convergence proof is based on the condition $C_{ii} \geq \rho(\mathbf{C})$, for all diagonal elements C_{ii} of matrix $\mathbf{C}$ [16]. However, it can be shown that $C_{ii} \leq \rho(\mathbf{C})$, for $i = 1, 2, \ldots, MN$, for any PD matrix, as is the case of $\mathbf{C}$. To prove this, notice that since $\mathbf{C}$ is PD, and using the definition of matrix norm induced by the Euclidean vector norm (which in the particular case of real symmetric matrices coincides with the spectral radius) [33], [34]

$$\rho(\mathbf{C}) = \|\mathbf{C}\|_2 = \max_{\|\mathbf{x}\|_2=1} \{\mathbf{x}^T\mathbf{C}\mathbf{x}\} \geq \mathbf{e}_i^T\mathbf{C}\mathbf{e}_i = C_{ii} \tag{20}$$

for all i, where $\mathbf{e}_i$ is a vector with a 1 in position i and zero elsewhere, which clearly satisfies $\| \mathbf{e}_i \|_2 = 1$. So the condition imposed in [16] is never verified (except for the trivial diagonal matrix) and the presented theorem cannot be used to assure convergence.

We now present a different implementation of the algorithm of Network 2, which can be considered if certain conditions are verified. If the underlying Gauss-Markov random field is assumed stationary with free boundary conditions, then matrix $\mathbf{A}^{-1}$ is block Toeplitz [36]. If matrix $\mathbf{B}^T\mathbf{B}$ is also block Toeplitz (which is the case if the blur mechanism can be written as a convolution) then matrix $\mathbf{C}$ is block Toeplitz and the iterative process of Network 2, with $\mathbf{G}$ equal to $\varepsilon\mathbf{I}$, can be written as a successive convolution

$$\mathbf{x}(t+1) = \mathbf{x}(t) + (\mathbf{x}(t) \star \mathbf{h}) + \frac{\mathbf{b}}{\varepsilon} \tag{21}$$

where $\star$ stands for 2D convolution and $\mathbf{h}$ is a convolution kernel easily obtainable from $\mathbf{H}$. Iteration (21) can be implemented on convolution oriented image processing hardware, as depicted in Fig. 2.

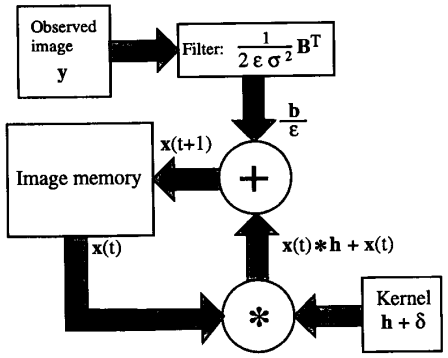


Fig. 2. Convolution-oriented implementation of the parallel algorithm of Network 2.

III. Sequential Algorithm: Implementation on a Standard Hopfield Neural Network

The two networks described, if implemented in hardware, require a special purpose design. The structure we introduce in this section, being standard, can be implemented on any general purpose neural architecture able to be configured as a Hopfield network of binary elements (there are now optical implementations of the Hopfield network which can be used if very fast operation is required [37]–[40]).

To obtain a standard binary Hopfield network a coding scheme has to be used. In this paper, we adopt the *bit density coding* (BDC) scheme in which each element of $\mathbf{x}$ is represented by the average of a set of L binary elements

$$x_i(t) = \frac{1}{L}\sum_{j=1}^{L} f_{ij}(t), \quad \text{for } i = 1, 2, \ldots, MN \tag{22}$$

where $f_{ij} \in \{0, 1\}$, as proposed by Takeda, *et al.* [25], and adopted by Zhou, *et al.* [18] for image restoration. Each x_i can take values in a discrete and bounded set, $x_i \in \{0, \frac{1}{L}, \frac{2}{L}, \ldots, 1\}$. As we have observed in Section II, the bounded nature of the available set of values poses no convergence problems as long as the solution vector coordinates belong to the interval $[0, 1]$; choosing other bounds is simply a matter of renormalization. The discrete nature of the set of values has numerical precision implications that will be studied in the next section.

A. An Auxiliary Subnetwork

Let us now introduce an interconnected set of binary elements that "knows" how to evolve towards a BDC representation of its input. Its structure has some resemblance to the Hamming network [14].

Network 3: Assume a standard Hopfield network of L binary elements $\{f_i \in \{0, 1\},\ i = 1, 2, \ldots, L\}$. Let the autoconnection weights be zero and all other interconnection weights be equal to $-1/L$. All elements have a common bias input of $b - 1/(2L)$. Assume that $b \in [0, 1]$ and that the standard Hopfield updating rule (see Appendix A) is followed.

Let $\mathcal{R}(x)$ stand for the operator that yields the nearest integer to its argument x. If $x = n + 0.5$, n being an integer, then take $\mathcal{R}(x) = n$. The behavior of Network 3 is described by the next theorem (which is proved in Appendix C).

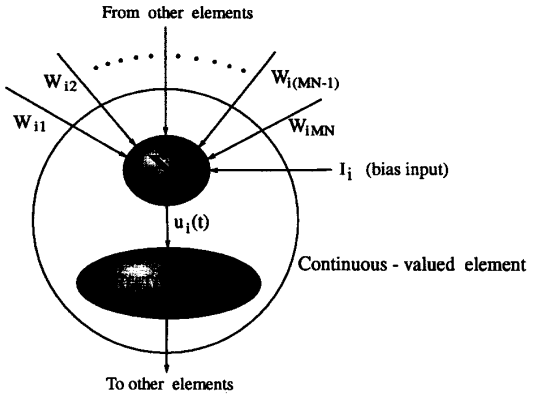


Fig. 3. Continuous valued elements of Network 1.

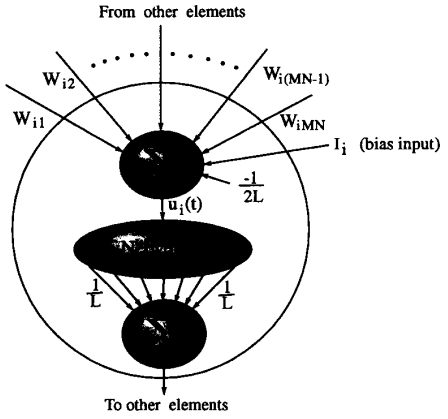


Fig. 4. Subset of elements of Network 4 interpreted as the replacement of each graded element of Network 1 by one Network 3.

Theorem 2: Network 3 evolves until exactly $\mathcal{R}(bL)$ elements are in state 1.

Network 3 is thus an automatic BDC calculator which is used as a building block of the complete Hopfield network we are proposing for image restoration.

B. The Proposed Network

The complete binary network is built by replacing each continuous element x_i of Network 1, represented in Fig. 3, by a subnet equal to a full Network 3 $\{f_{ij},\ j = 1, 2, \ldots, L\}$, with each x_i represented as in (22). In formal terms, the network is defined as follows:

Network 4: Define a network of MNL binary valued elements $\{f_{ij} \in \{0,1\},\ i = 1, 2, \ldots, MN,\ \ j = 1, 2, \ldots, L\}$. Let $T_{ij,kl}$, the interconnection weight between elements (ij) and (kl), be given by

$$T_{ij,kl} = \begin{cases} 0 & \Leftarrow \quad i = k \text{ and } j = l, \\ -1/L & \Leftarrow \quad i = k \text{ and } j \neq l, \\ W_{ik}/L & \Leftarrow \quad i \neq k \end{cases} \tag{23}$$

and the bias input to element (ij) be $H_{ij} = I_i - 1/(2L)$, where I_i and W_{ik} are as given in Network 1. The network is initialized with any state and follows the standard Hopfield updating rule (see Appendix A).

Observe that convergence is guaranteed by the fact that the autoconnections are zero. The total input to each element can be divided into two parts, $u_{ij}(t) = u_{ij}^E(t) + u_{ij}^I(t)$, the first one from outside its own subset, and the second one from its own subset. The external part is given by

$$\begin{aligned} u_{ij}^E(t) &= \sum_{\substack{k=1 \\ k \neq i}}^{MN} \sum_{l=1}^{L} T_{ij,kl} f_{kl}(t) + H_{ij} \\ &= \sum_{\substack{k=1 \\ k \neq i}}^{MN} \frac{W_{ik}}{L} \sum_{l=1}^{L} f_{kl}(t) + H_{ij} \\ &= \sum_{\substack{k=1 \\ k \neq i}}^{MN} W_{ik} x_k(t) + I_i - \frac{1}{2L} \\ &= x_k(t+1) - \frac{1}{2L} \end{aligned}$$

where $x_k(t+1)$ is the updated value that would be computed by Network 1. This total external input $u_{ij}^E(t) = x_k(t+1) - 1/(2L)$, being the same to all elements of the i^{th} subnet $\{f_{ij},\ j = 1, \ldots, L\}$, can be viewed as a common bias input (referred to as $b - 1/(2L)$ in the definition of Network 3). The subnetwork will then take a step towards the BDC representation of $x_i(t+1)$, as would be given by (11) in Network 1. That is, each time an element f_{ij} changes state, the value x_i represented by its subset according to (22) moves one $1/L$-step in the direction of the BDC representation of $x_i(t+1)$, as would be given by (11) in Network 1. The steps of Network 4 follow the same direction as those of Network 1, towards the solution, but have $1/L$ length. The interpretation of the ith subnet of Network 4, as a replacement for the continuous valued elements of Network 1 is depicted in Fig. 4. The summation blocks, represented in Fig. 4, do not really exist, since in a standard Hopfield network each element is responsible for computing its own total input; their inclusion in the figure is intended to illustrate the relation between Networks 1 and 4. However, it is possible to implement a network using exactly the structure represented in Fig. 4, thus using much fewer connections at the cost of not having a fully standard Hopfield network.

In conclusion, the proposed binary network is a $1/L$-step-size version of Network 1. Fault-tolerance is assured since the BDC scheme has many possible solutions, as is clear from (22). Should one element fail, another element would, later, automatically turn itself on to replace the damaged one. In other words, the energy of the binary network has many minima (since the autoconnections are zero, all the fixed points are minima of the standard Hopfield energy, see Appendix A) all of them corresponding to different BDC representations of the unique solution with respect to $\mathbf{x}$.

IV. The Effect of Finite Numerical Precision

In this section we examine the effect of finite numerical precision on the proposed algorithms. Consider that intermediate

computations can be made with infinite precision but that each neuron (pixel) can only store values using some finite word length. We will assume that the elements of Networks 1 and 2 can store L equidistant different values between 0 and 1. For Network 4, this is true by construction. As noted before, the bounded nature of the available set of values poses no convergence problem and other bounds are simply a matter of renormalization. The important numerical limitation is related to the discreteness of the set of values and not to its bounded nature.

A. Sequential Algorithms

Since both Network 4 and the discrete version of Network 1 implement versions of the same basic algorithm, they can be simultaneously analyzed. Both structures follow the same update direction which is as given by (11). The evolution of the discrete versions of Networks 1 and 4, can be written, respectively, as

$$x_i(t+1) = x_i(t) + \frac{1}{L}\mathcal{R}\left(L\sum_{j=1}^{MN} W_{ij}x_j(t) + L\,I_i\right) \quad (24)$$

and

$$x_i(t+1) = x_i(t) + \frac{1}{L}\,\mathrm{Sgn}\left(\mathcal{R}\left(L\sum_{j=1}^{MN} W_{ij}x_j(t) + L\,I_i\right)\right). \quad (25)$$

In (24) and (25), $\mathcal{R}(\cdot)$ stands for the already mentioned operator that returns the nearest integer to its argument, and $\mathrm{Sgn}(\cdot)$ stands for the *sign function* which is defined as

$$\mathrm{Sgn}(x) = \begin{cases} 1 & \Leftarrow \quad x > 0 \\ 0 & \Leftarrow \quad x = 0 \\ -1 & \Leftarrow \quad x < 0. \end{cases} \quad (26)$$

From (24) and (25) it is clear that the fixed points of both schemes are characterized by

$$\left|\sum_{j=1}^{MN} W_{ij}x_j + I_i\right| < \frac{1}{2L}, \qquad \text{for} \quad i = 1, 2, ..., MN. \quad (27)$$

This condition underlies the following theorem (which is proved in Appendix D).

Theorem 3: Let $\mathbf{x}$ be a fixed point of Network 4 and of the L-levels discrete version of Network 1. The square root of the mean square error (RMSE) of $\mathbf{x}$ with respect to the exact solution $\mathbf{x}^* = \mathbf{C}^{-1}\mathbf{b}$, is upper bounded as follows:

$$\mathrm{RMSE} \equiv \frac{\|\mathbf{x} - \mathbf{x}^*\|_2}{\sqrt{MN}} < \frac{\kappa(\mathbf{C})}{2L}. \quad (28)$$

Theorem 3 states that the RMSE bound depends, in a natural expectable way, on the number of quantization levels L, and on the condition number of matrix $\mathbf{C}$. A better conditioned matrix $\mathbf{C}$ leads to a lower bound for the RMSE. We can also interpret inequality (28) in terms of the degradation model parameters. Let us multiply $E(\mathbf{x})$ by σ^2, which does not affect the problem and makes the following discussion simpler; having done so, we get $\mathbf{C} = \sigma^2\mathbf{A}^{-1} + \mathbf{B}^T\mathbf{B}$, and $\mathbf{b} = \mathbf{B}^T\mathbf{y}$. Inspecting matrix $\mathbf{C}$ makes clear that, from a numerical point of view, what MAP estimation (and regularization) does is add to an ill-conditioned matrix $\mathbf{B}^T\mathbf{B}$ another matrix $\sigma^2\mathbf{A}^{-1}$ which is well conditioned, thus producing a better conditioned matrix. For a given fixed $\mathbf{B}$, the condition number of $\mathbf{C}$ (and thus the RMSE bound) gets better when the noise variance increases. Assume now that we have noise-free data ($\sigma^2 = 0$) in order to focus our analysis on matrix $\mathbf{B}$. Assume also that $\mathbf{B}$ is square and that the blur is symmetric ($\mathbf{B}$ is a symmetric matrix); then, the eigenvalues of $\mathbf{C} = \mathbf{B}^T\mathbf{B}$ are the squares of those of $\mathbf{B}$ and the same is true for the condition number

$$\kappa(\mathbf{B}^T\mathbf{B}) \equiv \frac{\rho(\mathbf{B}^T\mathbf{B})}{\lambda_{\min}(\mathbf{B}^T\mathbf{B})} = \left(\frac{\rho(\mathbf{B})}{\lambda_{\min}(\mathbf{B})}\right)^2 = \kappa^2(\mathbf{B}) \quad (29)$$

with $\lambda_{\min}(\mathbf{B})$ here standing for the minimum absolute eigenvalue of $\mathbf{B}$. Milder blurs, i.e., those for which the diagonal elements of $\mathbf{B}$ are much larger then the off-diagonal elements, have a smaller range of eigenvalues (recall Gerschgorin's theorem [34]) and thus a smaller (better) condition number than stronger blurs; these are characterized by having relatively large off-diagonal elements, when compared to the diagonal elements, and consequently a larger range of eigenvalues and a worse condition number. In conclusion, the numerical error is directly proportional to the severity of the blur (as measured by the condition number of matrix $\mathbf{B}^T\mathbf{B}$) and inversely proportional to the noise power. Although at first sight this fact may seem counterintuitive, it does make sense: When the noise power increases, the restoration process becomes fundamentally low-pass, thus more numerically robust; on the other hand, the deblurring process is basically high-pass and consequently less numerically robust. Recall that we are talking about the RMSE with respect to the exact solution $\mathbf{C}^{-1}\mathbf{b}$, and not to the true original image.

B. Parallel Algorithm

For Network 2, the parameters ε_i can be adjusted and this exerts influence on the numerical error bound. Let us again consider the simpler case where we have all the ε_i equal to ε. The RMSE bound depends on ε as stated in the following theorem (which is also proved in Appendix D):

Theorem 4: Let $\mathbf{x}$ be a fixed point of the L-levels discrete version of Network 2, with $\varepsilon_i = \varepsilon$, for $i = 1, 2, ..., MN$. The square root of the mean square error (RMSE) of $\mathbf{x}$ with respect to the exact solution $\mathbf{x}^* = \mathbf{C}^{-1}\mathbf{b}$, is upper bounded as follows:

$$\mathrm{RMSE} \equiv \frac{\|\mathbf{x} - \mathbf{x}^*\|_2}{\sqrt{MN}} < \frac{\varepsilon}{2L\lambda_{\min}(\mathbf{C})}. \quad (30)$$

The numerical error bound depends, again in an intuitively logical way, on the number of quantization levels, and on the minimum eigenvalue of $\mathbf{C}$; badly conditioned systems, which have very small minimum eigenvalues, present worse error bounds. On the other hand, when parameter ε is increased up to $\varepsilon_{\mathrm{opt}}$, according to (19), in order to maximize the convergence rate, the error bound also increases; there is a trade-off between convergence rate and numerical error in the interval $\rho(\mathbf{C})/2 = \varepsilon_{\min} < \varepsilon < \varepsilon_{\mathrm{opt}}$; range $\varepsilon > \varepsilon_{\mathrm{opt}}$, where both convergence rate and the RMSE bound deteriorate as ε is increased, should be avoided; for $\varepsilon < \varepsilon_{\min}$, convergence can no longer be guaranteed by Theorem 4.

To establish a comparison with the RMSE bound for the sequential algorithm, assume that ε is chosen equal to ε_{opt} and rewrite (30) as

$$\text{RMSE} < \frac{\varepsilon_{\text{opt}}}{2L\lambda_{\min}(\mathbf{C})} = \frac{\kappa(\mathbf{C})}{2L}\left(\frac{\kappa(\mathbf{C})+1}{2\kappa(\mathbf{C})}\right). \tag{31}$$

For well conditioned matrices, $\kappa(\mathbf{C}) \simeq 1$ and the error bound is the same as for the sequential scheme. However, for ill-conditioned matrices, $\kappa(\mathbf{C}) \gg 1$ and the error bound is two times better than the one given by (28).

Finally, recall that the bounds that were derived in this section are only valid if we are working with pixel values in $[0; 1]$. If other ranges of values are desired, the bounds will be affected by an appropriate multiplicative constant, e.g., 256 if the image values are in [0; 256].

V. Examples

All examples presented were obtained by simulating the networks on a conventional computer using (512 × 512 pixels / 8 bits-per-pixel) digital images. We have verified that the results produced by all the studied schemes are visually similar, which is natural since they result from different implementations of the same restoration criterion; accordingly, we will only present the ones corresponding to Network 1 (in fact a 256-levels discrete version) which is clearly the best suited for implementation on a conventional computer. To assess the importance of the visiting schedule, three options were tested: cyclic; cyclic with alternating directions; and random. We concluded that there was no difference in the visual results nor in the convergence rate.

Instead of modeling the original image as a Gauss-Markov random field via the specification of its covariance matrix $\mathbf{A}$, its inverse $\mathbf{A}^{-1}$ (called in [36] the *potential matrix*) which is highly structured and sparse, is specified. We considered $\mathbf{x}$ as a first-order field, for which matrix $\mathbf{A}^{-1}$ is block tridiagonal with tridiagonal blocks [36].

Fig. 5 presents the original undegraded image used in the following examples. The image of Fig. 6(a) is artificially blurred by a 9 × 9 uniform low-pass filter, and Fig. 6(b) presents its restored version. An example with a 9 × 9 Gaussian-shaped blur (variance 3.0) is exhibited in Fig. 7(a)–(b), while Fig. 8(a)–(b) shows a test with a large uniform motion blur (1 × 31).

In the preceding cases, no noise was artificially added; however, the algorithm assumed noise with standard deviation $\sigma = 1$. In the next two examples, noise was artificially added, after a mild Gaussian-shaped blur (3 × 3). Figs. 9(a) and 10(a) present the degraded images while Figs. 9(b) and 10(b) present the ones resulting from the restoration process. In the cases where noise is the main degradation present, the restored images do not look visually good, i.e., they appear too smooth. This is due to the fact that, in this case, the restoration process is basically a low-pass filter and the human visual system would accept more noise in exchange for the high-image frequency components (detail) lost [1].

The algorithm was also tested on nonartificially degraded data: a side-scan sonar image of the ocean bottom. The original and restored images are shown in Fig. 11(a)–(b), respectively. To adjust the parameters, several values were tested and those providing the visually better results were selected; the result shown was obtained by assuming a 5 × 5 Gaussian-shaped blur and $\sigma = 3$.

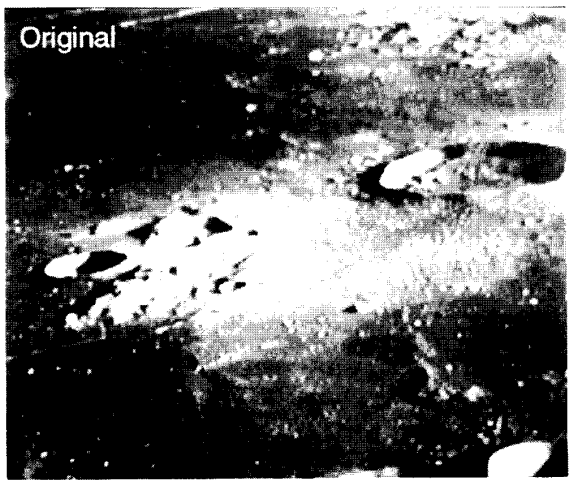


Fig. 5. Original undegraded image.

In all the preceding examples, Network 1 always reached convergence after about $15 \sim 20$ iterations, as is clear from the typical energy evolution along the iterations, presented in Fig. 12 (relative to the uniform blur case of Fig. 6). By iterations we mean a set of visits to all elements of the network, in order to perform meaningful comparisons with the parallel algorithm.

Network 2, being related to the Jacobi algorithm, is predictably slower and convergence is usually attained after $30 \sim 40$ iterations, depending on the choice of the parameters ε_i. Fig. 13 plots the energy evolution of Network 2, for three different choices of ε: 0.26 (which is the minimum value given by Theorem 1 as guaranteeing convergence in this case); 0.36; and 0.46. The plot reveals that faster convergence can be obtained by using ε =0.36, but the scheme becomes slower if ε is further increased up to 0.46. This is in accordance with (19), which states that the optimal value for ε, in terms of convergence rate, is slightly above the minimum value that guarantees convergence.

Network 4, due to its unit step size, has a much slower convergence rate than Networks 1 and 2. For this network, convergence is highly dependent on the number of neurons per pixel (parameter L) and on the initial condition. If the network is initialized with some BDC representation of the observed image, convergence is reached after about $L \sim 2L$ iterations. Fig. 14 shows a typical example of the energy evolution for this network. Here, one iteration stands for one visit to all subnets, i.e., MN updates.

Figs. 12–14 try to give an idea of the time evolution of the studied schemes. The exact time, however, is completely dependent on the particular hardware used and so it is not very significant. The proposed schemes are specially suited to be implemented on neural hardware and, as pointed out in Section I–C, are under a special restriction: the updating of

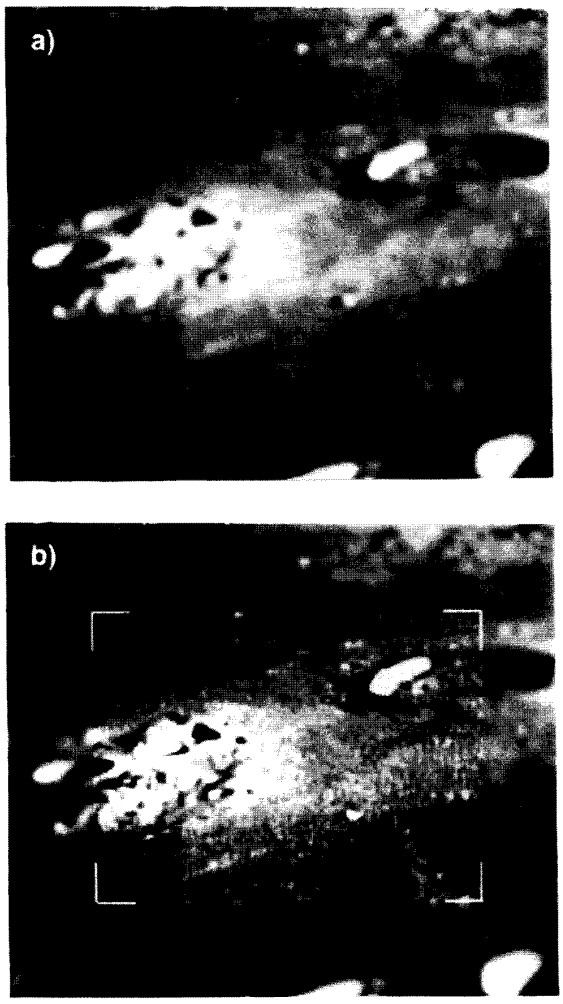


Fig. 6. (a) Image degraded by a 9 × 9 uniform low-pass filter; (b) restored image.

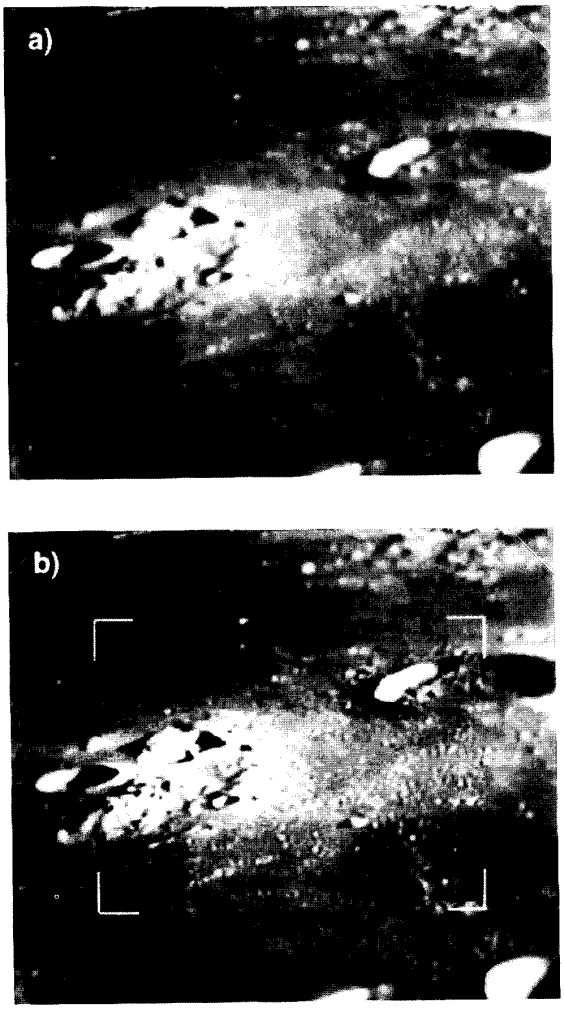


Fig. 7. (a) Image degraded by a 9 × 9 Gaussian-shaped low-pass filter; (b) restored image.

each element (pixel) depends only on local information. If this restriction was lifted, other better algorithms (e.g., optimal step gradient descent) could be adopted in which information has to be shared by all the pixels. For these reasons, we think it is unfair to compare the proposed neural algorithms with standard algorithms using a standard machine which is an unfavorable environment for the former. As a final comment, note that the quality of the images resulting from the restoration processes is fundamentally determined by the restoration criterion adopted and not by the particular algorithm used to implement it. We do not propose new restoration criteria, but rather introduce new neural implementations of a well known classical approach.

VI. Conclusion

In this paper neural network implementations of iterative image restoration algorithms were developed. Starting with a general family of iterative schemes, two classes of network implementations were proposed: one using graded neurons and the other one using binary neurons. We can conclude that the adopted strategy, which consists in implementing known algorithms instead of directly mapping the problem into a neural network, has an important advantage: all results concerning the implemented schemes can be directly imported and used to study convergence and other numerical issues. Two networks of graded elements were proposed (one operating in sequential mode and the other in parallel mode), proved to converge, and studied in terms of numerical precision. The binary network, which is a zero autoconnection standard Hopfield network with its well known convergence property, was also analyzed from the numerical point of view.

In all the examples presented, except the one with the side-scan sonar image, there is knowledge of the degradation process parameters. In realistic situations, this is seldom true. This fact stresses the importance of adaptive algorithms able to perform, simultaneously, both restoration and parameter es-

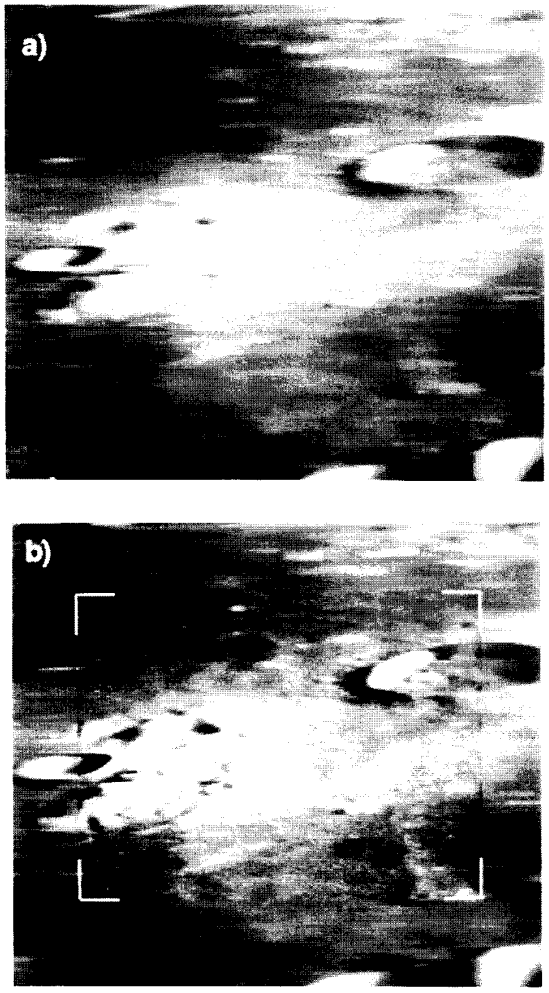


Fig. 8. (a) Image degraded by a 1 × 31 uniform motion blur; (b) restored image.

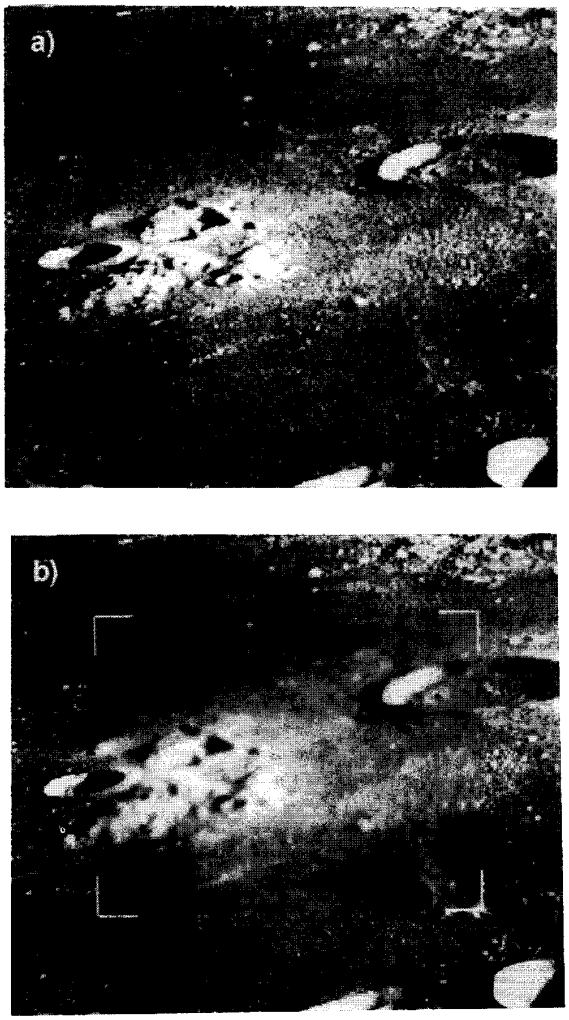


Fig. 9. (a) Image contaminated by (σ = 30) white Gaussian noise after a 3 × 3 Gaussian-shaped blur; (b) restored image.

timation. One approach to this problem in which the proposed algorithms can be imbeded as part of a more complex scheme is the (already mentioned) *expectation-maximization* (EM) algorithm [28]–[30]. There are also other possible directions such as adaptive versions of the *iterated conditional modes* algorithm [41].

Appendix A
The Hopfield Network

The Hopfield network is formed by a number, say M, of binary elements $\{n_i \in \{0,1\}, \quad i = 1,2,...,M\}$, whose outputs are fed back to all other elements via a weight matrix denoted by $\mathbf{T}$ (T_{ij} is the connection weight between element i and element j). Each element also has a bias input I_i. At each iteration, a single element i changes its state according to

$$n_i(t+1) = \begin{cases} 1, & u_i(t) \geq 0 \\ 0, & u_i(t) < 0 \end{cases}$$

where

$$u_i(t) \equiv \sum_{j=1}^{M} T_{ij} n_j(t) + I_i$$

is referred to as the *total input* (TI) to element i. As shown in [13], if the weights are symmetric ($T_{ij} = T_{ji}$) and the autoconnections zero ($T_{ii} = 0$), this network converges to fixed points which are local minima of the energy function

$$E(t) = -\frac{1}{2} \sum_{i=1}^{M} \sum_{j=1}^{M} T_{ij} n_i(t) n_j(t) - \sum_{i=1}^{M} I_i n_i(t).$$

The zero autoconnections condition was further studied by Ye, *et al.* [17], leading to the following conclusion: If the autoconnections are nonnegative ($T_{ii} \geq 0$) the network keeps its energy reduction property; on the other hand, the fixed

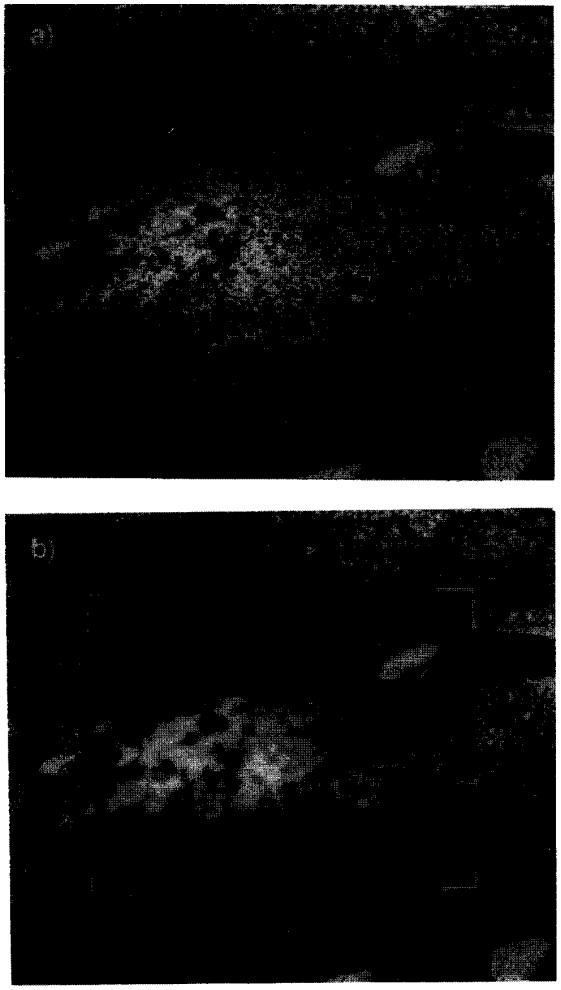


Fig. 10. (a) Image contaminated by (σ = 90) white Gaussian noise after a 3 × 3 Gaussian-shaped blur; (b) restored image.

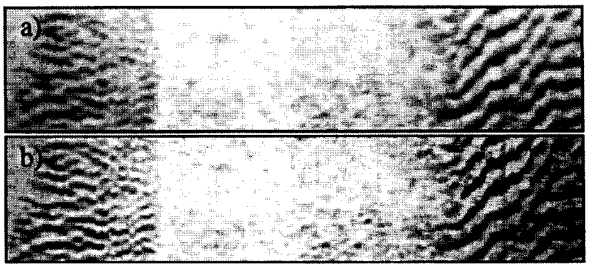


Fig. 11. (a) Original side-scan sonar image of the ocean floor; (b) restored image.

points of the network's evolution are local minima of the energy if and only if the autoconnections are nonpositive ($T_{ii} \leq 0$). So, the configurations suitable for optimization applications must have zero autoconnections.

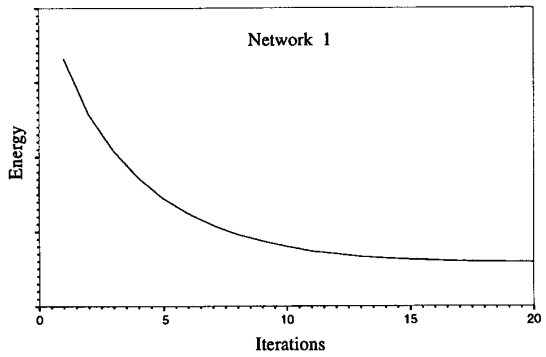


Fig. 12. Typical evolution of the energy E(x) for the algorithm of Network 1.

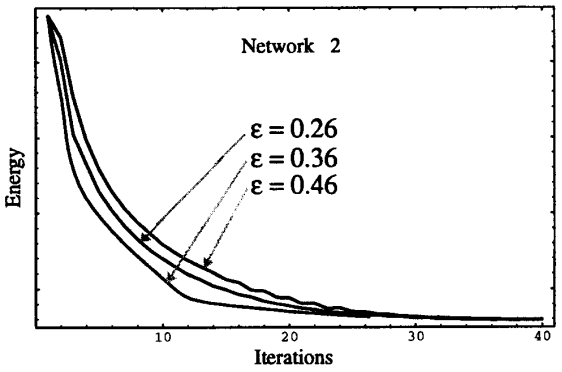


Fig. 13. Evolution of the energy E(x) for the algorithm of Network 2, using three different values for parameter ϵ.

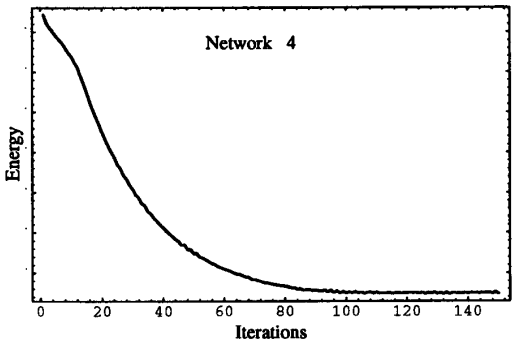


Fig. 14. Typical evolution of the energy E(x) for the algorithm of Network 4.

APPENDIX B
PROOF OF THEOREM 1

The proof of Theorem 1 is based on the following theorem (see [34]):

Theorem 5: Let $\mathbf{G}$ be a non-singular and symmetric matrix, and $\mathbf{C} = \mathbf{G} - \mathbf{H}$ be positive definite (PD). Then, $\mathbf{M} = \mathbf{G}^{-1}\mathbf{H}$ is convergent if and only if $\mathbf{Q} \equiv \mathbf{G} + \mathbf{H} = 2\mathbf{G} - \mathbf{C}$ is PD.

According to Theorem 5, $\mathbf{M}$ is convergent if and only if $\mathbf{Q}$ is PD. A sufficient condition is diagonal dominance. Since

the elements of $\mathbf{Q}$ are

$$Q_{ij} = \begin{cases} 2\varepsilon_i - C_{ii} & \Leftarrow \quad i = j \\ -C_{ij} & \Leftarrow \quad i \neq j \end{cases}$$

the diagonal dominance condition on matrix $\mathbf{Q}$ is

$$2\varepsilon_i - C_{ii} > \sum_{j=1,\ j\neq i}^{MN} |C_{ij}|$$

or

$$\varepsilon_i > \frac{1}{2}\sum_{j=1}^{MN} |C_{ij}|$$

since $C_{ii} > 0$. This concludes the proof of Theorem 1.

In the particular case of taking all ε_i equal to some value ε, we have $\mathbf{G} = \varepsilon\mathbf{I}$, and $\mathbf{Q} = 2\varepsilon\mathbf{I} - \mathbf{C}$. For this case the PD condition on $\mathbf{Q}$ becomes

$$\lambda_{\min}(\mathbf{Q}) = \lambda_{\min}(2\varepsilon\mathbf{I} - \mathbf{C}) = 2\varepsilon - \lambda_{\max}(C) = 2\varepsilon - \rho(\mathbf{C}) > 0$$

since a matrix is PD if and only if all its eigenvalues are positive. Finally, $2\varepsilon - \rho(\mathbf{C}) > 0$ is equivalent to $\varepsilon > \rho(\mathbf{C})/2$.

Appendix C
Proof of Theorem 2

To prove Theorem 2, let n_1 be the number of elements in state 1, and rewrite it as $n_1 = bL + \delta$. The total input to each element in state 0, denoted by u^0, and to each element in state 1, denoted by u^1, is equal to, respectively

$$u^0 = -\frac{1}{L}(bL + \delta) + b - \frac{1}{2L} = -\frac{1}{L}\left(\frac{1}{2} + \delta\right)$$

$$u^1 = -\frac{1}{L}(bL + \delta - 1) + b - \frac{1}{2L} = \frac{1}{L}\left(\frac{1}{2} - \delta\right).$$

Three situations have to be analyzed (recall that only one element is allowed to change state at each iteration):

- If $\delta \leq -\frac{1}{2}$, then $u^0 \geq 0$ and $u^1 \geq 0$ and so state 1 elements keep their 1 state, and state 0 elements switch to 1.
- If $-\frac{1}{2} < \delta \leq \frac{1}{2}$, then $u^0 < 0$ and $u^1 \geq 0$ and so both state 1 and state 0 elements keep their values.
- If $\delta > \frac{1}{2}$, then $u^0 < 0$ and $u^1 < 0$ and so state 0 elements keep their 0 state, and state 1 elements switch to 0.

This proves that the only fixed points of the network's evolution verify $n_1 = \mathcal{R}(bL)$.

Appendix D
Proof of Theorems 3 and 4

Proof of Theorem 3. The starting point of the proof is the fixed-point expression (27). Given the definitions of W_{ij} and of I_i, and knowing that $C_{ii} > 0$, the conditions stated in (27) can be rewritten as

$$\left| b_i - \sum_{j=1}^{MN} C_{ij}\, x_j(t) \right| < \frac{C_{ii}}{2L} \text{ for } i = 1, 2, \ldots, MN. \qquad (32)$$

Invoking the definition of *maximum norm* of a vector, $\|\mathbf{x}\|_\infty = \max\{|x_i|\}$, (32) implies that

$$\| \mathbf{b} - \mathbf{Cx} \|_\infty \leq \frac{\max\{C_{ii}\}}{2L}.$$

The bound can now be derived as

$$\begin{aligned} \text{RMSE} = \frac{\| \mathbf{C}^{-1}\mathbf{b} - \mathbf{x} \|_2}{\sqrt{MN}} &= \frac{\| \mathbf{C}^{-1}(\mathbf{b} - \mathbf{Cx}) \|_2}{\sqrt{MN}} \\ &\leq \frac{\| \mathbf{C}^{-1} \|_2 \| \mathbf{b} - \mathbf{Cx} \|_2}{\sqrt{MN}} \\ &\leq \| \mathbf{C}^{-1} \|_2 \| \mathbf{b} - \mathbf{Cx} \|_\infty \\ &< \frac{\rho(\mathbf{C})}{2L\,\lambda_{\min}(\mathbf{C})} = \frac{\kappa(\mathbf{C})}{2L} \end{aligned}$$

where we have invoked the following facts:

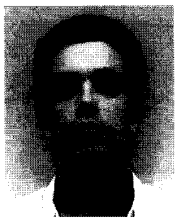
a) For any n-dimensional vector $\mathbf{x}$, $\|\mathbf{x}\|_2 \leq \sqrt{n}\|\mathbf{x}\|_\infty$.
b) As stated in (20), $\max\{C_{ii}\} \leq \rho(\mathbf{C})$.
c) Since $\mathbf{C}$ is a PD matrix, $\|\mathbf{C}^{-1}\|_2 = \frac{1}{\lambda_{\min}(\mathbf{C})}$.

Proof of Theorem 4. The proof of Theorem 4 follows exactly the same lines as that of Theorem 3, with ε taking the place of $\max\{C_{ii}\}$.

References

[1] H. C. Andrews and B. R. Hunt, *Digital Image Restoration.* Englewood Cliffs, NJ: Prentice Hall, 1977.
[2] M. Bertero, T. Poggio, and V. Torre, "Ill-posed problems in early vision," *Proc. IEEE*, vol. 76, pp. 869–889, Aug. 1988.
[3] A. Katsaggelos *et al.*, "A regularized iterative image restoration algorithm," *IEEE Trans. Signal Processing*, vol. 39, pp. 914–929, Apr. 1991.
[4] A. Tikhonov, A. Goncharsky, and V. Stepanov, "Inverse problems in image processing," in *Ill-Posed Problems in the Natural Sciences,* A. Tikhonov and A. Goncharsky, Eds. Moscow: Mir, 1987, pp. 220–232.
[5] J. Besag, "On the statistical analysis of dirty pictures," *J. R. Stat. Soc. B*, vol. 48, pp. 259–302, 1986.
[6] M. Figueiredo and J. Leitão, "Sequential and parallel iterative image restoration," in *Proc. 13th GRETSI Symp. Signal, Image Processing* (Juan-les-Pins, France), Sept. 1991, pp. 789–792.
[7] N. Karayiannis and A. Venetsanopoulos, "Regularization theory in image restoration—the stabilizing functional approach," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 1155–1179, July 1990.
[8] K. M. Hanson, "Bayesian and related methods in image reconstruction from incomplete data," in *Image Recovery: Theory and Applications*, H. Stark, Ed. New York: Academic, 1987, pp. 79–125.
[9] G. T. Herman, *Image Reconstruction from Projections.* New York: Academic, 1980.
[10] B. P. Medoff, "Image reconstruction from limited data. Theory and applications in computerized tomography," *Image Recovery: Theory and Applications*, H. Stark, Ed. New York: Academic, 1987, pp. 321–368.
[11] J. Paik and A. Katsaggelos, "Image restoration using a modified Hopfield network," *IEEE Trans. Image Processing*, vol. 1, pp. 49–63, Jan. 1992.
[12] A. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.
[13] J. Hopfield and D. Tank, "Neural computation of decisions in optimization problems," *Biolog. Cybernet.*, vol. 52, pp. 141–152, 1985.
[14] R. P. Lippmann, "An introduction to computing with neural networks," *IEEE Acoust., Speech, Signal Processing Magazine*, vol. 4, pp. 4–22, Apr. 1987.
[15] J. Abbiss, B. Brames, and M. Fiddy, "Superresolution algorithms for a modified Hopfield neural network," *IEEE Trans. Signal Processing*, vol. 39, pp. 1516–1523, July 1991.
[16] J. Paik and A. Katsaggelos, "Image restoration using the Hopfield network with nonzero autoconnections," in *Proc. Int. Conf. ASSP-ICASSP* (Albuquerque, NM), 1990, pp. 1909–1912.
[17] S. Yeh, H. Stark, and M. Sezan, "Hopfield-type neural networks," in *Digital Image Restoration*, A. Katsaggelos, Ed. New York: Springer Verlag, 1991, pp. 57–88.

[18] Y. Zhou *et al.*, "Image restoration using a neural network," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 1141–1151, July 1988.
[19] N. Nasrabadi and C. Choo, "Hopfield network for stereo vision correspondence," *IEEE Trans. Neural Networks*, vol. 3, pp. 5–13, Jan. 1992.
[20] Y. Zhou and R. Chellappa, "Stereo matching using a neural network," in *Proc. Int. Conf. ASSP-ICASSP*, 1988, pp. 940–1943.
[21] J. Paik and A. Katsaggelos, "Edge detection using a neural network," in *Proc. Int. Conf. ASSP-ICASSP* (Albuquerque, NM), 1990, pp. 2145–2148.
[22] C. Huang, "Parallel image segmentation using a modified Hopfield network," *Pattern Recognit. Letts.*, vol. 13, pp. 345–353, May 1992.
[23] T. Wang, X. Zhuang, and X. Xing, "Robust segmentation of noisy images using a neural network model," *Image, Vision Computing*, vol. 10, pp. 233–240, May 1992.
[24] D. Chen, R. Jain, and B. Schunck, "Surface reconstruction using neural networks," in *IEEE Conf. Comput. Vision, Pattern Recognit.* (Champaign, IL), 1992, pp. 815–817.
[25] M. Takeda and J. Goodman, "Neural networks for computation. Number representation and programing complexity," *Appl. Optics*, vol. 25, pp. 3033–3046, 1986.
[26] A. Papoulis, "A new algorithm in spectral analysis and band-limited extrapolation," *IEEE Trans. Circuits, Syst.*, vol. CAS-22, pp. 735–742, 1975.
[27] T. Simchony, R. Chellappa, and Z. Lichtenstein, "Pyramid implementation of optimal-step conjugate-search algorithms for some low-level vision problems," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 1408–1425, Nov. 1989.
[28] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood estimation from incomplete data via the EM algorithm," *J. R. Stat. Soc. B*, vol. 39, pp. 1–38, 1977.
[29] A. Katsaggelos and K. Lay, "Maximum likelihood identification and restoration of images using the expectation-maximization algorithm," in *Digital Image Restoration*, A. Katsaggelos, Ed. New York: Springer-Verlkag, 1991, pp. 143–176.
[30] R. Lagendijk, J. Biemond, and D. Boekee, "Blur identification using the expectation-maximization algorithm," in *Proc. Int. Conf. ASSP-ICASSP*, 1989, pp. 1397–1400.
[31] M. Figueiredo and J. Leitão, "Simulated tearing: An algorithm for discontinuity preserving visual surface reconstruction," in *IEEE Conf. Comput. Vision, Pattern Recognit.* (New York), June 1993, pp. 28–33.
[32] D. Geiger and F. Girosi, "Parallel and deterministic algorithms from MRF's: Surface reconstruction," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 401–412, May 1991.
[33] D. Bertsekas and J. Tsitsiklis, *Parallel and Distributed Computation. Numerical Methods.* Englewood, NJ: Prentice Hall, 1989.
[34] E. Isaacson and H. Keller, *Analysis of Numerical Methods.* New York: Wiley, 1966.
[35] M. Figueiredo and J. Leitão, "Image restoration using neural networks," in *Proc. Int. Conf. ASSP* (San Francisco), Mar. 1992, pp. II.409–II.412.
[36] J. Moura and N. Balram, "Recursive structure of noncausal Gauss-Markov random fields," *IEEE Trans. Inform. Theory*, vol. 38, pp. 334–354, Mar. 1992.
[37] A. David and B. Saleh, "Optical implementation of the Hopfield algorithm using correlation," *Appl. Optics*, vol. 29, pp. 1063–1064, 1990.
[38] N. Farhat *et al.*, "Optical implementation of the Hopfield model," *Appl. Optics*, vol. 24, pp. 1469–1475, 1985.
[39] J. Ohta *et al.*, "Optical implementation of an associative neural network model with a stochastic process," *Appl. Optics*, vol. 28, pp. 2426–2428, 1985.
[40] D. Psaltis and N. Farhat, "Optical information processing based on an associative memory model of neural nets with thresholding and feedback," *Appl. Optics*, vol. 28, pp. 98–100, 1985.
[41] M. Figueiredo and J. Leitão, "Bayesian estimation of ventricular contours in angiographic images," *IEEE Trans. Med. Imag.*, vol. 11, pp. 416–429, Sept. 1992.



Marío A. T. Figueiredo was born in Luanda, Angola, in 1962. He received both the E.E. degree and the M.S. degree in electrical and computer engineering from Instituto Superior Técnico (IST), Technical University of Lisbon, Portugal, in 1989 and 1990, respectively.

He is currently a Teaching Assistant at the Department of Electrical and Computer Engineering of IST, where he is also completing his Ph.D., and is a Research Assistant at Centro de Análise de Sinais, Technical University of Lisbon. His research interests include image processing, medical imaging, low-level computer vision, neural networks, detection and estimation theory.

José M. N. Leitão was born in Aldeia Velha-Avis, Portugal, in 1946. He received the E.E. degree and the Ph.D. in electrical engineering, in 1970 and 1983, respectively, both from Instituto Superior Técnico (IST), Technical University of Lisbon. He received the "Agregado" degree in electrical and computer engineering, also from IST, in 1992.

He was with the Laboratory of Physiology of the Instituto Gulbenkian de Ciência, Oeiras, Portugal, from 1970 to 1972. After spending three years at the University of Tübingen, Germany, he joined the faculty of IST in 1976, where he is currently an Associate Professor with the Department of Electrical and Computer Engineering, teaching courses on telecommunications and communication theory. He is also the coordinator of the Communication Theory Research Group in the Centro de Análise e Processamento de Sinais, Technical University of Lisbon. His main research interests are communication theory, pattern recognition, and signal and image processing.