Biol. Cybern. 66, 151–158 (1991)

Biological Cybernetics

Image segmentation using a neural network

A. Ghosh, N. R. Pal, and S. K. Pal

Electronics and Communication Sciences Unit, Indian Statistical Institute, 203 Barrackpore Trunk Road, Calcutta 700 035, India

Received December 10, 1990/Accepted in revised form June 15, 1991

Abstract. An object extraction problem based on the Gibbs Random Field model is discussed. The Maximum a'posteriori probability (MAP) estimate of a scene based on a noise-corrupted realization is found to be computationally exponential in nature. A neural network, which is a modified version of that of Hopfield, is suggested for solving the problem. A single neuron is assigned to every pixel. Each neuron is supposed to be connected only to all of its nearest neighbours. The energy function of the network is designed in such a way that its minimum value corresponds to the MAP estimate of the scene. The dynamics of the network are described. A possible hardware realization of a neuron is also suggested. The technique is implemented on a set of noisy images and found to be highly robust and immune to noise.

1 Introduction

Image segmentation and object extraction play a key role in image analysis and computer vision. Most of the existing techniques, both classical (Gonzalez and Wintz 1977, Rosenfeld and Kak 1982) and fuzzy set theoretic (Pal and Dutta Majumder 1986), are sequential in nature and the segmented output can not be obtained in real time.

In order to get the output in real time by parallel processing, some researchers are trying to develop neural network (NN) based information processing systems (Hopfield 1984; Hopfield and Tank 1985; Kohonen 1989; Rumelhart et al. 1986). Here the basic aim is to emulate the human neural information processing system, thereby making the system (artificially) intelligent. This NN based processing is suitable even when information is ill-defined and/or defective/partial. The approach is highly robust and noise insensitive.

The use of statistical techniques for modelling and processing image data is very common in computer vision. A specific example is the use of a Markov Random Field (MRF) to model real life images. Cohen and Cooper (1983); Derin and Elliot (1987); and Geman and Geman (1984); used Gibbs Distribution (GD) for characterizing an MRF, since a Gibbs Random Field (GRF) can be viewed as an MRF for a large neighbourhood system (Besag 1974; Kinderman and Snell 1980; Splitzer 1971). Most of the above mentioned authors use maximization of a'posteriori probability (MAP) criterion for segmentation/restoration of noisy images modelled as a GRF. But the computation of MAP estimate of the scene in this type of problem is very hard. Use of GD for segmentation and restoration of a noisy image requires $M^{N_1 \times N_2}$ possible combinations of its structure to be searched to get the best possible solution (M = number of discrete levels that the random variable can have, $N_1 \times N_2$ are the dimensions of the image). Thus the problem has exponential complexity. Normally, in such cases, heuristic solutions (Derin and Elliott 1987) are suggested. Since they cannot be implemented in parallel, the output still cannot be obtained in real time. However, by using parallel processing and exploiting the collective computational abilities of the NNs, the search space can be reduced drastically even after getting a substantially good solution.

The present work is an attempt to solve the computationally hard problem of finding the MAP estimate of a scene with a Neural Network. An NN architecture which is a modified version of that of Hopfield's continuous model (Hopfield 1984) is used in the proposed work. A single neuron is assigned for every pixel. Every neuron is assumed to be connected only to its neighbours, which may vary depending on the problem. The neurons have negative self feed back. The input bias of a neuron is a transformed version of the gray level of the corresponding pixel. Simple hardware able to realize a neuron for this problem, is suggested.

The simulation study was done using a synthetic image corrupted by noise, as well as a real (noisy) image. The synthetic image was corrupted by adding noise with $N(0, \sigma^2)$. The results obtained were satisfactory even when the SNR was 0.75, where SNR is defined as

$$\frac{\text{range of gray levels}}{\sigma}.$$

This verifies the robustness and the noise immunity of the proposed technique. It was also found that the level of precision used for testing the convergence affected the result to some extent, but not severely.

2 Background on Gibbs distribution

Here we present the basic definition of a particular class of Gibbs Distribution (GD) (Derin and Elliott 1987) which are mainly used in the modelling of images. For an extensive treatment on GD, related to the MRF and the equivalence of the two see (Besag 1974; Kinderman and Snell 1980; Splitzer 1971).

2.1 Basic definition of GD

We focus our attention on 2-D random fields defined over a finite $N_1 \times N_2$ rectangular lattice of points (pixels) characterized by

$$L = \{(i,j) : 1 \leqslant i \leqslant N_1, 1 \leqslant j \leqslant N_2\}. \quad (1)$$

Let us first define a neighbourhood system on the lattice L and the associated cliques.

Definition 1. The d^{th} order neighbourhood system (N^d) on L is described as

$$N^d = \{N_{ij}^d : (i,j) \in L, N_{ij}^d \subseteq L\} \quad (2)$$

such that

a) $(i,j) \notin N_{ij}^d$, and
b) if $(k,l) \in N_{ij}^d$, then $(i,j) \in N_{kl}^d$ for any $(i,j) \in L$.

Different ordered neighbourhood systems can be defined considering different sets of neighbouring pixels of (i,j). $N^1 = \{N_{ij}^1\}$ can be obtained by taking the four nearest neighbour pixels. Similarly, $N^2 = \{N_{ij}^2\}$ consists of the eight pixels neighbouring (i,j) and so on (as shown in Fig. 1a). Due to the finite size of the lattice (the size of the image being fixed), the neighbourhood of the pixels on the boundaries are necessarily smaller unless a periodic lattice structure is assumed.

The 'cliques' associated with the d^{th} order neighbourhood system N^d are defined as follows.

Definition 2. A clique for the d^{th} order neighbourhood system N^d defined on the lattice L, denoted by c, is a subset of L such that

a) c consists of a single pixel, or
b) for $(i,j) \neq (k,l)$, $(i,j) \in c$ and $(k,l) \in c$ implies that $(i,j) \in N_{kl}^d$.

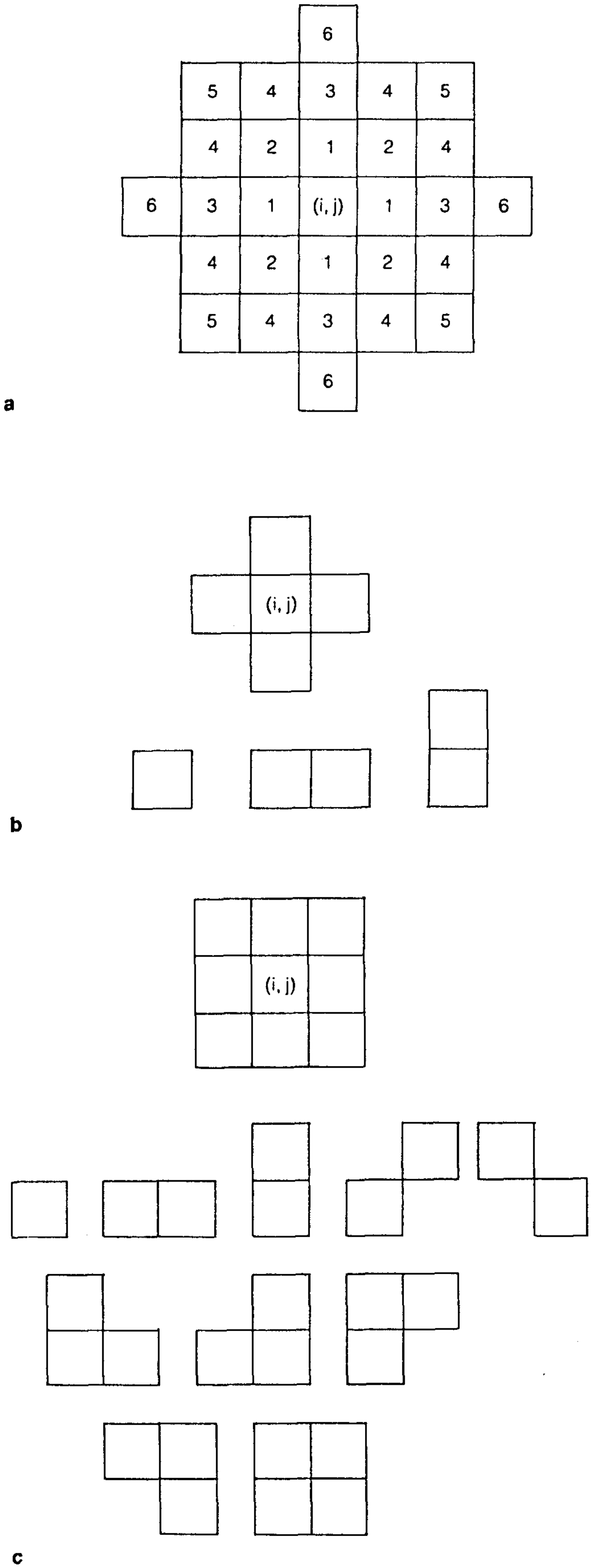

Fig. 1a–c. Neighbourhood system and clique types for a GRF. **a** d^{th} order neighbourhood system N^d for the pixel (i,j). **b** Neighbourhood system N^1 and the associated clique types. **c** Neighbourhood system N^2 and the associated clique types

For illustration, first and second order neighbourhood systems (N^1 and N^2) and the associated cliques are shown in Fig. 1b and c.

Now a Gibbs Distribution (GD) can be defined as follows.

Definition 3. Let N^d be the d^{th} order neighbourhood system defined on a finite lattice L. A random field $X = \{X_{ij}\}$ defined on L has a GD or equivalently is a Gibbs Random Field (GRF) with respect to N^d if, and only if, its joint distribution is of the form

$$p(X = x) = \frac{1}{Z} \mathrm{e}^{-U(x)} \tag{3}$$

with

$$U(x) = \sum_{c \in C} V_c(x) , \tag{4}$$

where $V_c(x)$ is the potential associated with the clique c, C is the collection of all possible cliques and

$$Z = \sum_x \mathrm{e}^{-U(x)} . \tag{5}$$

Here Z, the partition function, is a normalizing constant, and $U(x)$ is called the energy of the realization. The clique potential $V_c(x)$ is assumed to depend on the gray values of the pixels in c and on the clique types and nothing else.

The joint distribution expression in (3) says that with a decrease in $U(x)$ the probability $p(X = x)$ increases. In other words, the smaller $U(x)$, the energy of the realization x, the more likely is the realization [i.e., larger $p(X = x)$] and vice versa.

In this context it may be mentioned that any random field can be viewed as a Markov Random Field (MRF) when

$$N^d_{ij} = L \qquad \text{for all } (i,j) \in L$$

and every MRF with

$$p(X = x) > 0 \qquad \text{for all } x$$

is a GRF and vice versa (Besag 1974). Hence a scene realization can be taken as a GRF. Here, it is assumed that the random field X consists of M-valued discrete random variables $\{X_{ij}\}$ taking values in $Q = \{q_1, q_2, \ldots, q_m\}$. To define GD it suffices to define the neighbourhood system N^d and the associated clique potentials $V_c(x)$. It has been further found that the 2^{nd} order neighbourhood system is sufficient for modelling the spatial dependencies of a scene consisting of several objects. Hence we shall concentrate on the cliques associated with N^2. Extension to higher order neighbourhood systems and restriction to lower ones are self-evident.

3 Image model and problem formulation

In this section the formulation of the object extraction problem which finds maximum a'posteriori probability (MAP) estimate of a scene is discussed.

3.1 Gibbsian model for image data

A digital image $y = \{y_{ij}\}$ can be described as an $N_1 \times N_2$ matrix of observations. It is assumed that the matrix y is a realization from a random field $Y = \{Y_{ij}\}$. The lattice associated with this is the collection of $N_1 \times N_2$ pixels: $\{(i,j)\}$. The random field Y is defined in terms of the underlying random field $X = \{X_{ij}\}$ (scene). The scene random field X is a discrete valued one, where X_{ij} can take values in $Q = \{q_1, q_2, \ldots, q_m\}$ for each $(i,j) \in L$. A realization $X = x$ is a partitioning of the lattice into M region types such that $x_{ij} = q_k$, if the pixel (i,j) belongs to the k^{th} region type. Each region type can occur in more than one location in the lattice.

The Gibbsian Distribution basically characterizes spatial clustering of pixels into regions. The distribution emphasizes spatial continuity, i.e., if a pixel belongs to region type k, then the probability of its neighbouring pixels to belong to the same region type is very high. The novelty of the GRF lies in the fact that it takes into account the statistical information pertaining to size, shape, orientation and frequency of the regions.

Regions are assumed to have uniform intensities and the realized image is a corrupted version of the actual one, corrupted by white (additive i.i.d., i.e., independent and identically distributed) noise. So Y can be written as

$$Y_{ij} = F(X_{ij}) + W_{ij} \qquad (i,j) \in L \tag{6}$$

where $\{W_{ij}\}$ is i.i.d. noise. It is also assumed that $W_{ij} \sim N(0, \sigma^2)$. The function $F(\cdot)$ is a simple mapping of the region type to the corresponding gray level. In other words,

$$F(X_{ij}) = q_m \qquad \text{if } X_{ij} = m . \tag{7}$$

3.2 The object extraction algorithm

Here, the object extraction problem is simply the determination of the scene realization x that has given rise to the actual noisy image y. The realization x can not be obtained deterministically from y. So the problem is to estimate $\hat{x}$ of the scene X, based on the realization y. The statistical criterion of maximum a'posteriori probability (MAP) can be used to estimate the scene. The objective in this case is to have an estimation rule which yields $\hat{x}$ that maximizes the a'posteriori probability $p(X = x \mid Y = y)$.

From Bayes rule we can write

$$p(X = x \mid Y = y) = \frac{p(Y = y \mid X = x) p(X = x)}{p(Y = y)} . \tag{8}$$

Since y does not affect the maximization process, it is equivalent to maximize only the numerator of the RHS of (8), or its logarithm

$$\ln p(Y = y \mid X = x) + \ln p(X = x) . \tag{9}$$

Let the scene random field X be a second order (N^2) GD. Then the second component of (9) can be written as (from Eq. (3))

$$\ln p(X=x) = C_1 - \sum_{c \in C} V_c(x) \tag{10}$$

where $C_1 = -\ln Z = \text{constant}$.

The first term of (9) can be shown to be

$$\ln p(Y=y|X=x) = C_2 - \sum_{m=1}^{M} \sum_{(i,j) \in S_m} \frac{1}{2\sigma^2}(y_{ij} - q_m)^2 \tag{11}$$

where

$$C_2 = -\frac{N_1 \times N_2}{2} \ln(2\pi\sigma^2) = \text{constant}$$

and

$$S_m = \{(i,j) \in L : x_{ij} = q_m\} .$$

The difficulty in determining $\hat{x}$ is due to the fact that the maximization of equation (8/9) requires $M^{N_1 \times N_2}$ possible scene configurations to be searched, making the problem computationally hard. In the following sections, we shall exploit the collective computational abilities of a Neural Network (NN) to get the MAP estimate of a scene.

4 Description of neural networks

4.1 General description of neural networks

Neural networks are designated by the network topology, connection strength between pairs of neurons (weights), node characteristics and the status updating rules. Node characteristics mainly specify the primitive types of operations they can perform, like summing the weighted inputs coming to them and then amplify them. The updating rules may be for the weights and/or states of the processing elements (neurons). Normally an objective function is defined which represents the complete status of the network and its set of minima gives the stable states of the network. The neurons operate asynchronously (the status of any neuron can be updated at random times independent of the other) and in parallel thereby providing output in real time. Since there are interactions among all the neurons the collective property inherently reduces the computational task. Our study will be concerned with networks similar to that of Hopfield's continuous model (Hopfield 1984), for which a brief description is given below.

The model is based on continuous variables and responses. Let the output variable V_i for i^{th} neuron lie in the range $[-1, +1]$, and be a continuous and monotonically increasing (non linear) function of its instantaneous input U_i. The typical input/output relation

$$V_i = g(U_i) \tag{12}$$

is sigmoidal with asymptotes -1 and $+1$. It is also necessary that the inverse of g exists i.e., $g^{-1}(V)$ is defined.

A typical choice of the function g is

$$g(x) = 2 \cdot \frac{1}{1 + e^{-(x-\theta)/\theta_0}} - 1 . \tag{13}$$

Here, the parameter θ controls the shifting of the function g along the x axis and θ_0 determines the steepness (sharpness) of the function. Positive values of θ shift $g(x)$ towards positive x and vice versa. Lower values of θ_0 make the function steeper while higher values make it broader. The value of $g(x)$ lies in $[-1, 1]$ with 0.0 at $x = \theta$.

For the realization of the gain function g we use a non linear operational amplifier with negligible response time. A possible electrical circuit for the realization of a neuron is shown in Fig. 2a. In the diagram $V_1, V_2, V_3, \ldots, V_n$ represent the output voltage (status) of the amplifiers (neurons) to which the i^{th} amplifier is connected. $R_{ij}(=1/W_{ij})$ is the connecting resistance between the j^{th} and the i^{th} amplifiers. I_i is the input bias current for the i^{th} amplifier. R is the input resistance of the amplifier and C its input capacitance.

Suppose U_i is the total input voltage (potential at the point A, Fig. 2a) to the amplifier having a gain function g. Then applying Kirchoff's current law at the point A we get,

$$\sum_{j=1}^{n} \frac{V_j - U_i}{R_{ij}} + I_i = \frac{U_i}{R} + C\frac{dU_i}{dt}$$

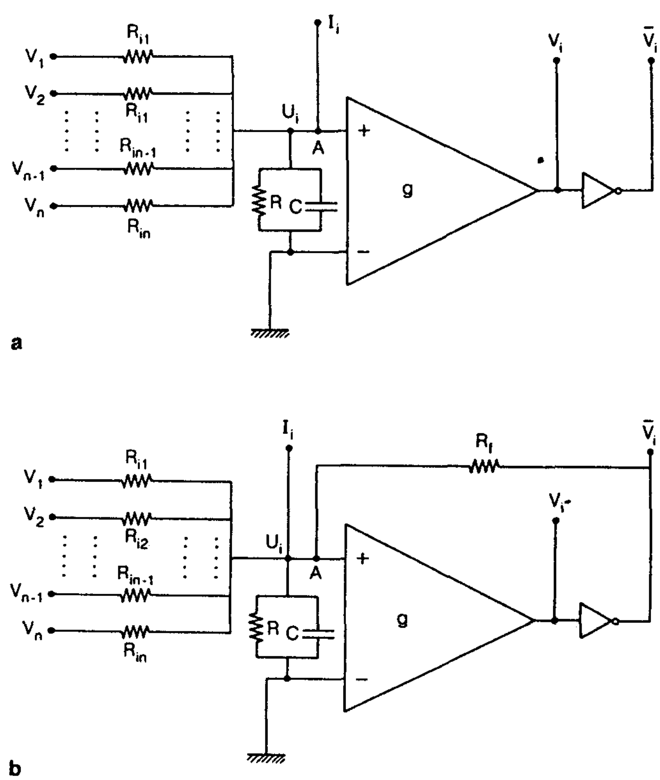

Fig. 2a,b. Hardware realization of neuron. **a** Electrical circuit equivalent of a neuron (Hopfield). **b** Electrical circuit equivalent of a neuron presently used or

$$C\frac{\mathrm{d}U_i}{\mathrm{d}t}=\sum_{j=1}^{n}\frac{V_j}{R_{ij}}-\frac{U_i}{R_i}+I_i \qquad \left[\frac{1}{R_i}=\sum_{j=1}^{n}\frac{1}{R_{ij}}+\frac{1}{R}\right]$$

$$=\sum_{j=1}^{n}W_{ij}\cdot V_j-\frac{U_i}{R_i}+I_i \tag{14}$$

Here R_i is the total input resistance of the amplifier, the output impedance of the amplifier being considered negligible. Equation (14) will be used later to find different equations governing the dynamics of the proposed network.

Hopfield (1984) showed that

$$E=-\sum_i\sum_j W_{ij}V_iV_j-\sum_i V_iI_i+\sum_i\frac{1}{R_i}\int_0^{V_i}g^{-1}(V)\,\mathrm{d}V \tag{15}$$

is a Liapunov function for the above system and $\mathrm{d}E/\mathrm{d}t\leqslant 0$. He also showed that since E is bounded, the time evolution of the system is a motion in the state space that seeks out and stops at minima in E.

4.2 Description of the network used in the present case

The topology of the neural network used in the present case is such that each neuron is connected to all of its N^2 neighbours. The network can therefore be thought of as a modified version of Hopfield's network (Hopfield 1984) with the connection strength to all neurons outside N^2 being zero.

From equation (13) we notice that

$$\operatorname*{Lt}_{x\to-\alpha} g(x)=-1$$

and

$$\operatorname*{Lt}_{x\to+\alpha} g(x)=+1$$

In other words, as $x\to\pm\alpha$, the function asymptotically approaches the limiting values of ± 1. In the present case, since the number of neighbours are fixed (8), for normalized values of connection strengths the input value to a neuron lies in $[-8, 8]$. A polynomial function defined over a finite domain may also be used as an input/output transfer function. One such choice might be,

$$\begin{aligned} g(x)&=-1 && x\leqslant a\\ &=2^n\left(\frac{x-a}{c-a}\right)^n-1 && a\leqslant x\leqslant b\\ &=1-2^n\left(\frac{c-x}{c-a}\right)^n && b\leqslant x\leqslant c\\ &=1 && x\geqslant c \end{aligned} \tag{16}$$

for $n\geqslant 2$ and $b=(a+c)/2$. In this case $g(x)$ lies in $[-1, 1]$ with $g(x)=0.0$ at $x=b$. The domain of x is $[a, c]$. The value of n controls the sharpness (steepness) of the function. The above function is nothing but a generalized version of the standard 'S' function as defined by Zadeh (Zadeh 1965).

For the present case, the domain of x is $[-8, 8]$ i.e., $a=-8$, $c=8$. The function g then takes the form

$$\begin{aligned} g(x)&=-1 && x\leqslant -8\\ &=\frac{1}{2^{3n}}(x+8)^n-1 && -8\leqslant x\leqslant 0\\ &=1-\frac{1}{2^{3n}}(8-x)^n && 0\leqslant x\leqslant 8\\ &=1 && x\geqslant 8\,. \end{aligned} \tag{17}$$

However, for quick convergence one can use the domain $[-1, 1]$, i.e., one may use the function

$$\begin{aligned} g(x)&=-1 && x\leqslant -1\\ &=(x+1)^n-1 && -1\leqslant x\leqslant 0\\ &=1-(1-x)^n && 0\leqslant x\leqslant 1\\ &=1 && x\geqslant 1\,. \end{aligned} \tag{18}$$

5 Algorithm for solving the MAP estimation problem with NN

5.1 Mathematical formulation of the MAP estimation problem

From (9–11), the problem is to maximize

$$\ln p(X=x)+\ln p(Y=y|X=x) \tag{19}$$

where

$$\ln p(X=x)=-\sum_{c\in C}V_c(x)+C_1 \tag{20}$$

and

$$\ln p(Y=y|X=x)=-\sum_{m=1}^{M}\sum_{(i,j)\in S_m}\frac{1}{2\sigma^2}(y_{ij}-q_m)^2+C_2 \tag{21}$$

where $S_m=\{(i,j)\in L: x_{ij}=m\}$, and C_1 and C_2 are constants.

In this study we shall consider only binary images corrupted by noise. In other words, the proposed method confines itself to the case when M = 2. Under this situation, without loss of generality we can assume that the gray values of the realized image lie in $[-1, 1]$. It has already been discussed in Sect. II that even for the 2nd order neighbourhood system (N^2) different types of cliques are possible. For the sake of simplicity we shall restrict ourselves only to cliques with two pixels. Further, we will not distinguish between the different possible types of cliques with two pixels. It has already been mentioned that if a pixel belongs to region type k, the probability of its neighbouring pixels being in the same region type is very high. This suggests that if a pair of adjacent pixels have similar values then the potential of the corresponding clique should increase the value of $p(X=x)$, i.e., the potential of the clique associated with these two pixels should be negative. If the values V_i and V_j, of two adjacent pixels are of the same sign, then $V_c(x)$ should be negative, otherwise positive.

One possible choice of the clique potential $V_c(x)$, of a clique c of the realization x, containing i^{th} and j^{th}

pixels is

$$V_c(x) = -W_{ij} V_i V_j \tag{22}$$

where W_{ij} is constant for particular i and j, and $W_{ij} > 0$. This W_{ij} can be viewed as the connection strength between the i^{th} and j^{th} neurons and V_i as the output of the i^{th} neuron.

In (11) y_{ij} is the realization of the modelled value x_{ij} of the $(i,j)^{\text{th}}$ pixel and q_m is the value of the $(i,j)^{\text{th}}$ pixel in the underlying scene, i.e., $x_{ij} = q_m$. For an NN one can easily interpret x_{ij} as the present status of the $(i,j)^{\text{th}}$ neuron and y_{ij} as the initial bias of the same neuron. Thus we can write

$$\sum_m \sum_{i,j} (y_{ij} - q_m)^2 = C_3 - 2 \sum_m \sum_{i,j} I_{ij} V_{ij} + \sum_m \sum_{i,j} V_{ij}^2 \tag{23}$$

using $y_{ij} = I_{ij}$ and $q_m = V_{ij}$. Here $C_3 = \Sigma_m \Sigma_{i,j} (y_{ij})^2 = \text{constant}$.

This can also be written in the form (replacing the two dimensional indices by one dimensional index)

$$C_3 - 2 \sum_i I_i V_i + \sum_i V_i^2 . \tag{24}$$

From (20–24) it can be said that maximization of (19) is identical to **maximization** of

$$C_1 + \sum_{i,j} W_{ij} V_i V_j - \frac{1}{2\sigma^2} \left(C_3 - 2 \sum_i I_i V_i + \sum V_i^2 \right) + C_2$$

i.e., **minimization** of (excluding the constant terms and multiplying by -1)

$$-\sum_{i,j} W_{ij} V_i V_j - \frac{1}{\sigma^2} \sum_i I_i V_i + \frac{1}{2\sigma^2} \sum V_i^2 \tag{25}$$

5.2 Realization with neural network

The problem is now reduced to the minimization of the expression in (25) with an NN. To do that, it suffices to establish an equivalence between (25) and the energy of an NN, and to provide an updating rule so that the convergence is guaranted. The state of the network changes with time t, because of the interactions of the neurons. The dynamics of the network are governed by differential equations which are obtained from

$$-\frac{\partial E}{\partial Vi} = \frac{\mathrm{d}U_i}{\mathrm{d}t} . \tag{26}$$

The gain function g can be realized by a non-linear operational amplifier with negligible response time. Comparing (15) and (25), one can see that a neural network whose neurons are as shown in Fig. 2a can not be used for the minimization of (25) due to the presence of an extra term $1/2\sigma^2 \Sigma V_i^2$. In order to realize the expression in (25) with a neural network let us consider the electrical circuit of Fig. 2b, which is similar to Fig. 2a except for the feed back connection. The feed back can account for the extra term in (25).

Let U_i be the total input to the amplifier (potential at point A, Fig. 2b) with gain function g. Then applying Kirchoff's current law at point A we get,

$$\sum_{j=1}^{n} \frac{V_j - U_i}{R_{ij}} + I_i + \frac{-V_i - U_i}{R_f} = \frac{U_i}{R} + C \frac{\mathrm{d}U_i}{\mathrm{d}t}$$

or

$$C \frac{\mathrm{d}U_i}{\mathrm{d}t} = \sum_{j=1}^{n} W_{ij} V_j - \frac{U_i}{R_i} - \frac{V_i}{R_f} + I_i \tag{27}$$

where

$$W_{ij} = 1/R_{ij}$$

and

$$\frac{1}{R_i} = \sum_{j=1}^{n} \frac{1}{R_{ij}} + \frac{1}{R} + \frac{1}{R_f} .$$

Now considering the quantity

$$E = -\sum_i \sum_j W_{ij} V_i V_j - \sum_i V_i I_i + \frac{1}{2R_f} \sum V_i^2 + \sum \frac{1}{R_i} \int_0^{V_i} g^{-1}(V)\,\mathrm{d}V \tag{28}$$

Now,

$$\frac{\mathrm{d}E}{\mathrm{d}t} = \sum_i \frac{\partial E}{\partial V_i} \cdot \frac{\mathrm{d}V_i}{\mathrm{d}t}$$

$$= \sum_i \left(-\sum_j W_{ij} \cdot V_j - I_i + \frac{1}{R_f} V_i + \frac{U_i}{R_i} \right) \cdot \frac{\mathrm{d}V_i}{\mathrm{d}t}$$

$$= \sum_i \left(-C \cdot \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \cdot \frac{\mathrm{d}V_i}{\mathrm{d}t} \qquad \text{[from 27]}$$

$$= -\sum_i \left\{ C \cdot (g^{-1})' \cdot \left(\frac{\mathrm{d}V_i}{\mathrm{d}t} \right)^2 \right\}$$

Since g is a monotonic, increasing function and C is a positive constant, every term in $\{\cdot\}$ of the above expression is non negative, and hence

$$\frac{\mathrm{d}E}{\mathrm{d}t} \leqslant 0 , \tag{29}$$

and

$$\frac{\mathrm{d}E}{\mathrm{d}t} = 0 \Rightarrow \frac{\mathrm{d}V_i}{\mathrm{d}t} = 0 \qquad \text{for all } i .$$

E is easily seen to be bounded. As $\mathrm{d}E/\mathrm{d}t \leqslant 0$, searching in the gradient direction will lead us to a minimum of E. So the evolution of the system is a motion in the state space that seeks and stops at minima in E. So, E can be considered as an energy function of the network with neurons electrically equivalent to Fig. 2b. The last term in (28) is the energy loss of the system. In the high gain limit (Hopfield 1984), the stable states of the network correspond to the local minima of the quantity

$$E = -\sum_i \sum_j W_{ij} V_i V_j - \sum_i V_i I_i + \frac{1}{2R_f} \sum V_i^2 . \tag{30}$$

The expressions in (25) and (30) are equivalent with proper adjustment of coefficients. So the minimal values of the expression in (25) can be easily determined with an NN having the energy function in (28) and neuron realization of Fig. 2b.

In order to get the equations governing the dynamics of the network, let us consider the energy function

$$E = -\sum_i \sum_j C W_{ij} V_i V_j - \frac{C}{\sigma^2} \sum_i V_i I_i + \frac{C}{2\sigma^2} \sum_i V_i^2 + \sum_j \frac{1}{R_i} \int_0^{V_i} g^{-1}(V)\, \mathrm{d}V \qquad (31)$$

Here

$$-\frac{\partial E}{\partial V_i} = \sum_j C W_{ij} V_j + \frac{C}{\sigma^2} \cdot I_i - \frac{C}{\sigma^2} V_i - \frac{1}{R_i} U_i \qquad (32)$$

Differentiating (28) and equating with (27) we notice that

$$-\frac{\partial E}{\partial V_i} = C \cdot \frac{\mathrm{d}U_i}{\mathrm{d}t}$$

Hence

$$\frac{\mathrm{d}U_i}{\mathrm{d}t} = \sum_j W_{ij} V_j + \frac{1}{\sigma^2} \cdot I_i - \frac{1}{\sigma^2} V_i - \frac{1}{\tau} U_i \qquad (33)$$

where $\tau = RC$.

Thus differential equation (33) will govern the state of the system.

To find a solution, we select an initial state of the network at random, and let it evolve according to (33). It will eventually reach a steady state (minimum E) and stop. To get further improvement in the solution one can use the Simulated Annealing (Kirkpatric et al. 1983) type of approach. However, from the experimental results it will be seen that, for this type of problem annealing is not required.

6 Computer simulation and results

To check the validity and effectiveness of the proposed technique, a computer simulation was done using synthetic and real images. The synthetic image was an 'ELLIPSE' corrupted by $N \sim (0, \sigma^2)$ noise. The real image was a noise corrupted version of a tank ('NOISY TANK'). The original synthetic image had only two levels which were corrupted by additive noise with different σ values (15, 32). It is to be noted from (33) that in order to get the input to all neurons of a network (of sizes $N_1 \times N_2$) at an instant $(t + \Delta t)$ one has to solve $N_1 \times N_2$ differential equations with given initial values at time t. For this the Euler method was used here, i.e., we iterated (from 33)

$$U_i(t + \Delta t) = U_i(t) + \Delta t\left(\sum_j W_{ij} V_j(t) + \frac{1}{\sigma^2} \cdot I_i - \frac{1}{\sigma^2} V_i(t) - U_i(t)\right) \qquad (34)$$

until convergence. Here τ is set equal to 1. A numerical solution for these differential equations requires a stopping criterion, which was

$$|U_i(t + \Delta t) - U_i(t)| < \varepsilon, \qquad \text{for all } i \qquad (35)$$

where ε is a preassigned small positive quantity. In the present simulation study $\Delta t \approx 10^{-5}$ and $\varepsilon \approx 10^{-6}$.

The network was assumed to attain a stable state if for every neuron i, $|V_i(t) - V_i(t + \Delta t)| < \varepsilon^1$, where ε^1 is a preassigned small positive quantity. If the precision level was increased (value of ε^1 decreased) the network would take more time to converge. Quicker convergence could be obtained by increasing Δt or ε^1.

For network simulation the input bias was taken to be linearly proportional to the gray level of the corresponding pixels mapped in the range $[-1, 1]$. Thus, i being the gray value of a pixel, L the maximum available gray level, the input bias I is taken as

$$I = 2\frac{i}{L} - 1 \,. \qquad (36)$$

The input/output transfer function used was a second order polynomial (18 with $n = 2$). The connection strengths were assigned as

$$\left.\begin{aligned} W_{ij} &= 1 \quad \text{if } j \in N_i^2 \\ &= 0 \quad \text{otherwise} \end{aligned}\right\}.$$

The results obtained by the technique are shown in Figs. 3–4. In Fig. 3, the results are for a synthetic

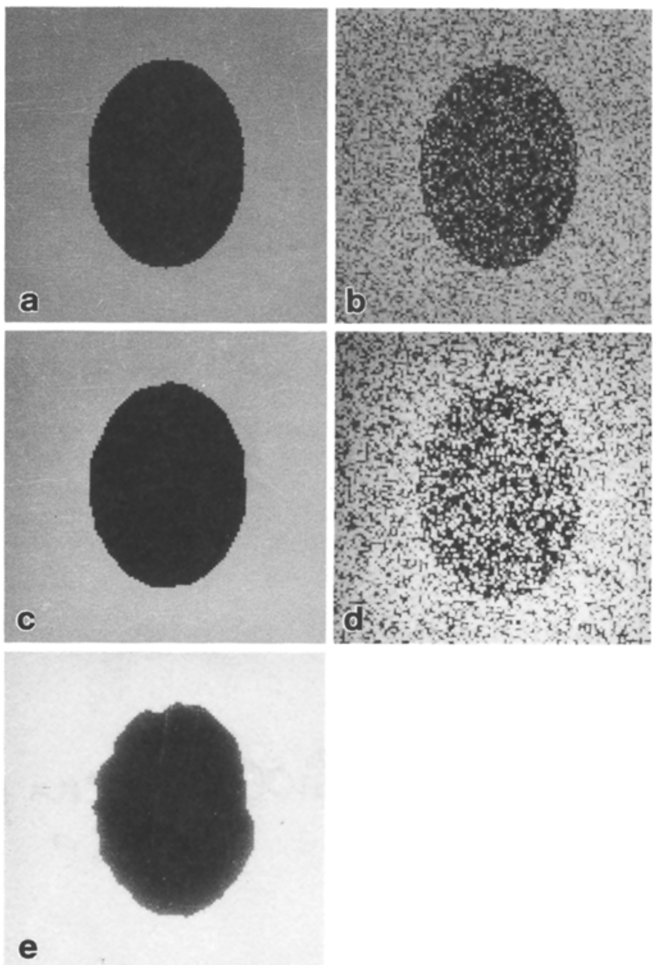

Fig. 3a–e. Simulation results for a synthetic image of Ellipse. **a** Original image. **b** Noisy version with SNR = 1.6. **c** Extracted object when SNR = 1.6. **d** Noisy version with SNR = 0.75. **e** Extracted object when SNR = 0.75

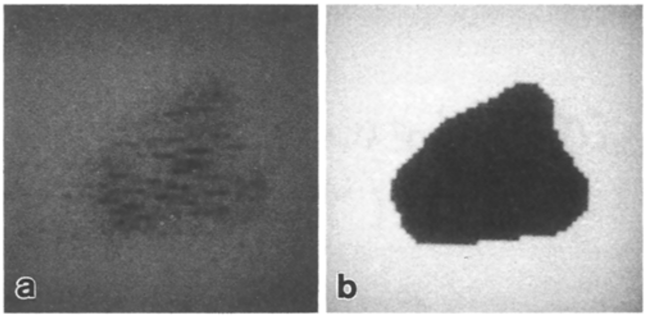

Fig. 4a,b. Simulation results for a real image of noisy tank. **a** Original image of noisy tank. **b** Extracted object of the noisy tank

image corrupted by noise. It is evident that with lower value of σ (high SNR) the object extracted was close to the actual one; whereas with higher value of σ (low SNR) there was a slight deterioration in the result. However, even for low SNR the key features of the original input were preserved. From Fig. 4 it is seen that for a real image ('NOISY TANK') the extracted object was compact and preserved its approximate outline. These results show the robustness and noise immunity of the technique.

7 Discussion and conclusion

The present work demonstrates a method for solving a computationally hard object extraction problem with a modified Hopfield neural network architecture with a single neuron assigned to each pixel. The input/output transfer function used was a polynomial over a fixed domain, instead of an asymptotic function. The energy function of the network was designed so that its minimum value corresponded to the MAP estimate of the scene. The dynamics of the network were described and a possible hardware realization of a neuron was suggested.

The proposed technique was implemented and tested on a synthetic image corrupted by noise and on a real image. The results obtained were satisfactory, even with high degree of noise.

Acknowledgement. We thank the anonymous reviewer for his suggestions. We also thank Prof. D. Dutta Majumder for his interest in this work.

References

Besag J (1974) Spatial interaction and statistical analysis of lattice systems. J R Statist Soc B 36:192–236

Cohen FS, Cooper DB (1983) Real time textured image segmentation based on noncausal Markovian random field models. Proc SPIE Conf Intell Robots, Cambridge, Mass

Derin H, Elliott H (1987) Modeling and segmentation of noisy and textured images using Gibbs random fields. IEEE Trans Pattern Anal Machine Intell PAMI-9:39–55

Geman S, Geman D (1984) Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. IEEE Trans Pattern Anal Machine Intell PAMI-6:721–741

Gonzalez RC, Wintz P (1977) Digital image processing. Addison-Wesley, Reading, Mass

Hopfield JJ (1982) Neural networks and physical systems with emergent computational abilities. Proc Natl Acad Sci USA 79:2554–2558

Hopfield JJ (1984) Neurons with graded response have collective computational properties like those of two state neurons. Proc Natl Acad Sci USA 81:3088–3092

Hopfield JJ, Tank DW (1985) Neural computation of decisions in optimization problems. Biol Cybern 52:141–152

Kinderman R, Snell JL (1980) Markov random fields and their applications. American Mathematical Society, Providence

Kirkpatric S, Gellat CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680

Kohonen T (1989) Self-organization and associative memory, 3rd edn. Springer, Berlin Heidelberg New York

Pal SK, Dutta Majumder D (1986) Fuzzy mathematical approach to pattern recognition. Wiley (Halsted Press), New York

Parsi BK, Parsi BK (1990), On problem solving with Hopfield neural networks. Biol Cybern 62:415–423

Rosenfeld A, Kak AC (1982) Digital picture processing. Academic Press, New York

Rumelhart DE, McClelland J, PDP Research Group (1986) Parallel distributed processing: explorations in the microstructure of cognition, vol 1. MIT Press, Cambridge, Mass

Splitzer F (1971) Markov random fields and Gibbs ensembles. Am Math Mon 78:142–154

Zadeh LA (1965) Fuzzy sets. Inf Control 8:338–353

Dr. Nikhil R. Pal
Indian Statistical Institute
203 Barrackpore Trunk Road
Calcutta 700 035
India