Extended $\alpha\beta$ associative memories

J.H. Sossa Azuela* and R. Barrón Fernández
Centro de Investigación en Computación, Instituto Politécnico Nacional,
Av. Juan de Dios Batíz and M. Othón de Mendizábal, Col. Nueva Industrial Vallejo, México, D.F. 07738. MÉXICO,
Tel. 5729 6000 ext, 56512. Fax 5729 6000 ext. 56607,
e-mail: hsossa@cic.ipn.mx, rbarron@cic.ipn.mx

Recibido el 11 de septiembre de 2006; aceptado el 22 de noviembre de 2006

The $\alpha\beta$ associative memories recently developed in Ref. 10 have proven to be powerful tools for memorizing and recalling patterns when they appear distorted by noise. However they are only useful in the binary case. In this paper we show that it is possible to extend these memories now to the gray-level case. To get the desired extension, we take the original operators α and β, foundation of the $\alpha\beta$ memories, and propose a more general family of operators. We find that the original operators α and β are a subset of these extended operators. For this we first formulate a set of functional equations in terms of the original properties of operators α and β. Next we solve this system of equations and find a family of solutions. We show that the α and β originally proposed in Ref. 10 are just a particular case of this new family. We present the properties of the new operators. We then use these operators to build a new set of extended memories. We also give the conditions under which the extended memories are able to recall a pattern either from the pattern's fundamental set or from altered versions of them. We give real examples with images where the proposed memories show their efficiency. We compare the proposal with other similar works, and show the ours performs much better.

Keywords: Computer science and technology; neural engineering; image quality; contrast; resolution; noise.

Las memorias $\alpha\beta$ recientemente desarrolladas en Ref. 10 han mostrado ser herramientas poderosas para memorizar y recobrar patrones cuando estos aparecen distorsionados por ruido. Sin embargo, son sólo útiles en el caso binario. En este artículo mostramos que es posible extender estas memorias para trabajar en el caso de niveles de gris. Para obtener esta extensión tomamos los operadores originales α yβ, fundamento de las memorias $\alpha\beta$ y proponemos una familia más general, de la cual forman parte los originales. Para esto formulamos un conjunto de ecuaciones funcionales en términos de las propiedades de los operadores originales. Enseguida, resolvemos este sistema y encontramos una familia de soluciones. Mostramos que los operadores originales α y β son un caso particular de esta nueva familia. Damos algunas de las propiedades de los nuevos operadores. Usamos entonces los nuevos operadores para construir un conjunto de memorias extendidas. Damos las condiciones bajo las cuales estas memorias son capaces de recuperar patrones del conjunto fundamental o a partir de versiones ruidosas de las mismas. Damos también ejemplos reales con imágenes donde las memorias propuestas muestran su eficiencia. Comparamos la propuesta con otras propuestas similares y demostramos que la nuestra tiene un mucho mejor desempeño.

Descriptores: Ciencias de la computación y tecnología; Ingeniería neuronal; calidad de imagen; contraste; resolución; ruido.

PACS: 89.20.Ff; 87.80.Xs; 87.57.Ce

1. Introduction

Associative memories [3] together with neural networks [4], fuzzy models and genetic algorithms today form a group of techniques and methodologies of Artificial Intelligence, known as soft computing techniques. The importance of these techniques is their capacity to solve particular problems in a more efficient way than classical techniques. In the solution of a given problem, soft computing techniques should collaborate to produce better solutions, giving as a result a hybrid-computing scheme.

The main feature of an associative memory (AM) consists in recalling a pattern from a distorted version of it or from another one related to the goal pattern. This property can be useful, for example, in pattern recognition problems or information retrieval [5,6].

From a technical point of view, an AM can be seen as a dynamic system [1] where the patterns can be considered as states (auto-associative case) [7] or outputs dependent on the states (hetero-associative case). Seen as a discrete dynamic system, the problem consists in building the dynamic system by taking into account a set of reference patterns (training or learning phase) and given a reference pattern distorted by noise find the original pattern (recalling phase). This is equivalent to a transition of the dynamic system from an initial state until arriving to the nearest stable state; if this perturbation is not too large, this transition should correspond to the desired pattern.

In greater detail, an associative memory **M** is a system that relates input vectors and outputs vectors as follows:

$$\mathrm{x} \xrightarrow{\mathrm{M}} \mathrm{y}$$

with **x** and **y**, respectively the input and output pattern vectors. Each input vector forms an association with a corresponding output vector. An association between input pattern **x** and output pattern **y** is denoted by (x,y). For a positive k integer, the corresponding association will be denoted by $\left(\mathrm{x}^k,\mathrm{y}^k\right)$ [8,9] .

An associative memory, **M**, is represented by a matrix whose *ij*-th component is m_{ij}. **M** is generated from a finite *a priori* set of known associations, known as the *fundamental set of associations,* or simply the *fundamental set*

(FS). If ξ is an index, the fundamental set is represented as: $\left\{\left(\mathrm{x}^{\xi},\mathrm{y}^{\xi}\right)|\xi=1,2,\ldots,p\right\}$ with p the cardinality of the set. The patterns that form the fundamental set are called *fundamental patterns*. If it holds that $\mathrm{x}^{\xi}=\mathrm{y}^{\xi}\ \forall\ \mu\in\{1,2,\ldots p\}$, **M** is auto-associative, otherwise it is hetero-associative. A distorted version of a key pattern **x** to be used to recall a pattern will be denoted by $\tilde{x}$. If when feeding a distorted version of x^{w} with $w\in\{1,2,\ldots,p\}$ to an associative memory **M** it happens that the output corresponds exactly to the associated pattern y^{w}, we say that recall is perfect.

Many ways to build an associative memory have been reported in the literature. For several examples, refer to Refs. 2 to 19. Recently in Ref. 10, the authors describe a new class of associative memories, the so-called $\alpha\beta$memories. Their functioning is based on two binary operators: α and β. In the properties of these two important operators lies the power of the above-mentioned memories. These memories work very well in the presence of additive or subtractive noise; its domain is however restricted to the binary one. In this work we describe how to extend this class of associative memories to the case of gray-level patterns. A first attempt in this direction was reported in Refs. 13 and 16. The results obtained are however far from desired.

It is worth mentioning that many of the results in the binary case are based on properties that, though obtained by taking into account a particular domain, do not strictly depend on that domain. Once generalized, many of these results in the binary case can be naturally extended to more interesting ranges such as the gray-level case.

The central point we need to solve in this research is how to generalize the operators α and β to get the desired new extended associative memories. This can be done in two ways:

1. Either by finding two more operators, say α' and β' functionally different, with similar properties, probably not identical but with the same goal, or
2. To change the domain of α and β for a more interesting domain from (a more realistic one) and see which are the changes that must done to the original operators so that they will maintain their properties even in a different domain.

In both cases, to tackle the problem, it helps at the beginning to have an analytical expression for the original α and β operators. This can be obtained, as we shall later see, by setting up a system of *functional equations* by using some of the central properties of the original operators α and β. Once expounded the systems of functional equations, we look for a family of solutions factorizing additively the binary operators, for example:

$$\alpha(x,y)=f(x)+g(y)$$
$$\beta(x,y)=p(x)+q(y) \qquad (1)$$

This simplifies the problem. It allows us to explore a wide variety of operators, among these, as we shall later see: α and β. Something important in this perspective is that given operator α, operator β is totally determined. In this work we shall try option 2). In future works we shall explore option 1).

2. Foundations of $\alpha\beta$ Memories

In this section we present the development of the proposed extended memories. Firstly, we give a survey of $\alpha\beta$ memories. We then explain the details about the development of the proposed memories.

2.1. Survey of $\alpha\beta$ Memories

$\alpha\beta$ memories are based on the operation of two operators α and β , defined as:

$$\alpha: A\times A\rightarrow B \qquad (2)$$
$$\beta: B\times A\rightarrow A, \qquad (3)$$

where $A=\{0,1\}$ and $B=\{0,1,2\}$. In tabular form, $\alpha: A\times A\rightarrow B$ and $\beta: B\times A\rightarrow A$ are defined as shown in Tables I and II.

Both operations were found by extensive research by taking as their foundation the **max** and **min** operations of the morphological associative memories.

2.1.1. *Matrix operations* $\vee_\alpha$, $\wedge_\alpha$, $\vee_\beta$, $\wedge_\beta$

Let $P=[p_{ij}]_{m\times r}$ and $Q=[q_{ij}]_{r\times n}$ be two matrices. The following matrix operations are defined in [10]:

- α**max** operation: $P_{m\times r}\vee_\alpha Q_{r\times n}=\left[f_{ij}^{\alpha}\right]_{m\times n}$ where $f_{ij}^{\alpha}=\bigvee_{k=1}^{r}\alpha(p_{ik},q_{kj})$,

TABLE I. Values of $\alpha(x,y)$.

x	y	α(x,y)
0	0	1
0	1	0
1	0	2
1	1	1

TABLE II. Values of $\beta(x,y)$.

x	y	α(x,y)
0	0	1
0	1	0
1	0	0
1	1	1
2	0	1
2	1	1

- β**max** operation: $P_{m\times r} \vee_\beta Q_{r\times n} = \left[f_{ij}^{\beta}\right]_{m\times n}$ where $f_{ij}^{\beta} = \overset{r}{\underset{k=1}{\vee}} \beta\left(p_{ik}, q_{kj}\right)$,
- α**min** operation: $P_{m\times r} \wedge_\alpha Q_{r\times n} = \left[f_{ij}^{\alpha}\right]_{m\times n}$ where $f_{ij}^{\alpha} = \overset{r}{\underset{k=1}{\vee}} \alpha\left(p_{ik}, q_{kj}\right)$,
- β**min** operation: $P_{m\times r} \wedge_\beta Q_{r\times n} = \left[h_{ij}^{\beta}\right]_{m\times n}$ where $h_{ij}^{\beta} = \overset{r}{\underset{k=1}{\wedge}} \beta\left(p_{ik}, q_{kj}\right)$,

where $\vee$ and $\wedge$ denote the **max** and **min** operators, respectively. These four matrix operations are similar to the morphological matrix operations described in Ref. 8. When applied between vectors we have that:

- If $x \in A^n$ and $y \in A^m$, then $y \vee_\alpha x^t$ is an m×n matrix, and it also holds that

$$\mathrm{y} \ \vee_\alpha \mathrm{x}^t = \mathrm{y} \wedge_\alpha \mathrm{x}^t = \begin{pmatrix} \alpha(y_1,x_1) & \alpha(y_1,x_2) & \cdots & \alpha(y_1,x_n) \\ \alpha(y_2,x_1) & \alpha(y_2,x_2) & \cdots & \alpha(y_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \alpha(y_m,x_1) & \alpha(y_m,x_2) & \cdots & \alpha(y_m,x_n) \end{pmatrix}_{m\times n}.$$

  Symbol $\otimes$ is used to represent both operations, when operating on column vectors: $y \vee_\alpha x^t = y \otimes x^t = y \wedge_\alpha x^t$.

- If $x \in A^n$ and P is an m×n matrix, operations $P_{m\times r} \vee_\beta$ x and $P_{m\times r} \wedge_\beta x$ give as a result two vectors of dimension m, with an i-th component $\left(P_{m\times r} \vee_\beta x\right)_i = \overset{n}{\underset{j=1}{\vee}} \beta\left(p_{ij}, x_j\right)$ and $\left(P_{m\times r} \wedge_\beta x\right)_i = \overset{n}{\underset{j=1}{\wedge}} \beta\left(p_{ij}, x_j\right)$.

2.1.2. *αβ memories*

Two kinds of associative memories are described in Ref. 10: hetero-associative and auto-associative. Due to limitations of space limitations, we shall talk about auto-associative memories only. If to an hetero-associative memory fulfils the condition that $\mathrm{y}^\xi = \mathrm{x}^\xi \ \forall \ \xi \in \{1, 2, \cdots, p\}$, according to Section 1, the memory becomes an auto-associative one. In this case it is obvious that:

- The fundamental set takes the form $\left\{\left(\mathrm{x}^\xi, \mathrm{x}^\xi\right) | \xi = 1, 2, \ldots, p\right\}$.
- The input and output patterns have the same dimension, for example n.
- The memory is a square matrix.

Two auto-associative $\alpha\beta$ memories, **M** and **W** are fully described in Ref. 10. **M** memories are useful for coping with additive noise; **W** memories, on the contrary, are useful for coping with subtractive noise. Due to space limitations, only **M** memories are described.

Auto-associative $\alpha\beta$ memories type M:

<u>**TRAINING PHASE:**</u>

Step 1: For each $\xi = 1, 2, \cdots, p$, from each couple $\left(\mathbf{x}^\xi, \mathbf{x}^\xi\right)$ build the matrix: $\left[\mathbf{x}^\xi \otimes \left(\mathbf{x}^\xi\right)^t\right]_{n\times n}$.

Step 2: Apply binary **max** operator $\vee$ to the matrices obtained in Step 1 to get matrix **M** as follows:

$$\mathbf{M} = \overset{p}{\underset{\xi=1}{\vee}} \left[\mathbf{x}^\xi \otimes \left(\mathbf{x}^\xi\right)^t\right]. \tag{4}$$

The *ij*-th component **M** is given as follows:

$$m_{ij} = \overset{p}{\underset{\xi=1}{\vee}} \alpha\left(x_i^\xi, x_j^\xi\right). \tag{5}$$

<u>**RECALLING PHASE:**</u> We have two cases:

Case 1: Recovering of a fundamental pattern. A pattern $\mathbf{x}^\omega$, with $\omega \in \{1, 2, \cdots, p\}$, is presented to the auto-associative memory **M** and the following operation is done:

$$\mathbf{M} \wedge_\beta \mathbf{x}^\omega. \tag{6}$$

The result is a column vector of dimension n, with the i-th component given as:

$$\left(\mathbf{M} \wedge_\beta \mathbf{x}^\omega\right)_i = \overset{n}{\underset{j=1}{\wedge}} \beta\left(m_{ij}, x_j^\omega\right). \tag{7}$$

In this case the recalling conditions are always satisfied since **M** always has 1's along its main diagonal.

Case 2: Recovering of a pattern from an altered version of it. A pattern $\tilde{\mathbf{x}}$ (altered version with additive noise of a pattern $\mathbf{x}^\omega$) is presented to the auto-associative memory **M** and the following operation is carried out:

$$\mathbf{M} \wedge_\beta \tilde{\mathbf{x}}. \tag{8}$$

Again, the result is a column vector of dimension n, with the i-th component given as:

$$\left(\mathbf{M} \wedge_\beta \tilde{\mathbf{x}}\right)_i = \overset{n}{\underset{j=1}{\wedge}} \beta\left(m_{ij}, \tilde{x}_j\right). \tag{9}$$

In this case a sufficient condition for obtaining $\mathbf{x}^\omega$ from $\tilde{\mathbf{x}}$ is that for each row of matrix **M** one of its elements by less than or equal to a corresponding element in matrix $\mathbf{x}^\omega \otimes (\tilde{\mathbf{x}})^t$. For more details, refer to Ref. 10.

3. Development of the Extended Memories

In this section the extended memories are developed. Firstly we derive the extended operators A and B. The new memories are then presented. We also investigate the conditions under which the extended memories are able to recall patterns either from the fundamental set or from altered versions of them. We firstly analyze the case of the hetero-associative memories type **M** and **W**. We then perform the same analysis for auto-associative memories type **M** and **W**.

3.1. Operators A and B

When the $\alpha\beta$ memories make use of the α and β operators described in Ref. 10, its range of usability is $[0,1]$; when they make use of the extended operators A and B, introduced in this section, its range (as we shall see) is the gray-level range: $[0, L-1]$, with L the number of gray levels. We shall use the symbols, A and B to denote the extended operators, the capitals of α and β. Taking into account the properties of the original binary operators α and β described in Sec. 2.1, one way to find a generalization consists in formulating a system of functional equations [21]. It can be shown that, in this case, the system should have the following form:

$$\begin{aligned} \mathrm{B}\left(\mathrm{A}\left(x,y\right),y\right) &= x \\ \mathrm{B}\left(\left(x\vee y\right),z\right) &= \mathrm{B}\left(x,z\right)\vee \mathrm{B}\left(y,z\right) \end{aligned} \tag{10}$$

As we shall see later, these two properties of A and B are all that we need to characterize most of the solutions of interest. Once a family of solutions is found, one should show that the α and β, originally proposed in Ref. 10, are just a particular case of this new family. By taking into account Cauchy's functional equation [22], we propose, as an initial solution, the following:

$$\begin{aligned} \mathrm{A}\left(x,y\right) &= f\left(x\right)+g\left(y\right) \\ \mathrm{B}\left(x,y\right) &= p\left(x\right)+q\left(y\right). \end{aligned} \tag{11}$$

If A and B have this form, the matter of being increasing or decreasing with respect to x or $y$reduces to the requirement that *f, g, p, q,* be also increasing or decreasing. As we shall also see, distributivity to the left with respect to the **max** operator is equivalent to the fact that B is increasing with respect to its first element. As can be seen, the problem is already solved by means of the second equation. With this, we can see that we only need to focus on the inverse left relation between A and B. By taking into account the proposed structure for A and B, we have:

$$\mathrm{B}\left(\alpha\left(x,y\right)\right) = p\left(f\left(x\right)+g\left(y\right)\right)+q\left(y\right) = x \tag{12}$$

p distributes in its argument only if it is a solution of Cauchy's equation [22], this is p is a homotecy of the form $p(x) = cx$ where c is an arbitrary constant. Then

$$pf\left(x\right)+pg\left(y\right)+q\left(y\right)=x. \tag{13}$$

This implies that

$$pf\left(x\right)=x, \left(pg+q\right)\left(y\right)=0. \tag{14}$$

Thus, $p=f^{-1}$, $q=-f^{-1}g$

As a first observation, we have the fact that, once the parameters of A are established, those of B are determined. Also, as p is a homotecy, f, its inverse, is also. Taking f as the most simple homotecy (the identity), we have:

$$f\left(x\right)=x, p\left(x\right)=x, q=-g. \tag{15}$$

In addition g is minus the identity, A and B are as follows:

$$\begin{aligned} \mathrm{A}\left(x,y\right) &= x-y \\ \mathrm{B}\left(x,y\right) &= x+y. \end{aligned} \tag{16}$$

On the other hand, it can be shown that when adding a constant, let us say k, to A, this generates a non-linearity on B, that is if:

$$\mathrm{A}\left(x,y\right)=x-y+k. \tag{17}$$

This implies that $\mathrm{B}\left(x,y\right) = \phi\left(x+y\right)$ where ϕ is nonlinear function.

The family of binary operators generated this way allows us to obtain, on the one hand, the expression for binary operators α and β, where ϕ (Fig. 1) is the step function centered at 1:

$$\begin{aligned} \alpha\left(x,y\right) &= x-y+1 \\ \beta\left(x,y\right) &= \phi\left(x+y\right) \end{aligned} \tag{18}$$

$$\phi\left(x\right)=\begin{cases} 0 & x\leq 1 \\ 1 & x>1 \end{cases}. \tag{19}$$

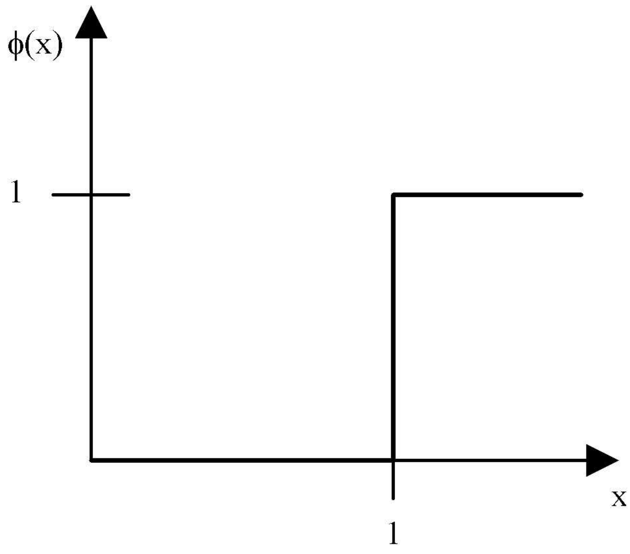

FIGURE 1. Graph of ϕ.

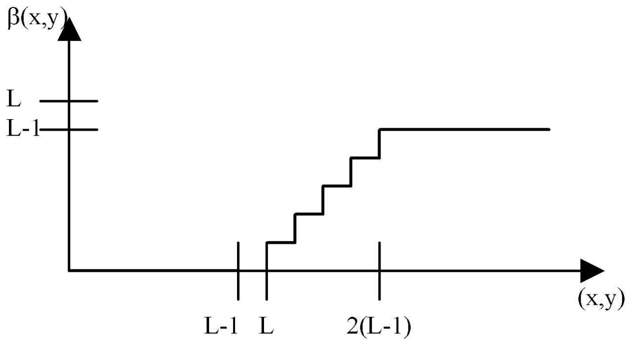

FIGURE 2. Graph of B.

On the other hand, the same family of operators allows us to obtain an expression for patterns with L gray levels, whose discrete domain is, for example, $A = \{0, 1, 2, \ldots, L-1\}$; in this case A and B take the form

$$\mathrm{A}(x,y) = x - y + L - 1$$
$$\mathrm{B}(x,y) = \phi(x+y), \tag{20}$$

where now ϕ is, in this case, the staircase function (Fig. 2), defined as follows:

$$\mathrm{B}(x,y) = \begin{cases} 0 & \text{if} \quad (x+y) \le L-1 \\ x+y-(L-1) & \text{if} \quad L \le x+y < 2(L-1) \\ L-1 & \text{if} \quad (x+y) \ge 2(L-1) \end{cases}. \tag{21}$$

Functions A and B have the same form as in the binary case, namely: $\mathrm{A} : A\times A \to B$ and $\mathrm{B} : B\times A \to A$, where now $A = \{0, 1, 2, \ldots, L-1\}$ and $B = \{0, 1, 2, \ldots, 2(L-1)\}$. The properties of these two important operations are much like those of the binary operators. They are not listed here due to space limitations.

3.2. Extended AB Memories

In this section we introduce the extended memories able to recall gray-level patterns. The proposed memories have the same structure as the $\alpha\beta$ memories presented in Ref. 10. The difference between the extended memories and the standard $\alpha\beta$ memories is of course the use of operators A and B instead of operators α and β. We provide the conditions under which the extended memories are able to recall patterns either from the fundamental set or from altered versions of them. Due to space limitations we only analyze the extended auto-associative memories of type **M** (EAS **M** memories). We give several numerical examples to better illustrate the operation of these devices. To operate an EAS **M** memory, we again use first the operator $\otimes$, then the **max** operator $\vee$. During training, Eq. (**4**) changes as follows:

$$v_{ij} = \bigvee_{\xi=1}^{p} \mathrm{A}\left(x_i^\xi, x_j^\xi\right). \tag{22}$$

In the same way, during pattern recall, Eqs. (**7**) and (**9**) change to:

$$(\mathrm{M} \wedge_{\mathrm{B}} \mathrm{x}^\omega)_i = \bigwedge_{j=1}^{n} \mathrm{B}\left(m_{ij}, x_j^\omega\right). \tag{23}$$

$$(M \wedge_{\mathrm{B}} \tilde{x})_i = \bigwedge_{j=1}^{n} \mathrm{B}\left(m_{ij}, \tilde{x}_j\right). \tag{24}$$

Next we give the conditions under which an EAS **M** memory provides correct recall. We first give the results concerning the correct recall of a pattern of the fundamental set. We then do the same thing, but an altered version of a pattern of the fundamental set is presented to the memory. The results presented here are essentially the same as those given in Ref. 10. Instead of using operators α and β, they take into account operators A and B. Their proof is not included due to space limitations.

Example 1. *Suppose we want to first memorize and then recall the following fundamental set, with L=8:*

$$\mathrm{x}^1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \quad \mathrm{x}^2 = \begin{pmatrix} 4 \\ 2 \\ 3 \end{pmatrix} \text{and } \mathrm{x}^3 = \begin{pmatrix} 6 \\ 5 \\ 5 \end{pmatrix}.$$

TRAINING PHASE:

$$\mathrm{x}^1 \vee_{\mathrm{A}} \left(\mathrm{x}^1\right)^T = \begin{pmatrix} 7 & 8 & 7 \\ 6 & 7 & 6 \\ 7 & 8 & 7 \end{pmatrix},$$

$$\mathrm{x}^2 \vee_{\mathrm{A}} \left(\mathrm{x}^2\right)^t = \begin{pmatrix} 7 & 9 & 8 \\ 5 & 7 & 6 \\ 6 & 8 & 7 \end{pmatrix},$$

$$\mathrm{x}^3 \vee_{\mathrm{A}} \left(\mathrm{x}^3\right)^T = \begin{pmatrix} 7 & 8 & 8 \\ 6 & 7 & 7 \\ 6 & 7 & 7 \end{pmatrix}.$$

Thus

$$\mathrm{M} = \begin{pmatrix} 7 & 9 & 8 \\ 6 & 7 & 6 \\ 7 & 8 & 7 \end{pmatrix}.$$

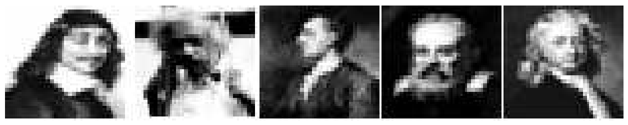

FIGURE 3. Set of images used in the experiments.

RECALLING PHASE:

$$M \wedge_{\mathrm{B}} x^1 = \begin{pmatrix} 7 & 9 & 8 \\ 6 & 7 & 6 \\ 7 & 8 & 7 \end{pmatrix} \wedge_{\mathrm{B}} \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$
$$= \begin{pmatrix} \mathrm{B}(7,1) \wedge \mathrm{B}(9,0) \wedge \mathrm{B}(8,1) \\ \mathrm{B}(6,1) \wedge \mathrm{B}(7,0) \wedge \mathrm{B}(6,1) \\ \mathrm{B}(7,1) \wedge \mathrm{B}(8,0) \wedge \mathrm{B}(7,1) \end{pmatrix}$$
$$= \begin{pmatrix} 1 \wedge 2 \wedge 2 \\ 0 \wedge 0 \wedge 0 \\ 1 \wedge 1 \wedge 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

One can easily verify that the other two patterns are also correctly recalled.

The following proposition provides conditions for correct recall of a pattern of the fundamental set when an altered version of it is presented to EAS **M** memory.

Proposition 1. *Let* $\left\{ \left(\mathrm{x}^\xi, \mathrm{x}^\xi \right) | \xi = 1, 2, \ldots, p \right\}$ *the fundamental set of an* EAS **M** *memory and L the number of levels the elements of each* x^ξ *can take. Let* $\tilde{x}$ *be an altered version with additive noise. If* $\tilde{x}$*is presented as input to the* EAS **M** *memory, and if besides for each* $i \in \{1, \ldots, n\}$ *it holds that* $\exists j = j_0 \in \{1, \ldots, n\}$*, which depends on w and i such as* $v_{ij_0} \leq \mathrm{A}\left(x_i^w, \tilde{x}_{j_0} \right)$*, then we have correct recall, namely* $\mathrm{M} \wedge_{\mathrm{B}} \tilde{x} = x^w$.

Example 2. *Let us take an altered version by additive noise of pattern* $\mathrm{x}^2 = \begin{pmatrix} 4 \\ 2 \\ 3 \end{pmatrix}$*, for example* $\tilde{x}^2 = \begin{pmatrix} 4 \\ 3 \\ 3 \end{pmatrix}$:

$$M \wedge_{\mathrm{B}} \tilde{x}^2 = \begin{pmatrix} 7 & 9 & 8 \\ 6 & 7 & 6 \\ 7 & 8 & 7 \end{pmatrix} \wedge_{\mathrm{B}} \begin{pmatrix} 4 \\ 3 \\ 3 \end{pmatrix}$$
$$= \begin{pmatrix} \mathrm{B}(7,4) \wedge \mathrm{B}(9,3) \wedge \mathrm{B}(8,3) \\ \mathrm{B}(6,4) \wedge \mathrm{B}(7,3) \wedge \mathrm{B}(6,3) \\ \mathrm{B}(7,4) \wedge \mathrm{B}(8,3) \wedge \mathrm{B}(7,3) \end{pmatrix}$$
$$= \begin{pmatrix} 4 \wedge 5 \wedge 4 \\ 3 \wedge 3 \wedge 2 \\ 4 \wedge 4 \wedge 3 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \\ 3 \end{pmatrix}.$$

The reader can easily verify that this example satisfies the conditions given by Proposition 1 for perfect recall in the presence of additive noise.

4. Experimental results

In this section, the proposed extended associative memories are tested with more realistic patterns. Images of five famous mathematicians (Descartes, Einstein, Euler, Galileo and Newton) were used are shown in Fig. 3. The images are 32 by 29 pixels and 256 gray levels. Only the EAS associative memories of type **M** and **W** were tested.

4.1. Construction of the Association Matrix

The images shown in Fig. 3 were first converted to image vectors with 968 elements (32 times 29) each. For this standard scanning procedure is used. These vectors were then used to construct the corresponding matrices **M** and **W**, by using the techniques described in Sec. 2.

4.2. Recalling of the fundamental set

In this first experiment, the five images were fed to matrices **M** and **W** already built. To all of them were applied the procedures described in Sec. 3.2. In all cases, of course, the five patterns were perfectly recalled.

4.3. Recalling of a pattern by a corrupted version of it

4.3.1. Case of a M memory

Three groups of images were generated: the first one with additive noise, the second one with saturated noise of the salt type, and the third one with manually saturated noise. In the first case, to the gray-value $f(x,y)$ of pixel with coordinates (x,y), an integer v was added, such that $f(x,y) + v \leq (L-1)$. In the second case, the gray-value of a pixel was simply saturated to the value $(L-1)$.

In the first case, the value v was first randomly selected. It was then added to the gray-value $f(x,y)$ of the pixel if $s < t$. $s \in [0,1]$ is an uniformly randomly distributed random variable, t is the parameter controlling how much of the image is corrupted. This way, the bigger the value of t, the more of the image pixels should be corrupted. If $t = 0$, no pixel value is modified. In the contrary, if $t = 1$, all pixels values should be changed. In the second case, the gray-value $f(x,y)$ was simply saturated to $(L-1)$ if $s < t$. In the third case, Microsoft PAINT utility was used was used to manually modify the gray-levels of the pixels.

A quantitative measure as to how good the recall is all cases was chosen as follows. Let $f_{recalled}(x,y), f_{original}(x,y)$ the gray levels of a pixel in the original image and the corresponding pixel in the recalled image. Let *NMP* the number of modified pixels in the recalled image with respect to the original image when subtracting pair by pair their gray levels. If *NPI* is total number of pixels of the image, then percentage of modified pixels *PP* of the recalled image with respect to the original image is given as:

$$PP = 100 \left(\frac{NMP}{NPI} \right) \tag{25}$$

4.3.1.1. Performance in the presence of additive noise

Twenty five images were obtained as explained. Parameter t was varied form 0.1 to 0.5 in steps of 0.1. Figure 4a shows the obtained images. The number (percentage) of modified pixels at each image is shown above each image. Recalled versions are shown in Fig. 4b. The number (percentage) of non-recalled pixels in the recalled image with respect to the original image is shown above each image. Notice also how as the level of noise increases, the recalled versions math well with the original images.

4.3.1.2. Performance in the presence of salt noise

Again, 25 images were obtained as explained. Parameter $t$was again varied from 0.1 to 0.5 in steps of 0.1. The value of the pixel was saturated to $(L-1)$. Figure 5a shows the obtained images. The number (percentage) of modified pixels at each image is shown above each image. Recalled versions are shown in Fig. 5b. Above each recalled image it is indicated the number (percentage) of non-recalled pixels with respect to the original image is shown above each image. Notice how despite the level of noise introduced to the images is bigger than in the first case, recalled versions match much better with the original images.

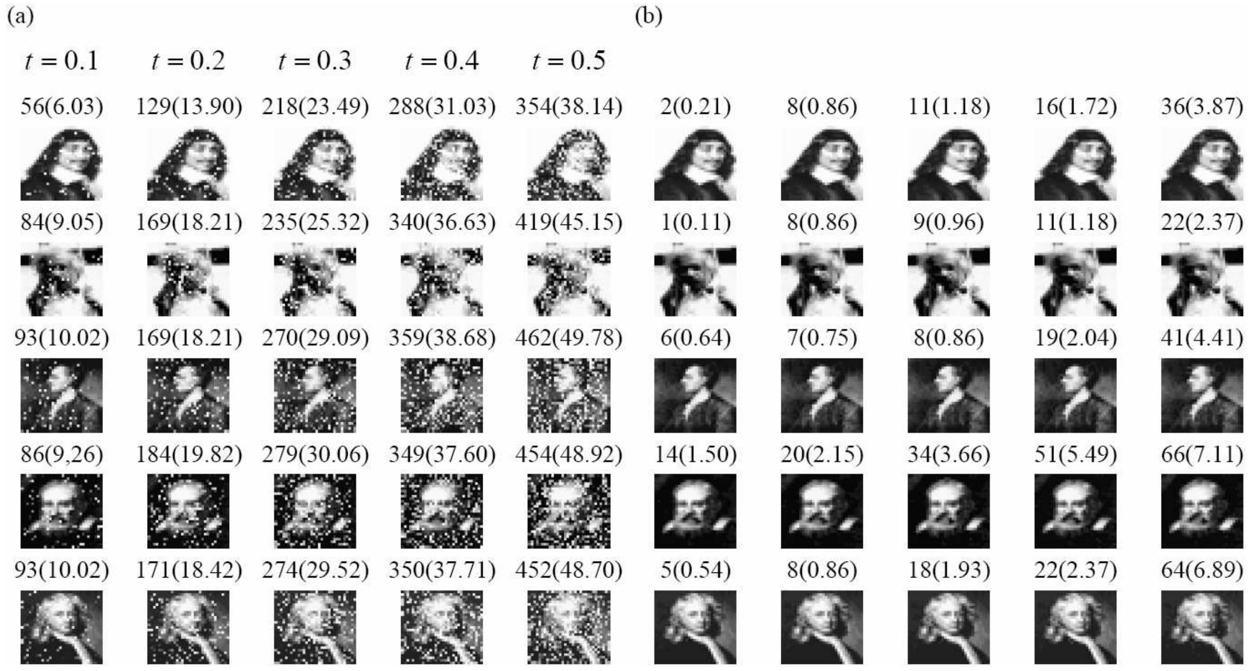

FIGURE 4. (a) Versions with additive noise, t varied from 0.1 to 0.5 with steps of 0.1. (b) Recalled images versions. The number of pixels here is the number of non-recalled pixel values.

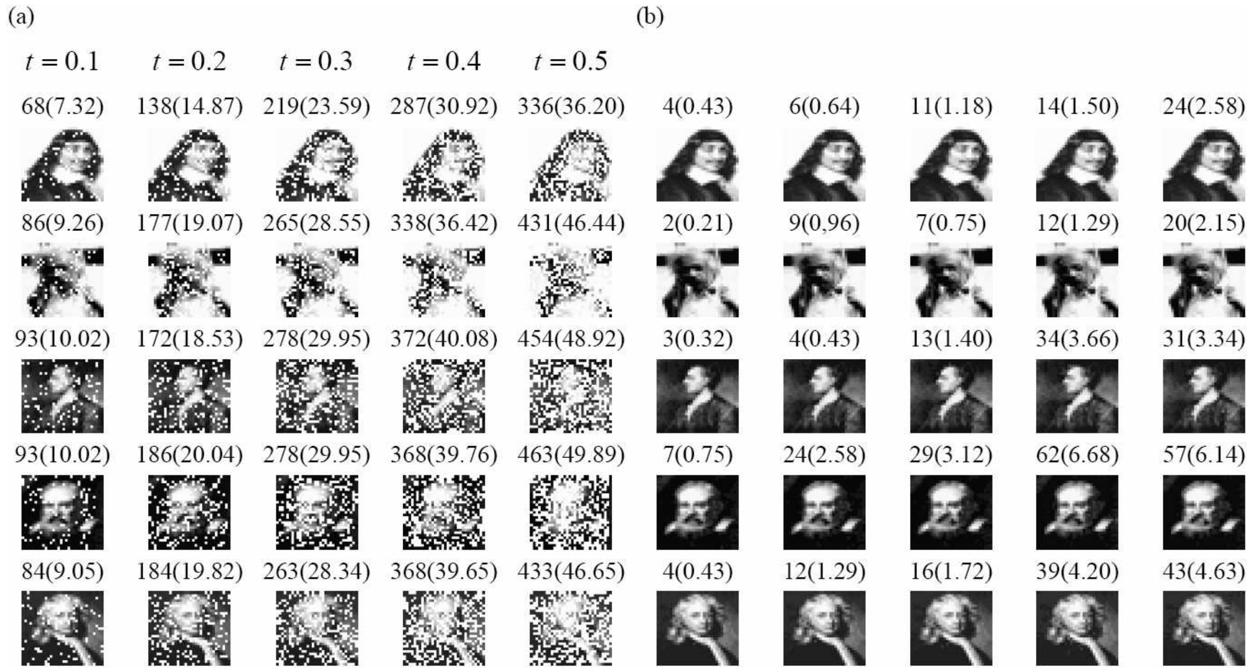

FIGURE 5. (a) Versions with salt noise, t varied from 0.1 to 0.5 with steps of 0.1. (b) Recalled versions. The number of pixels here is the number of non-recalled pixel values.

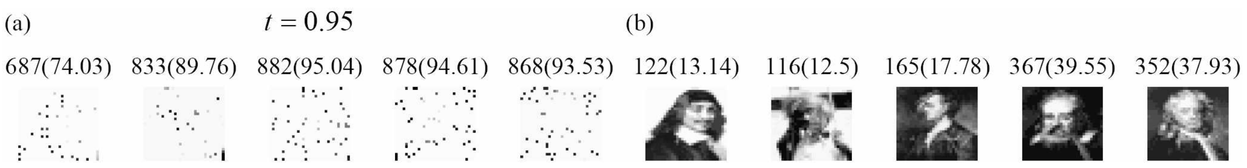

FIGURE 6. a) Versions with additive noise with $t = 0.95$. (b) Recalled versions.

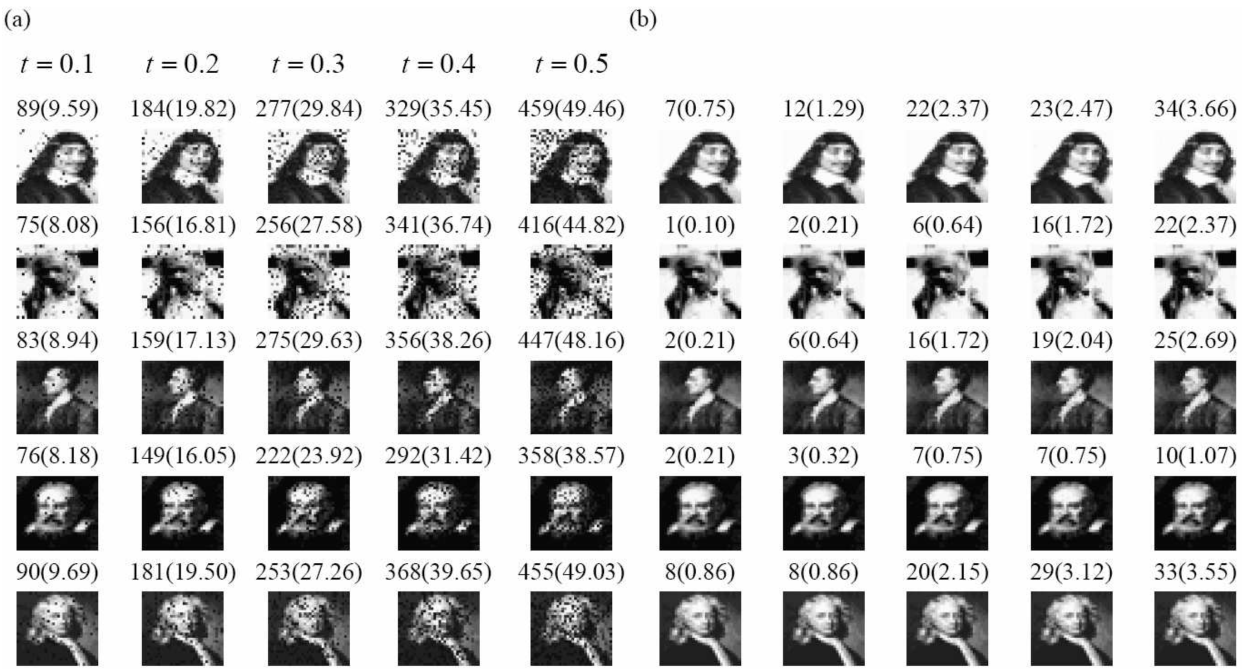

FIGURE 7. (a) Versions with subtractive noise, t varied from 0.1 to 0.5 with steps of 0.1. (b) Recalled images versions. The number of pixels here is the number of non-recalled pixel values.

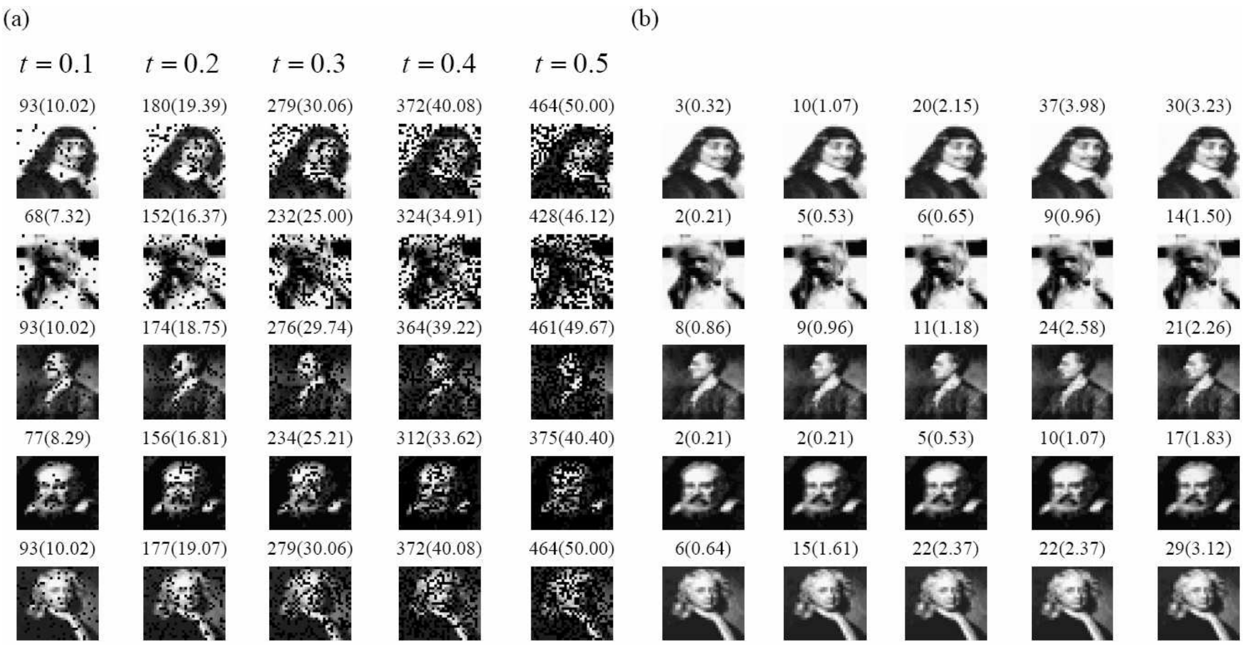

FIGURE 8. (a) Versions with pepper noise, t varied from 0.1 to 0.5 with steps of 0.1. (b) Recalled versions. The number of pixels here is the number of non-recalled pixel values.

4.3.1.3 Performance in the presence of big quantities of additive noise

We wanted to test the performance of the proposal with big quantities of additive noise. Five noisy, versions, one for each original image with $t = 0.95$ were generated. These are shown in Fig. 6a. Notice the level of noise introduced to each image. Even for us humans is impossible to re-built the original image from such distorted pattern. Recalled versions from these extremely noisy versions are shown in Fig. 6b. Notice how despite the level of noise introduced to the images, the recalled versions well match their originals.

The average time to recall an image in all cases, when using the proposed model, is 0.4 seconds in a Pentium 4 at 1.3 GHz.

4.3.2. Case of a W memory

Again, three groups of images were generated. The first one with subtractive noise, the second one with saturated noise of type pepper, and the third one with manually added saturated pepper noise. In the first case, to the gray-value $f(x,y)$ of pixel with coordinates (x,y), an integer v was subtracted, such that $f(x,y) - v \leq (L-1)$. In the second case, again to the gray-value of a pixel an integer was subtracted, such that $f(x,y) - v = (L-1)$. The value of v was chosen as in the case additive noise.

4.3.2.1 Performance in the presence of subtractive noise

Twenty-five images were obtained as explained. Parameter t was varied from 0.1 to 0.5 in steps of 0.1. Figure 7a shows the obtained images. The number (percentage) of modified pixels at each image is shown above each image. The recalled versions are shown in Fig. 7b. The number (percentage) of non-recalled pixels in the recalled image with respect to the original image is shown above each image. Notice also however the level of noise added, the recalled versions math well the original images.

4.3.2.2 Performance in the presence of pepper noise.

Twenty-five images were obtained as explained. Parameter t was varied from 0.1 to 0.5 in steps of 0.1. The value of the pixel was saturated to 0. Figure 8a shows the obtained images. The number (percentage) of modified pixels at each image is shown above each image. The recalled versions are shown in Fig. 8b. The number (percentage) of non-recalled pixels in the recalled image with respect to the original image is shown above each image. Notice also however the level of noise added, the recalled versions math well the original images.

4.3.2.3 Performance in the presence of big quantities of subtractive noise

We wanted to test the performance of the proposal with big quantities of subtractive noise. Five noisy, versions, one for each original image with $t = 0.95$ were generated. These are shown in Fig. 9a. Notice the level of noise introduced to each image. As in the case of highly distorted images with positive saturating noise, even for us humans it is impossible to rebuilt the original image from such distorted pattern. Recalled versions from these extremely noisy versions are shown in Fig. 9b. Notice how despite the level of noise introduced to the images, the recalled versions well match their originals.

The average time to recall an image in all cases, when using the proposed model, is 0.4 seconds in a Pentium 4 at 1.3 GHz.

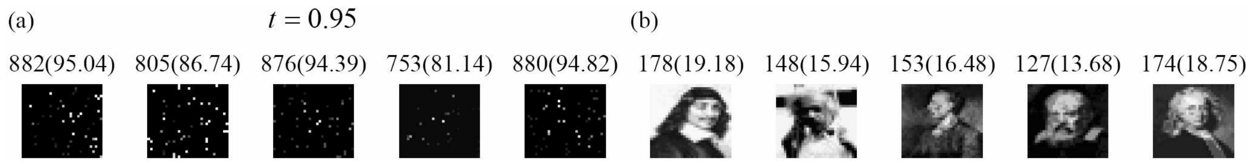

FIGURE 9. (a) Versions with subtractive noise with $t = 0.95$. (b) Recalled versions.

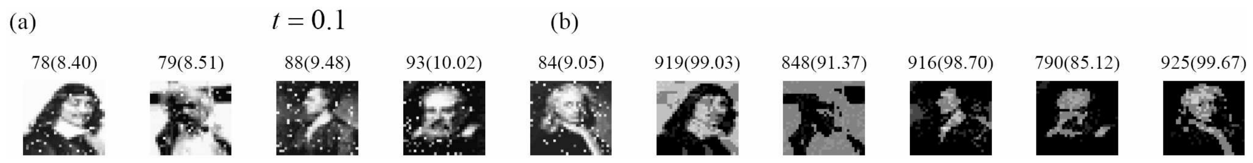

FIGURE 10. (a) Versions with additive noise with $t = 0.1$. (b) Recalled versions with M memory. Note how in this case most of pixel values are not correctly recalled. Compare with the results obtained with the new proposal with the value of $t = 0.1$ (Figure 4.)

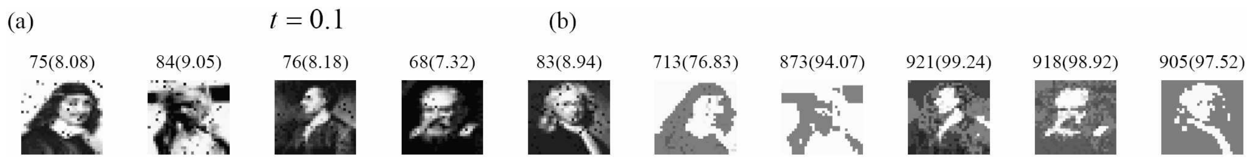

FIGURE 11. (a) Versions with subtractive noise with $t = 0.1$. (b) Recalled versions with W memory. Note how in this case most of pixel values are not correctly recalled. Compare with the results obtained with the new proposal with the same value of $t = 0.1$ (Figure 7).

4.3.3. *Comparison with other approaches*

The most similar work to the one presented in this work is the one reported in Refs. 13 and 16. In Refs. 13 and 16, the authors describe how a binary image can be used to recall gray-level patterns. The idea consists on that given a set of gray-level patterns to be first memorized: (1) Decompose them into their corresponding binary patterns, and (2) Build the corresponding binary associative memory (one memory for each binary layer) with each training pattern set (by layers). A given pattern or a distorted version of it, is recalled in three steps: (1) Decomposition of the pattern by layers into its binary patterns, (2) Recalling of each one of its binary components, layer by layer also, and (3) Reconstruction of the pattern from the binary patterns already recalled in step 2.

One of the problems of the proposal given in Refs. 13 and 16 is that when positive or subtractive noise is added to the pattern, at the moment of decomposing the pattern into its binary patterns, mixed noise is introduced. This is well known to be one of the main drawbacks of these kind of memories. This of course will tend to affect the performance of the memory. To show this fact in Figs. 10a and 11a we show one distorted of each pattern of Fig. 3 with positive noise and with subtractive noise, respectively, and with $t = 0.1$. Recalled versions are shown in Figs. 10b and 11b, respectively. Note how in both cases most of the pixel values were not correctly recalled, due to the problems already described. Compare the results obtained with the new proposal as shown in Figs. 4 and 7 and see how our proposal provides a much better recalling results. The reader can easily show that the more noise (additive or subtractive) is added to the pattern the worse will be the recall of the pattern if the proposal presented in Refs. 13 and 16 is used.

One last thing that is worth to be mentioning is that with the proposal described in Refs. 13 and 16 the recalling times in the same platform is of 7 seconds in average, while with the new proposal recalling times are of 0.4 seconds.

5. Conclusions

In this paper we have proposed a new set of associative memories able to work with gray-level patterns. The proposed set of memories is an extension of the associative $\alpha\beta$ memories recently introduced in Ref. 10. While $\alpha\beta$ memories work in the binary case, AB memories work in the gray-level case. To derive the set of extended memories, we first take operators α and β, the foundation of the functioning of $\alpha\beta$ memories, and solve a set of functional equations to get the extended operators, A and B. It is shown that the operators α and β are a special case if the general operators A and B. We give the necessary and sufficient conditions under which the proposed set of memories is able to recall first the fundamental set of patterns, and second a pattern from an altered version of it when additive or subtractive noise is added to the pattern.

The proposed extension was tested with several real patterns (images of five known mathematicians) with very satisfactory results. Even in the case of severe noise, the proposed extended memories are able to recall patterns from distorted versions of them.

Compared to other similar approaches the proposal shows a much better performance in recalling results as demonstrated in Sec. 4.3.3. Also the times for pattern recall with the new proposal are much better than the obtained with other similar works.

Nowadays, we are working through the solution of the following more general problem of patterns distorted with mixed noise. One of the main drawbacks of classical associative memories including morphological and $\alpha\beta$ memories is that they cannot efficiently deal with mixed noise. Their performance is highly affected when mixed noise appears. Mixed noise is a more realistic noise than either subtractive or additive noise. In this case we are searching for new ways to build the associative matrix in such way that a pattern affected by mixed noise is efficiently recalled. For some seminal investigations, refer for example to the works reported in Refs. 11, 12, 14, 15, and 17 to 20.

Acknowledgements

This work was economically supported by SIP-IPN under grants 20050156, 20060517 and 20071438, and CONACYT by means of grant 46805.

$*$. Corresponding author.

1. M.H. Hassoun, *Associative Neural Memories: Theory and Implementation* (Oxford University Press, 1993).
2. J.A. Anderson, *Mathematical Biosciences*, **14** (1972) 197.
3. J.A. Anderson and T. Kohonen (Eds.) *Neurocomputing: Foundations of Research* (Cambridge, MIT Press, 1990).
4. M.H. Hassoun, *Fundamentals of Artificial Neural Networks* (The MIT Press. 1995).
5. K. Steinbuch, *Kybernetik* **1** (1961) 26.
6. K. Steinbuch and H. Frank, *Kybernetik*, **1** (1961) 117.
7. J.J. Hopfield, *Proceedings of the National Academy of Sciences* **79** (1982) 2554.
8. G.X. Ritter, P. Sussner, and J.L. Díaz de León, *IEEE Transactions on Neural Networks* **9** (1998) 281.
9. G.X. Ritter, J.L. Díaz de León, and P. Sussner *Neural Networks* **12** (1999) 851.
10. C. Yáñez, Associative Memories based on Order Relations and Binary Operators (In Spanish), PhD Thesis, Center for Computing Research, February of 2002, (2002).

11. H. Sossa *et al.*, *Lecture Notes on Computer Science* **3287** (2004) 187.
12. H. Sossa *et al.*, *Lecture Notes on Computer Science* **3287** (2004) 195.
13. H. Sossa *et al.*, *Lecture Notes on Artificial Intelligence* **3315** (2004) 656.
14. H. Sossa *et al.*, *Lecture Notes on Artificial Intelligence* **3315** (2004) 687.
15. H. Sossa *et al.*, *Lecture Notes on Artificial Intelligence* **3315** (2004) 646.
16. H. Sossa *et al.*, *Neural Processing Letters* **22** (2005) 85.
17. R.A. Vázquez, H. Sossa, and R. Barrón, *LNAI* **3789** (2005) 317.
18. H. Sossa and R. Barrón, *Lecture Notes on Computer Science* **3773** (2005) 1036.
19. R.A. Vázquez, H. Sossa, and B. Garro, *Lecture Notes on Artificial Intelligence* **4293** (2006) 367.
20. B. Cruz, H. Sossa, and R. Barrón, to appear in *Neural Processing Letters* (2006).
21. A. Dhombres, *Functional Equations Containing Several Variables* (Cambridge University Press, 1991).
22. J. Aczel, *Functional Equations: History, Applications and Theory* (Kluwer Academic Pub. Group, 1984).