DOI:10.2298/CSIS091114026B

A new method for constructing kernel vectors in morphological associative memories of binary patterns

Yiannis.S. Boutalis[1]

[1] Department of Electrical and Computer Engineering,
Democritus University of Thrace,
GR-67100 Xanthi, Greece
ybout@ee.duth.gr

Abstract. : Kernel vectors represent an elegant representation for the retrieval of pattern associations, where the input patterns are corrupted by both erosive and dilative noise. However, their action completely fails when a particular kind of erosive noise, even of very low percentage, corrupts the input pattern. In this paper, a theoretical justification of this fact is given and a new method is proposed for the construction of kernel vectors for binary patterns associations. The new kernels are not binary but 'gray', because they contain elements with values in the interval [0, 1]. It is shown, both theoretically and experimentally that the new kernel vectors carry the good properties of conventional kernel vectors and, at the same time, they can be easily computed. Moreover, they do not suffer from the particular noise deficiency of the conventional kernel vectors. The recalling result is in general a gray pattern, which in the sequel undergoes a simple thresholding action and passes through a simple Hamming network to produce high recall rates, even in heavily corrupted patterns. Retrieval of pattern associations is very significant for a variety of scientific disciplines including data analysis, signal and image understanding and intelligent control.

Keywords: Neural networks, Associative memory, Kernel vectors, Noise Robustness.

1. Introduction

Morphological Neural Networks (MNNs) represent artificial neural networks whose neurons perform an elementary operation of mathematical morphology [1], [2]. Unlike Hopfield network [3,4], MNNs provide the result in one pass through the network, without any significant amount of training. A number of researchers devised MNNs for a range of applications like those appearing, for example, in [5-13].

Artificial neural network models are specified by the network topology, unit characteristics, and training or learning rules. The underlying algebraic system used in these models is the set of real numbers R together with the operations of addition and multiplication and the laws governing these operations. This algebraic system, known as ring, is commonly denoted by (R,+,x). The basic computations occurring in morphological networks are based on the algebraic lattice structure $(R,+,\wedge,\vee)$, where $\wedge,\vee$ denote the binary operations for minimum and maximum, respectively. The basic axiomatic operations of lattice algebra [14] are:

$$\begin{aligned}
&(\alpha \vee -\infty) = (-\infty \vee \alpha) = \alpha \quad \forall \alpha \in R_{-\infty} \\
&\alpha \wedge \infty = \infty \wedge \alpha = \alpha \quad \forall \alpha \in R_{\infty} \\
&\alpha + 0 = \alpha \quad \forall \alpha \in R \\
&0 + \alpha = \alpha \quad \forall \alpha \in R
\end{aligned} \qquad (1.1)$$

An *associative memory* (AM) is an input-output system that describes a relation $R \subseteq \mathrm{R}^m \times \mathrm{R}^n$. If ($\mathbf{x}$, $\mathbf{y}$)$\in R$, i.e. if the input $\boldsymbol{x}$ produces the output $\mathbf{y}$, then the associative memory is said to *store* or *record* the *memory association* ($\mathbf{x}$, $\mathbf{y}$). NN models serving as associative memories are generally capable of retrieving a complete output pattern **y** even if the input pattern **x** is corrupted or incomplete. The purpose of *auto-associative* memories is the retrieval of **x** from corrupted or incomplete versions $\tilde{\mathbf{x}}$. If an artificial associative memory stores associations (**x**, **y**), where **x** cannot be viewed as a corrupted or incomplete version of **y**, then we speak about *hetero-associative* memory.

Research on neural associative memories goes back in the 1950s [15-17]. *Linear associative memory* or *correlation memory* [17-19] is one well known neural associative memory. Association of patterns $\mathbf{x} \in \mathrm{R}^n$ and $\mathbf{y} \in \mathrm{R}^m$ is achieved by means of a matrix-vector product $\mathbf{y} = \mathbf{W} \cdot \mathbf{x}$. If we suppose that the goal is to store k vector pairs ($\mathbf{x}^1$, $\mathbf{y}^1$),...,($\mathbf{x}^k$, $\mathbf{y}^k$), where $\mathbf{x}^r \in \mathrm{R}^n$ and $\mathbf{y}^r \in \mathrm{R}^m$ for all r = 1,...,k, then **W** is an $m \times n$ matrix given by the following outer product rule:

$$\mathbf{W} = \sum_{r=1}^{k} \mathbf{y}^r \cdot (\mathbf{x}^r)' \qquad (1.2)$$

If **X** denotes the matrix whose column are the input patterns $\mathbf{x}^1,...,\mathbf{x}^k$ and **Y** denotes the matrix whose columns are the output patterns $\mathbf{y}^1,...,\mathbf{y}^k$, then this equation can be written in the simple form **Y·X'**. If the input patterns $\mathbf{x}^1,...,\mathbf{x}^k$ are orthonormal, then

$$\mathbf{W} \cdot \mathbf{x}^r = ((\mathbf{y}^1 \cdot \mathbf{x}^1)' + ... + (\mathbf{y}^k \cdot \mathbf{x}^k)') \cdot \mathbf{x}^r = \mathbf{y}^r \qquad (1.3)$$

Thus, we have perfect recall of the output patterns $\mathbf{y}^1,...,\mathbf{y}^k$. In case the patterns are not orthonormal the capacity of the memory is extremely limited and its ability to retrieve associations is further reduced when input patterns are noisy [20].

Morphological associative memories (MAMs) are based on the algebraic lattice structure $(R,+,\wedge,\vee)$. They have been initially proposed in [14], [21], [22] for associating binary pattern vectors and are far more robust to noise than the conventional linear associative memories. Moreover, their memory capacity in stored associations is significantly larger than the capacity of conventional linear associative memories. These properties and the fact that a number of notable features such as optimal absolute storage capacity and one-step convergence have been shown to hold in the general case for real-valued patterns [14], motivated for their extension to gray-coded vector associations [23] or recently in general gray-scale MAMs and fuzzy oriented treatments [24], [25],[26 - 28]. Recent works report also on recall and storage performance of various types of MAMs in color patterns under various types and percentages of noise [29], [30], while other works present new dynamic MAMs and test their performance in recalling gray and color pattern associations [31], [32], [33]. According to [14], there are two alternative approaches to construct MAMs. In the first approach the constructed memory is very robust in the presence of input patterns containing erosive noise. In the dual approach the constructed memory is very robust in the presence of input patterns containing dilative noise. In case the input pattern $\tilde{\mathbf{x}}^r$ is corrupted by both erosive and dilative noise none of the memories is able to recall $\mathbf{y}^r$.

To overcome this problem Ritter and his coworkers [34-36] proposed the idea of two-step MAMs and the production of the so-called kernel vectors, as an elegant representation of the associations ($\mathbf{x}^r$, $\mathbf{y}^r$). These vectors, in conjunction with appropriate morphological operations**,** are suitable to recall $\mathbf{y}^r$ when an arbitrarily corrupted (both eroded and dilated) input pattern $\tilde{\mathbf{x}}^r$ appears at the input of the MAM. A binary vector **z** is said to be a kernel of the association (**x**, **y**) between the binary vectors **x** and **y** if it is a subset of **x** and satisfies some conditions given in [34], [35]. More recent developments on reconstructing patterns from noisy inputs extend the definition of kernels and associate them with the property of strong morphological independence which should be present in the stored patterns [37-40].

There are two major issues related to the kernel method. The first is the very selection of representative kernel vectors. The conditions for a vector **z** to be a kernel vector, given in [34], [35], they do not provide with a fast method for selecting such vectors. As it will be pointed out in section 3 the procedure of selecting kernel vectors is quite time consuming, especially when the pattern vectors are of large dimension. Additional conditions for kernels and selection procedures, which alleviate in some extent this problem, are proposed in [35],[41-43] while in [37-40] the procedure of selecting the kernels emerges through the steps of the proof of the relevant theorems, provided that the patterns are initially modified to meet the property of strong morphological independence.

The second major issue, which is of this paper concern, is related to the behavior of kernel method in the presence of noisy input patterns. Although this issue has received only minor attention in the relevant literature [43], the kernel method is not entirely robust to noise. It can be experimentally verified that, although kernel vectors are quite satisfactory in retrieving associations based on noisy (both dilated and eroded) data, in some cases, they entirely fail even in the presence of very small noise percentages. This happens when one or more nonzero element of the input pattern, which correspond to respective nonzero elements of the kernel vector are “hit” by erosive noise, which converts them to zero-valued elements. In this case, the retrieval procedure recalls nothing, even if the amount of noise is quite limited.

In this paper, a theoretical proof of this observation is established and the mechanism of producing this failure is explained. Next, a new kernel definition method is proposed which overcomes this problem and is more robust to such kind of noise than the traditional kernel vectors. The new kernels are not anymore binary but they contain elements of variable values in the range [0, 1]. The values of the elements of each selected kernel are related to the frequency of appearance of the corresponding elements of the input training patterns. Different alternatives of selecting kernels based on frequencies are proposed and are compared in respect to their robustness in noise. All frequency based kernel vectors no longer suffer from the particular noise deficiency of conventional kernels. Moreover, the procedure of selecting the kernel vectors is quite simple and well defined; therefore it is not time consuming. The development is given for an auto-associative scheme, but hetero-association can be tackled with the same procedure. The pattern association recall is performed in three steps. The first step is the (*input pattern -> kernel*) and (*kernel -> output pattern*) recall. The result is an output pattern, which in general is not binary but gray and keeps in a large extend the shape of the output pattern to be recalled. Therefore a simple thresholding action is the second step and may produce the corresponding binary pattern. The third step increases the performance of the proposed method by passing the recalled binary pattern through a simple Hamming network equipped with index selection. The performance of the new kernel in binary pattern recall is demonstrated by using extensive character recognition experiments under various types and percentages of noise.

The paper is organized as follows. First, in section 2, a brief but complete introduction to the morphological associative memories is given. The kernel method is also reported in the same section and its weaknesses are demonstrated. In the sequence, in section 3, the theoretical justification of the failure of the kernel method in the presence of the particular kind of erosive noise is given. In section 4, the new kernel definition method is introduced and its validity and noise robustness is theoretically established. Section 5 provides experimental results regarding the capacity and noise robustness of the proposed kernels, presenting also comparisons between the alternative versions of the frequency based kernels. Conclusions and discussion are finally given in section 6.

2. Morphological associative memories and the kernel method

There are two basic approaches to record k vector pairs ($\mathbf{x}^1$, $\mathbf{y}^1$),...,($\mathbf{x}^k$, $\mathbf{y}^k$) using a *morphological associative memory* (MAM) [14]. The first approach consists of structuring an $m \times n$ matrix $\mathbf{W}_{XY}$ with elements computed by

$$w_{ij} = \bigwedge_{r=1}^{k} (y_i^r - x_j^r), \quad i = 1,...,m \quad j = 1,...,n \tag{2.1}$$

The dual approach consists of constructing an $m \times n$ matrix $\mathbf{M}_{XY}$ with elements computed by

$$m_{ij} = \bigvee_{r=1}^{k} (y_i^r - x_j^r), \quad i = 1,...,m \quad j = 1,...,n \tag{2.2}$$

If matrix $\mathbf{W}_{XY}$ receives a vector $\mathbf{x}^r$ as input, the product $\mathbf{y}^r = \mathbf{W}_{XY} \otimes \mathbf{x}^r$ is formed. The product is called *max product* and each element of the resulting vector $\mathbf{y}^r$ is computed by the formula

$$y_i^r = \bigvee_{j=1}^{n} (w_{ij} + x_j^r), \; i = 1,...,m \tag{2.3}$$

Likewise, if matrix $\mathbf{M}_{XY}$ receives $\mathbf{x}^r$ as input the so-called *min product* $\mathbf{y}^r = \mathbf{M}_{XY} \oplus \mathbf{x}^r$ is formed, where each element of the resulting vector $\mathbf{y}^r$ is computed by the formula.

$$y_i^r = \bigwedge_{j=1}^{n} (m_{ij} + x_j^r), \; i = 1,...,m \tag{2.4}$$

Matrices $\mathbf{W}_{XY}$ and $\mathbf{M}_{XY}$, computed by (2.1) and (2.2) respectively, constitute the memory of the MAM. The only required training of the network is simply the computation of either of the two matrices **W** or **M**. The difference between the two memories arises when noisy patterns appear at their input. Memory $\mathbf{M}_{XY}$ is able to retrieve $\mathbf{y}^r$ in case a noisy vector $\tilde{\mathbf{x}}^r$ corrupted by dilative noise appears at its input, while memory $\mathbf{W}_{XY}$ is able to retrieve $\mathbf{y}^r$ in case an eroded version of the input pattern appears at its input. The nature of dilative and erosive noise in binary patterns, as well as the retrieval results of $\mathbf{M}_{XY}$ and $\mathbf{W}_{XY}$ respectively is depicted in Fig. 2.1 and 2.2. In case the input pattern $\tilde{\mathbf{x}}^r$ is corrupted by both erosive and dilative noise none of the memories is able to recall $\mathbf{y}^r$.

Other types of memories have also been proposed in the recent years. The *median* operator (instead of min or max) is initially proposed in [31] to produce a memory that is robust in strictly mixed type of noise, which presents perfect recall if this noise is of median zero and if the fundamental (uncorrupted) pattern set fulfills some strict conditions. To overcome these restrictions the authors in [31] propose the construction of the memory using specially constructed surrogates of the fundamental pattern set and an algorithm for the training and recall phase using these surrogates. The idea of using surrogates appears also in [32], [33], where a dynamic type of memory is presented and the *mid* operator is employed. Although successful in storing and recalling general pattern associations, dynamic associative memories are not appropriate for binary pattern associations, because an essential part of their algorithm, namely the determination of active region [32], fails when binary patterns are involved.

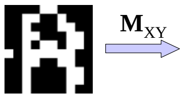

Fig. 2.1. Pattern recall of a dilated input using the min product and memory matrix M.

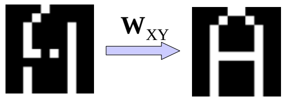

Fig. 2.2. Pattern recall of an eroded input using the max product and memory matrix W.

Another type of memories for binary pattern associations follows from fuzzy oriented treatments [25 - 28]. A new set theoretic interpretation of recording and recall of binary AMM is given in [25-27] and a generalization is provided using fuzzy set theory. This results in the definition of the so-called fuzzy morphological associative memories and the use of the *fuzzy max product* of **W** and **x** and the *fuzzy min product* of **M** and **x.** When a corrupted binary pattern appears as input to memory **W** or **M** the outcome is a gray-level pattern. The respective binary recall is obtained by using appropriate normalized threshold, which however has to be different depending on the type of memory used. An iterative algorithm is also proposed in [25] for the determination of the appropriate threshold. The recall scores are further improved in [28], where an enhanced fuzzy autoassociative morphological memory is presented and is combined with a discrete Hamming neural network that increases the accuracy of the recall. Experiments with patterns corrupted with various noise levels are presented, however the pattern set used is very limited and is not sufficient for drawing clear conclusions.

A well established approach to overcome the problem of the complete failure of memories **W** (eq. 2.1) and **M** (eq. 2.2) in dilative and erosive noise respectively is the *kernel* method. To overcome the problem Sussner, Ritter and coworkers [34], [35] proposed the idea of two-step MAMs and the production of the so-called *kernel vectors*, as an elegant representation of the associations ($\mathbf{x}^r$, $\mathbf{y}^r$). These vectors, in conjunction with matrices **M** and **W,** are suitable to recall $\mathbf{y}^r$ when an arbitrarily corrupted (both eroded and dilated) input pattern $\tilde{\mathbf{x}}^r$ appears at the input of the MAM. A binary vector **z** is said to be a kernel of the association (**x**, **y**) between the vectors **x** and **y** if it is a subset of ***x*** and satisfies the following conditions.

$$\mathbf{M}_{zz} \oplus \mathbf{x} = \mathbf{z} \qquad (2.5)$$

$$\mathbf{W}_{ZY} \otimes \mathbf{z} = \mathbf{y}, \qquad (2.6)$$

where matrices $\mathbf{M}_{ZZ}$ and $\mathbf{W}_{ZY}$ are computed according to (2.1), (2.2). In case of autoassociation (2.5) and (2.6) are written as

$$\mathbf{M}_{ZZ} \oplus \mathbf{x} = \mathbf{z} \qquad (2.7)$$

and

$$\mathbf{W}_{ZX} \otimes \mathbf{z} = \mathbf{x} \qquad (2.8)$$

respectively. Fig. 2.3 shows a sample of binary patterns (letters) and their corresponding kernel patterns, which satisfy equations (2.7) and (2.8). When an arbitrarily corrupted input pattern $\tilde{x}$ appears at the input of a morphological autoassociative network, the original uncorrupted input pattern $\mathbf{x}$ is recalled by the following equation:

$$\mathbf{W}_{ZX} \otimes \left(\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}}\right) = \mathbf{x} \qquad (\mathbf{2.9})$$

More recent developments on reconstructing patterns from noisy inputs extend the definition of kernels and associate them with the property of strong morphological independence which should be present in the stored patterns [37-40]. A procedure for selecting such kernels is proposed in [37], which also involves a mechanism of altering the initial patterns so that they fulfill the property of strong morphological independence.

Fig. 2.3. Input patterns and underneath their respective kernels

There are two major issues related to the traditional kernel method. The first is the very selection of representative kernel vectors. Equations (2.5), (2.6) (or (2.7), (2.8) for autoassociation) provide the necessary and sufficient conditions for a vector **z** to be a kernel vector but they do not provide with a fast method for selecting such vectors. An apparent procedure is the random selection of kernel vectors and their acceptance or rejection on the basis of (2.5) and (2.6). This procedure is quite time consuming, especially when the pattern vectors are of large dimension. Additional conditions for kernels and selection procedures are proposed in [35],[41-43].

The second major issue, which is of this paper concern, is related to the behavior of kernel method in the presence of noisy input patterns. Although this issue has received only minor attention in the relevant literature [43], the kernel method is not entirely robust to noise. It can be experimentally verified that, although kernel vectors are quite satisfactory in retrieving binary patterns associations based on noisy (both dilated and eroded) data, in some cases of erosive noise, they entirely fail even in the presence of very small noise percentages. This happens when one or more nonzero elements of the input pattern, which correspond to respective nonzero elements of the kernel vector are "hit" by noise, which converts them to zero-valued elements. In this case, the retrieval equation (2.9) recalls nothing. A theoretical explanation of this fact is given in the following section.

Closing this section, it is worth noting that even in the dynamic MAM presented in [32], [33], there is a part of the memory, namely a one column vector of the respective matrix of the memory, that the authors call *kernel* of the associative memory. The elements of such a kernel play an important role in recalling patterns altered by some kind of noise. The recall procedure will fail if *kernel* changes, leading to “loss of memory” [32], [33]. However, in this paper, we will not further refer to this type of memories, because dynamic MAM is not appropriate for storing and recalling binary pattern associations. This is due to the failure of determining the so-called “active region” [32] when binary patterns are involved. Another reason is that, with binary patterns, the low accuracy condition (see [32], section 4.2) is very often activated.

3. Noise deficiency of the kernel method

The basic idea of kernel method is to construct a vector (the kernel vector), which is a subset, or otherwise a sparse version, of the binary input vector it represents. Actually, the stricter the subsethood it is the better the performance of the method is. Two very useful conditions for binary kernels are proposed in [34] and in [35]. They are described by the following equations

$$\mathbf{z}^r \subset \mathbf{x}^r \qquad (3.1)$$

$$\mathbf{z}^\gamma \wedge \mathbf{z}^\xi = \mathbf{0} \qquad (3.2)$$

Equation (3.1) expresses the condition of subsethood described above, while equation (3.2) expresses the demand that different kernels should not have common nonzero points. A pictorial interpretation of the subsethood condition can be drawn from Fig. 2.3. Each binary letter of dimension (10x10) can be scanned row by row and be represented by a (100x1) vector containing only the values 0 and 1. Here, 1 represents the black pixel and 0 represents the white pixel. Similarly, the corresponding kernel patterns can be represented by the (100x1) kernel vectors. The 1s in the kernel vectors are much fewer than the 1s of the corresponding patterns and this is a direct interpretation of (3.1). Moreover, the kernel patterns do not have common non-zero elements and therefore condition (3.2) is fulfilled.

Since each kernel vector satisfies (3.1), each input vector $\mathbf{x}^r$ can be considered as a dilated version of its corresponding kernel vector $\mathbf{z}^r$. Therefore, since memory $\mathbf{M}_{ZZ}$ is very capable in retrieving patterns corrupted by dilative noise it can recall $\mathbf{z}^r$ by using equation (2.5) (or (2.7)) and the vector $\mathbf{x}^r$ as its input. In the sequel, each recalled kernel vector $\mathbf{z}^r$ can be used as the input to the memory $\mathbf{W}_{ZX}$ or $\mathbf{W}_{ZY}$ to retrieve $\mathbf{x}^r$ or $\mathbf{y}^r$ by using equation (2.6) or (2.8) respectively. The combination of these two steps is expressed in equation (2.9), In case an arbitrarily corrupted version $\tilde{\mathbf{x}}^r$ of the input pattern is considered, equation (2.9) can still provide reasonably good results, provided that $\tilde{\mathbf{x}}^r$ can be considered as a dilated version of $\mathbf{z}^r$. If, however, $\tilde{\mathbf{x}}^r$ can be considered as a corrupted version $\mathbf{z}^r$, which contains even the slightest amount of erosive noise, the recall procedure entirely fails and equation (2.5) and consequently (2.9) recalls nothing. The theorems that follow theoretically explain this situation. A pictorial representation of the situation is also given in Fig. 3.1. It can be observed that if the input pattern is hit by erosive noise, which affects even one pixel that corresponds to a nonzero element of the corresponding kernel pattern, then the recall procedure completely fails.

For notational convenience, we call as kernel point any element of the kernel vector which has nonzero value (value 1). We also recall that, kernel vectors satisfy (3.1) and (3.2).

Theorem 3.1. The rows of **M**zz are vectors with values

1. All zeros ($m_{ij} = 0, \forall j$) if the row index (i) does not correspond to any kernel point of any kernel vector.
2. All ones ($m_{ij} = 1, \forall j$) if the row index (i) corresponds to a kernel point belonging to any kernel vector z^{γ} , except when the column index (j) corresponds to another kernel point of the same kernel vector z^{γ} . In this case $m_{ij} = 0$

The proof of Theorem 3.1 is given in the Appendix. A pictorial representation of **M**zz for the kernels of Fig. 2.3 is given in Fig. 3.2, where the values of **M**zz are displayed in the form of a binary image (0 = black, 1= white). In this example, **M**zz is of dimension (100x100).

Theorem 3.2. If there is at least one index *i*, such that $\tilde{x}_i^r = 0$ and $z_i^r = 1$, then equation (2.5) recalls nothing. That is

$$\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}} = \mathbf{0} \tag{3.3}$$

and consequently

$$\mathbf{W}_{ZX} \otimes \left(\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}}\right) = \mathbf{0} \tag{3.4}$$

The proof of Theorem 3.2 is given in the Appendix. It explains why there is such a complete failure in the presence of this particular kind of erosive noise, even if the noise percentage is very low.

4. The new kernel definition method

In the sequel we propose a new method for recalling associations between binary patterns, by constructing kernel vectors which does not present the noise deficiency of the conventional kernel vectors described in the previous section. Moreover, the procedure of constructing the kernels requires only moderate processing time and therefore is computationally much more efficient than the conventional kernel method. In our approach kernel vectors are not binary but their elements are allowed to take values in the interval [0, 1]. Each kernel vector **z** is constructed to be a subset of an input vector **x** (see equation (3.1)), but the subsethood is defined in a fuzzy-like manner.

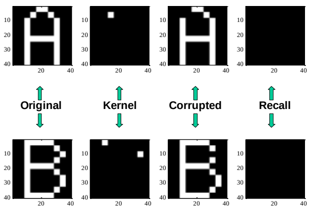

Fig. 3.1. Failure of the recall procedure when even a slight amount of erosive noise affects pixels, which correspond to nonzero kernel elements.

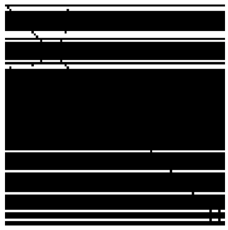

Fig. 3.2. Pictorial repre-sentation of the values of **M**zz. A black pixel cor-responds to $m_{ij} = 0$, while a white pixel corresponds to $m_{ij} = 1$.

Definition 4.1. The new kernel vectors are formed based on the following conditions.

1. Kernel vectors contain values that are in the interval [0,1].
2. The values of the elements of each kernel vector are zero when the corresponding values of the input vector are zero and nonzero when the corresponding input vector values are nonzero.
3. The nonzero values of each kernel vector are formed based on the frequency, f_j, of appearance of the corresponding nonzero input vector element in all the input vectors according to the following ***method 1***:
4. $z_j^r = 1$ if $f_j = 1$, $z_j^r = 0.8$ if $f_j = 2$, $z_j^r = 0.7$ if $f_j = 3$, $z_j^r = 0.4$ if $f_j \geq 4$

We call the non-zero elements of a kernel vector ***kernel points***. Each kernel point is associated with its strength. The kernel points with $z_j^r = 1$ are the *strongest kernel points*, while kernel points with $z_j^r = 0.8, 0.7, 0.4$ are weaker kernel points. In using the frequencies to construct kernels there are more than one alternative. Two such alternatives, termed ***method 2*** and ***method 3,*** are presented later on in this section. However, ***method 1*** is enough to demonstrate the idea and draw the relevant conclusions**.**

Fig. 4.1 shows examples of kernel vectors constructed according to *method 1* and associated with the input patterns (letters) of Fig. 2.3. Different gray levels represent the kernel points of different strength with the stronger kernel points corresponding to the brighter intensity. The algorithm for constructing kernel vectors is quite simple and obvious and therefore its computational burden is quite low in comparison with the conventional kernel method. Indeed, to construct a kernel vector **z** that corresponds to an input vector **x** we proceed as follows: If the value of the element of the input vector is 0 the corresponding element of the kernel vector is set to 0. For each nonzero element of the input vector we count the nonzero occurrences of the same element in the other input vector. We call this frequency of appearance. Next we assign a nonzero value to the corresponding element of the kernel vector according to the counted frequency and according to *method 1*of definition 4.1. The same procedure is applied for the construction of all the kernel vectors.

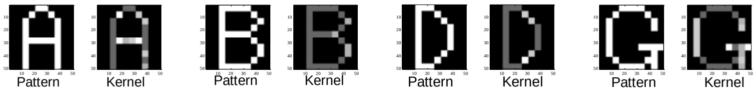

Fig. 4.1. Pictorial representation of patterns and their corresponding kernel patterns constructed using the proposed new method. The brighter pixels correspond to stronger kernel points.

Each new kernel vector is a subset of its corresponding input vector according to equation (3.1), but the subsethood is now defined in a fuzzy like manner since for each element of the kernel vector the following relation holds

$$\mathbf{z}_i^r \leq \mathbf{x}_i^r, \forall i \qquad (4.1)$$

The new kernel definition comes as an intuitive extension of conventional binary kernels. In meeting (3.1) and (3.2), denoting subsethood and uniqueness, the resulting binary kernels are sparse vectors, carrying all information in the few nonzero values. All these nonzero values are crucial and, as pointed out in section 3, even if one of them is missing (has zero value) in the input pattern the whole recalling procedure collapses. In the new definition, the "gray" kernel vectors are allowed to include also other supporting non-zero values, associated with non-zero pattern elements appearing in more than one pattern. Due to the definition of the new kernel vectors, which is based on the frequencies of the elements of the corresponding input vectors, the form of memory matrix **M**zz is now completely different than the corresponding matrix produced by conventional kernels. The new form of **M**zz allows the recalling procedure to be successful even if the input vector is corrupted by the special erosive noise described in section 3. The particular *method 1* for selecting the gray values {1, 0.8, 0.7, 0.4} to be attributed in kernel points associated with the respective pattern non-zero element frequencies {1, 2, 3, ≥ 4}, came out of author's observations and is sufficient for demonstrating the idea and drawing the conclusions. As it is proved in the next theorems, the proposed *method 1* serves the purpose of removing the particular noise deficiency of traditional kernels. Other frequency based methods may also be valid. Two such alternative methods (*method 2* and *method 3*) are presented at the end of this section, which carry the good property of *method 1,* and perform better in large pattern sets and are more robust in mixed noise.

Theorem 4.1. The rows of **M**zz are vectors with values

- All zeros ($m_{ij} = 0, \forall j$) if the row index (i) does not correspond to any kernel point of any kernel vector (frequency 0).

- $m_{ij} \in I_1 = \{1, 0.6, 0.3, 0.2, 0\}$ if the row index i corresponds to kernel value $z_i^r = 1$ (frequency 1).
- $m_{ij} \in I_2 = \{0.8, 0.4, 0.1, 0\}$ if the row index i corresponds to kernel value $z_i^r = 0.8$ (frequency 2).
- $m_{ij} \in I_3 = \{0.7, 0.3, 0\}$ if the row index i corresponds to kernel value $z_i^r = 0.7$ (frequency 3)
- $m_{ij} \in I_4 = \{0.4, 0\}$ if the row index i corresponds to kernel value $z_i^r = 0.4$ (frequency ≥ 4).

The proof of Theorem 4.1 is given in the Appendix.

Theorem 4.2. If there are one or more indices *i*, such that $\tilde{x}_i^r = 0$ and $z_i^r \neq 0$, then equation (2.5) recalls the kernel vector or an eroded version of it. That is

$$\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}} = \tilde{\mathbf{z}} \quad (4.2)$$

and consequently

$$\mathbf{W}_{ZX} \otimes \left(\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}}\right) = \mathbf{x} \quad (4.3)$$

The proof of Theorem 4.2 is given in the Appendix. Fig. 4.2 shows examples of perfect recall using the new kernel vectors constructed by *method 1* and as input vectors uncorrupted patterns. Figure 4.3 shows examples of perfect recall using the new kernel vectors and as input vectors patterns corrupted by the erosive noise described in section 3.

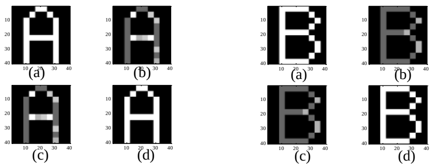

Fig. 4.2. Examples of perfect recall using the new kernel vectors and input vectors the uncorrupted patterns: (a) the input, uncorrupted vector (b) the original kernel vector (c) perfect recall of the kernel vector by (2.5) (d) perfect recall of the pattern by (2.9)

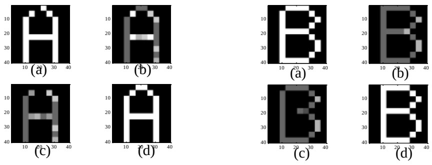

Fig. 4.3. Examples of perfect recall using the new kernel vectors and input vectors corrupted by the erosive noise described in section 3. (a) the corrupted vector used as input (b) the original kernel vector (c) non-perfect recall of the kernel vector by (2.5) (d) perfect recall of the pattern by (2.9)

It has to be noted that term $\left(\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}}\right)$ in (4.3) will produce $\tilde{\mathbf{z}}$, which can be considered as an eroded version of $\mathbf{x}$. Memory $\mathbf{W}_{ZX}$ is robust in erosive noise but its robustness depends also on the amount of the erosion. Therefore, with the exception of experiments involving small pattern sets with probably small percentages of mixed noise, eq. (4.3) will not produce a binary pattern but a gray surrogate of it, having a shape quite close to it. A binary version of the recall can be obtained after a simple thresholding action. Similar to [28], the recalling results can be further improved if the binary recall passes through a simple Hamming network to produce an index pointing to a pattern of the initial uncorrupted pattern base. The overall recalling scheme is shown in Fig. 4.4 and its performance is demonstrated by the experiments appearing in the next section.

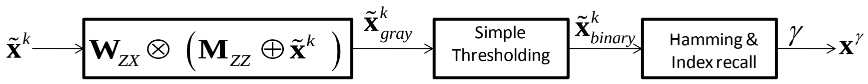

Fig. 4.4. The three step recall procedure. The corrupted binary pattern $\tilde{\mathbf{x}}^k$ passes through eq. (4.3) to produce the gray recall $\tilde{\mathbf{x}}^k_{gray}$, which after simple thresholding will produce a probably corrupted binary pattern recall. The third step will produce an index γ to be used for retrieving the uncorrupted binary pattern $\mathbf{x}^y$. If $\gamma = k$ we have perfect recall.

In definition 4.1 the values of the elements of kernel vectors are determined by their frequency of appearance of the corresponding element in the patterns. However, in ***method 1*** only 4 different values are employed, as all elements having frequency greater than 4 receive the same value $z_j^r = 0.4$. Two alternative frequency based methods are presented below.

Method 2: A plausible extension of method 1 is to allow for a finer determination of kernel point values taking into account all available frequencies of appearance. Since the maximum frequency of a non-zero pixel cannot be greater than the number of available patterns (*nop*) the kernel elements receive the following grading

$$z_j^r = 1 \text{ if } f_j = 1, \quad z_j^r = 1 - \frac{f_j}{nop} \text{ if } f_j \geq 2$$

Regarding the special noise deficiency of kernels, *method 2* carries the same good properties of *method 1*, because theorems 4.1 and 4.2 can be easily extended to cover this method by simply increasing the number of different cases examined to at most *nop* cases. In this approach the sets I_i in Theorem 4.1 now contain more and different values than those of *method 1*, but the rationale of their development and their usage in Theorem 4.2 remain the same. Regarding noise robustness, in the experiments carried out in the next section, *method 2* proves to be more robust than *method 1*.

Method 3: A slightly different approach for constructing the gray kernels is the frequency dependent two-step kernel construction method, which is described here in detail. In the first step frequency patterns are created. Then the kernel vectors are determined by giving values to the kernel points according to the strength of common frequencies in the frequency patterns. More specifically,

<u>Step 1.</u> For each initial pattern $\mathbf{x}^r$ a corresponding frequency pattern p^r is constructed. The elements of the frequency pattern are constructed according to the following rule

$p_j^r = 0$ if $x_j^r = 0$, $p_j^r = f_j$ if $x_j^r \neq 0$

Where, as usual, f_j is the frequency of appearance of the non-zero j^{th} element in all patterns. Apparently the minimum value f_j can take is 1.

<u>Step 2.</u> Based on the set of “frequency” patterns, the set of common frequencies F is determined. That is, F contains all the distinct frequencies that appear in ***all*** the “frequency’ patterns. Apparently, the cardinality c_F of F is at most equal to the number of patterns *nop*; usually it is smaller. Let also $f_{\min} = \min f_i \in F$ and $f_{\max} = \max f_i \in F$ denote the minimum and maximum of the frequencies participating in F. Next, for each pattern, its kernel vector z^r is constructed according to the following rule.

$z_j^r = 0$ if $p_j^r = 0$, or $p_j^r \notin F$

$z_j^r = 1$ if $p_j^r = f_{\min}$, $z_j^r = 1 - \dfrac{p_j^r - f_{\min}}{f_{\max} - f_{\min}}$ if $p_j^r > f_{\min}$

That is, the nonzero elements of each kernel vector correspond to elements of the “frequency” patterns having frequencies that appear in all other “frequency” patterns. The concept of strong and weaker kernel points remain, because the values of the kernel points are determined by the corresponding frequencies, with the largest value (=1) taken by elements that correspond to the minimum common frequency and the other values (<1) are gradually reducing according to the increase of the corresponding common frequency.

Regarding the special noise deficiency of kernels, *method 3* carries the same good properties of *method 1* and *method 2,* because the construction of **M**zz is analogous to that of the other methods. Theorems 4.1 and 4.2 can be easily extended to cover this method by simply considering different number of cases examined (the number of cases is $c_F + 1$, counting also the $m_{ij} = 0$ case). The sets I_i in Theorem 4.1 now contain different values than those of the other methods, but the rationale of their development and their usage in Theorem 4.2 remain the same. Regarding noise robustness, in the experiments carried out in the next section, *method 3* proves to be much more robust than the other two methods.

5. Experimental results

To test the performance of the proposed method, a number of experiments were carried out. In order to have a meaningfully sized data set, 26 uncorrupted binary patterns were initially created. The set contains the 35x34 binary images of the capital letters of the Latin alphabet and is shown in Fig. 5.1. Some examples of gray kernels produced by *method 1, 2* and *3* respectively are shown in Fig. 5.2.

Fig. 5.1. The complete pattern set consisting of 35x34 binary images of the 26 capital letters of English alphabet.

Fig. 5.2. Sample of kernel vectors produced by the three frequency based methods. (a) Kernels of *method* 1 (b) Kernels of *method* 2 (c) Kernels of *method* 3.

All methods (1, 2 &3) of constructing kernels were initially tested on their recall performance of *uncorrupted* patterns, when the three step procedure of Fig. 4.4 was applied. All methods had a 100% success in recalling the uncorrupted pattern.

Fig. 5.3 shows three examples (one for each method) of the overall procedure. The pattern interacts with memory **M** according to (4.2) to produce **z.** Then, **z** interacts with memory **W** according to (4.3) to produce a gray pattern $\tilde{\mathbf{x}}$ having the shape of **x**. Next, the gray outcome passes from a simple thresholding scheme to produce a binary pattern. In this simple thresholding scheme every gray value greater than zero receives value 1. In the sequel, the binary recall may pass through a Hamming network to recall an index pointing to a member of the initial pattern set. Finally, the uncorrupted pattern associated with this index appears as the final recall of this procedure. It has to be noted that, in the experiments carried out, we did not use a Hamming network. Instead the Hamming distances

$$H_r = \sum_{i=1}^{n} \left| \mathbf{x}_i^r - \tilde{\mathbf{x}}_i \right| \quad r = 1, \cdots, k \qquad (5.1)$$

between the binary recall and each one of the patterns of the initial pattern space were computed and the index of the pattern giving the smallest distance was selected. The pattern recalled is $\mathbf{x}^{\gamma}$, where

$$\gamma = \arg\min_{r} \{H_r\}, \quad r = 1, \cdots, k \tag{5.2}$$

In (5.1), *n* denotes the size (in pixels) of a pattern, *i* denotes the i^{th} element of the pattern vector and *k* is the number of patterns in the pattern space.

Fig. 5.3. Steps of uncorrupted pattern recall according to the procedure of Fig. 4.4 using the three frequency based kernel methods. **(a)** recall using *method 1*, **(b)** recall using *method 2*, **(c)** recall using *method 3*. From left to right: **1st column** - input pattern, **2nd column** - kernel recall using (4.2), **3rd column** - pattern recall using (4.3), **4th column** – binarization of column 3 recall using simple thresholding, **5th column** – final recall after computing the minimum Hamming distance.

The use of (5.1) and (5.2) provides also a means for indirectly evaluating the storage capacity of the overall scheme. Starting from a small sized pattern space (k = 5) and gradually increasing the size by 2 until k = 26, the average minimum distance of all pattern recalls at each pattern space for each method is recorded. Figure 5.4 shows the plot of the average minimum distance (AMD) of recalls in respect to the size of the pattern space. It is evident that the AMD increases in the beginning but it approaches an upper bound as the number of patterns increases. That is, after a certain number of patterns the AMD and indirectly the storage capacity of the scheme is not affected by the size of the pattern space. Moreover, Fig. 5.4 demonstrates the superiority of method 3, since it provides always more confident results (smaller AMD).

The next set of experiments is concerned with the robustness of all methods under the presence of various percentages of mixed type (both dilative and erosive) noise. Mixed noise was randomly applied on each binary pattern. The noise percentage is computed by counting the number of the altered (noisy) pixels and divide them with the total number of image pixels. Figure 5.5 shows three examples (one for each method) of the overall recall procedure, when a corrupted by 30% mixed noise pattern appears in the input of the proposed 3-step recall scheme.

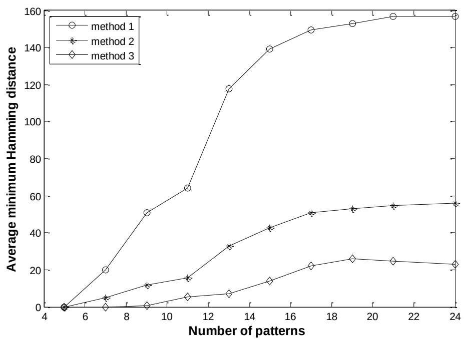

Fig. 5.4. Average minimum Hamming distance for the three frequency based kernel methods in respect to the size of the pattern space.

Fig. 5.5. Steps of successful pattern recall of a corrupted pattern according to the procedure of Fig. 4.4 using the three frequency based kernel methods. **(a)** recall using *method 1*, **(b)** recall using *method 2*, **(c)** recall using *method 3*. From left to right: **1st column** – corrupted input pattern, **2nd column** - kernel recall using (4.2), **3rd column** - pattern recall using (4.3), **4th column** – binarization of column 3 recall using simple thresholding, **5th column** – final recall after computing the minimum Hamming distance.

The various recall stages are similar with those appearing in Fig. 5.3, however the effect of noise is now evident in the intermediate results. It is also clear that *method 2* is more robust to noise than *method 1*, because the intermediate results are closer to the actual ones. Similarly, *method 3* performs much better than the other two.

Finally, Fig. 5.6 depicts the robustness of each method in mixed noise. The failure recall rate of each method is displayed in respect to the percentage of the mixed noise. Since the noise is randomly applied, in order to have statistically reliable results, each experiment corresponding to a different noise percentage was carried out 50 times and the average failure recall rate is actually recorded and displayed. It is evident that *method 3* is far more robust than the other two methods and *method* 2 is significantly more robust than *method 1.* Therefore, although all frequency based kernel methods do not present the particular noise deficiency of traditional kernel methods, they present significant differences regarding their noise robustness. This in turn may prompt for future seeking of alternative frequency based kernel construction methods with better performance in recall of noisy patterns.

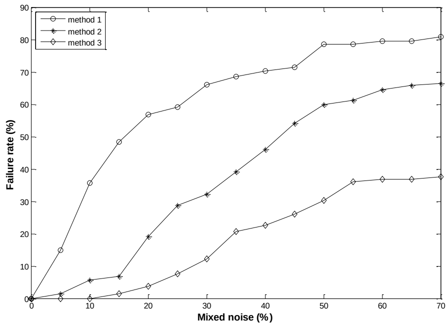

Fig. 5.6. Failure recall rates of the three frequency based kernel vector methods in respect to mixed noise percentage.

6. Conclusion

A new method for constructing kernel vectors was proposed in this paper to be used for recalling associations between binary patterns. The need for a new kernel definition arose from a special noise deficiency of the conventional binary kernel vectors of the relevant literature. The new kernels are not binary but 'gray', because they contain elements with values in the interval [0, 1]. These values are determined by the frequency of appearance of nonzero elements of the input pattern vectors. Alternative schemes of producing kernels based on frequency were presented and it was shown, both theoretically and by character recall examples that the new kernel vectors carry the good properties of conventional kernel vectors and, at the same time, they can be easily computed. Moreover, they do not suffer from the particular noise deficiency of the conventional kernel vectors. Experiments, performed on large set of binary patterns corrupted by various degrees of mixed noise demonstrate the robustness and the limits of the proposed approach. Future extension of this work might extend the proposed approach to cover more frequency dependent kernel alternatives as well as an investigation regarding the applicability of the proposed concepts in gray and possibly color pattern associations.

Acknowledgement. The author wants to acknowledge the contribution of Mr. S. Chatzichristofis in performing the experiments carried out.

References

1. Peter Sussner and Marcos Eduardo Valle, Morphological and certain fuzzy morphological associative memories for classification and prediction. In Vassilis Kaburlasos and Gerhard Ritter, (eds), Computational Intelligence Based on Lattice Theory, chapter 8, Springer-Verlag, Heidelberg, Germany, 149–172, (2007).
2. J.L.Davidson, G.X.Ritter, A theory of morphological neural networks, in: Digital Optical Computing II, Vol. 1215, Proc. SPIE, 378–388, (July 1990).
3. J.J.Hopfield, Neural networks and physical systems with emergent collective computational abilities, Vol. 79, Proc. of the National Academy of Sciences, U.S.A., 2554–2558, (April 1982).
4. J.J.Hopfield, Neurons with graded response have collective computational properties like those of two-state neurons, Vol. 81, Proc. of the National Academy of Sciences, U.S.A., 3088–3092, (May 1984).
5. J.L.Davidson, Simulated annealing and morphological neural networks, in: Image Algebra and Morphological Image Processing III, Vol. 1769, Proc. SPIE, San Diego, 119–127, (1992).
6. J.L.Davidson, F.Hummer, Morphology neural networks: an introduction with applications, Circuits, Systems and Signal Processing, Vol. 12, No. 2, 177–210, (1993).
7. C.P.Suarez–Araujo, Novel neural network models for computing homothetic invariances: an image algebra notation, Journal of Mathematical Imaging and Vision, Vol. 7, No. 1, 69–83, (1997).
8. Y. Won, P. D. Gader, Morphological shared weight neural network for pattern classification and automatic target detection, In Proc. of the IEEE International Conference on Neural Networks, Perth, Australia, 2134–2138, (November 1995).
9. P. D. Gader, M. Khabou, and A. Koldobsky, Morphological regularization neural networks, Pattern Recognition., Special Issue on Mathematical Morphology and Its Applications, Vol. 33, No. 6, 935–945, (2000).
10. L. Pessoa and P. Maragos, Neural networks with hybrid morphological/ rank/linear nodes: A unifying framework with applications to handwritten character recognition, Pattern Recognition, Vol. 33, 945–960, (2000).
11. Handbook of Neural Computation, E. Fiesler and R. Beale, (Eds)., IOP Publishing and Oxford University Press, pp. C1.8:1–C1.8:14. W. Armstrong, M. Thomas, “Adaptive logic networks”, (1997).
12. B. Raducanu, M. Graña, and X. F. Albizuri, Morphological scale spaces and associative morphological memories: Results on robustness and practical applications, J. Math. Imaging and Vision, Vol. 19, No. 2, 113–131, (2003).
13. M. Graña, J. Gallego, F. J. Torrealdea, and A. D’Anjou, On the application of associative morphological memories to hyperspectral image analysis, Lecture Notes in Computer Science, Vol. 2687, 567–574, (2003).
14. G.X.Ritter, P.Sussner, J.L.Diaz de Leon, Morphological associative memories, IEEE Trans. Neural Networks, Vol. 9, No. 2,, 281–293, (1998).
15. W.K. Taylor. Eletrical simulation of some nervous system functional activities, Information Theory, Vol. 3, 314–328, (1956).
16. K. Steinbruch. Die lernmatrix. Kybernetick, Vol. 1, 36–45, (1961).
17. J. A. Anderson and E. Rosenfeld, (Eds), Neurocomputing: Foundations of Research, Vol. 1, MIT Press, Cambridge, MA, (1989).
18. T.Kohonen, Correlation matrix memory, IEEE Trans. Comput. Vol. 21, 353–359, (1972).

19. James A. Anderson, Andras Pellionisz, and Edward Rosenfeld, (Eds). Neurocomputing: Directions for Research, Vol. 2, MIT Press, Cambridge, MA, (1990).
20. Toshiyuki Tanaka, Shinsuke Kakiya, and Yoshiyuki Kabashima, Capacity Analysis of Bidirectional Associative Memory, In Proc. Seventh Int. Conf. Neural Information Processing, Taejon, Korea, Vol. 2,, 779-784, (Nov. 2000).
21. G. X. Ritter and P. Sussner. An introduction to morphological neural networks, In Proceedings of the 13th International Conference on Pattern Recognition, Vienna, Austria, 709–717, (1996).
22. G. X. Ritter and P. Sussner, Morphological neural networks In Intelligent Systems: A Semiotic Perspective, In Proceedings of the 1996 International Multidisciplinary Conference, Gaithersburg, Maryland, 221–226, 1996.
23. G. Costantini, D. Casali, and R. Perfetti, Neural associative memory storing gray-coded gray-scale images, IEEE Trans. Neural Networks, Vol. 14, No. 3, 703–707, (2003).
24. P. Sussner and M. E. Valle, Gray-Scale Morphological Associative Memories, IEEE Trans. Neural Networks, Vol. 17, No. 3, 559-570, (2006).
25. P. Sussner, Generalizing operations of binary morphological autoassociative memories using fuzzy set theory, J. Math. Imaging and Vision, Special Issue on Morphological Neural Networks, Vol. 9, No. 2, 81–93, (2003).
26. P. Sussner, A relationship between binary autoassociative morphological memories and fuzzy set theory, In Proceedings of the Int. Joint Conf. on Neural Networks, Washington DC, (2001).
27. P. Sussner, A Fuzzy autoassociative morphological memory, In Proceedings of the Int. Joint Conf. on Neural Networks, Portland OR, (2003).
28. G. Urcid, J.A. Nieves-V., and C.A. Reyes-G, Enhanced fuzzy autoassociative morphological memory for binary pattern recall, In Proceedings 4th IASTED Int. Conf. on Signal Processing, Pattern Recognition, and Applications, Innsbruck, Austria, (2007).
29. Roberto A. Vazquez and H. Sossa, Behavior of morphological associative memories with true-color image patterns, Neurocomputing, Vol. 73, N0. 1-3, 225-244, (2009).
30. M.E. Valle: A Class of Sparsely Connected Autoassociative Morphological Memories for Large Color Images, IEEE Transactions on Neural Networks, Vol. 20, No. 6, 1045 – 1050, (2009).
31. H. Sossa, R. Barron and Roberto A. Vazquez, New associative memories to recall real-valued patterns, Lecture Notes in Computer Science, Springer, Vol. 3287, 195-202, (2004).
32. Roberto A. Vazquez and H. Sossa, A new associative model with dynamical synapses, Neural Processing Letters, Vol. 28, No. 3, 189-207, (2008).
33. Roberto A. Vazquez and H. Sossa, A bidirectional heteroassociative memory for true-color patterns, Neural Processing Letters, Vol. 28, No. 3, 131-153, (2008).
34. G.X.Ritter, J.L.Diaz de Leon and P.Sussner, Morphological bidirectional associative memories, Neural Networks, Vol. 12, 851-867, (1999).
35. P. Sussner, Observations on morphological associative memories and the kernel method, Neurocomputing, Vol. 31, 167-183, (2000).
36. P. Sussner, Associative morphological memories based on variations of the kernel and dual kernel methods, Neural Networks, Vol. 16, No. 5, 625–632, (2003).
37. G.X. Ritter, G.Urcid, and L. Iancu, Reconstruction of noisy patterns using morphological associative memories, Journal of Mathematical Imaging and Vision, Vol. 19, 95-111, (2003).

38. [G. Urcid and G.X. Ritter, Kernel computation in morphological associative memories for grayscale image recollection, In Proceedings of 5th IASTED Int. Conf. on Signal and Image Processing, Honolulu, Hawaii, (2003).
39. G.X. Ritter and P. Gader, Fixed Points of Lattice Transforms and Lattice Associative Memories, in P. Hawkes (Ed.) Advances in Imaging and Electron Physics Elsevier, Vol. 144, San Diego, California, 2006.
40. G. Urcid, G.X. Ritter, L. Iancu, Kernel computation in morphological bidirectional associative memories, Progress in Pattern Recognition, Speech, and Image Analysis, Springer LNCS 2905 Proceedings, 563-570, (2003).
41. T. Ida, S. Ueda, M. Kashima, T. Fuchida and S. Murashima, On a method to decide kernel patterns of morphological associative memory, Systems and Computers in Japan, Vol. 33, No. 7, 85-94, (2002).
42. M. Hattori, A. Fukui and H. Ito, A fast method of constructing kernel patterns for morphological associative memory, In Proceedings of 9th International Conference on Neural Information Processing, 1058-1063, (Singapore 2002).
43. H. Hashiguchi and M. Hattori, Robust morphological associative memory using multiple kernel patterns, In Proceedings 10th International Conference on Neural Information Processing, (2003).

7. Appendix (Proof of Theorems)

Notation: In the following theorems an element of a conventional binary kernel vector is said to be a kernel point if its value is 1. An element of a new, "gray" kernel vector is said to be a kernel point if its value is > 0.

Proof of Theorem 3.1. To prove the theorem we start from the formula of computing the elements of **M**zz. According to equation (2.2) the elements m_{ij} of **M**zz are computed by

$$m_{ij} = \bigvee_{r=1}^{p} (z_i^r - z_j^r), \quad i = 1,...,n \quad j = 1,...,n \qquad (\mathbf{A.1})$$

where n is the length of kernel vector z.

In general the internal term $(z_i^r - z_j^r)$ can take the following values

(a) $z_i^r - z_j^r = 0$. This happens when for a specific *r* both z_i^r and z_j^r are kernel points or both are not kernel points

(b) $z_i^r - z_j^r = 1$ This happens when for a specific *r* z_i^r is a kernel points and z_j^r is not a kernel points

(c) $z_i^r - z_j^r = -1$ This happens when for a specific *r* z_i^r is not a kernel point and z_j^r is a kernel point

Taking now into account that (A.1) is computed by taking the maximum of all the internal terms $(z_i^r - z_j^r)$ for $r = 1 \ldots p$, we conclude the following.

(1) If index i does not correspond to any kernel point of any kernel vector then $z_i^r = 0, \forall r$. Consequently, for each j $(z_i^r - z_j^r), r = 1 \dots p$ takes the values 0 and –1. Therefore, the maximum of these values is 0 and therefore $m_{ij} = 0, \forall j$.

(2) In case index i equals the index of a kernel point in any kernel vector z^γ, then taking into account situation (b) above, $m_{ij} = 1, \quad \forall j$ except when $z_j^\gamma = 1$ (that is situation (a) above holds) and simultaneously $z_i^r = z_j^r = 0, \forall r \neq \gamma$, where $m_{ij} = 0$. This means that when the kernel points of a kernel vector do not coincide with the kernel points of another kernel vector (or otherwise kernel points appear only once) (which holds true due to condition (3.2)), then $m_{ij} = 0$ when both i and j correspond to kernel points.

Proof of Theorem 3.2. We take into account that in computing equation (2.5), that is,

$M_{zz} \oplus x = z$, the i^th element of z is computed using the elements of the i^th row of Mzz according to equation (2.4) as follows

$$z_i^r = \overset{n}{\underset{j=1}{\wedge}} (m_{ij} + x_j^r)\,,\; i = 1, \dots, n$$

We distinguish the following cases

1. Index of line, *i*, of **M**zz <u>does not correspond to any kernel point</u> of any kernel vector. In this case according to Theorem 3.1 $m_{ij} = 0, \forall j$, and therefore $\overset{n}{\underset{j=1}{\wedge}} (m_{ij} + x_j^r) = \overset{n}{\underset{j=1}{\wedge}} x_j^r = 0$. The last equation holds because there is the plausible assumption that at least one element of the input vector $\mathbf{x}^r$ is of zero value.
2. Index of line, *i*, of **M**zz <u>corresponds to a kernel point</u> of kernel vector z^r. In this case, according to Theorem 3.1, ($m_{ij} = 1, \forall j$), except when the column index (j) corresponds to a kernel point of the same kernel vector z^γ. In this case $m_{ij} = 0$ .Therefore, $\overset{n}{\underset{j=1}{\wedge}} (m_{ij} + x_j^r) =$

$$\wedge\Big(\underset{j \notin \ker npo\text{int}\, s}{\wedge} (1 + x_j^r)\,,\; \underset{j \in \ker npo\text{int}\, s}{\wedge} (0 + x_j^r) \Big) = \begin{cases} 0 & if\ \exists\, x_j^r = 0 : j \in \ker npo\text{int}\, s \\ 1 & otherwise \end{cases}$$

In the above notation an index, *j*, is said to belong to the kernel points if the element of the kernel vector having the same index, *j*, is a kernel point. The above equation proves that a kernel vector $\mathbf{z}^r$ can be recalled by equation (2.5) only if all the elements of the input vector $\mathbf{x}^r$, which correspond to kernel points, have the value 1. In other words, if even one element of the input vector, which corresponds to a kernel point, is hit by erosive noise then equation (2.5) recalls nothing and so does equation (2.9).

Proof of Theorem 4.1. To prove the theorem we start from the formula (A.1), which computes the elements of **M**zz. Now, the new kernel vectors are used.

In general the internal term $(z_i^r - z_j^r)$ can take values from the set {-1, -0.8, -0.7, -0.6, -0.4, -0.2, -0.1, 0,, 0.1, 0.2, 0.3, 0.4, 0.6, 0.8, 1} with the extreme values to appear as follows

(a) $z_i^r - z_j^r = 0$. This happens when for a specific *r* both z_i^r and z_j^r are kernel points of the same value (strength).

(b) $z_i^r - z_j^r = 1$ This happens when for a specific *r* z_i^r is one of the strongest kernel points (value 1) and z_j^r is not a kernel point (value 0)

(c) $z_i^r - z_j^r = -1$ This happens when for a specific *r* z_i^r is not a kernel points and z_j^r is one of the strongest kernel points

Taking now into account that (A.1) is computed by taking the maximum of all the internal terms $(z_i^r - z_j^r)$ for $r = 1 \ldots p$ we conclude the following.

(1) If index i does not correspond to any kernel point of any kernel vector then $z_i^r = 0, \forall r$. Consequently, for each j $(z_i^r - z_j^r), r = 1 \ldots p$ takes the values 0, -0.4, -0.7, -0.8, –1. Therefore, the maximum of these values is 0 and therefore $m_{ij} = 0, \forall j$.

(2) If index i corresponds to a kernel point with value $z_i^r = 1$ (frequency 1), then $m_{ij} \in I_1 = \{1, 0.6, 0.3, 0.2, 0\}$ with the value $m_{ij} = 1$ being the most frequent. It has to be noted that the values of m_{ij} are formed by $(z_i^r - z_j^r)$ of kernel r only. The values $(z_i^\gamma - z_j^\gamma), \gamma \neq r$ are always smaller since $z_i^\gamma = 0 \; \gamma \neq r$ (Frequency 1 means only $z_i^r = 1$) The values of m_{ij} are resolved as follows: $m_{ij} = 1, if \; z_j^r = 0$, $m_{ij} = 0.6, if \; z_j^r = 0.4$, $m_{ij} = 0.3, if \; z_j^r = 0.7$, $m_{ij} = 0.2, if \; z_j^r = 0.8$, $m_{ij} = 0, if \; z_j^r = 1$

(3) If index i corresponds to a kernel point with value $z_i^r = 0.8$ (frequency 2), then $m_{ij} \in I_2 = \{0.8, 0.4, 0.1, 0\}$ with the value $m_{ij} = 0.8$ being the most frequent. For the formation of m_{ij} only two kernel vectors are actually responsible, which have $z_i^r = 0.8$. We call this set of (two) kernels as forming ser and denote it as ***F***. The values of $(z_i^\gamma - z_j^\gamma), \gamma \notin F$ are always smaller since $z_i^\gamma = 0 \; \gamma \notin F$ (Frequency 2 means only two $z_i^r = 0.8, r \in F$) The values of m_{ij} are resolved as follows:

$m_{ij} = 0.8 \quad if \; z_j^\gamma = 0, \text{at least for one} \gamma \in F$

$m_{ij} = 0.4 \quad if \; z_j^\gamma = 0.4, \text{at least for one} \gamma \in F \text{ and for the remaining } z_j^\gamma \geq 0.4$

$m_{ij} = 0.1 \quad if\ z_j^\gamma = 0.7,\text{at least for one}\gamma \in F \text{ and for the remaining } z_j^\gamma \geq 0.7$

$m_{ij} = 0 \quad if\ i = j$ or

$m_{ij} = 0 \quad if\ z_j^\gamma = 0.8,\text{at least for one}\gamma \in F \text{ and for the remaining } z_j^\gamma \geq 0.8$

(4) If index i corresponds to a kernel point with value $z_i^r = 0.7$ (frequency 3), then $m_{ij} \in I_3 = \{0.7, 0.3,\ 0\}$ with the value $m_{ij} = 0.7$ being the most frequent. For the formation of m_{ij} only three kernel vectors are actually responsible, which have $z_i^r = 0.7$. We call this set of (three) kernels as forming ser and denote it as ***F***. The values of $(z_i^\gamma - z_j^\gamma), \gamma \notin F$ are always smaller since $z_i^\gamma = 0\ \gamma \notin F$ (Frequency 3 means only three $z_i^r = 0.7, r \in F$). The values of m_{ij} are resolved as follows:

$m_{ij} = 0.7 \quad if\ z_j^\gamma = 0,\text{at least for one}\gamma \in F$

$m_{ij} = 0.3 \quad if\ z_j^\gamma = 0.4,\text{at least for one}\gamma \in F \text{ and for the remaining } z_j^\gamma \geq 0.4$

$m_{ij} = 0 \quad if\ i = j$

$m_{ij} = 0 \quad if\ z_j^\gamma = 0.7,\text{at least for one}\gamma \in F \text{ and for the remaining } z_j^\gamma \geq 0.7$

(5) If index i corresponds to a kernel point with value $z_i^r = 0.4$ (frequency≥ 4), then $m_{ij} \in I_4 = \{0.4, 0\}$ with the value $m_{ij} = 0.4$ being the most frequent. For the formation of m_{ij} only the kernel vectors, which have $z_i^r = 0.4$ are actually responsible. We call this set of kernels as forming set and denote it as ***F***. The values of $(z_i^\gamma - z_j^\gamma), \gamma \notin F$ are always smaller since $z_i^\gamma = 0\ \gamma \notin F$ The values of m_{ij} are resolved as follows:

$m_{ij} = 0.4 \quad if\ z_j^\gamma = 0,\text{at least for one}\gamma \in F$

$m_{ij} = 0 \quad if\ i = j$

$m_{ij} = 0 \quad if\ z_j^\gamma = 0.4,\text{at least for one}\gamma \in F \text{ and for the remaining } z_j^\gamma \geq 0.4$

Proof of Theorem 4.2. Similarly to the proof of theorem 3.1, we take into account that in computing equation (2.5), that is, $\mathbf{M}_{zz} \oplus \mathbf{x} = \mathbf{z}$, the ith element of **z** is computed using the elements of the ith row of **M**zz according to equation (2.4) as follows

$$z_i^r = \bigwedge_{j=1}^{n} (m_{ij} + x_j^r)\ ,\ i = 1, ..., n$$

We distinguish the following cases

1. Index of line, *i*, of **M**zz <u>does not correspond to any kernel point</u> of any kernel vector. In this case $m_{ij} = 0, \forall j$, therefore $\bigwedge_{j=1}^{n} (m_{ij} + x_j^r) = \bigwedge_{j=1}^{n} x_j^r = 0$. The last equation holds because there is the plausible assumption that at least one element of the input vector $\mathbf{x}^r$ is of zero value.

2. Index of line, i, of **M**zz <u>corresponds to a kernel point</u> of any strength of kernel vector $\mathbf{z}^r$. In this case ($m_{ij} \in [0, m_{str}]$), where $m_{str} = \{1, 0.8, 0.7, 0.4\}$ depending on the strength of the kernel point with index i. Therefore, if $x_j^r = 1, \forall j$ being an index corresponding to a kernel point of $\mathbf{z}^r$ of any strength then $\bigwedge_{j=1}^{n}(m_{ij} + x_j^r) = m_{str} = z_i^r$. On the other hand if $\exists\, j \in \{\ker npo\operatorname{int} s\}^r : x_j^r = 0$ then $\bigwedge_{j=1}^{n}(m_{ij} + x_j^r) \in [0, m_{str}) \leq z_i^r$

In the above notation an index, j, is said to belong to the kernel points if the element of the kernel vector having the same index, j, is a kernel point. The above equation proves that a kernel vector $\mathbf{z}^r$ can be exactly recalled by equation (2.5) if all the elements of the input vector $\mathbf{x}^r$, which correspond to kernel points, have the value 1. In case, however, one or more points of the input vector, which correspond to kernel points, are hit by erosive noise then (2.5) does not entirely fail. In this case (2.5) recalls an eroded version of $\mathbf{z}^r$. Here however erosion of the value of one pixel means the replacement of its value by another smaller value. This fact is expressed by equation (4.2), which is repeated here

$$\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}} = \tilde{\mathbf{z}}$$

Where $\tilde{\mathbf{z}}$ denotes the eroded version of **z**. Since memory $\mathbf{W}_{ZX}$ is robust in erosive noise, equation (2.9) results in recalling the original, uncorrupted pattern according to equation (4.3) as follows

$$\mathbf{W}_{ZX} \otimes \left(\mathbf{M}_{ZZ} \oplus \tilde{\mathbf{x}}\right) = \mathbf{x}$$

Yiannis Boutalis received the diploma of Electrical Engineer in 1983 from Democritus University of Thrace (DUTH), Greece and the PhD degree in Electrical and Computer Engineering (topic Image Processing) in 1988 from the Computer Science Division of National Technical University of Athens, Greece. Since 1996, he serves as a faculty member, at the Department of Electrical and Computer Engineering , DUTH, Greece, where he is currently an Associate Professor and director of the Automatic Control Systems lab. Currently, he is also a Visiting Professor for research cooperation at Friedrich-Alexander University of Erlangen-Nuremberg, Germany, chair of Automatic Control. He served as an assistant visiting professor at University of Thessaly, Greece, and as a visiting professor in Air Defence Academy of General Staff of airforces of Greece. He also served as a researcher in the Institute of Language and Speech Processing (ILSP), Greece, and as a managing director of the R&D SME Ideatech S.A, Greece, specializing in pattern recognition and signal processing applications. His current research interests are focused in the development of Computational Intelligence techniques with applications in Control, Pattern Recognition, Signal and Image Processing Problems.

Received: November 14, 2009; Accepted: July 27, 2010.