Geometric Associative Memories and Their Applications to Pattern Classification

Benjamin Cruz, Ricardo Barron, and Humberto Sossa

Abstract Associative memories (AMs) were proposed as tools usually used in the restoration and classification of distorted patterns. Many interesting models have emerged in the last years with this aim. In this chapter a novel associative memory model (Geometric Associative Memory, GAM) based on Conformal Geometric Algebra (CGA) principles is described. At a low level, CGA provides a new coordinate-free framework for numeric processing in problem solving. The proposed model makes use of CGA and quadratic programming to store associations among patterns and their respective class. To classify an unknown pattern, an inner product is applied between it and the obtained GAM. Numerical and real examples to test the proposal are given. Formal conditions are also provided that assure the correct functioning of the proposal.

1 Introduction

Two main problems in pattern recognition are pattern classification and pattern restoration. One approach usually used to restore or classify desired patterns is by means of an associative memory. Lots of models of associative memories have emerged in the last 40 years, starting with the Lernmatrix of Steinbouch [25], then the Linear Associator of Anderson [1] and Kohonen [19], and the well-known model proposed by Hopfield, the Hopfield Memory [18]. For their operation, all of these models use the same kind of algebraic operations. Later there appeared the so-called *Morphological Associative Memories* (MAMs) [23] that are based on the mathematical morphology paradigm.

An associative memory **M** is a device whose main function is associating input patterns with output patterns. The notation for a pattern association between two

R. Barron (✉)
Center of Computing Research, Juan de Dios Batiz s/n Col. Nueva Industrial Vallejo Gustavo A. Madero, C.P. 07738, Mexico DF, Mexico
e-mail: rbarron@cic.ipn.mx

E. Bayro-Corrochano, G. Scheuermann (eds.), *Geometric Algebra Computing*,
DOI 10.1007/978-1-84996-108-0_11,

vectors x and y can be seen as an ordered pair (x, y). The whole set of all associations that form the associative memory is called *fundamental patterns set* or simply *fundamental set* (FS). Patterns belonging to the FS are called *fundamental patterns*.

Associations are completely stored in a weighted matrix. This matrix can be used to generate output patterns using the associated input patterns. This weighted matrix is the associative memory **M**. The process by which **M** is built is called *learning* or *training phase,* and the process through which an output pattern is generated using an input pattern is called *restoration* or *classification phase*.

When by means of an associative memory **M** a specific fundamental pattern is correctly classified, then **M** presents a *perfect recall* for that pattern [5]. An associative memory that presents perfect recall for all patterns of the FS is called a *perfect recalling memory*. When an associative memory **M** recovers or classifies patterns affected with noise correctly, it is said that **M** presents a *robust recall* or *robust classification*.

1.1 Classic Associative Memory Models

In 1961, a first work of associative memories was developed by Karl Steinbouch, the so-called *Die Lernmatrix*. This memory was proposed in 1961 and is capable of both pattern classification and pattern association [25]. In 1972, two papers by James A. Anderson [1] and Teuvo Kohonen [19] proposed the same model of associative memory, the so-called *Linear Associator* model for associative memories.

In the same year, a new associative memory device was presented by Kaouru Nakano, *the Associatron* [22]. This device was able to store entities represented by bit-patterns in a distributed form. It was able to restore complete patterns using a portion of them. Ten years later, John J. Hopfield introduced the so-called *Hopfield Memory* [18]. Hopfield considers this model as a physical system described by x that has locally stable points.

Almost 20 years after the Hopfield Memory, a new set of lattice algebra-based associative memories appeared, the *Morphological Associative Memories* (MAMs) [23]. Minima or maxima of sums are used for their operation, in contrast to the sums of products used in previous models. A variant of these MAMs are the *Alpha–Beta* Associative Memories ($\alpha\beta$) [27]. For these memories, two new operators are defined: α (alpha) and β (beta). These are detailed and discussed in [5].

There are two types of MAMs and $\alpha\beta$, the *min memories* that can cope with patterns altered with subtractive noise and the *max memories* that can cope with patterns altered with additive noise. However, contrary to what one might think, the performance of these models in the presence of patterns altered with mixed noise (most common in real situations) is too deficient [26].

In the literature, three ways to face the problem of mixed noise by means of an associative memory can be highlighted. In [26], a way to solve this problem by means of the so-called *kernels* is given. A kernel is a reduced version of an original pattern; the basic idea is to use two associative memories, one for recalling the kernel

using the distorted pattern and the other one for recalling the original pattern using the obtained kernel. Kernels for MAMs are however difficult to find, and if new patterns have to be added to the fundamental set, the kernels need to be computed again [11].

Another approximation is by means of the so-called *median memories* [24]. These memories use the well-known *median* operator widely used in signal processing instead of the maximum or minimum operators. Due the characteristics of the median operator, these memories can cope with mixed noise directly. However, the conditions for obtaining a perfect recall are difficult to achieve.

Finally in [11], it is shown how to solve the problem of the mixed noise by decomposing a pattern into parts (*sub-patterns*). This method is feasible due to the locality of the noise. Some parts of the pattern are less affected by noise than the other ones. However this method can consume a lot of computing time.

2 Basics of Conformal Geometric Algebra

In the XIX century many scientists worked on the development of algebraic systems. Among these, William K. Clifford (1845–1879) introduced *Geometric Algebras* (GA) called *Clifford Algebras* by mathematicians. They were completely described in his paper *Applications of Grassmann's Extensive Algebra* [10].

A geometric algebra is a priori coordinate-free [14]. In GA, geometric objects and operators over these objects are treated in a single algebra [13]. A special characteristic of GA is its geometric intuition. Another important feature is that the expressions in GA usually have low symbolic complexity [17].

The Conformal Geometric Algebra (CGA) is a (3,2)-dimensional coordinate-free theory and provides a conformal representation for 3D objects. Spheres and circles are both algebraic objects with a geometric meaning. In CGA, points, spheres, and planes are easily represented as *multivectors* [15]. A multivector is the outer product of various vectors [20].

CGA provides a great variety of basic geometric entities to compute with [17]. Intersections between lines, circles, planes, and spheres are directly generated. The creation of such elementary geometric objects simply occurs by algebraically joining a minimal number of points in the object subspace. The resulting multivector expressions completely encode in their components positions, orientations, and radii [13].

The main products of Geometric Algebra are the *geometric* or *Clifford product*, the *outer product* and the *inner product*. The inner product is used for the computation of angles and distances.

For notation purposes, Euclidean vectors will be noted by lowercase italic letters ($p, q, s, \ldots$), with the exception of the letters i, j, k, l, m, n that will be used to refer to indices. The corresponding conformal points will be noted by italic capital letters (P, Q, S, $\ldots$). A Euclidean matrix will be noted by bold capital letters ($\mathbf{M}$). To denote that an element belongs to an object (vector), a subscript will be used. To refer that an object belongs to a set of objects of the same type, a superscript will be used.

For example, if S is a sphere, then S_k is the kth component of it, and S^k is the kth sphere of a set of spheres. To denote scalars Greek letters will be used.

In particular, an original Euclidean point $p \in \mathbb{R}^n$ is extended to an $(n+2)$-dimensional conformal space [16] as

$$P = p + \frac{1}{2}(p)^2 e_\infty + e_0, \tag{1}$$

where p is a linear combination of the Euclidean base vectors. e_0 and e_∞ represent the Euclidean origin and the point at infinity, respectively, such that $e_0^2 = e_\infty = 0$ and $e_0 \cdot e_\infty = -1$ [13].

Equation (1) expresses a homogeneous relationship between both Euclidean and conformal domains since, given a scalar α and a conformal point P, αP and P both represent the same Euclidean point p. When the coefficient of e_0 is equal to 1, then P has a canonic representation. In this section, the algebra works in the conformal domain, while the geometric semantics lies in the Euclidean domain. In the same way, the sphere has the canonical form

$$S = C - \frac{1}{2}(\gamma)^2 e_\infty = c + \frac{1}{2}\big((c)^2 - (\gamma)^2\big)e_\infty + e_0, \tag{2}$$

where C is the central point in conformal form as defined in (2), where γ is the radius of the sphere. Also, a sphere can be easily obtained by four points that lie on it [13], as follows:

$$S = P^1 \wedge P^2 \wedge P^3 \wedge P^4. \tag{3}$$

In this case, it is said that (3) is a dual representation of (2). In the same way, a plane can be defined by three points that lie on it and the point at infinity [13] as follows:

$$T = P^1 \wedge P^2 \wedge P^3 \wedge e_\infty. \tag{4}$$

From (3) and (4) we can see that a plane is a sphere that passes through the point at infinity [15].

A distance measure between two conformal points P and Q can be defined with the help of the inner product [16] as follows:

$$P \cdot Q = p \cdot q - \frac{1}{2}(p)^2 - \frac{1}{2}(q)^2 = -\frac{1}{2}(p-q)^2 \quad \Longleftrightarrow \quad (p-q)^2 = -2(P \cdot Q), \tag{5}$$

resulting in the square of the Euclidean distance. In the same way, a distance measure between one conformal point P and a sphere S can be defined with the help of the inner product [16] as

$$P \cdot S = p \cdot s - \frac{1}{2}\left((s)^2 - \frac{1}{2}(\gamma)^2\right) - \frac{1}{2}(p)^2 = \frac{1}{2}\big((\gamma)^2 - (s-p)^2\big) \tag{6}$$

or in a simplified form as

$$2(P \cdot S) = (\gamma)^2 - (s-p)^2. \tag{7}$$

Based on (7), if $P \cdot S > 0$, then p is inside of the sphere; if $P \cdot S < 0$, then p is outside of the sphere; and if $P \cdot S = 0$, then p is on the sphere. In pattern classification, therefore, if a CGA spherical neighborhood is used, the inner product makes it possible to know when a pattern is inside or outside of the neighborhood.

3 Geometric Algebra Classification Models

While classic models all use the same kind of algebraic operations, MAMs make use of the mathematical morphological paradigm (min and max operations). Next, a description of how geometric algebra operations can be used to store the association among a subset of patterns and their corresponding index classes is shown. It is worth mentioning that the idea of using geometric algebra in classification is not new.

In [2], a *Quaternionic Multilayer Perceptron* (QMLP) in *Quaternion Algebra* is developed. A QMLP is a Multilayer Perceptron (MLP) in which both the weights of connections and the biases are *quaternions*, as well as input and output signals. With the help of the QMLP, the number of parameters of an MLP needed to perform a multidimensional series prediction decreases [2].

A new set of *Geometric Algebra Neural Networks* was introduced in [7]. Real, complex, and quaternionic neural networks can be further generalized in the geometric algebra framework [7]. The weights, the activation functions, and the outputs are represented by multivectors. The geometric product is used to operate these multivectors.

Bayro and Vallejo extended the McCulloch–Pitts neuron [21] to the *geometric neuron* by substituting the scalar product with the Clifford or geometric product. A *feed-forward geometric neural network* is then built, where the inner vector product is extended to the geometric product, and the activation functions are a generalization of the function proposed in [2].

In [7], a new approach is also proposed, the *Support Multivector Machines*. The basic idea is generating neural networks using *Support Vector Machines* (SVM) for the processing of multivectors in geometric algebra. The use of geometric algebra in SVMs offers both new tools and new understanding of SVMs for multidimensional learning [7].

In [4], a special higher-order neuron, the so-called *Hyper-sphere neuron* was introduced. A hypersphere neuron may be implemented as a *perceptron* with two bias inputs. In that work, a perceptron based on conformal geometric algebra principles was described. An iterative hypersphere neuron was also proposed. The decision surface of the perceptron presented is not a hyperplane but a hypersphere. An advantage of this representation is that only a standard scalar product needs to be evaluated in order to decide whether an input vector is inside or outside a hypersphere.

So-called *Clifford Neurons* are introduced in [8], the weights and the threshold of a classical neuron are replaced by multivectors, and the real multiplication is replaced by the Clifford product. Two types of Clifford Neurons are described, the

Basic Clifford Neuron, which can be viewed as a *Linear Associator*, and the *Spinnor Clifford Neuron*. Both types of neurons can be the starting point to fast second-order training methods for Clifford and Spinnor MLPs in the future [9].

In the following section, a new idea that has never been used before to develop an associative memory model based on the conformal geometric algebra principles will be explained.

4 Geometric Associative Memories

Definition 1 When two sets of points in $\mathbb{R}^n$ can be completely separated by a hyperplane, they are said to be *linearly separable*.

Linear separation is important for pattern classification; that hyperplane works as a *decision surface*; it can be used for deciding to which class an unclassified will be assigned by finding which side of the hyperplane the pattern is located. Many classification models (i.e., *neural networks*) have better results when the patterns are linearly separable.

In the same way, the next definition can be enunciated.

Definition 2 When two sets of points in $\mathbb{R}^n$ can be completely separated by a hypersphere, they are said to be *spherically separable*.

In this case the decision is made by finding if the pattern is located *inside* or *outside* of the sphere. Thus, the following theorem can be established:

Theorem 1 *Any two sets of linearly separable points in* $\mathbb{R}^n$ *are spherically separable too*.

Proof Consider any two sets of linearly separable points in $\mathbb{R}^n$. From (3) and (4) the hyper-plane that separate them is a sphere that passes through the point at infinity. Then, there is a sphere that separates them. Therefore, the two sets are spherically separable. □

It is worth mentioning that Theorem 1 does not guarantee that two sets of spherically separable points are linearly separable.

Spherical neighborhoods are usually difficult to handle, but in the context of geometric algebra, this is not a problem. In [4], a method for building a hypersphere neuron in an iterative way is described. In the following, three one-shot methods to build a sphere are explained.

4.1 Creating Spheres

The goal of a *Geometric Associative Memory* (GAM) is to classify a pattern as belonging to a specific class if and only if the pattern is inside of the support region

(hypersphere) of that class. Building spherical neighborhoods implies to find the center of each sphere and then a suitable radius. Some procedures to achieve this with CGA have been reported in the literature. Three of them are described in the following.

In [12], a *one-shot* method is described, where given a set of points $\{p^i, i = 1, \ldots, m\}$, a spherical neighborhood is constructed. The center is computed as

$$c = \sum_{i=1}^{n} p^i / m. \tag{8}$$

In other words, the center is the average among all the patterns of each class. To compute the radius, the following expression is used:

$$\gamma = \min\left[\left(C \cdot P^i\right),\ i = 1, \ldots, m\right], \tag{9}$$

where C and P^i are the conformal representations of c and p^i, respectively. This procedure guarantees that all the patterns in the respective class will be inside of a sphere of class. A disadvantage of this procedure is its high computational cost.

In [17], a second approach is presented: planes or spheres are fitted into point sets by using a least squares approach. The algorithm uses the distance measure between points and spheres with the help of the inner product. It performs a least squares approach to minimize the square of the distances between a point and a sphere.

With the help of this approach, spherical neighborhoods that fit a set of patterns can be created. In this case, the spheres work as attractors with their corresponding class patterns as centers. The drawback for this method is that generally some points might appear located outside the resulting sphere.

In [17], bounding a sphere of cloud points is presented. The case for one and two points is described, and the case of expanding an existing bounding sphere when adding more points (or spheres) is presented. Using both cases, bounding a sphere of a set of points can be easily performed.

In [6], a method to construct a smallest enclosing hypersphere using quadratic programming and conformal geometric algebra was presented. The method combines characteristics of the first two methods, i.e., fit an optimal sphere that contains all the points.

The above methods can be used to build spherical neighborhoods for a specific class by using the points (patterns) of that class. But they do not take into account the patterns of other classes or the separation between classes.

In the following a new approach will be presented. It is inspired by ideas from [12]. The proposal can be used to find an optimal spherical neighborhood taking into account the patterns of the class that the sphere covers and the patterns of the other classes.

Let $P = \{P^i \cup P^j \mid i = 1, \ldots, l,\ j = l + 1, \ldots, m\}$ be a set of spherically separable points in $\mathbb{R}^n$, where $\{P^i \mid i = 1, \ldots, l\}$ are points belonging to one class, and $\{P^j \mid j = l + 1, \ldots, m\}$, are points belong to the other class. The problem is to find

an optimal sphere S with the least square error, such that P^i are inside S and P^j are outside of it or, in other words, to solve

$$\min_S \sum_{i=1}^{m} (P^i \cdot S), \tag{10}$$

subject to (11) for points inside of the sphere and (12) for points outside of it

$$P^i \cdot S \geq 0, \quad i = 1, \ldots, l, \tag{11}$$

$$P^j \cdot S < 0, \quad j = l+1, \ldots, l. \tag{12}$$

In order to find an optimal solution, a quadratic programming algorithm must be applied. Therefore this problem must be changed into a Euclidean problem of optimization, starting from (10) and considering that all the spheres are in canonical form such that the term S_{n+2} of them can be omitted:

$$\begin{aligned}\sum_{i=1}^{m} (P^i \cdot S)^2 &= \sum_{i=1}^{m} \left(p^i \cdot s - S_{n+1} - \frac{1}{2}(p^i)^2 \right)^2 \\ &= \sum_{i=1}^{m} \left([p^i \cdot s - S_{n+1}] - \frac{1}{2}(p^i)^2 \right)^2 \\ &= \sum_{i=1}^{m} \left([p^i \cdot s - S_{n+1}]^2 - [p^i \cdot s - S_{n+1}](p^i)^2 + \frac{1}{4}(p^i)^4 \right), \end{aligned} \tag{13}$$

where $S_{n+1} = \frac{1}{2}(p^i - s)^2$. Thus,

$$\sum_{i=1}^{m} (P^i \cdot S)^2 = \sum_{i=1}^{m} (p^i \cdot s - S_{n+1})^2 + \sum_{i=1}^{m} (-p^i \cdot s + S_{n+1})(p^i)^2 + \frac{1}{4} \sum_{i=1}^{m} (p^i)^4. \tag{14}$$

Here, the third term is irrelevant because it does not depend on parameter S, and thus it can be omitted. Without losing generality it can be rewritten in Euclidean notation as in (15), where $\mathbf{W}$ and $\mathbf{F}$ are matrices whose components are (16) and (17) respectively, and $x = [S_1, \ldots, S_{n+1}]$.

$$\sum_{i=1}^{m} (P^i \cdot S)^2 = \sum_{i=1}^{m} \left(\sum_{k=1}^{n+1} \mathbf{W}_{i,k} x_k \right) + \sum_{i=1}^{m} \left(\sum_{k=1}^{n+1} \mathbf{F}_{i,k} x_k \right), \tag{15}$$

$$\mathbf{W}_{i,k} = \begin{cases} p_k^i & \text{for } k = 1, \ldots, n, \\ -1 & \text{otherwise}, \end{cases} \tag{16}$$

$$\mathbf{F}_{i,k} = \begin{cases} -(p_k^i)(p^i)^2 & \text{for } k = 1, \ldots, n, \\ (p^i)^2 & \text{otherwise.} \end{cases} \tag{17}$$

Let $w_i = [\mathbf{W}_{i,1}, \ldots, \mathbf{W}_{i,n+1}]$; then for the first term of the right side of expression (15) and considering that w_i^t is the transpose of the vector w_i, we have

$$\begin{aligned} \sum_{i=1}^{m}\left(\sum_{k=1}^{n+1}\mathbf{W}_i^t S_k\right)^2 &= \sum_{i=1}^{m}\left(w_i^t x\right)^2 \\ &= \left(w_1^t x + \cdots + w_m^t x\right)^2 \\ &= w_1^t x\, w_1^t x + \cdots + w_m^t x\, w_m^t x \\ &= \left(w_1^t x\, w_1^t + \cdots + w_m^t x\, w_m^t\right)x \\ &= \left(\left[x^t w_1 + \cdots + x^t w_m\right]\mathbf{W}\right)x \\ &= x^t \mathbf{W}^t \mathbf{W} x. \end{aligned} \tag{18}$$

By considering that $\mathbf{H} = \mathbf{W}^t\mathbf{W}$ we have

$$\sum_{i=1}^{m}\left(\sum_{k=1}^{n+1}\mathbf{W}_i^t S_k\right)^2 = x^t \mathbf{H} x. \tag{19}$$

For the second term of the right side of the expression (15), let $y_k = \sum_{i=1}^{m}\mathbf{F}_{i,k}$:

$$\begin{aligned} \sum_{i=1}^{m}\left(\sum_{k=1}^{n+1}\mathbf{F}_{i,k}x_k\right) &= \sum_{k=1}^{n+1}\left(\sum_{i=1}^{m}\mathbf{F}_{i,k}x_k\right) \\ &= \sum_{k=1}^{n+1}(y_k x_k). \end{aligned} \tag{20}$$

Let $y = [y_1, \ldots, y_{n+1}]$; then

$$\sum_{i=1}^{m}\left(\sum_{k=1}^{n+1}\mathbf{F}_{i,k}S_k\right) = y^t x. \tag{21}$$

With the help of (19) and (21), expression (10) can be converted into Euclidean matrix notation as follows:

$$x^t \mathbf{H} x + y^t x. \tag{22}$$

The constraint (11) for points inside of the sphere will change into

$$\begin{aligned} P^i \cdot S &\geq 0, \\ p^i \cdot s - S_{n+1} - \frac{1}{2}\left(p^i\right)^2 &\geq 0, \\ p^i \cdot s - S_{n+1} &\geq \frac{1}{2}\left(p^i\right)^2, \\ \sum_{k=1}^{n+1}(-\mathbf{W}_{k,i})S_k &\leq -\frac{1}{2}\left(p^i\right)^2, \end{aligned} \tag{23}$$

where $\mathbf{W}_{i,k}$ was defined in (16). Equation (23) can be rewritten as

$$-\mathbf{W}x \leq -\frac{1}{2}p_i^2, \tag{24}$$

where $x = [S_1, \ldots, S_{n+1}]$, and the constraint (12) for points outside of the sphere will be

$$\begin{aligned} P^i \cdot S &< 0, \\ p^i \cdot s - S_{n+1} - \frac{1}{2}\left(p^i\right)^2 &< 0, \\ p^i \cdot s - S_{n+1} &< \frac{1}{2}\left(p^i\right)^2, \\ \sum_{k=1}^{n+1}(-\mathbf{W}_{i,k})S_k &< \frac{1}{2}\left(p^i\right)^2, \end{aligned} \tag{25}$$

$$\mathbf{W}x < \frac{1}{2}p_i^2. \tag{26}$$

Let $\mathbf{A} = [-\mathbf{W}, \mathbf{W}]$, and b be a vector whose ith component is $-\frac{1}{2}(p^i)^2$ for $i = 1, \ldots, l$ and $\frac{1}{2}(p^i)^2 - \varepsilon$ for $i = l + m, \ldots, m$. The ε is a smallest positive quantity used to change the "$<$" of (26) to be the "$\leq$". Then the constraints (11) and (12) can be converted to a Euclidean matrix notation,

$$\mathbf{A}x \leq b. \tag{27}$$

Finally, the problem of solving (10) has changed to a classical optimization problem with constraints

$$\begin{aligned} &\min_x \left(x^t \mathbf{W} x + y^t x\right), \\ &\text{s.t.} \quad \mathbf{A}x \leq b. \end{aligned} \tag{28}$$

Thus, the optimal sphere S is given by solving (28), where $S_k = x_k$ for $k = 1, \ldots, n + 1$ and $S_{n+2} = 1$. It is clear that by including in the restrictions all the

points that stay out of the sphere, the solution S results in a separation surface that allows differentiating between two classes (i.e., inner and outer points).

The procedure works perfectly for two spherically separable classes. In a multiclass situation, the procedure is similar. In this case, the subset $\{P^i, i = 1, \ldots, l\}$ will be all patterns for class k, and $\{P^j,\ j = l+1, \ldots, m\}$ will be all patterns for the other classes. The kth sphere S_k is then found by solving (28). The same procedure must be applied for all the other classes.

4.2 Pattern Learning and Classification

The learning phase of an associative memory consists on storing associations among input patterns and output patterns. In the case of Geometric Associative Memories (GAMs), the learning phase consists on creating the spherical neighborhoods for each class. A GAM $\mathbf{M}$ is thus a matrix of size $m \times (n+2)$ (m is the number of classes, and n is the dimension of the space). The kth row are the components of the kth sphere, as it can be seen in (29). C^k and γ^k are the center and radius of the kth sphere, respectively.

$$\mathbf{M} = \begin{bmatrix} S^1 \\ S^2 \\ \vdots \\ S^m \end{bmatrix}. \tag{29}$$

Classification of a pattern can be performed by using the idea from [4], an inner product between the conformal notation of an unclassified pattern $x \in \mathbb{R}^n$ and $\mathbf{M}$ must be applied to getting, as a result, a vector u. This vector will contain all the inner products between the unclassified pattern and the spheres of class. This vector is given as

$$u_k = \mathbf{M}_k \cdot X = S^k \cdot X. \tag{30}$$

When X is inside a sphere, (30) returns a positive number (or zero) and a negative number otherwise. Note that the classification phase is independent of the training phase.

In some cases (mainly noisy patterns), X could be inside two or more spheres or could be outside of all spheres. To decide to which sphere a given pattern belongs, the following remapping must be used:

$$v_k = \begin{cases} -\infty & \text{if } u_k < 0, \\ u_k - (r^k)^2 & \text{otherwise.} \end{cases} \tag{31}$$

This must be done for $k = 1, \ldots, m$. The class identifier j can be obtained as follows:

$$j = \arg\max_k [v_k \mid k = 1, \ldots, m]. \tag{32}$$

As it can be seen, when x is outside of the k-sphere, expression (31) returns $-\infty$, and when x is inside of the k-sphere, the same expression returns the distance (with minus sign) between x and c^k. By doing this, x will be classified by a class sphere covering its conformal representation; with the help of expression (32), x will be classified by the sphere with center closest to it.

Note that, in some cases, $v_k = -\infty$ for $k = 1, \ldots, m$, that is, x is outside of all the spheres. Then, when expression (32) is applied, it cannot return a value. At this point, two choices can be taken. First, x does not belong to any class. Second, using expression (32) directly on $u_k - (r^k)^2$. In this case, the GAM works as a minimum distance classifier, but the use of neighborhoods is relegated.

The classification phase is independent of the training phase. The proposed method works perfectly when the classes are spherically separable.

4.3 Conditions for Perfect Classification

In associative memories, when an associative memory $\mathbf{M}$ recovers or classifies the fundamental set correctly, it is said that $\mathbf{M}$ presents perfect recall or perfect classification. Let $\mathbf{M}$ be a trained GAM, as it was presented in the previous section.

Theorem 2 *Assume m sets of spherically separable classes in $\mathbb{R}^n$, and let $\boldsymbol{M}$ be a trained GAM for those classes. Then $\boldsymbol{M}$ presents perfect classification.*

Proof Let k be an index class whose sphere S^k is the kth component in $\mathbf{M}$, and let p be a fundamental pattern of class k, and let j be an index $j = 1, \ldots, m$ such that $j \neq k$, S^k having being obtained using expression (10). Then according to condition (11), $P \cdot S^k \geq 0$ because it is a pattern of class k and $P \cdot S^j \geq 0$ for some $j \neq k$.

When (31) is applied to P, v has a positive number or zero in position k and $-\infty$ in the other positions. Therefore, (32) returns k. This covers all patterns in all classes. □

4.4 Conditions for Robust Classification

In associative memories, when an associative memory $\mathbf{M}$ recovers or classifies correctly patterns affected with noise, it is said that $\mathbf{M}$ presents *robust recall* or *robust classification*. The robustness in a GAM depends on the size of its radius; the GAM can classify any noise pattern as belonging to its class when that pattern is located inside of it. Patterns located outside of a specific sphere (i.e., some noise patterns) will not be classified as belonging to that class sphere.

The quantity of noise that can admit a fundamental pattern depends on the position of it with respect to the center and the border of the sphere. Patterns nearest to the center can admit more quantity of noise than patterns located near of the border.

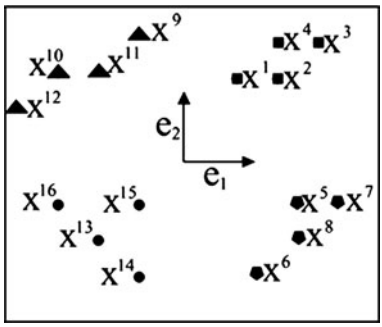

Fig. 1 Example 1, sets of patterns. *Square*, *pentagon*, *triangle*, and *circle* shapes are patterns belonging to classes 1, 2, 3, and 4, respectively

5 Numerical Examples

In this section, two illustrative examples for the problem of classifying sets of patterns are presented. For simplicity, in order to clarify the results, a 2D and 3D Euclidean space for the geometric problem are used.

In both cases, a function of the Optimization Toolbox of MatLab was used to solve the minimization problem. Function *quadprog* solves quadratic programming problems. It finds an initial feasible solution by first solving a linear programming problem.

Example 1 The following are linearly separable patterns set in $\mathbb{R}^n$:

$$\begin{aligned}
&\text{Class 1} \quad x^1 = [1 \quad 1], \quad x^2 = [2 \quad 1], \quad x^3 = [3 \quad 2], \quad x^4 = [2 \quad 2],\\
&\text{Class 2} \quad x^5 = [2 \quad -1], \quad x^6 = [1 \quad -3], \quad x^7 = [3 \quad -1],\\
&\qquad\quad x^8 = [2 \quad -2],\\
&\text{Class 3} \quad x^9 = [-1 \quad 3], \quad x^{10} = [-3], \quad x^{11} = [-2 \quad 2],\\
&\qquad\quad x^{12} = [-4 \quad 1],\\
&\text{Class 4} \quad x^{13} = [-2 \quad -2], \quad x^{14} = [-1 \quad -3], \quad x^{15} = [-1 \quad -1],\\
&\qquad\quad x^{16} = [-3 \quad -1].
\end{aligned} \tag{33}$$

Figure 1 shows a graphical representation of these patterns. By using (10), the corresponding spheres (in this case, circles) are obtained. Their respective centers and radii are

$$\begin{aligned}
c^1 &= [0.65 \quad 1.11], \quad \gamma^1 = 2.51,\\
c^2 &= [1 \quad -1], \quad \gamma^2 = 2,\\
c^3 &= [-1.5 \quad 1.5], \quad \gamma^3 = 2.55,\\
c^4 &= [-0.8 \quad -0.6], \quad \gamma^4 = 2.41.
\end{aligned} \tag{34}$$

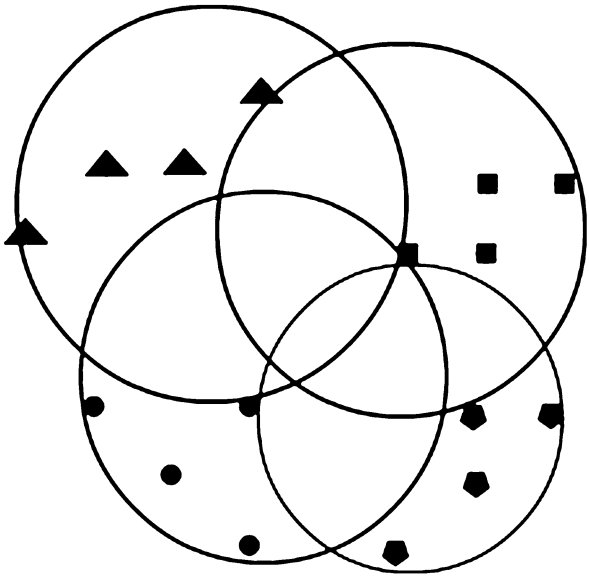

Fig. 2 Circles obtained using the method of Sect. 4.2. They function as separation surfaces

The value used for ε in this example was 0.0001. Finally, the GAM $\mathbf{M}$ is

$$\mathbf{M} = \begin{bmatrix} S^1 = C^1 - (\frac{1}{2})(\gamma^1)^2 e_\infty \\ S^2 = C^2 - (\frac{1}{2})(\gamma^1)^2 e_\infty \\ S^3 = C^3 - (\frac{1}{2})(\gamma^1)^3 e_\infty \\ S^4 = C^4 - (\frac{1}{2})(\gamma^1)^4 e_\infty \end{bmatrix}$$
$$= \begin{bmatrix} S^1 = 0.65e_1 + 1.11e_2 - 2.31e_\infty + e_0 \\ S^2 = e_1 - e_2 - e_\infty + e_0 \\ S^3 = -1.5e_1 + 1.5e_2 - e_\infty + e_0 \\ S^4 = -0.8e_1 - 0.6e_2 - 2.4e_\infty + e_0 \end{bmatrix}. \tag{35}$$

In Fig. 2, the corresponding circles are presented. Note that the circle of class 1 is optimal, because if it grows a bit more, then X^{11} could fall inside of it, and if it decreases a bit more, X^3 could fall outside of it. The same happens with the other circles. Now, let the following set of noisy patterns to be classified:

$$\begin{aligned} \tilde{x}^1 &= x^1 + [0 \quad 2] = [1 \quad 3], \\ \tilde{x}^8 &= x^8 + [-2 \quad 0] = [0 \quad -2], \\ \tilde{x}^9 &= x^9 + [-1 \quad 1] = [-2 \quad 4], \\ \tilde{x}^{15} &= x^{15} + [0 \quad -1] = [-1 \quad -2]; \end{aligned} \tag{36}$$

they have been affected with noise. If (30) is applied, the result is

$$u^1 = \begin{bmatrix} 1.31 \\ -6 \\ -1 \\ -5.2 \end{bmatrix}, \qquad u^8 = \begin{bmatrix} -1.92 \\ 1 \\ -4 \\ 1.6 \end{bmatrix},$$
$$u^9 = \begin{bmatrix} -4.53 \\ -15 \\ 0 \\ -8.4 \end{bmatrix}, \qquad u^{15} = \begin{bmatrix} 1.31 \\ -6 \\ -1 \\ -5.2 \end{bmatrix}. \tag{37}$$

The next step is to apply expression (31). By doing this, the following expressions are obtained:

$$v^1 = \begin{bmatrix} -1.83 \\ -\infty \\ -\infty \\ -\infty \end{bmatrix}, \qquad v^8 = \begin{bmatrix} -\infty \\ -1 \\ -\infty \\ -\infty \end{bmatrix},$$
$$v^9 = \begin{bmatrix} -\infty \\ -\infty \\ -3.25 \\ -\infty \end{bmatrix}, \qquad v^{15} = \begin{bmatrix} -\infty \\ -\infty \\ -\infty \\ -1 \end{bmatrix}. \tag{38}$$

The class index is then obtained by means of (32) for $\tilde{x}^1$, $\tilde{x}^8$, $\tilde{x}^9$, and $\tilde{x}^{15}$, $j = 1, 2, 3, 4$, respectively. Note that in these cases, classification is correct even when they are affected with noise and although $\tilde{x}^8$ falls inside of two spheres. However, consider the following pattern:

$$\tilde{x}^3 = x^3 + \begin{bmatrix} 0.05 & 0 \end{bmatrix} = \begin{bmatrix} 3.05 & 2 \end{bmatrix}. \tag{39}$$

Note that, in this case, the noise is minimum, but when expressions (30) and (31) are applied,

$$u^3 = \begin{bmatrix} -0.12 \\ -4.6 \\ -7.23 \\ -7.89 \end{bmatrix}, \qquad v^3 = \begin{bmatrix} -\infty \\ -\infty \\ -\infty \\ -\infty \end{bmatrix}, \tag{40}$$

and in this case, expression (32) cannot classify it, due to that it is located outside of all spheres.

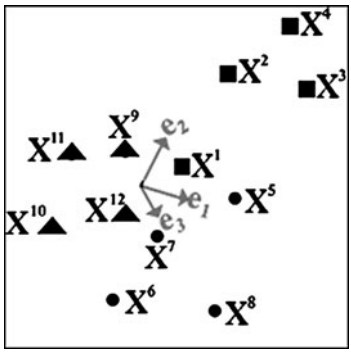

Fig. 3 Example 2, sets of patterns in 3D. *Square*, *circle*, and *triangle* shapes are patterns belonging to classes 1, 2, and 3 respectively

Example 2 Consider the following set of nonlinearly separable patterns in $\mathbb{R}^3$:

$$\begin{aligned}
\text{Class 1} \quad & x^1 = [0.5 \quad 0.5 \quad 0.5], \qquad x^2 = [-0.5 \quad 2.5 \quad -1], \\
& x^3 = [2.5 \quad 2 \quad -1], \qquad x^4 = [1 \quad 3 \quad 0], \\
\text{Class 2} \quad & x^5 = [2 \quad 0 \quad -0.5], \qquad x^6 = [0.5 \quad -2 \quad 0], \\
& x^7 = [2 \quad -1.5 \quad 0.5], \qquad x^8 = [1 \quad -1 \quad -0.5], \\
\text{Class 3} \quad & x^9 = [-0.5 \quad 0.5 \quad 0], \qquad x^{10} = [-1 \quad -1 \quad -0.5], \\
& x^{11} = [-1 \quad 0 \quad -0.5], \qquad x^{12} = [0 \quad -0.5 \quad 0].
\end{aligned} \tag{41}$$

Figure 3 shows a graphical representation of these patterns.

By using (10), the corresponding class spheres were obtained. Their respective centers and radii are

$$\begin{aligned}
c^1 &= [1.19 \quad 1.60 \quad 0.61], \qquad \gamma^1 = 2.11, \\
c^2 &= [1.75 \quad -0.75 \quad 1.72], \qquad \gamma^2 = 2.47, \\
c^3 &= [-0.29 \quad -0.20 \quad 0.05], \qquad \gamma^3 = 1.06;
\end{aligned} \tag{42}$$

the value used for ε in this example was 0.0001. Finally, the GAM $\mathbf{M}$ is

$$\mathbf{M} = \begin{bmatrix} S^1 = 1.19e_1 + 1.6e_2 + 0.61e_3 - 0.04e_\infty + e_0 \\ S^2 = 1.75e_1 - 0.75e_2 + 1.72e_3 + 0.25e_\infty + e_0 \\ S^3 = -0.29e_1 - 0.2e_2 + 0.05e_3 - 0.5e_\infty + e_0 \end{bmatrix}. \tag{43}$$

In Fig. 4, the corresponding spheres are presented. As in the previous example, the spheres are optimal.

Now, let the following set of noisy patterns to be classified:

$$\begin{aligned}
\tilde{x}^1 &= x^1 + [0.5 \quad -0.5 \quad -1] = [0.5 \quad 0.5 \quad 0], \\
\tilde{x}^7 &= x^7 + [-1 \quad 0.5 \quad -0.5] = [1 \quad -1 \quad 0],
\end{aligned} \tag{44}$$

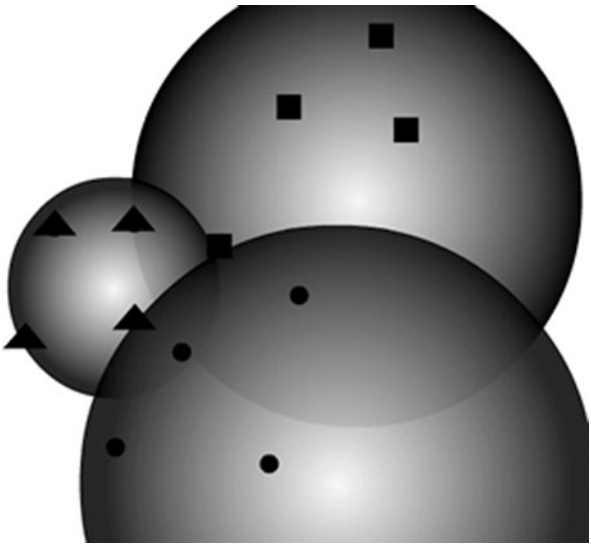

Fig. 4 Spheres obtained using the method of Sect. 4.2. They function as separation surfaces

$$\tilde{x}^{10} = x^{10} + \begin{bmatrix} 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}.$$

If (30) is applied, the result is

$$u^1 = \begin{bmatrix} 1.25 \\ -0.27 \\ -0.65 \end{bmatrix}, \qquad u^7 = \begin{bmatrix} -1.36 \\ 1.25 \\ -0.59 \end{bmatrix}, \qquad u^{10} = \begin{bmatrix} 0.04 \\ -0.25 \\ 0.5 \end{bmatrix}. \tag{45}$$

The next step is to apply expression (31). By doing this, the following expressions are obtained:

$$v^1 = \begin{bmatrix} -0.97 \\ -\infty \\ -\infty \end{bmatrix}, \qquad v^7 = \begin{bmatrix} -\infty \\ -1.79 \\ -\infty \end{bmatrix}, \qquad v^{10} = \begin{bmatrix} -2.19 \\ -\infty \\ -0.06 \end{bmatrix}. \tag{46}$$

The class index is then obtained by means of (32) for $\tilde{x}^1$, $\tilde{x}^7$, and $\tilde{x}^{10}$, $j = 1, 2, 3$, respectively. As in Example 1, the classification is correct for some patterns altered with noise.

As can be observed, although GAMs can classify some patterns altered with noise, they are very sensitive in some other cases, mainly in the case of patterns located at the border of the sphere. This problem might be fixed by adding a positive value to restriction (24) and a negative value to restriction (26).

6 Real Examples

To test the potential of the proposal, two trials with real data were performed. In the first trial, the best known database used by the pattern recognition community

Table 1 Results of the classification phase

Data Set	Patterns used in FS	Classification of FS	Classification of TS
Iris Plant	15	100%	90%
Iris Plant	25	100%	93.3%
Iris Plant	30	100%	94.6%
Wines	15	90.6%	71.9%
Wines	25	91.1%	80.9%
Wines	30	97.7%	84.8%

was adopted, the *Iris Plant Data Base*. This data set contains three classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other two; the latter two classes are NOT linearly separable from each other. Each instance has four numeric, predictive attributes (sepal length, sepal width, petal length, and petal width).

For the second trial, the *Wine Recognition Data Base* was used. These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivations. The analysis determined the quantities of 13 constituents found in each of the three types of wines. This data set contains three classes of 59, 48, and 71 instances, respectively, where each class refers to a type of wine. Each instance has 13 continuous attributes.

Both data bases were obtained from [3]. As in the previous section, the function *quadprog* of the optimization toolbox of Matlab was used in order to solve the quadratic programming problem.

15, 25, and 30 instances, respectively, were used to form the Fundamental Set (FS) for each type of problem. All of the instances were used to form the Test Set (TS). In the training phase, patterns of each FS were used to build the GAMs. In the classification phase, patterns of the TS were classified; the results are shown in Table 1. The first column shows the number of patterns used for the learning phase, and the second and third columns show the percentages of the patterns correctly classified using the FS and the TS, respectively. In the case of the GAM from Iris Plant data set, perfect recall was obtained because the patterns used for the FS are spherically separable. In the case of the GAM from the Wine data set, perfect recall is not obtained because the patterns are not spherically separable. Thus some patterns of a specific class may fall outside their respective class sphere.

It can be observed that classification rate of the patterns of the TS (for both data sets) increases when the number of patterns used in the FS increases. When the FS has more patterns, the corresponding spheres grow, and then more patterns of the TS could fall inside of the class spheres.

7 Conclusions and Future Work

Geometric Algebra allows one to model situations and to formulate problems in terms of high-level symbolic expressions. Nevertheless, it is possible to achieve an

implementation working in an elementary coordinate system. Of course, in some cases, it is possible to find a solution by purely handling symbolic expressions, but this is rare for realistic problems.

In this work, a new associative memory model based on Conformal Geometric Algebra has been described, the Geometric Associative Memory (GAM). The training phase is done by finding an optimal sphere with quadratic programming. GAMs can perfectly operate when the classes are spherically separable.

For classification purposes, an inner product between the unclassified pattern and the GAM was applied. Then a minimum function is used to obtain the index class.

Numerical and real examples were given to show the potential of the proposal. As shown, the method can operate both with linearly and nonlinearly separable patterns. The proposed model can also cope with distorted patterns.

Patterns located on the border of the sphere might not be well classified. At this moment, a way to extend the radius of the sphere is been developing. The basic idea is to change the restrictions of the optimization problem.

Formal conditions under which the proposed model can work were also given and proven. In particular, the case of the perfect classification was presented. A brief explanation about the functioning of the GAMs against the noise was presented; the GAMs can cope with noise patterns when the noise version falls inside of the class sphere.

Nowadays, we are also interested to test our method in more realistic situations and in comparison (in computing time and performance) between the proposed model and other geometric classification models. We are working too in GAMs that work with separation surfaces other than spheres, like ellipses, squares, or other irregular shapes; then, the GAMs can work with nonspherically separable classes.

Acknowledgements The authors thank the National Polytechnic Institute of Mexico (SIP-IPN) under grants 20090620 and 20091421. Authors also thank the European Union, the European Commission, and CONACyT for the economical support. This paper has been prepared for economical support of the European Commission under grant FONCICYT 93829. The content of this paper is an exclusive responsibility of the CIC-IPN, and it cannot be considered that it reflects the position of the European Union. We thank also the reviewers for their comments for the improvement of this paper.

References

1. Anderson, J.: A simple neural network generating an interactive memory. Math. Biosci. **14**, 197–220 (1972)
2. Arena, P., Baglio, S., Fortuna, L., Xibilia, M.: Chaotic time series prediction via quaternionic multilayer perceptrons. Syst., Man and Cybern., Intell. Syst. for the 21st Century, IEEE Int. Conf., vol. 2, pp. 1790–1794 (1995)
3. Asuncion, A., Newman, D.: UCI machine learning repository (2007). http://www.ics.uci.edu/mlearn/MLRepository.html
4. Banarer, V., Perwass, C., Sommer, G.: The hypersphere neuron. 11th Eur. Symp. on Artif. Neural Netw. Evere, Belgium: d-side publ., pp. 469–474 (2003)
5. Barron, R.: Memorias asociativas y redes neuronales morfologicas para la recuperacion de patrones. Ph.D, thesis, Mexico, DF: National Institute Politechnic—Center of Computing Research (2006)

6. Barron, R., Cruz, B., Sossa, H., Laguna, G.: Conformal geometric algebra for spherical convex hull optimization. In: Proc. 3rd Internat. Conf. on Appl. of Geom. Algebras in Comput. Sci. and Eng., AGACSE 2008 (2008)
7. Bayro, E., Vallejo, R.: Geometric feedforward neural networks and support vector machines. In: Bayro-Corrochano, E., Sobczyk, G. (eds.) Geometric Algebra with Applications in Science and Engineering, pp. 309–325. Birkhäuser, Basel (2001)
8. Buchholz, S.: A theory of neural computation with Clifford algebras. Thesis, Kiel: Christian-Albrechts-Universitat (2005)
9. Buchholz, S., Tachibana, K., Hitzer, E.: Optimal learning rates for Clifford neurons. In: Proc. of ICANN 2007, Part I. LNCS, vol. 4668, pp. 864–873. Springer, Berlin (2007)
10. Clifford, W.: Applications of Grassmann's extensive algebra. Am. J. Math. **1**(4), 350–358 (1878)
11. Cruz, B., Sossa, H., Barron, R.: A new two level associative memory for efficient pattern restoration. Neural Process. Lett. **25**, 1–16 (2007)
12. Cruz, B., Barron, R., Sossa, H.: Geometric associative memory model with application to pattern classification. In: Proc. 3rd Internat. Conf. on Appl. of Geom. Algebras in Comput. Sci. and Eng., AGACSE 2008 (2008)
13. Hitzer, E.: Euclidean geometric objects in the Clifford geometric algebra of origin, 3-space, infinity. Bull. Belg. Math. Soc. **11**(5), 653–662 (2004)
14. Hestenes, D., Sobczyk, G.: Clifford Algebra to Geometric Calculus. Springer, Berlin (1984)
15. Hestenes, D.: Old wine in new bottles. In: Bayro-Corrochano, E., Sobczyk, G. (eds.) Geometric Algebra: A Geometric Approach to Computer Vision, Quantum and Neural Computing, Robotics, and Engineering, pp. 498–520. Birkhäuser, Basel (2001)
16. Hestenes, D., Li, H., Rockwood, A.: New algebraic tools for classical geometry. In: Sommer, G. (ed.) Geometric Computing with Clifford Algebras. vol. 40, pp. 3–23. Springer, Heidelberg (2001)
17. Hildebrand, D.: Geometric computing in computer graphics using conformal geometric algebra. Tutorial, TU Darmstadt, Germany: Interact. Graph. Syst. Group (2005)
18. Hopfield, J.: Neural networks and physical systems with emergent collective computational abilities. Proc. Natl. Acad. Sci. **79**, 2554–2558 (1982)
19. Kohonen, T.: Correlation matrix memories. IEEE Trans. Comput. **C-21**(4), 353–359 (1972)
20. Li, H., Hestenes, D., Rockwood, A.: Generalized homogeneous coordinates for computational geometry. In: Sommer, G. (ed.) Geometric Computing with Clifford Algebras. vol. 40, pp. 27–52. Springer, Heidelberg (2001)
21. McCulloch, W., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**, 115–133 (1943)
22. Nakano, K.: Associatron a model or associative memory. IEEE Trans. Syst., Man Cybern **12**, 380–388 (1972)
23. Ritter, G., Sussner, P., Diaz-de-Leon, J.: Morphological associative memories. IEEE Trans. Neural Netw. **9**(2), 281–293 (1998)
24. Sossa, H., Barron, R.: New associative model for pattern recall in the presence of mixed noise. IASTED Fifth Int. Conf. on Signal and Image Process. (SIP 2003), pp. 485–490 (2003)
25. Steinbouch, K.: Die Lernmatrix. Kybernetik **1**(1), 26–45 (1961)
26. Sussner, P.: Observations on morphological associative memories and the kernel method. Neurocomputing **31**, 167–183 (2003)
27. Yañez, C., Diaz-de-Leon, J.: Introducción a las memorias asociativas. Mexico: Res. in Comp. Sci. (2003)