Translation, Rotation, and Scale Invariant Pattern Recognition by High-Order Neural Networks and Moment Classifiers

Stavros J. Perantonis and Paulo J. G. Lisboa

Abstract—**The classification and recognition of two-dimensional patterns independently of their position, orientation, and size by using high-order networks are discussed. A method is introduced for reducing and controlling the number of weights of a third-order network used for invariant pattern recognition. The method leads to economical networks that exhibit high recognition rates for translated, rotated, and scaled, as well as locally distorted, patterns. The performance of these networks at recognizing typed and handwritten numerals independently of their position, size, and orientation is compared with and found superior to the performance of a layered feedforward network to which image features extracted by the method of moments are presented as input.**

I. Introduction

PATTERN classification and recognition invariant under translation, rotation, and scale transformation are usually achieved by following a two-stage process. First, a suitable set of features with the desired invariance properties is extracted from the patterns selected for classification; this step is then followed by presentation of the extracted features to a classifier whose purpose is to partition the space of features into decision regions corresponding to each pattern class [1], [2].

A class of neural-network-based invariant pattern recognition models (hereinafter class A) conforms to this approach: conventional feature extraction methods are used to define appropriate invariant features, and the role of the neural network is simply to act as a classifier for the independently selected features. In a second class of models (class B), however, the desired invariances are incorporated in the structure of dynamics of the neural network, which is then able without external assistance both to refer a pattern to a standard configuration and to recognize it [3]–[8]. Models in class A are generally better suited for practical applications, because they combine the strength of conventional feature extraction methods with the classifying power of neural networks, while models in class B often encounter difficulties, especially when combining different types of invariance.

In this paper we study the invariant pattern recognition properties of *high-order networks,* which can be viewed as a link between models in classes A and B. These networks process products of the pixel values in an image. High-order networks can be configured in such a way as to encode in the values of their synaptic weights the property of invariance under coordinate transformations in two dimensions. In this way they can be viewed as equivalent to ordinary first-order networks acting upon invariant features equal to appropriate sums of products of pixel values. From this point of view, the evaluation of these sums represents a preprocessing stage, and the neural network assumes the conventional role of the feature classifier. We show that high-order networks can, when implemented appropriately, surpass in performance class A networks which use any number of moments as feature representations of the image.

A problem that is inherent in the implementation of high-order neural networks is the combinatorial increase in the number of weights with the order of the network [9]. Although it is known that this problem is partly alleviated in networks designed for invariant pattern recognition [10], [11], this has not been done in a way which gives control over the size of the network.

We discuss in detail the implementation of third-order networks for pattern recognition with invariance under translation, rotation, and scaling and suggest a method of controlling the size of the network independently of the screen resolution, a method which also leads to improved robustness against image distortion. The patterns used to test the networks are typed as well as handwritten numerals including images that are distorted and corrupted by pixel noise.

This paper is organized as follows: In Section II we introduce the general formalism for the construction of invariant high-order networks. In Section III we discuss a method of reducing the number of weights in an invariant network of the third order. Section IV serves as an introduction to pattern recognition by the method of moments with particular reference to the application of this method in conjunction with a neural network classifier. In Section V we describe our simulations for the recognition of numerals, the results of which we present in Section VI. Finally, Section VII is an account of our conclusions.

II. High-Order Networks for Invariant Pattern Recognition

We consider a pattern represented by its pixel map s_i, $i = 1, 2, \cdots, N$, on a N-pixel screen. Let us now regard each pixel i as a point on a regular square grid of lattice. The

Manuscript received February 14, 1991; revised September 6, 1991.

S. J. Perantonis was with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, U.K. He is now with the N.S.P.R.C. Domokritos, GR-15310 Aghia Paraskevi, Athens, Greece.

P. J. G. Lisboa is with the Department of Electrical Engineering and Electronics, University of Liverpool, Liverpool, L69 3BX, U.K.

IEEE Log Number 9104308.

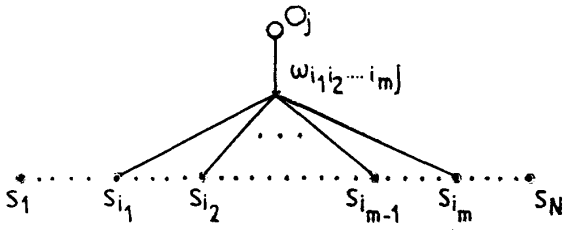


Fig. 1. Connectivity in a high-order network.

position of the pixel i is determined by the position vector $\boldsymbol{r}(i)$ with its origin at the center of the screen. Consider now a neural network with a single layer of weights which processes products of the form $s_{i_1}s_{i_2}\cdots s_{i_p}$. In the notation established in Fig. 1, the output of a node j is of the form

$$O_j(\boldsymbol{s}) = f\left(\sum_{i_1}\sum_{i_2}\cdots\sum_{i_p} w_{i_1 i_2\cdots i_p j}\, s_{i_1}s_{i_2}\cdots s_{i_p}\right) \tag{1}$$

where f is the activation function for the formal neurons. We now consider a group $\mathcal{G}$ of coordinate transformations realizable on the pixel lattice, under which we wish O_j to be invariant. Examples of such groups of transformations are the group of translations by vectors whose components are multiples of the spacing between pixels and the group of inversions about a principal lattice direction. Under a transformation R belonging to $\mathcal{G}$, the pixel s_i will be transformed to a pattern represented by s'_i such that

$$s'_i = s_{i'} \qquad \text{with } \boldsymbol{r}(i') = R\boldsymbol{r}(i). \tag{2}$$

The transformed output of the network is of the form

$$\begin{aligned} O'_j &= f\left(\sum_{i_1}\sum_{i_2}\cdots\sum_{i_p} w_{i_1 i_2\cdots i_p j}\, s'_{i_1}s'_{i_2}\cdots s'_{i_p}\right) \\ &= f\left(\sum_{i_1}\sum_{i_2}\cdots\sum_{i_p} w_{i_1 i_2\cdots i_p j}\, s_{i'_1}s_{i'_2}\cdots s_{i'_p}\right) \end{aligned} \tag{3}$$

where $\boldsymbol{r}(i'_m) = R\boldsymbol{r}(i_m)$, $m = 1, 2, \cdots, p$. Equation (3) can be rewritten as follows:

$$O'_j = f\left(\sum_{i_1}\sum_{i_2}\cdots\sum_{i_p} w_{k_1 k_2\cdots k_p j}\, s_{i_1}s_{i_2}\cdots s_{i_p}\right) \tag{4}$$

where $\boldsymbol{r}(k_m) = R^{-1}\boldsymbol{r}(i_m)$, $m = 1, 2, \cdots, p$. Since R^{-1} is an element of $\mathcal{G}$, it follows that O_j will be invariant under the action of all elements in $\mathcal{G}$ if the weights are chosen so that

$$w_{i_1 i_2\cdots i_p j} = w_{k_1 k_2\cdots k_p j} \tag{5}$$

provided that there exists an element of $\mathcal{G}$ through which $\boldsymbol{r}(i_m)$ and $\boldsymbol{r}(k_m)$ are related for all $m = 1, 2, \cdots, p$. Thus the desired invariance is built into the architecture of the network through the imposition of appropriate constraints on the synaptic weights.

For linear coordinate transformations, (5) means that in order to ensure invariance, we must set $w_{i_1 i_2\cdots i_p j} = w_{k_1 k_2\cdots k_p j}$ whenever the relative coordinates of any pair (i_a, i_b) $(a, b = 1, \cdots, p)$ can be obtained from the relative coordinates of the corresponding pair (k_a, k_b) through a transformation of $\mathcal{G}$.

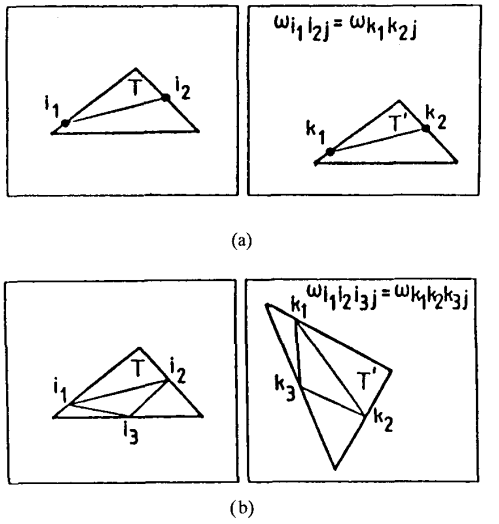


Fig. 2. Rules for constructing high-order networks for invariant pattern recognition. In all cases the triangle T represents an image which the network is called upon to recognize. (a)(i) Translation-invariant recognition by a second-order network: For the parallel and equal segments i_1i_2 and k_1k_2 the weights $w_{i_1i_2j}$ and $w_{k_1k_2j}$ are set equal to each other. (a)(ii) Translation and scale invariant recognition by a second-order network: For the parallel segments i_1i_2 and k_1k_2 the weights $w_{i_1i_2j}$ and $w_{k_1k_2j}$ are set equal to each other. (b) Translation, rotation and scale invariant recognition by a third-order network: For the similar triangles $i_1i_2i_3$ and $k_1k_2k_3$ the weights $w_{i_1i_2i_3j}$ and $w_{k_1k_2k_3j}$ are set equal to each other. In cases (a)(ii) and (b) the effective inputs of the network must be normalized as discussed in the text.

For example, to obtain a second-order network which produces output invariant under translation by lattice vectors, we must set equal all weights $w_{i_1i_2j}$ and $w_{k_1k_2j}$ for which the line segments i_1i_2 and k_1k_2 can be transformed to one another through such a translation, i.e., weights for which these segments are equal in length and parallel to each other (Fig. 2(a)).

Note that this procedure for building into the network architecture invariances under a group of transformations, assigns p-tuples of points in the plane to equivalence classes $\mathcal{C}_h$, $h = 1, 2, \cdots$ defined by the classification rule of (5). We can thus substitute the notation $w_{i_1i_2\cdots i_pj}$ for the weights with the notation w_{hj}, where h is a collective index corresponding to the class $\mathcal{C}_h$. The output of a node in the top layer of the high-order network can be now written as

$$O_j = f\left(\sum_h w_{hj} \sum_{(i_1,\cdots,i_p)\in\mathcal{C}_h} s_{i_1}s_{i_2}\cdots s_{i_p}\right) \tag{6}$$

so that we are effectively dealing with a single-layered feed-forward network (perceptron) with weights w_{hj} and *effective inputs*

$$I_h = \sum_{(i_1,\cdots,i_p)\in\mathcal{C}_h} s_{i_1}s_{i_2}\cdots s_{i_p} \tag{7}$$

representing the invariant features of the images which the network is called upon to classify. Bias terms clearly do not affect the invariance properties of the network; nor do subsequent layers of weights.[1] In this way, the network architecture is the same as that of multilayered first-order feedforward networks, widely studied elsewhere, and additional programming tasks are confined to the determination of the equivalence classes and the calculation of the effective inputs I_h from $\boldsymbol{s}$.

It is evident that exact transformations corresponding to translation by arbitrary vectors, rotation by arbitrary angles, and scaling by arbitrary factors are not realizable on a square lattice, because under these transformations a lattice point does not necessarily transform onto another lattice point. However, we can make use of schemes involving the lattice points closest to the continuum transform. It should be noted that such "lattice transformations" do not necessarily form a group, but rather have the mathematical structure of a "loop" [12], so that the above considerations are not directly applicable. For example, neighboring lattice points may be mapped onto the same lattice point under a lattice scaling transformation which is thus noninvertible. We can nevertheless utilize some of the main points in the discussion above to build networks which ensure approximate invariance under translation, scaling, and rotation.

For example, a second-order network can produce output approximately invariant to translation and scaling transformations provided that

a) weights $w_{i_1 i_2 j}$ and $w_{k_1 k_2 j}$ for which the line segments $i_1 i_2$ and $k_1 k_2$ are parallel to each other (they can thus be transformed to each other through a combined translation and scaling operation) are set equal [11] (Fig. 2(a)(ii)); and

b) all images are appropriately normalized to compensate for the fact that by scaling an image we may be mapping points which are originally distinct to the same point or vice versa.

If we consider images represented by binary input (0 for the blanks on the screen and 1 for the "active" image pixels), a linear scaling by a factor of ρ brings about a scaling of the number of active pixels approximately by ρ^2 and of the effective network inputs I_h by ρ^4. We can compensate for this effect by normalizing the effective input vector $(I_1, I_2, \cdots)$ to a standard Euclidean length.

To implement combined translation, scaling, and rotation invariance, we notice that any two line segments in the plane can be transformed into each other by a combination of those transformations. Consequently, if a second-order network were used, all its weights would have to be set equal to each other, leaving no degrees of freedom for learning how to classify distinct patterns. A third-order network, however, will produce output approximately invariant to all three transformations if any two weights $w_{i_1 i_2 i_3 j}$ and $w_{k_1 k_2 k_3 j}$ are set equal whenever the triangles with vertices at i_1, i_2, i_3 and k_1, k_2, k_3 respectively can be transformed into each other by a combination of the three transformations [11]. This means that the two triangles must be similar to each other with their equal angles encountered in the same order when the perimeters of the triangles are traversed in a counterclockwise manner (Fig. 2(c)). In addition, all images must be normalized to compensate for a multiplicative factor ρ^6 on the effective inputs I_h brought about by scaling transformations as discussed above.

Among the networks described in this section, second-order networks with outputs invariant under translation have been implemented for associative memory [13] and pattern recognition applications [10]. In addition, Reid *et al.* [11] have produced a second-order neural network with automatic translation and scaling invariance able to recognize the characters "T" and "C" irrespective of position and size with 100% accuracy. In this work we concentrate on invariance under all three affine transformations simultaneously. We shall thus implement a third-order network suitably augmented by a method for the reduction of the number of weights. This we discuss in the next section.

III. Reduction of the Number of Weights

A problem inherent in high-order networks is the combinatorial increase of the number of weights with the order of the network. Indeed, this was the reason why Minsky and Papert dismissed high-order networks as impractical [9]. However, building invariances into these networks partially helps alleviate this problem, since it amounts to substituting a number of weights corresponding to p-tuples of points in the plane classified in the same equivalence class by just one weight. The resulting network configuration is then capable of practical realization in hardware [14].

In the case of the third-order network designed to implement invariance under translation, scaling, and rotation, this reduction in the number of weights is achieved by assigning similar triangles to the same equivalence class. Equivalently, we can say that this classification results in the reduction of the number of effective network inputs N_I defined in (7) (and appropriately normalized) which represent the invariant features by which we choose to characterize our images.

It should be noted, however, that this classification scheme does not allow the user to vary the number of effective inputs or the size of the network independently of the number of pixels on the screen. Even on relatively coarse screens, the scheme leads to a large number of effective inputs and weights. Moreover, it is very sensitive to the mild distortions that a triangle defined on the screen can suffer if subjected to an approximate lattice rotation of scaling transformation. This is clearly not a satisfactory situation. What is needed is a classification scheme which can lead to a relatively small number of invariant features controllable independently of the screen resolution and which is less sensitive to the distortion of images brought about through lattice coordinate transformations.

A simpler scheme adopted here involves classifying "approximately similar" triangles into the same equivalence class. Approximately similar triangles will be characterized by the same value of their two smallest angles (the value of the third

[1] We shall refer to these multilayered networks as high-order networks, although products of the form $s_{i_1} s_{i_2} \cdots s_{i_p}$ are processed only at the first hidden layer of nodes.

angle being dependent on the value of the other two) within a finite tolerance ω.

More specifically, let us consider the set of all triangles which can be defined on the screen. We denote their angles by α, β, and γ with $\alpha \leq \beta \leq \gamma$. The set $\mathcal{T}_1$ of triangles for which α and β are encountered in immediate succession and in that order when the triangle is traversed in a counterclockwise manner must be considered separately from the set $\mathcal{T}_2$ consisting of the inverted images of these triangles, if we do not want to impose additional inversion invariance on our system. The following relations are obeyed by α and β:

$$\alpha \leq \beta, \quad 0 \leq \alpha \leq \pi/3 \qquad 0 \leq \beta < \pi/2$$
$$0 \leq \beta + a/2 \leq \pi/2 \tag{8}$$

the last of which is a direct consequence of the relations $\alpha + \beta + \gamma = \pi$ and $\beta \leq \gamma$. We choose an angular tolerance ω so that $W = \pi/(3\omega)$ and $Q = \pi/(2\omega)$ are integers, and partition the sets of possible values of α and β into bins defined by

$$(k-1)\omega \leq \alpha < k\omega \qquad (l-1)\omega \leq \beta < l\omega$$
$$1 \leq k \leq W, \quad 1 \leq l \leq Q. \tag{9}$$

Finally, we assign all triangles whose angles satisfy relations (9) for given values of k and l to the same equivalence class.

This classification scheme is illustrated in Fig. 3 for $\omega = \pi/12$ and for the triangles belonging to $\mathcal{T}_1$. The segment AB represents the side of the triangle ABC adjacent to the smallest and next-to-smallest angles α and β. The lines emerging from points A and B define the boundaries of the bins in which the range of α and β is partitioned. The intersections of these lines define triangular and quadrilateral regions, so that all triangles in a given region are classified in the same equivalence class. Note that relations (8) must be obeyed, so that nonempty classes correspond only to regions in the area bounded by the bold line in Fig. 3, which consists of the segment AB, its perpendicular bisector (ε), and the arc BD of the circle with its center at A, corresponding to an angle of $\pi/3$.

To find the number of effective inputs N_I of our network, we must calculate the number N_{cl} of nonempty classes of triangles as a function of ω. Evidently,

$$N_i = 2N_{cl} = 2(N_{\text{sec}} - N_{\text{left}}) \tag{10}$$

where N_{sec} is the number of regions in the circular sector DAB, and N_{left} is the number of regions whose interior lies entirely on the left of (ε) (for set $\mathcal{T}_1$). The factor of 2 is inserted to take account of both sets $\mathcal{T}_1$ and $\mathcal{T}_2$. We now consider the regions corresponding to a certain value of k but all possible values of l (eq. (9)) and calculate their contributions $\Delta N_{\text{sec}}(k)$ and $\Delta N_{\text{left}}(k)$ to N_{sec} and N_{left}. For $k = 0$, we get $\Delta N_{\text{sec}}(0) = Q$ and $\Delta N_{\text{left}}(0) = 0$. Noting that the lines corresponding to $\beta = \pi/2 - \omega, \pi/2 - 2\omega, \cdots$ intersect the lines corresponding to $\alpha = 2\omega, 4\omega, \cdots$, respectively, at points belonging to the arc BD, we readily conclude that, starting from $k = 0$, every increase of k by 2 corresponds to

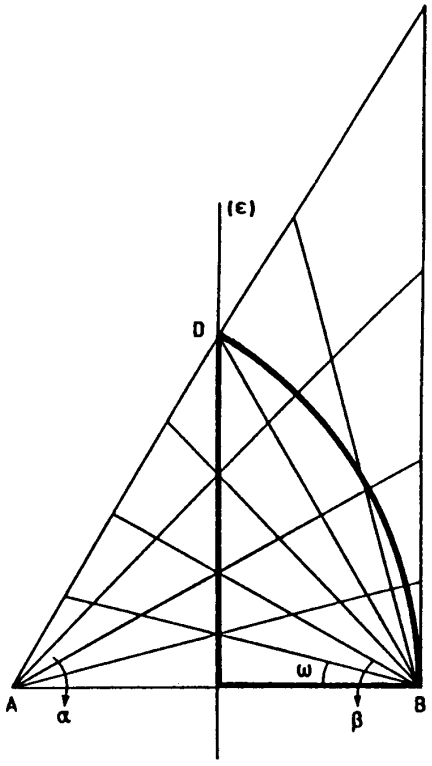


Fig. 3. Schematic representation of the partitioning of the set of triangles defined on the screen into classes of approximately similar triangles for the purpose of invariant pattern recognition using a third-order network.

a decrease of $\Delta N_{\text{sec}}(k)$ by 1, so that

$$N_{\text{sec}} = \sum_{k=0}^{2Q/3-1} \Delta N_{\text{sec}}(k) = 2 \sum_{m=0}^{Q/3-1} (Q - m)$$
$$= Q(5Q+3)/9. \tag{11}$$

Similarly, starting from $k = 0$, every increase of k by 1 corresponds to an increase of $\Delta N_{\text{left}}(k)$ by 1, so that

$$N_{\text{left}} = \sum_{k=0}^{2Q/3-1} \Delta N_{\text{left}}(k) = \sum_{m=0}^{2Q/3-1} m = Q(2Q-3)/9. \tag{12}$$

From (10), (11), and (12) we finally obtain for the number of effective inputs as a function of the angular tolerance ω

$$N_I = \frac{2Q(Q+2)}{3} = \frac{\pi(4\omega+\pi)}{6\omega^2}. \tag{13}$$

Finally, we note that as ω increases and N_I decreases, the average areas of the regions in Fig. 3 corresponding to each class increase. Consequently, it becomes more unlikely that mild distortions of the shape of a triangle will push it out of the equivalence class to which it originally belongs. The effective inputs of the network therefore become less sensitive to mild image distortions, including the distortions brought about by lattice scaling and rotation transformations.

However, this conclusion applies only to structured distortion which affects the shape of the image. It does not apply to noise randomly distributed on the screen.

IV. Invariant Pattern Recognition by the Method of Moments

In selecting a suitable invariant pattern recognition system with which to compare the performance of high-order neural networks, we decided to make use of a system utilizing the so-called Zernike moments—suitably normalized—as the invariant features and a multilayered feedforward neural network as the classifier.

Moments and invariant functions of moments have been extensively used for invariant feature extraction in a wide range of two- and three-dimensional pattern recognition applications [15]–[21]. Pattern recognition results from moment-based methods compare well with the results of other popular invariant feature extraction schemes, e.g., Fourier descriptor analysis [19]. Of the various types of moments defined in the literature (regular, Zernike, pseudo-Zernike, Legendre, rotational, and complex moments), Zernike and pseudo-Zernike moments have been shown to be superior to the others in terms of their insensitivity to image noise, information content, and ability to provide faithful image representation [21]. Moreover, a scheme of image representation by the moduli of Zernike moments of images appropriately normalized by means of low-order regular moments was studied by Khotanzad and Hong [20] and shown to yield superior results to other moment-based methods in an invariant character recognition task. We therefore adopt this scheme as our basis for comparison with invariant high-order neural networks.

As regards the choice of classifier, it is by now established that multilayered neural networks are able to match, and often improve upon, the performance of conventional classifiers [1] in a large number of applications, including invariant character recognition via the method of moments [22]. Moreover, the use of a common neural network architecture facilitates the comparison between the two approaches (high-order networks versus moments) to the problem of invariant patter recognition addressed in this paper.

In the remainder of this section we briefly review the formalism of regular and Zernike moments and describe the method of feature extraction based on them which we shall use in our pattern recognition experiments.

Given a two-dimensional image represented by a continuous function $s(x,y)$ of the Cartesian coordinates x and y in the plane, moments are generically functions of the form

$$\iint s(x,y)\, g(x,y)\, dx\, dy \tag{14}$$

where $g(x,y)$ is an appropriate weighting function. For discrete images represented by functions defined at discrete points on a lattice, sums are used instead of integrals so that the generic expression (14) becomes

$$\sum_i s(x_i,y_i)\, g(x_i,y_i)\, \Delta x\, \Delta y \tag{15}$$

where $\Delta x\, \Delta y$ is the area of the unit cell of the lattice, and the index i runs over all lattice points. We shall henceforth use the notation for continuous images keeping in mind that discrete images can be similarly treated by substituting the integrals with the appropriate sums.

The regular moments of an image represented by $s(x,y)$ are defined by the relation

$$M_{ab} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} s(x,y)\, x^a y^b\, dx\, dy \tag{16}$$

where a and b are nonnegative integers.

The Zernike moment of order n with repetition m [23] is a complex number given by the equation

$$A_{nm} = \frac{n+1}{\pi} \iint_{x^2+y^2\le 1} s(x,y)\, [V_{nm}(x,y)]^*\, dx\, dy \tag{17}$$

with n a nonnegative integer and m an integer such that $n - |m|$ is nonnegative and even. The complex-valued functions $V_{nm}(x,y)$ are defined by

$$V_{nm}(x,y) = V_{nm}(r,\vartheta) = R_{nm}(r)\, \exp(im\vartheta) \tag{18}$$

where r and ϑ represent polar coordinates over the unit disk and R_{nm} are polynomials of r (Zernike polynomials) given by

$$R_{nm}(r) = \sum_{l=0}^{(n-|m|)/2} (-1)^l \cdot \frac{(n-1)!}{l!\,[(n+|m|)/2-1]!\,[(n-|m|)/2-l]!}\, r^{n-2l}. \tag{19}$$

A main feature of the Zernike polynomials is that they are orthogonal. Consequently, the image $s(x,y)$ can be reconstructed directly from the coefficients as follows:

$$s(x,y) = \sum_n \sum_m A_{nm} V_{nm}(r,\vartheta). \tag{20}$$

From the point of view of pattern recognition, it is useful to approximate $s(x,y)$ by using a finite number of Zernike moments, with order less than or equal to n^*. The value of n^* can be chosen so that a substantial proportion of the image (say 90%) is reconstructed through (20).

By virtue of (17) and (18), the Zernike moments of a rotated version of the image $s(x,y)$ by an angle ϑ_o are related to the Zernike moments of the original image by $A'_{nm} = A_{nm} \exp(-im\vartheta_o)$, so that the moduli $|A_{nm}|$ are invariant under rotations and can be used as appropriate features for rotation-invariant pattern recognition. Note that $|A_{nm}| = |A_{n,-m}|$ (by (18)), so that only moments with $m \ge 0$ need be considered. Since, however, the moduli of the Zernike moments are not invariant under translation or scaling, the original image has to be adjusted for the effect of these transformations before the $|A_{nm}|$ are calculated. Here we adopt the scheme also used by Khotanzad and Hong [20], whereby the original image is adjusted for translation and scaling using a standard normalization procedure involving low-order regular

moments. More specifically, the image $s(x,y)$ is transformed as follows:

$$s(x,y) \rightarrow s'(x,y) = s\Big[x(M_{00}/d)^{1/2} + M_{10}/M_{00}, \; y(M_{00}/d)^{1/2} + M_{01}/M_{00}\Big] \tag{21}$$

where d is a predetermined value. This operation transforms the image $s(x,y)$ to a standard image whose centroid coincides with the origin and whose zeroth-order regular moment is set to the constant value d. The moduli of the Zernike moments for this image are invariant under rotation, translation, and scaling and can be used as features suitable for invariant pattern recognition. (For discrete images only approximate "lattice" rotation and scaling transformations are realizable and the moduli of the Zernike moments are only approximately invariant under these transformations.) It should be noted that with the normalization scheme described here the moduli of the lowest-order Zernike moments, A_{00} and A_{11}, are independent of $s(x,y)$ ($|A_{00}| = d/\pi$ and $|A_{11}| = 0$); therefore we ignore them. Excluding these moments and the moments with negative m, we are left with a number of Zernike features for image processing equal to

$$N_{\text{Zer}} = \begin{cases} \ell^2 + 2\ell - 1 & \text{for } n^* = 2\ell, \quad \ell = 1,2,\cdots \\ \ell^2 + 3\ell & \text{for } n^* = 2\ell + 1, \quad \ell = 1,2,\cdots. \end{cases} \tag{22}$$

In summary, the scheme for invariant pattern classification described in this section involves the following steps:

a) Adjustment of all images for translation and scaling through (21).
b) Calculation of the moduli $|A_{nm}|$ of the Zernike moments for the adjusted images ($2 \le n \le n^*, 0 \le m \le n, n-m$ even). The moduli of the Zernike moments are used as invariant features representing the original images.
c) Classification of the images by inputting the invariant features $|A_{nm}|$ to a (possibly multilayered) feedforward network with N_{Zer} input nodes, an adjustable number of hidden nodes, and a number of output nodes equal to the total number of classes to which the images belong.

V. Network Implementation and Experiments

We have implemented the following types of networks to recognize images representing typed and handwritten digits "0" to "8"[2] (Figs. 4 and 5) on an $N = 20^2$ pixel screen.

1) Third-Order Networks (TON's): These are for a range of values of the angular tolerance ω (from $\pi/144$ to the highest possible value of $\pi/6$). The networks are implemented in the feature extractor–classifier mode as follows. First, the triangles on the screen have to be scanned, their angles calculated, and a class membership number $C(i_1, i_2, i_3)$ has to be assigned to each triangle with vertices i_1, i_2, and i_3 and stored for subsequent use. An array of dimension $\binom{N}{3}$ is required for this task ($\cong 10^7$ for a 400 pixel screen), which creates a storage problem. To overcome the problem, we only store class membership numbers for triangles with one vertex at a certain point G on the screen. The class membership number of any other triangle can be calculated each time it is needed by translating the triangle so that its origin coincides with G.

[2] No image is included to represent "9," which can obviously be obtained from "6" by a rotation.

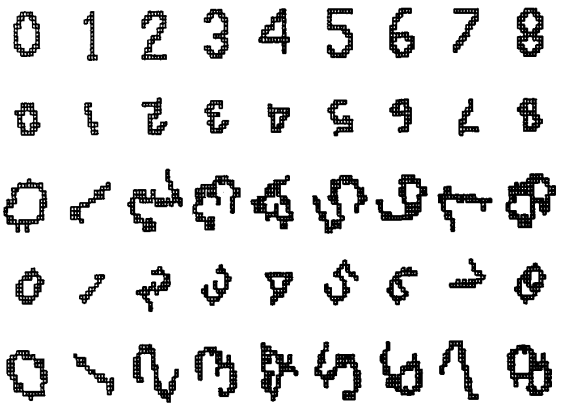
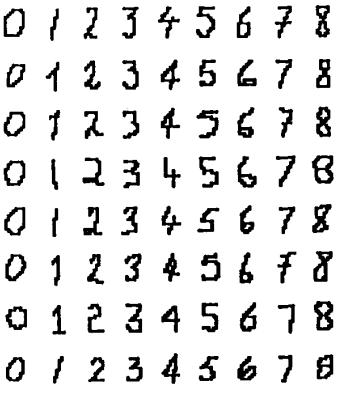
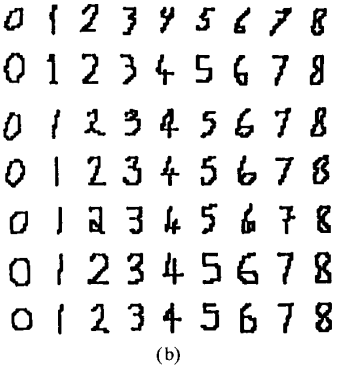


Fig. 4. A set of typed numerals used to train TON's and ZFC's. All digits are rotated and scaled versions of those shown in the first row.



(a)



(b)

Fig. 5. Two sets of handwritten numerals. The digits in (a) as well as four rotated and scaled versions for each digit are used for training TON's and ZFC's. (b) The digits in (b) as well as arbitrarily rotated and scaled versions of these digits are used for testing.

This is a fast operation involving no multiplications and takes a very small proportion of the time for training and recall. Next, the effective input vectors (cf. (7) with $p = 3$) for the nominated patterns are calculated, normalized to a standard length, and presented as input to a (possibly multilayered) feedforward network with nine output nodes corresponding to the nine classes ("0"–"8"), which is trained by gradient back-propagation [24], [25]. The logistic function $f(z) = 1/(1 + \exp(-z))$ is used as the activation function for the formal neurons.

2) First-Order Feedforward Networks: These have a variable number of hidden nodes which are used as classifiers for the moduli of the Zernike moments of the nominated images appropriately scaled and rotated as discussed in Section IV. We shall refer to these networks as Zernike feature classifiers (ZFC's). For the images representing the typed and handwritten numerals in our experiments, we have checked that performance results are not sensitive to the value of d (eq. (21)) for a range of values, and $d = 0.075$ corresponding to images with 30 "active" pixels was selected. The maximum order n^* if the Zernike moments used for feature extraction was chosen in such a way that the ratio of the Hamming distance between the original and the reconstructed image to the number of pixels in the unit disk was at most 0.10 for all images in the training sets. For our typed and handwritten digits, this corresponding to $n^* = 11$. However, we have also studied values of n^* in the range 9 to 12 (which is the maximum number used by Khotanzad and Hong [20] in a similar experiment involving typed characters of the Roman alphabet) to ensure optimal performance. Thus, on excluding $|A_{00}|$ and $|A_{11}|$ for the reasons discussed in Section IV, we arrive at networks with 28 to 47 inputs (cf. (22)). As before, the networks have nine output nodes and are trained by gradient back-propagation.

We have carried out extensive simulations involving the set of typed numerals to study and compare the ability of the networks to cope with transformed, distorted, and noisy images (experiment A). A more restricted number of simulations has been carried out to test the ability of the networks to recognize handwritten numerals (experiment B). In the rest of this section we describe these experiments before we go on to analyze their results in the next section.

Experiment A: Invariant Recognition of Typed Digits

A set of typed digits ranging from "0" to "8" represented by binary input on a 20 by 20 screen is shown in the first row of Fig. 4. These images are scaled and rotated to produce five different versions per image. The range of the scaling factor is between 0.7 and 1.3. The 45 resulting images, which are shown in Fig. 4, are used for training TON's and ZFC's.

The performance of the networks is then tested with respect to the following tasks.

1) Recognition of Transformed Patterns: Under a coordinate transformation, a point on the lattice is mapped to the lattice point closest to its actual continuum transform. However, for scaling transformations by factors $\rho > 1$, a slightly different prescription is used, to ensure that the number of pixel grows with the scaling factor ρ. For these transformations, the value of a pixel P in a transformed pattern is chosen equal to the value of the pixel in the original pattern closest to the image of P under the inverse of the exact continuum transformation. Under this scheme, it is only grid rotation and scaling transformations which introduce distortion to either the effective inputs of the TON or the Zernike features, while translations leave these features invariant. Thus both networks automatically exhibit 100% recognition rates for translations. Consequently, only their ability to recognize scaled and rotated patterns need be checked.

To this end, 50 scaled versions of each digit in the first row of Fig. 4 are prepared using scaling factors with a uniform random distribution between 0.7 and 1.3. These are then presented to the neural networks as input and assigned to the class corresponding to the network node with the highest output. The experiment is repeated with 50 rotated versions per pattern using angles of rotation with uniform distribution between 0 and 2π as well as with 50 versions that are both scaled and rotated for the same range of scaling factors and angles of rotation.

2) Recognition of Distorted Patterns: Both types of networks are tested for their ability to recognize 30 versions per pattern of the standard digits "0"–"8" locally distorted by the following algorithm: For each active pixel in every image, a pseudorandom number R_0 is generated with uniform probability density in $(0, 1)$. The pixel is shifted to one of the four nearest neighboring sites (randomly selected) if $R_0 \leq P_0$, where P_0 is a constant; otherwise it is not shifted. Results are presented for $P_0 = 0.25$ and 0.50.

The ability of the networks to recognize patterns both distorted and transformed is also checked. To this end, 30 versions per pattern scaled and rotated as in the task of recognizing transformed patterns and then distorted are presented to the networks and their rates of success in correctly recognizing these transformed and distorted versions are recorded.

3) Recognition of Patterns Transformed and Corrupted by Noise: Thirty versions per pattern are generated by superimposing noise to the original images "0"–"8." The noise is generated by randomly flipping a proportion of the pixels which are contained in the unit disk, to ensure a fair comparison between TON's and ZFC's, given that only these pixels are taken into account for the calculation of the Zernike features. The ability of the networks to recognize the corrupted images is recorded. The experiment is repeated by superimposing noise to scaled and translated versions of the original images. Noise corresponding to three, six, and nine flipped pixels is used.

Experiment B: Invariant Recognition of Handwritten Digits

The set of handwritten digits shown in Fig. 5 is split into two subsets, one of which is used for training TON's and ZFC's (Fig. 5(a)) while the other is used for testing (Fig. 5(b)). As before, each digit is presented to the networks in five differently oriented and scaled versions (including the version shown in Fig. 5(a)), for a total number of 360 states in the training set. The networks are tested for their ability to recognize versions of the digits in Fig. 5(a) rotated by arbitrary angles and scaled by factors between 0.8 and 1.2. In addition, they are tested for their ability to recognize the

TABLE I
AVERAGE STANDARD DEVIATION OVER MEAN VALUE RATIO FOR THE EFFECTIVE INPUTS OF THE TON NETWORK FOR ROTATED AND/OR SCALED VERSIONS OF THE DIGITS IN THE FIRST ROW OF FIG. 4

		$\langle\sigma/\mu\rangle$ (%)		
ω/π	N_I	Rotation	Scaling	Combined
1/144	3552	9.95	14.18	21.09
1/72	912	8.90	10.03	13.11
1/36	240	6.47	7.14	8.83
1/24	112	4.03	4.53	5.37
1/18	66	3.65	4.22	4.35
1/9	32	2.54	2.73	3.20
1/6	10	1.39	1.47	1.85

TABLE II
AVERAGE STANDARD DEVIATION OVER MEAN VALUE RATIO FOR THE MODULI OF ZERNIKE FEATURES FOR ROTATED, SCALED, AND SUBSEQUENTLY NORMALIZED VERSIONS OF THE DIGITS IN THE FIRST ROW OF FIG. 4

		$\langle\sigma/\mu\rangle$ (%)		
n^*	N_{Zer}	Rotation	Scaling	Combined
12	47	6.08	7.33	8.46
11	40	5.97	7.30	8.29
10	34	5.86	7.44	8.16
9	28	5.76	7.09	7.91

digits in Fig. 5(b), on which they have not been trained (but which are approximately of the same size and orientation as those in Fig. 5(a), as well as scaled and/or rotated versions of these digits.

VI. NUMERICAL RESULTS

A. Recognition of Typed Numerals

1) Rotation and Scaling: A look at the scaled and rotated images in Fig. 4 will convince the reader that rotating and especially scaling images on a 20 by 20 grid introduces a substantial amount of distortion. As a result, both the effective inputs of the TON and the inputs of the ZFC are only approximately invariant under these transformations. A measure of the sensitivity of these features to rotation and scaling on the grid is the ratio of the standard deviation σ to the mean μ of the value of each of these features for a large sample of rotated and scaled images. In Tables I and II we show the average σ/μ ratio,—evaluated using 50 scaled, rotated, or both scaled and rotated versions of the nine digits—for the effective inputs of the TON and for the inputs of the ZFC respectively. Note that the average σ/μ increases sharply with the number N_I of effective inputs of the TON. It is thus verified that the effective inputs are less sensitive to rotation and scaling for larger values of the angular tolerance ω. Note, moreover, that for the smaller values of N_I, the effective inputs of TON's are less sensitive to the transformations than the Zernike features for a range of values of n^*. Provided that these TON's can adequately separate the patterns, they should be expected to exhibit better performance than ZFC's.

TABLE III
PERFORMANCE OF TON'S FOR THE RECOGNITION OF ROTATED AND/OR SCALED VERSIONS OF THE TYPED DIGITS (FOR DIFFERENT VALUES OF ω)

One-Layered				
		Success Rate		
ω/π	N_I	Rotation	Scaling	Combined
1/144	3552	85	87	81
1/72	912	91	91	88
1/36	240	97	98	93
1/24	112	97	100	96
1/18	66	99	100	95
1/12	32	99	98	96
Best Two-Layered: 40 Hidden Nodes				
1/12	32	100	99	97

TABLE IV
SUCCESS RATE FOR RECOGNITION OF SCALED AND/OR ROTATED VERSIONS OF THE TYPED DIGITS BY ZFC'S

One-Layered				
	Number of	Success Rate		
n^*	moments	Rotation	Scaling	Combined
12	47	94	94	85
11	40	95	94	89
10	34	92	90	84
9	28	86	88	78
Best Two-Layered: 40 Hidden Nodes				
12	47	93	95	91

In Table III we list the success rates for recognition of scaled, rotated, and both scaled and rotated versions of the nine digits by TON's with different values of ω. For one-layered networks, as ω increases starting from $\omega = \pi/144$, the success rate increases, with the network reaching its optimal performance in the region $\omega = \pi/36$ to $\pi/12$, corresponding to $112 - 32$ effective inputs. The optimal performance corresponds to perfect or nearly perfect recollection of images separately scaled or rotated and to a success rate of 96% for images subjected to both transformations. It follows that a relatively small number of effective inputs corresponding to relatively large values of ω constitutes an adequate set of features for efficient invariant pattern recognition. Only for the largest possible value of $\omega = \pi/6$ do we find that the one-layered TON cannot be trained successfully, indicating that the resulting ten effective inputs cannot be separated by linear decision boundaries.

We have also investigated the effect on performance of adding a hidden layer with a variable number M of nodes (in the range 5–80) and find that this can lead to a marginal improvement in performance, corresponding to 97% maximum accuracy for both transformations (see Table III).

Finally, in Table IV we show the success rate for recognition of rotated and/or scaled patterns by ZFC's. Optimal

TABLE V
SUCCESS RATE FOR RECOGNITION OF TRANSFORMED, DISTORTED, AND NOISY VERSIONS OF THE TYPED DIGITS USING TON'S

		One-Layered									
		Distortion P_0		Trans. + Distor. P_0		Noise (pixels)			Transform. + Noise		
ω/π	N_I	0.25	0.50	0.25	0.500	3	6	9	3	6	9
1/144	3552	97	82	67	57	100	99	86	75	63	44
1/72	912	99	86	73	63	100	97	78	81	66	47
1/36	240	99	87	83	70	100	92	67	86	63	45
1/24	112	99	89	83	69	100	83	56	88	62	43
1/18	66	99	89	86	69	99	85	52	82	57	43
1/12	32	98	89	86	71	97	85	56	86	63	46
		Best Two-Layered: 40 Hidden Nodes									
1/12	32	99	91	89	77						

TABLE VI
SUCCESS RATE FOR RECOGNITION OF TRANSFORMED, DISTORTED, AND NOISY VERSIONS OF THE TYPED DIGITS USING ZFC'S

		Distortion P_0		Trans. + Distor. P_0		Noise (pixels)			Transform. + Noise		
n^*	N_{Zer}	0.25	0.50	0.25	0.50	3	6	9	3	6	9
12	47	90	82	74	63	83	81	75	81	67	66
11	40	87	83	73	66	86	82	71	84	68	66
10	34	86	79	73	61	81	74	64	80	65	63
9	28	79	64	62	60	80	65	59	69	59	52

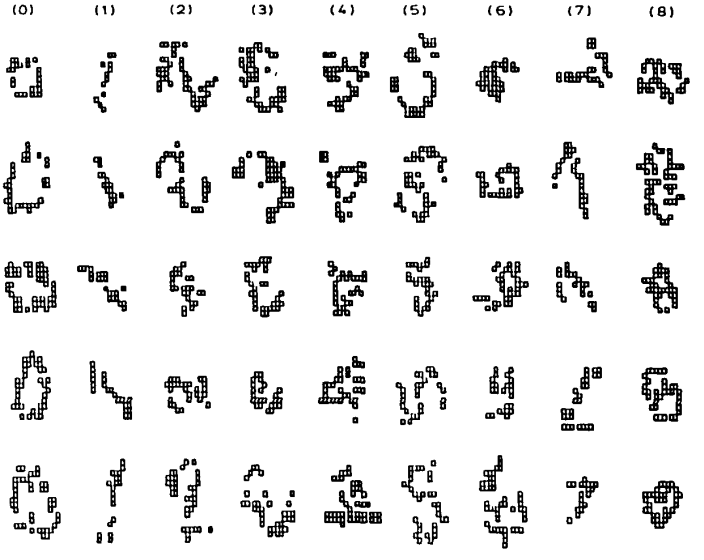


Fig. 6 Scaled, rotated, and locally distorted versions of the digits in the first row of Fig. 3 correctly recognized by a third-order network with 32 effective inputs and 40 hidden nodes. The local distortion to which the digits are subjected corresponds to $P_0 = 0.5$ (see text).

performance is achieved for n^* in the range 11–12. With a two-layered network, we can achieve a maximum recognition accuracy of 91% for both transformations. It is thus evident that TON's with a relatively small N_I exhibit performance superior to that of the optimum ZFC.

2) Distortion: Results for the recognition accuracy of TON's and ZFC's are presented in the second and third columns of Tables V and VI. It is verified that TON's with a relatively small N_I perform better than networks with a large number of inputs, especially when combined distortion, rotation, and

TABLE VII
SUCCESS RATES FOR RECOGNITION OF HANDWRITTEN DIGITS BY TON'S AND ZFC'S WITH RESPECT TO (a) ROTATED, (b) SCALED, AND (c) BOTH ROTATED AND SCALED VERSIONS OF THE DIGITS IN THE TRAINING SET OF FIG. 5(a) AS WELL AS WITH RESPECT TO (d) THE DIGITS IN THE TESTING SET OF FIG. 5(b) AND WITH (e) ROTATED, (f) SCALED, AND (g) BOTH ROTATED AND SCALED VERSIONS OF THESE DIGITS

	Training Set			Test Set			
Type of Network	rot. (a)	scal. (b)	rot. + scal. (c)	original (d)	rot. (e)	scal. (f)	rot. + scal. (g)
TON, $N_I = 66$, $M = 40$	93	97	91	90	82	83	79
TON, $N_I = 912$, $M = 0$	82	93	77	78	71	74	66
ZFC, $n^* = 11$, $M = 40$	89	89	79	71	68	72	63

scaling is considered. Adding a hidden layer to the flat TON leads to improved performance. Fig. 6 shows some of the distorted and transformed patterns a network with $N_I = 32$ and $\omega = \pi/6$ has been able to classify correctly. For the ZFC's we find that the addition of a hidden layer of nodes does not lead to an appreciable improvement in performance for the recognition of distorted images. All TON's except for the largest ($\omega = \pi/144$ and $\pi/72$) are superior to the ZFC's.

3) Noise: Results for the recognition of noisy images are shown in the last two columns of Tables V and VI for TON's and ZFC's respectively. In contrast to their behavior under distortion, TON's become less tolerant to noise as N_I decreases from infinity towards relatively small values (Table V, column 4). However, when images are scaled, rotated, and subjected to noise at the same time, the net effects of structured distortion brought about by the lattice transformations and of noise seem to balance each other, and similar success rates are obtained for all values of N_I (Table V, column 5).

TON's appear to be generally less noise-tolerant than ZFC's (the relatively low success rates of ZFC's for small amounts of noise can be attributed to the distortion brought about by the translation and scaling transformations in the image normalization process), although both can tolerate relatively small amounts of noise (see also [20], [21] for the noise sensitivity of Zernike features), especially when compared with non-invariant associative memories and pattern recognition systems [26], [27].

B. Recognition of Handwritten Numerals

From the typed numeral recognition tests described so far, it has become apparent that two-layered TON's show only a marginal improvement in performance compared with one-layered networks. However, it has also become clear that to ensure efficient invariant recognition, networks with a large number of effective inputs have to be abandoned in favor of smaller networks. The option of including hidden nodes in the network architecture has to be retained as a means of ensuring adequate capacity to deal with large amounts of data in the training set. Moreover, our strategy abandons the principle whereby the strength of high-order networks in dealing with tasks for which the formation of nonlinear decision boundaries is required lies in their sideways expansion to include enough "image-enhancing" terms [28]–[30], and the option of including a second layer of nodes to deal with these problems must be retained.

Our handwritten digit recognition experiments provide a good illustration of these points. We find it impossible to train successfully flat TON's with a relatively small number of nodes (in the range 32–112) on the 360-image training set. Although more extended flat networks can cope with the task, their performance is poor compared with that of two-layered networks with a relatively small number of effective inputs. Typical results are shown in Table VII, where the performances of a flat network with $N_I = 912$ and of a two-layered network with $N_I = 66$ and 40 hidden nodes are compared. Although the second network has approximately twice as many weights as the first one, it exhibits inferior recognition rates.

The performance of the TON with $N_I = 66$ and 40 hidden node also compares favorably with the performance of the ZFC (a typical result is also shown in Table VII). The 90% recognition rate for the digits of Fig. 5(b), on which the network has not been trained, is also superior to the recognition rate of 78% of feedforward networks to which the pixel representations of the digits are presented as input without any feature extraction [27].

VII. CONCLUSION

In this paper we have studied the invariant recognition properties of high-order feedforward networks. We have introduced a method for controlling the size of third-order networks designed for recognition of translated, rotated, and scaled images. The method involves partitioning the set of triangles on the image plane into classes of approximately similar triangles characterized by the same value of their two smallest angles within a finite angular tolerance. For relatively large values of the angular tolerance, the method leads to the construction of economical networks with a relatively small number of weights, which exhibit improved recognition performance compared with the large structures obtained by using small values of angular tolerance. The improvement is evident in recognizing transformed (translated, scaled, and rotated) as well as distorted patterns.

It was found that third-order networks are superior to the use of Zernike moments followed by a conventional neural network classifier in invariant pattern recognition tasks including recognition of distorted patterns, but not patterns corrupted by noise randomly distributed on the screen. In both cases, the computational overhead incurred in coding

the images is much less than the time required to train the classification network. One further attraction of using high-order networks, in this respect, is that of allowing practical hardware implementations which would considerably speed up the preprocessing stage.

Finally, further research is possible toward optimal ways of defining the boundaries between the equivalence classes of approximately similar triangles in order to minimize the sensitivity of third-order networks to distortion. Additional areas for investigation are the use of a large number of pixels, particularly in respect of the effect on robustness against pixel noise, and also comparisons with more elaborate descriptive character representations, as obtained, for instance, using morphological methods.

REFERENCES

[1] R. P. Lippmann, "Pattern classification using neural networks," *IEEE Communications Magazine,* pp. 47–64, Nov. 1989.

[2] R. Lenz, "Group invariant pattern recognition," *Pattern Recognition,* vol. 23, pp. 199–217, 1990.

[3] E. Bienenstock and C. Von der Malsburg, "A neural network for the retrieval of superimposed connection patterns, *Europhysics Lett.,* vol. 3, pp. 1243–1249, 1987.

[4] E. Bienenstock and C. Von der Malsburg, "A neural network for invariant pattern recognition," *Europhysics Lett.,* vol. 4, pp. 121–126, 1987.

[5] A. Fuchs and H. Haken, "Pattern recognition and associative memory as dynamical processes in nonlinear systems," in *Proc. IEEE 1st Int. Conf. Neural Networks,* 1987, pp. 217–224.

[6] V. S. Dotsenko, "Neural networks: Translation-, rotation-, and scale invariant pattern recognition," *J. Physics A: Mathematical and General,* vol. 21, pp. L783–L787, 1988.

[7] R. Kree and A. Zippelius, "Recognition of topological features of graphs and images in neural networks," *J. Physics A: Mathematical and General,* vol. 21, pp. L813–L818, 1988.

[8] A. C. C. Coolen and F. W. Kujik, "A learning mechanism for invariant pattern recognition in neural networks," *Neural Networks,* vol. 2, pp. 495–506, 1989.

[9] M. Minsky and S. Papert, *Perceptrons.* Cambridge, MA: MIT Press, 1969.

[10] H. H. Chen *et al.*, "High order correlation model for associative memory," in *AIP Conf. Proc.,* 1986, pp. 86–99.

[11] M. B. Reid, L. Spirkovska, and E. Ochoa, "Rapid training of higher-order neural networks for invariant pattern recognition," in *Proc. IJCNN Int. Conf. Neural Networks,* vol. 1, 1989, pp. 689–692.

[12] P. Lisboa and C. Michael, "Lattices in group manifolds: Applications to lattice gauge theory," *Nuclear Physics,* B210 [FS6], pp. 15–28, 1982.

[13] T. Maxwell, C. L. Giles, Y. C. Lee, and H. H. Chen, "Nonlinear dynamics of artificial neural systems," in *AIP Conf. Proc.,* 1986, pp. 299–304.

[14] D. Psaltis, C. H. Park, and J. Hong, "Higher order associative memories and their optical implementations," *Neural Networks,* vol. 1, pp. 149–163, 1988.

[15] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inform. Theory,* vol. IT-8, pp. 179–187, 1962.

[16] B. A. Dudani, K. J. Breeding, and R. B. McGhee, "Aircraft identification by moment invariants," *IEEE Trans. Comput.,* vol. C-26, pp. 39–46, 1977.

[17] S. S. Reddi, "Radial and angular moment invariants for image identification," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. PAMI-3, pp. 240–242, 1981.

[18] Y. S. Abu-Mostafa and D. Psaltis, "Image normalization by complex moments," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. PAMI-7, pp. 46–55, 1985.

[19] A. P. Reeves, R. J. Prokop, S. E. Andrews, and F. P. Kuhl, "Three-dimensional shape analysis using moment and Fourier descriptors," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. 10, pp. 937–943, 1988.

[20] A. Khotanzad and Y. H. Hong, "Invariant image recognition by Zernike moments," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. 12, pp. 489–497, 1990.

[21] C.-H. Teh and R. T. Chin, "On image analysis by the method of moments," *IEEE Trans. Pattern Anal. Machine Intell.,* vol. 10, pp. 496–513, 1988.

[22] A. Kotanzad and J. H. Lu, "Distortion invariant character recognition by a multi-layer perceptron and back-propagation learning," in *Proc. IEEE 2nd Int. Conf. Neural Networks,* vol. 1, 1988, pp. 625–631.

[23] M. Teague, "Image analysis via the general theory of moments," *J. Opt. Soc. Amer.,* vol. 70, pp. 920–930, 1980.

[24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,* D. E. Rumelhart and J. L. McLelland, Eds., vol. 1. Cambridge, MA: MIT Press, 1986, pp. 318–362.

[25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature,* vol. 323, pp. 533–536, 1986.

[26] J. K. O. Leung, "Experimental study on neural net recognition of handwritten numerals," M.Sc. thesis, Liverpool University, Liverpool, U.K., 1988.

[27] P. J. G. Lisboa, "Neural networks in vision," in *Neural Networks: Current Applications,* P. J. G. Lisboa, Ed. London: Chapman and Hall, 1991.

[28] C. L. Giles and T. Maxwell, "Learning invariance, and generalization in high-order neural networks," *Appl. Opt.,* vol. 26, pp. 4972–4978, 1987.

[29] D. Sobajic, "Neural nets for control of power systems," Ph.D. thesis, Computer Science Dept., Case Western Reserve University, Cleveland, OH, 1988.

[30] Y-H. Pao, *Adaptive Pattern Recognition and Neural Networks.* New York: Addison Wesley, 1989.

Stavros J. Perantonis was born in Athens, Greece, in 1960. He received the B.Sc. degree in physics from the University of Athens in 1984, the D.Phil. degree in theoretical physics from the University of Oxford in 1987, and the M.Sc. degree in computer science from the University of Liverpool in 1990.

From 1987 to 1990 he was a postdoctoral research associate in the Department of Theoretical Physics, University of Liverpool. He is currently with the Demokritos Research Center, Athens, Greece. He has published work on theoretical high-energy physics, lattice gauge theory, and more recently, neural network theory and applications. His current research interests in neural networks include optimization, pattern recognition, network stability, and training algorithms.

Paulo J. G. Lisboa carried out research in theoretical physics at the Universities of Liverpool and Bristol, followed by a post in control engineering with the electricity supply industry. He then returned to the University of Liverpool as a Lecturer in the Department of Electrical Engineering and Electronics. His main research interests are image analysis for telecommunications, optical character recognition, and process control, comprising standard signal processing methods and artificial neural networks.