Optical implementations of radial basis classifiers

Mark A. Neifeld and Demetri Psaltis

We describe two optical systems based on the radial basis function approach to pattern classification. An optical-disk-based system for handwritten character recognition is demonstrated. The optical system computes the Euclidean distance between an unknown input and 650 stored patterns at a demonstrated rate of 26,000 pattern comparisons/s. The ultimate performance of this system is limited by optical-disk resolution to 10^{11} binary operations/s. An adaptive system is also presented that facilitates on-line learning and provides additional robustness.

Key words: Neural networks, radial basis functions, pattern recognition, optical disk.

1. Introduction

We describe two optical architectures for the realization of distance-based classifiers; in particular, radial basis function classifiers. The first is an optical-disk-based implementation. The two-dimensional (2-D) storage format of the optical disk makes parallel access to data an attractive possibility. The optical disk can be thought of as a computer-addressed 2-D binary spatial light modulator (SLM) or storage medium with a space–bandwidth product (SBP) of 10^{10} pixels. A number of potential applications that take advantage of these characteristics exist and have been discussed elsewhere in the literature.[1,2] An optical-disk-based implementation of radial basis classifiers is quite natural owing to the large storage requirements typical of such pattern-recognition algorithms. In the system described here the optical-disk-based radial-basis-function classifier is demonstrated as a handwritten character recognition system. The second architecture is a parallel adaptive neural network that facilitates on-line learning and offers added robustness to noise and optical system imperfections.

2. Radial Basis Functions

The radial basis function (RBF) approach to pattern recognition differs from neural networks that are based on supervised, output error-driven learning algorithms such as back error propagation (BEP) in a number of respects.[3,4] It is typical for RBF-based systems to incur short learning times while requiring rather large network realizations. This approach therefore bears some similarity to memory-intensive sample-based systems such as K-nearest neighbor (KNN) classifiers.[5] In such systems, learning time and learning algorithm complexity are traded for classification time and memory requirements. The motivation for using a RBF network to perform pattern-recognition tasks comes from the relatively well established mathematical framework associated with regularization theory and hypersurface reconstruction.[6] In hypersurface reconstruction the problem is to construct an approximate function $\hat{f}(\mathbf{w}, \mathbf{x})$, which takes a vector $\mathbf{x}$ into a prescribed output $f(\mathbf{x})$. The vector $\mathbf{w}$ is a parameter vector used to tune the estimate $\hat{f}$. For simplicity we consider one-dimensional (1-D) outputs only. In order to construct $\hat{f}$ we provide a set of training samples $\{\mathbf{x}_i \rightarrow f(\mathbf{x}_i); i = 1, \ldots, M\}$ taken from $f(\mathbf{x})$ (i.e., the underlying hypersurface to be approximated). The problem then reduces to the choice of the form of $\hat{f}$ and the appropriate parameters $\mathbf{w}$, such that $\hat{f}(\mathbf{w}, \mathbf{x}_i) = f(\mathbf{x}_i)$ for $i = 1, \ldots, M$. This problem is identical to the pattern-recognition problem in which one is given a set of training patterns and is asked to find a classifier $\hat{f}$ with the appropriate parameters $\mathbf{w}$, such that the resulting machine classifies the training set correctly. In both cases we desire that future samples be mapped correctly and that the system behave well in the presence of noise. In order to obtain these desirable characteristics in hypersurface reconstruction, a criterion of smoothness is often placed on the estimator $\hat{f}$. The RBF approach may be derived as the optimal solution to the regularized problem for a

M. A. Neifeld is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, Arizona 85721. D. Psaltis is with the Department of Electrical Engineering, California Institute of Technology, Pasadena, California 91125.

Received 21 May 1992.
0003-6935/93/081370-10$05.00/0.

specific smoothing operator.[7] The RBF solution defines an approximating function $\hat{f}(\mathbf{w}, \mathbf{x})$ as a weighted sum of radially symmetric basis functions in $\mathscr{R}^N$. Given a training set $X = \{\mathbf{x}^i, f(\mathbf{x}^i);\ i = 1, \ldots, M\}$ comprising a set of M points $\{\mathbf{x}^i \in \mathscr{R}^N; i = 1, \ldots, M\}$ and the values of the unknown function $f(\mathbf{x})$ at those points, we see that the RBF approach specifies an estimator as

$$\hat{f}(\mathbf{w}, \mathbf{x}) = \sum_{i=1}^{\tilde{M}} a_i \exp(-|\mathbf{x} - \mathbf{t}^i|^2/\sigma_i^2), \qquad (1)$$

where the centers or templates $\{\mathbf{t}^i\}$, the widths $\{\sigma_i\}$, and the weights $\{a_i\}$ comprise the parameter vector $\mathbf{w} = \{\mathbf{t}^i, \sigma_i, a_i;\ i = 1, \ldots, \tilde{M}\}$ and are determined from the training set.

The RBF classifier seeks to approximate the underlying function as a sum of Gaussian bumps. According to the above expression $\hat{f}$ comprises $\tilde{M}$ of these bumps, each centered at $\mathbf{t}^i$ with width σ_i and weighted by a_i to form the final output. We may estimate the parameters $\mathbf{w}$ from the training set such that $\hat{f}(\mathbf{x}^i) \approx f(\mathbf{x}^i)$ by using any number of supervised and/or unsupervised algorithms.[3,7]

The RBF approach may also be considered as a neural-network architecture, as shown in Fig. 1. We define the RBF unit shown in Fig. 1(a) as a neuron, with a response given by

$$y_i = \exp(-|\mathbf{x} - \mathbf{t}^i|^2/\sigma_i^2),$$

where $\mathbf{t}^i$ is called the neuron center and σ_i the neuron width. These units are depicted in the second layer of Fig. 1(b). The output layer of the RBF network consists of a single linear unit whose output is simply the weighted sum of its inputs. The overall network mapping conforms then to Eq. (1), as desired. In Fig. 2(a) we show a RBF network for estimating a function of two input variables, and in Fig. 2(b) we depict an example of an input space configuration of the mapping induced by such a network. The dots in Fig. 2(b) represent the training samples, and the dashed circles represent the e^{-1} contours of the four Gaussian basis functions used to construct the RBF network. As a specific example of training such a network, we used a k-means algorithm with $k = 4$ to determine the centers of the basis functions.[8] This procedure results in determination of the four centers, shown as asterisks in the figure. In order to determine the widths associated with each center, we used a KNN algorithm. The five nearest neighbors to each center were chosen, and the average of these five distances was used as σ_i for the associated bump. Note that these procedures result in the determination of the centers $\mathbf{t}^i$ and the widths σ_i in a completely unsupervised fashion. In this way the first layer of a RBF network may be trained without using an error-driven procedure, thereby reducing training time. Training of the output layer can be accomplished through the use of either a mean-square error minimization procedure (e.g., adaline) or a relatively simple perceptron learning algorithm.[9]

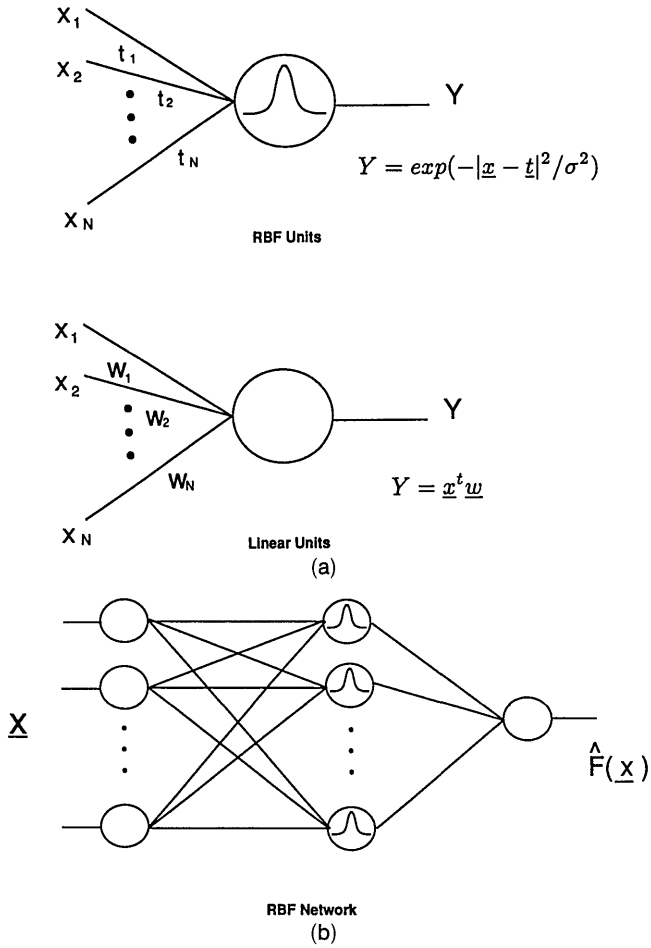


Fig. 1. (a) Definition and schematic of RBF units (top) and linear units (bottom). (b) General RBF network.

3. Radial-Basis-Function-Based Handwritten Character Recognition

In this section we describe the implementation of the RBF classifier trained to solve a handwritten character recognition problem. We consider the ten-class problem of identifying handwritten digits 0–9. Using a SUN3/60 workstation, several authors were asked to draw the numerals 0–9 on a 16 × 16 grid. The resulting database of 950 images (95 per class) was randomly separated into a 300-element testing set and a 650-element training set, forming our reference library. Examples of characters from the training and testing sets are shown in Fig. 3.

In order to provide both shift and scale invariance, we first preprocessed both training and testing sets so that each 16 × 16 image was centered (by repositioning each character within the 16 × 16 grid such that the number of blank rows or columns of pixels was the same on all sides of the character) and scaled to a 10 × 10 window (by stretching each character such that its maximum extent is 10 pixels). Following this preprocessing, we unraster the 10 × 10 pixel

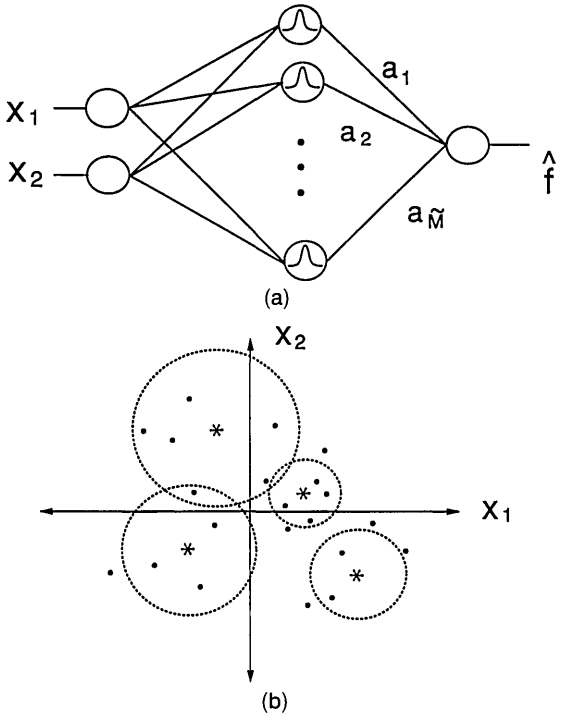


Fig. 2. (a) RBF network for estimating a scalar function of two variables. (b) Example input space configuration resulting from the network shown in (a).

input field to form a 100-bit binary vector, and each such vector $\mathbf{t}^i$ corresponding to each of the 650 preprocessed training or reference images is stored on the optical disk as a radial line. For each vector $\mathbf{t}^i$ we also store its complement $\bar{\mathbf{t}}^i$ in the adjacent position. This method of encoding permits us to simulate bipolar templates, using disks that store binary, unipolar reflectivity values. The template pixel size in this experiment is chosen to be 177 tracks by 116 pixels along the track. Track-to-track spacing is 1.5 μm and pixel separation is approximately 1.0 μm. This storage scheme permits us to record 1376 templates per disk.

The architecture we implemented is shown in Fig. 4. The preprocessed 100-bit binary vector $\mathbf{x}$ is presented to the system shown in Fig. 4, and the first layer of RBF units computes the RBF projections $y_i = \exp(-|\mathbf{x} - \mathbf{t}^i|^2/\sigma_i^2)$. We choose to use as RBF centers $\{\mathbf{t}^i\}$ all 650 reference images of the training set. This choice of centers also facilitated an earlier KNN-based handwritten character recognition system that has been reported elsewhere.[10] After the RBF projections are calculated in the middle layer, this 650-dimensional intermediate representation is then transformed by using the interconnection matrix $\mathscr{W}$ to arrive at a ten-dimensional output representation as shown. Each output neuron corresponds to one of the ten classes, and a winner-take-all network then performs the classification. Since we choose to use the entire 650-template training set as

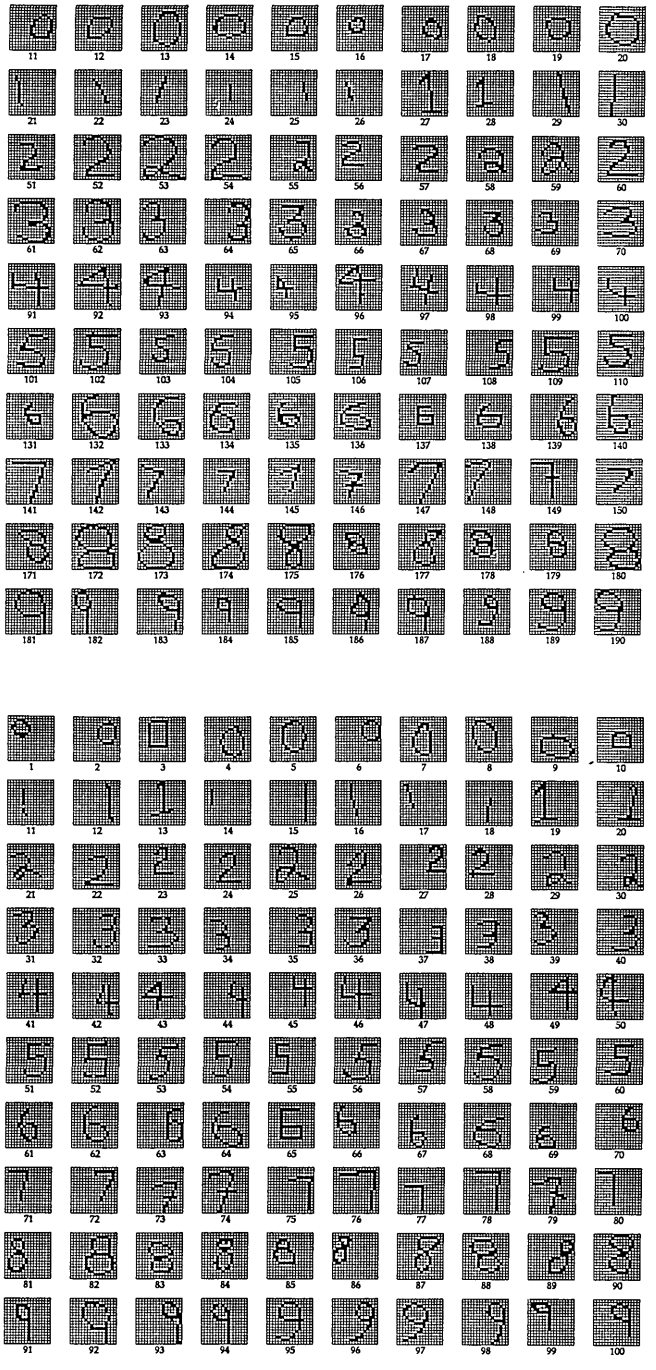


Fig. 3. Examples of handwritten numerals from the training set (top) and the testing set (bottom) used in the optical RBF experiment.

RBF centers, the only iterative learning required for the first layer of this network is for widths $\{\sigma_i\}$. Of course, the second layer must also be trained to perform the desired classification on the resulting RBF representations.

There are many potential training algorithms for $\{\sigma_i\}$ and $\mathscr{W}$. The most successful algorithm we found for computing the widths $\{\sigma_i\}$ was to make σ_i proportional to the distance between template $\mathbf{t}^i$ and its

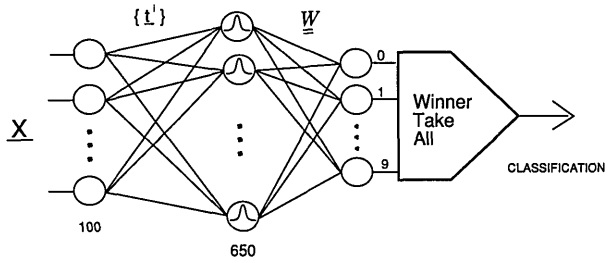


Fig. 4. RBF network for handwritten digit recognition.

nearest neighbor. That is,

$$\sigma_i = \tilde{\sigma} \min_{j \neq i} |\mathbf{t}^j - \mathbf{t}^i|,$$

where the proportionality constant $\tilde{\sigma}$ is selected *a priori*. Training of the output layer was most successful when $\mathcal{W}$ was initialized with a binary address algorithm and then trained by using the perceptron learning algorithm. The binary address algorithm does not require specific knowledge of the intermediate representations generated during training; it only requires knowledge of the class assignment of each of the 650 RBF centers. This reduces second-layer computation time and improves network performance. The binary address algorithm defines the initial $\mathcal{W}$ as

$$w_{ij} = \begin{cases} 1 & \text{if } \mathbf{t}^j \in \Omega_i \\ -1 & \text{otherwise} \end{cases}.$$

Following this initialization, we use the perceptron algorithm to incorporate detailed knowledge of the training representations into the output-layer weights.

Using these procedures for training the RBF network, we have in computer simulation a best RBF performance of 89%, as shown in Table 1. Although the trend with increasing $\tilde{\sigma}$ is an improvement in network performance, in general, we found that the broader the basis functions, the longer the perceptron algorithm takes to converge. For this reason, Table 1 does not contain any entries for $\tilde{\sigma} > 1.2$. We note here that the best RBF network performance of 89% is substantially better than the best KNN system performance of 83% obtained using the same template library. This performance can also be compared with a single layer of ten neurons, each trained with the perceptron algorithm by using the 650-image reference library. The recognition rate in this case is 75% on the 300-element testing set. In general, we would expect an improvement in RBF network performance with variable centers in which $\tilde{M}$, $\mathbf{t}^i$, and $\tilde{\sigma}_i$ are all optimized. This case is not studied here since we are interested primarily in the performance of the optical implementation.

Table 1. Classification Results Obtained with a One-Nearest Neighbor Rule for Training the Radial-Basis-Fuction Widths

$\tilde{\sigma}$	Training Set[a]	Testing Set[b]
0.5	650	226
0.7	650	257
1.0	650	266
1.2	650	267

[a]Number of elements correct out of a 650-element set.
[b]Number of elements correct out of a 300-element set.

In Fig. 5 we show the RBF widths computed by using the procedure described above. Each row in the figure represents the widths associated with centers in a single class. There are therefore 65 blocks per row and 10 rows in the figure. Each block is a gray-scale coding of the width associated with the corresponding template, with dark being zero width. With this encoding, each row of the figure corresponds to the values of σ_i for templates in a single class. Note that the second row in Fig. 5, corresponding to handwritten 1's, is particularly dark, which indicates that these vectors tend to be well clustered or are in general located close to other vectors. Also in Fig. 5 we see that the width associated with one particular template representing a handwritten 6 is quite broad, indicating that this vector is basically isolated in the input space. With the same display format as that in Fig. 5, Fig. 6 shows the ten weight vectors of the second layer generated for the best RBF network. The single bright row in each weight vector indicates that the weight vector is tuned to intermediate representations from essentially one class.

4. Radial-Basis-Function Optical Classifier

A schematic of the optical system used to compute the distance between an unknown preprocessed input image and each template stored on the disk in the format described above is shown in Fig. 7. In this

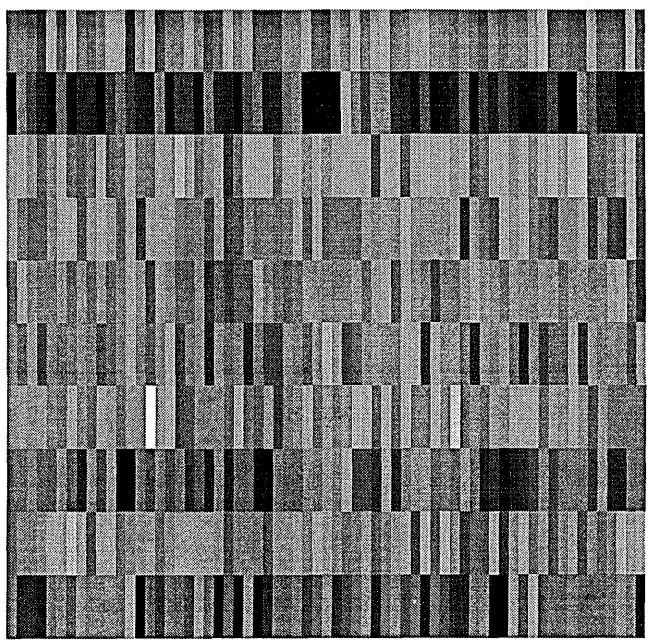


Fig. 5. RBF widths computed using the one-nearest-neighbor rule with $\tilde{\sigma} = 1.2$.

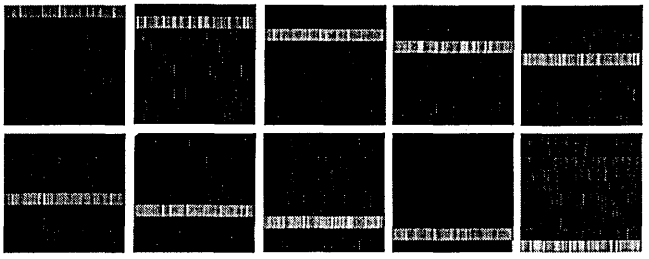


Fig. 6. Second-layer weights computed with the perceptron algorithm after initialization with the binary address algorithm. The weights for neurons 1–10 appear consecutively from left to right, top to bottom.

architecture an Epson liquid-crystal television is used as a 1-D SLM to present the unrastered input character to the system. An image of the input vector is formed as a radial line on the disk as shown, and the total diffracted intensity is collected by the output lens and measured with a Photodyne 1500XP detector. The detector output represents the inner product between the input vector and the illuminated template vector. The postprocessing system for this experiment consists of two parts. First, a sample–hold circuit is used to detect the peaks of the raw detector output. The amplitudes of these peaks represent the desired inner products. The sample–hold circuit is clocked by a signal that is phase locked to the sector markers that are recorded on the disk; the markers appear as 32 bright radial lines and provide a strong diffracted signal. The second stage of postprocessing consists of an analog-to-digital converter board in an IBM PC computer followed by software that implements the nonlinearity of the second layer and computes the final output.

The 650 reference images were preprocessed as described above and stored on the disk along with their complements as 100-bit binary vectors. With a disk rotation rate of 20 Hz, these 1300 vectors were processed at a rate of 26,000 inner products/s, equivalent to 2,600,000 binary operations/s. It should be pointed out that this relatively slow processing speed arises from a severe underutilization of disk capacity. In this experiment a large template pixel size was used (177 tracks by 116 pixels across track) in order to provide alignment simplicity. A system that utilizes the minimum disk resolution of roughly 1 μm pixels, together with a disk rotation rate of 100 Hz, would achieve an inner-product rate of 10^7/s, corresponding to a raw processing rate of 10^{11} binary operations/s. The 300 testing images were preprocessed as described in Section 3, and stored in an IBM PC computer, which drove the liquid-crystal television and provided input vectors to the system. An example of the raw detector output for the all 1's input vector is shown in Fig. 8. The two tallest peaks in this trace correspond to sector markers on the disk and represent the inner product between the all 1's vector and itself. From this data we can calculate the effective brightness per input pixel measured at the detector as 0.6 nW. This value is in good agreement with the known optical losses in the system. The other peaks in Fig. 8 provide normalization data that are stored in memory and read out during postprocessing. The IBM-PC samples the inner-product signal once per peak, averages four rotations worth of data (total acquisition time ≈0.2 s), and computes the Euclidean distances from the inner products as

$$|\mathbf{x} - \mathbf{t}^i|^2 = |\mathbf{x}|^2 + |\mathbf{t}^i|^2 - 2\mathbf{x} \cdot \mathbf{t}^i,$$

where $\mathbf{x}$ is the unknown input image and $\mathbf{t}^i$ is a stored template. Since our optical system actually measures $\mathbf{t}^i \cdot \mathbf{x}$ and $\overline{\mathbf{t}^i} \cdot \mathbf{x}$, we may form the distance for binary vectors as

$$|\mathbf{x}|^2 = (\mathbf{x} \cdot \mathbf{1}) \qquad [\mathbf{1} = (1, 1, \ldots, 1)]$$
$$= \mathbf{x} \cdot (\mathbf{t}^i + \overline{\mathbf{t}^i}),$$

so that

$$|\mathbf{x} - \mathbf{t}^i|^2 = |\mathbf{t}^i|^2 + \mathbf{x} \cdot \overline{\mathbf{t}^i} - \mathbf{x} \cdot \mathbf{t}^i.$$

Once again, $|\mathbf{t}^i|^2$ for $i = 1, \ldots, 650$ is stored in normalization memory and read out during the postprocessing stage.

The optical-disk-based inner-product calculations are collected by the postprocessing system that com-

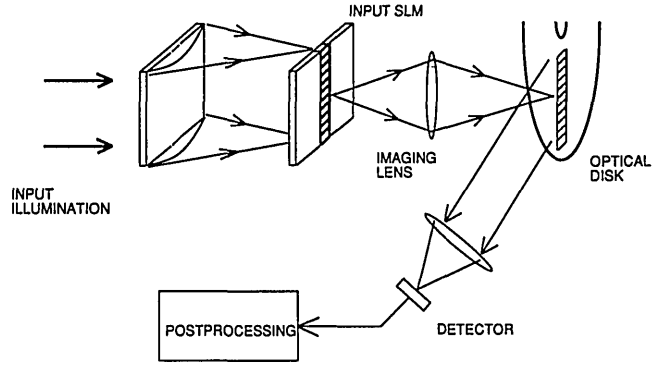


Fig. 7. Optical system used to compute the distance between an input and an array of stored templates.

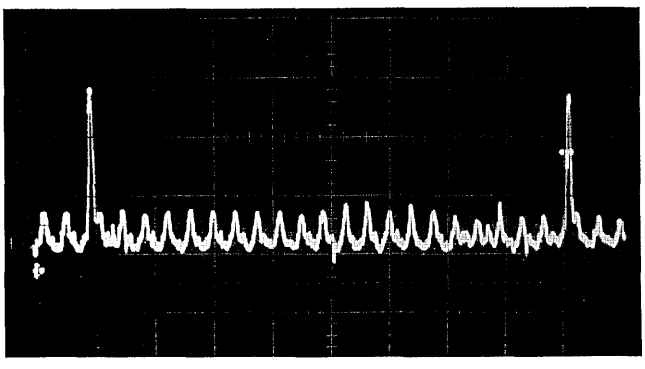


Fig. 8. Example of raw detector output indicating the optically computed inner products.

putes the required Gaussian weighting and simulates the output layer in which a classification is made. These postprocessing steps were carried out off line in software for our experiments. The classification rate for the optical RBF system was 83%. This is compared with a recognition rate of 79% obtained using an optical KNN network based on the same template data. A comparison between the performances of the optical system and a computer simulation is shown in Table 2. The table entries indicate the number of correct classifications out of 30 for each of the 10 classes of characters 0–9. The various noise sources in the optical system result in a 6% loss of recognition rate. In order to better understand the effect of these imperfections on the RBF network performance, we constructed a computer model that incorporates error sources such as finite contrast, nonuniform illumination profile, detector noise, and quantization noise. Using values for the error variables as measured from the optical apparatus, we found that nonuniformity of the illumination profile was the limiting factor in our experiment. A plot of classification rate versus log of the $1/e^2$ Gaussian profile width is shown in Fig. 9. We can see from Fig. 9 that for the measured profile parameter value of 1.8, the expected recognition rate drops to 86%. This rate is then the noise-limited optical system performance and is close to the experimentally demonstrated 83%. The cumulative effect of these errors can be measured a second way; directly from the distance calculations. In Fig. 10 we show the 650 distances computed for a single input image (a handwritten 3), obtained by using both the ideal computer simulation and the optical system. From the figure we see that there is a substantial variation between these two plots. This variation can be quantified by computing the rms distance error over the entire testing set as

$$\Delta D_{\mathrm{rms}} = \frac{\left[\frac{1}{M}\sum_{i=1}^{M}(d_i^{\mathrm{Opt}} d_i^{\mathrm{Sim}})^2\right]^{1/2}}{\frac{1}{M}\sum_{i=1}^{M} d_i^{\mathrm{Sim}}},$$

where d_i^{Sim} and d_i^{Opt} are the Euclidean distances between the 300 input images and the 650 templates calculated from the simulation and the optical system, respectively. There are $M = 195{,}000$ such measurements in our case. For the results presented here the rms distance error was found to be $\Delta D_{\mathrm{rms}} = 28.5\%$. Although this error is quite large, the recognition rate obtained by using the optical system is in satisfactory agreement with the expected rate, attesting to the robustness of the RBF approach.

5. Parallel Optical Distance Computation

In order to provide additional robustness to the optical system as well as to increase the computation speed, we propose a parallel nondisk implementation.

Table 2. Performance Comparison between Optical RBF Classifier and Simulation[a]

Class	Experiment	Simulation
0	25	29
1	28	29
2	28	28
3	25	27
4	28	23
5	20	25
6	25	24
7	23	28
8	24	25
9	22	29
Total	248	267
Overall recognition rate	83%	89%

[a]The table elements are the number of correct classifications out of 30 for each class.

We observe from the previous discussion that the hardware implementation of a RBF network comprises two primary components: the subsystem for computing M parallel Euclidean distances and the basis-function evaluation subsystem. Shown in Fig. 11 is an optical system that can be used to obtain the required parallel distance computation for the case of binary vectors. A similar system can be used to compute the distances for continuous-valued vectors; however, we concentrate on the binary system for now. In Fig. 11 an N-dimensional binary vector $\mathbf{x}$ is represented as a vertical intensity array in the input plane, and each center $\mathbf{t}^i$ is stored in a vertical column of the $\underline{\underline{t}}$ transparency shown. This system is dual rail since it requires $\mathbf{x}$ and $\underline{\underline{t}}$ and their complements $\bar{\mathbf{x}}$ and $\bar{\underline{\underline{t}}}$, respectively. We now show that by using this

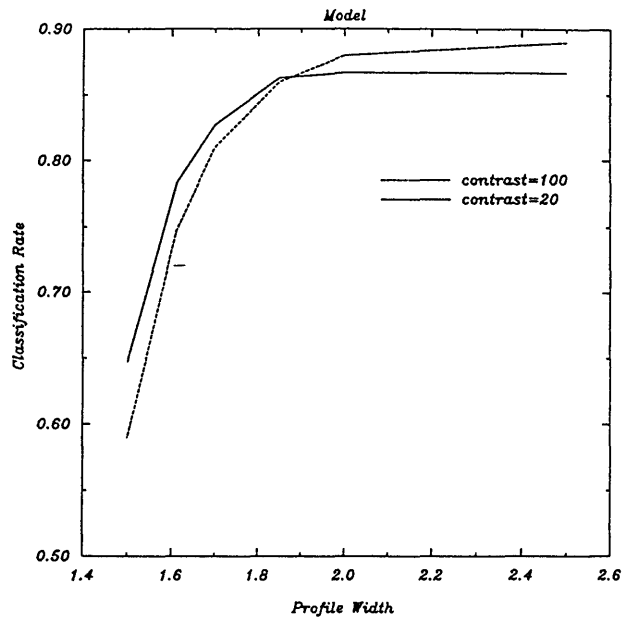
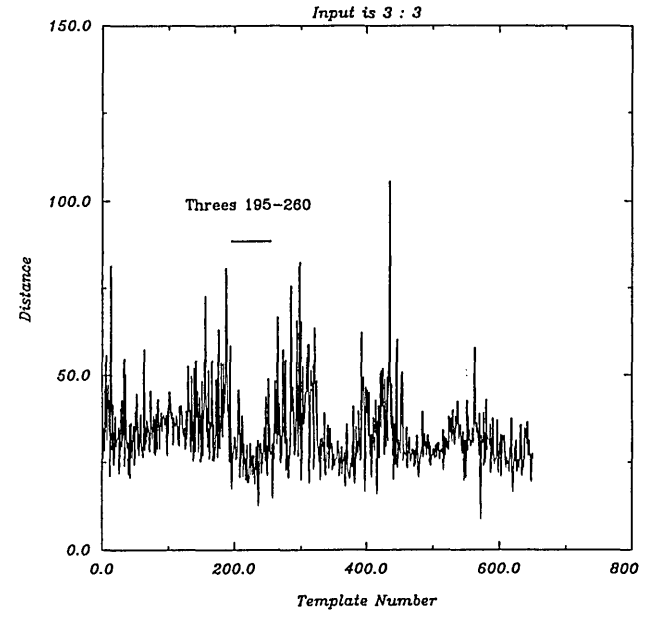
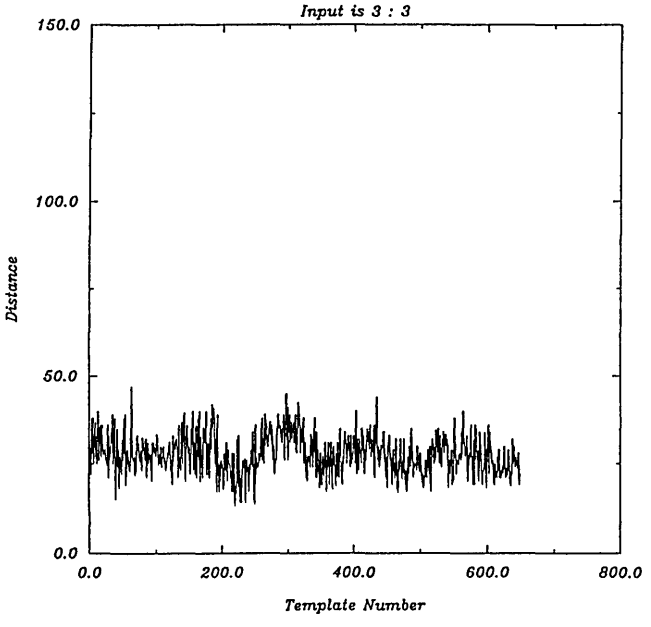


Fig. 9. Predicted recognition rate versus illumination profile width.



(a)



(b)

Fig. 10. (a) Experimental and (b) actual distance versus template number for a single input image (handwritten 3, number three). Template numbers 195–260 represent the class of handwritten 3's.

representation, we can perform the distance computation entirely optically.

Given an input $\mathbf{x}$ and a center $\mathbf{t}^i$, we can write the Euclidean distance between these two vectors as

$$d^i = |\mathbf{x} - \mathbf{t}^i|^2 = \sum_{j=1}^{N} (x_j - t_j^i)^2 = \sum_{j=1}^{N} d_j^i.$$

We can further write the componentwise distances d_j^i

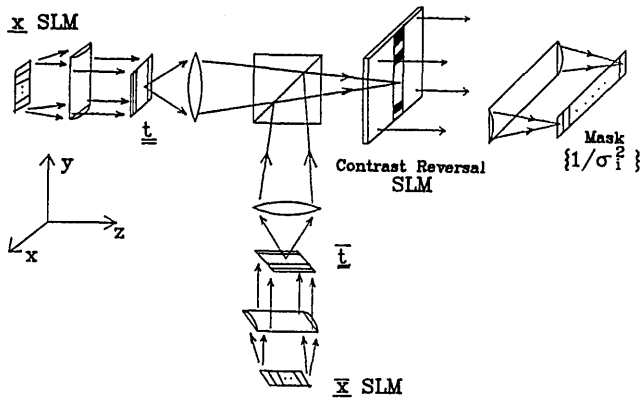


Fig. 11. Parallel optical distance computer.

in the binary case as the exclusive or function (XOR) of the component bits. That is,

$$\overline{d_j^i} = x_j t_j^i + \overline{x_j}\,\overline{t_j^i}.$$

Writing d_j^i in this complement form makes the optical realization more clear. Returning to the system shown in Fig. 11, we see that light from the $\mathbf{x}$ SLM is collimated in the x direction and imaged in the y direction so that, immediately to the right of the transparency $\underline{t}$, the componentwise product is formed between the input and all of the centers. That is, we generate the array $\{x_j t_j^i;\ i = 1, \ldots, M; j = 1, \ldots, N\}$. Similarly, in the lower arm of the system the complement array $\{\overline{x_j}\,\overline{t_j^i};\ i = 1, \ldots, M;\ j = 1, \ldots, N\}$ is formed, and these two arrays are simultaneously imaged upon a contrast-reversing SLM. This superposition combined with the contrast reversal yields the desired component distances d_j^i to the right of the contrast-reversing SLM. A good candidate for this contrast-reversal SLM is the optically addressed ferroelectric liquid-crystal SLM.[11]

Returning once again to Fig. 11, we see that, after the bitwise XOR's are computed as described above, the desired array of distances is obtained by summing in the y direction with the cylindrical lens shown. A simple 1-D SLM can be used to represent the desired widths so that, immediately to the right of the output plane shown, we obtain the desired terms $\{|\mathbf{x} - \mathbf{t}^i|^2/\sigma_i^2;\ i = 1, \ldots, M\}$.

Although the system described above operates on 1-D arrays of data, a 2-D version, which is better suited for operating on image data, is also possible. This 2-D extension is straightforward and involves the use of lenslet arrays for accessing the spatially multiplexed template images stored in the $\underline{t}$ and $\underline{\bar{t}}$ planes. The details of the 2-D system as well as those of an extended 1-D system capable of operating on continuous-valued vectors are the subject of future research and are not discussed here; however, in order to indicate the expected performance limitations of these systems, we consider the 2-D binary version. If we assume that the inputs to our system are 100 × 100 pixel images, then the number of

centers in the RBF network is limited by the SBP of the optical system. Obtaining 900 centers requires a template mask with 3000 × 3000 pixels, which in turn demands an optical system with SBP = 9×10^6. This would be the largest feasible implementation. Notice that although the contrast-reversal plane must have a large SBP, the output plane requires a SBP equal to only the number of centers. This is an attractive characteristic of the present system, since on-line learning will require a programmable SLM in this plane for σ_i adaptation. In the present example this SLM would be required to have only 30 × 30 pixels.

6. Parallel Basis-Function Evaluation

Having defined the optical distance computer in Section 5, we turn our attention to the second primary component of the optical RBF implementation. This component performs the basis-function evaluation. Notice that the basis-function evaluation requires only point operations in the plane of distances. Further, if we consider the case of a single output neuron [i.e., $f(\mathbf{x})$: $\mathcal{R}^N \rightarrow \mathcal{R}^1$], then completion of the RBF computation after the distance computer requires point operations only, followed by a global sum. With this observation in mind we propose the optoelectronic postprocessing chip shown in Fig. 12.

The chip shown consists of an array of modules, each module comprising a photodetector to detect the output of the optical distance computer, analog multipliers to realize the required width and output weighting, and an exponentiation unit to realize the basis-function evaluation. The output of each such module is summed on a common line to generate the network response. We should note here that all the required functions in a module are compactly achievable by using analog very-large-scale integration (VLSI) or, if more precision is required, the photodetector may be followed by an analog-to-digital converter, and then each module could be implemented in digital electronics. The 2-D extension of this postprocessing chip is once again straightforward, and, since there are no intermodule communication requirements, connectivity issues in the 2-D arrangement do not arise. All computation in this postprocessing chip is local, excepting the final sum. Also note that this implementation has the flexibility to permit the realization of a variety of different basis functions as well as to support a useful on-line learning algorithm, which is discussed further in Section 7.

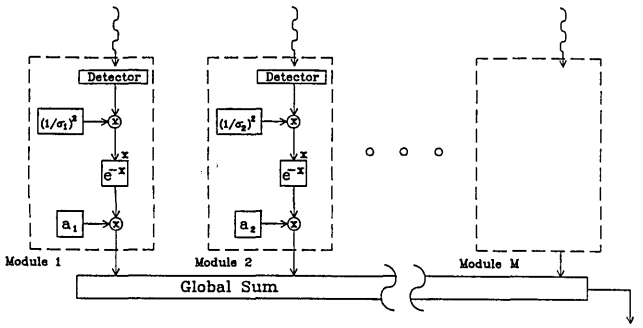


Fig. 12. Optoelectronic postprocessing chip.

The above all-electronic postprocessing chip is particularly well suited for the case of a network with one output only. For the case of multiple outputs we have two alternative systems. If the number of outputs is relatively small (~ 10), then the most attractive alternative is simply to use multiple postprocessing chips. This approach retains the simplicity and the flexibility of the VLSI implementation. Alternatively, if the number of outputs is large, we may consider a hybrid approach wherein each e^{-x} box in Fig. 12 is followed by a light-modulating element, permitting the exponentially weighted distances to be read out optically, with liquid-crystal modulators, for example.[12] In this way an efficient optical implementation of the output layer is facilitated. This approach has the advantage of providing scalability in terms of output units while retaining much of the convenience of the VLSI implementation. In this system, update of the output weights during a learning cycle is done optically through the use of photorefractive holograms in the output layer.[13]

7. Learning

The effective implementation of iterative learning algorithms is a common stumbling block in both electronic and optical neural-network architectures. Here we suggest a learning algorithm for RBF networks that is suitable for implementation with the optoelectronic hardware we described. Associated with each of the postprocessing stages (i.e., all-electronic and hybrid) is an implementation of the on-line learning algorithm. We describe the all-electronic single-output implementation here.

Referring to Eq. (1) for the RBF network response function, we can define a criterion function for the goodness of an RBF network as

$$E = \sum_{i=1}^{M} [\hat{f}(\mathbf{w}, \mathbf{x}^i) - f(\mathbf{x}^i)]^2 = \sum_{i=1}^{M} E_i^2,$$

where E_i is the error between the actual and desired network responses in the presence of training vector $\mathbf{x}^i$ only. E is the conventional sum-of-squared-error function evaluated over the entire training set. If we assume that the training set T is fixed and that each training vector is used as a single RBF center as before, then the learning procedure reduces to finding $\{\sigma_i\}$ and $\{a_i\}$ to minimize the error E. A simple gradient descent procedure is a candidate algorithm for the minimization of E. Using this procedure, we can write expressions for the update of the network parameters σ_p and a_p in response to the error measured for a single-input training vector $\mathbf{x}^l$. These are

$$\Delta(1/\sigma_p)^2 = -\alpha_\sigma E_l |\mathbf{x}^l - \mathbf{t}^p|^2 a_p \exp(-|\mathbf{x}^l - \mathbf{t}^p|^2/\sigma_p^2), \quad (2)$$

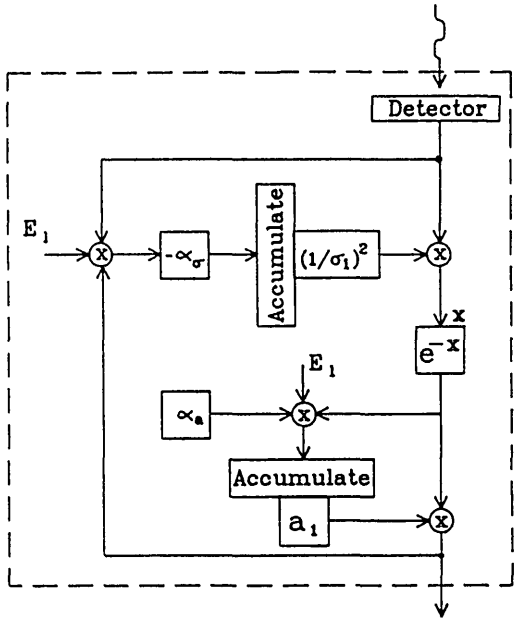


Fig. 13. On-line learning postprocessing module.

$$\Delta(a_p) = \alpha_a E_l \exp(-|\mathbf{x}^l - \mathbf{t}^p|^2/\sigma_p{}^2), \qquad (3)$$

where α_σ and α_a are acceleration constants for the width and output weight updates, respectively.[14] Notice that these expressions define a backward error propagation type of rule for RBF networks. In this learning algorithm, however, no special backward response function is required for the RBF units owing to the fact that the $\exp(x)$ function is its own derivative. All signals required to compute the updates defined in Eqs. (2) and (3) above are present in the forward path of the network. Furthermore, we observe that all required learning signals are present in the electronic portion of the proposed implementation and that no intermodule communication is necessary. The implication of these observations is that rapid, parallel update of all network parameters can be realized with a simple modification of the postprocessing module presented earlier. In Fig. 13 we show a block diagram of the modified postprocessing module. By incorporating the additional local connections shown and by adding accumulation registers to the $(1/\sigma_i)^2$ and a_i blocks, we can implement the on-line parallel RBF learning algorithm with little increase in overall circuit complexity over the nonadaptive system. In this way a 30 × 30 element adaptive postprocessing array, capable of facilitating on-line learning in a 900-center RBF network, should be possible.

8. Conclusions

We have demonstrated an optical system that can implement a RBF pattern classifier. The experimental system achieved a processing rate of 2,600,000 binary operations/s, corresponding to the computation of 13,000 Euclidean distances/s. The capability of the optical-disk-based system is limited by the maximum length of template vectors ($\sim 10^4$ bits), the maximum number of template vectors ($\sim 10^5$), and the maximum disk rotation rate (~ 100 Hz). These upper bounds correspond to a processing rate of $\sim 10^{11}$ binary operations/s.

This system was trained off line with the handwritten numerals 0–9, and it achieved a recognition rate in a computer simulation of 89% on a 300-element testing set. Similar performance (91% recognition rate) was achieved by using the same off-line training procedure in an RBF network with 2000 centers; the network was trained on segmented zip-code data obtained from the U.S. Postal Service database. The optical-disk-based 650-center system achieved a recognition rate of 83%. In this study it was found that factors such as nonuniform disk reflectivity, nonuniform illumination, and finite contrast were all significant contributors to a 28% rms error in the optical distance computation. Furthermore, since this large distance error resulted in only a 6% degradation in recognition performance, the RBF approach was seen to be robust in the presence of such errors.

We might expect an on-line learning scheme in which optical system inperfections are present during the learning phase to provide compensation for those imperfections and to result in a recognition rate closer to the simulation value. We have described such an adaptive optical RBF hardware implementation. If on-line learning, more careful system design, more powerful learning algorithms for learning the optimal center and width values, and a larger hidden layer (~ 2000 units) are combined, the optical system should be able to approach the 91% recognition rate obtained in simulation for the zip-code data.

This work was supported by the U.S. Army Research Office and the Defense Advanced Research Projects Agency. The authors thank Seiji Kobayashi of the Sony Corporation, Subrata Rakshit, and Alan Yamamura for assistance with the optical-disk recording system.

References

1. D. Psaltis, M. A. Neifeld, and A. A. Yamamura, "Image correlators using optical memory disks," Opt. Lett. **14,** 429–431 (1989).
2. D. Psaltis, M. A. Neifeld, A. A. Yamamura, and S. Kobayashi, "Optical memory disks in optical information processing," Appl. Opt. **29,** 2038–2057 (1990).
3. J. Moody and C. Darken, "Fast learning in networks of locally tuned processing units," Neural Computat. **1,** 281–294 (1989).
4. 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. McClelland, eds. (MIT, Cambridge, Mass., 1986), Vol. 1, pp. 318–362.
5. T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," IEEE Trans. Inf. Theory **IT-13,** 21–27 (1967).
6. T. Poggio and F. Girosi, "A theory of networks for approximation and learning," AI Memo No. 1140 (MIT Artificial Intelligence Laboratory, Cambridge, Mass., 1989).
7. T. Poggio and F. Girosi, "Networks for approximation and learning," Proc. IEEE **78,** 1481–1495 (1990).

8. J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, L. M. LeCam and J. Neyman, eds. (U. California Press, Berkeley, Calif., 1967), Vol. 1, pp. 281–297.
9. R. Duda and P. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973), Chap. 5, pp. 141–147.
10. M. A. Neifeld, S. Rakshit, A. A. Yamamura, and D. Psaltis, "Optical disk implementation of radial basis classifiers," in *Optical Information Processing Systems and Architectures, II*, B. Javidi, ed., Proc. Soc. Photo-Opt. Instrum. Eng. **1347,** 4–15 (1990).
11. D. Jared, K. M. Johnson, and G. Moddel, "Joint transform correlator using an amorphous-silicon ferroelectric liquid-crystal spatial light modulator," Opt. Commun. **76,** 97–102 (1990).
12. T. J. Drabik and M. A. Handschy, "Silicon VLSI ferroelectric liquid-crystal technology for micropower optoelectronic computing devices," Appl. Opt. **29,** 5220–5223 (1990).
13. D. Psaltis, D. J. Brady, and K. Wagner, "Adpative optical networks using photorefractive crystals," Appl. Opt. **27,** 1752–1759 (1988).
14. M. A. Neifeld, S. Rakshit, and D. Psaltis, "Handwritten zip code recognition using an optical radial basis function classifier," in *Applications of Artificial Neural Networks II*, S. K. Rogers, ed., Proc. Soc. Photo-Opt. Instrum. Eng. **1469,** 250–255 (1991).