Pergamon

Pattern Recognition, Vol. 29, No. 8, pp. 1369–1383, 1996
Elsevier Science Ltd
Copyright ©, 1996 Pattern Recognition Society
Printed in Great Britain, All right reserved
0031-3203/96 $15.00 + 00

0031-3203(95)00164-6

AUTOMATIC GENERATION OF RBF NETWORKS USING WAVELETS

SHAYAN MUKHERJEE[†]* and SHREE K. NAYAR[‡]
Department of Computer Science, Columbia University, New York, NY 10027, U.S.A.

(*Received for publication* 27 *November* 1995)

Abstract—Learning can be viewed as mapping from an input space to an output space. Examples of these mappings are used to construct a continuous function that approximates the given data and generalizes for intermediate instances. Radial-basis function (RBF) networks are used to formulate this approximating function. A novel method is introduced that automatically constructs a generalized radial-basis function (GRBF) network for a given mapping and error bound. This network is shown to be the smallest network within the error bound for the given mapping. The integral wavelet transform is used to determine the parameters of the network. Simple one-dimensional examples are used to demonstrate how the network constructed using the transform is superior to that constructed using standard *ad hoc* optimization techniques. The paper concludes with the automatic generation of GRBF networks for a multi-dimensional problem, namely, real-time 3D object recognition and pose estimation. The results of this application are favorable. Copyright © 1996 Pattern Recognition Society. Published by Elsevier Science Ltd.

Multivariate approximation   Radial-basis function networks   Wavelets
Supervised learning   3D object recognition   Pose estimation

1. INTRODUCTION

Often, learning can be equated to constructing a continuous function that maps a given set of inputs to outputs. This function is constructed from a set of example mappings, namely the training data. This smooth function effectively interpolates the known data. In this context, learning is equivalent to generating a continuous function that approximates the given data and generalizes for intermediary instances. A rigorous formulation of this approximating function results in a weighted sum of radial-basis functions (RBF). This type of approximating function can be cast as a class of neural networks termed RBF networks.[1] These networks are universal approximators, theoretically capable of approximating any function to a reasonable degree of precision[2] with only one layer of basis functions. (Neural networks with sigmoidal bases require two layers of basis functions to be universal approximators.) RBF networks have been used for a variety of practical applications ranging from the recognition of stick figures[3] to pricing derivative securities.[4]

Although RBF networks have a rigorous formulation, this advantage is lost in most practical implementations. The reason is the assumption that the network consists of as many basis functions as training examples. In practice, this proves to be a severe limitation as the number of training data is typically large. This is remedied by using generalized radial-basis function (GRBF) networks that have fewer basis functions than examples. However, the parameters of the GRBF network are typically set in an *ad hoc* manner. In this paper, an analytic method to construct GRBF networks using wavelets is introduced. The GRBF networks constructed using this method can be shown to be the smallest network for a given mapping and error bound.

This result is achieved through a novel construction and application of the integral wavelet transform.[5] Wavelet bases are constructed from approximations of radial basis functions. The magnitude of the coefficient corresponding to each basis determines the importance to the mapping of the radial basis functions comprising that wavelet. The wavelet coefficients along with Parseval's identity are used to determine the number of bases required and their parameters. In order to compute the wavelet coefficients, we extended the fast wavelet algorithm[6] for multi-dimensional sparse data.

The paper is organized as follows. The formulation of an input–output mapping as a RBF network is described in Section 2. The relation between the integral wavelet transform and RBF networks is presented in Section 3. The computational complexity and memory requirements of the automatically generated network with respect to the dimensionalities of the input and output spaces and number of basis functions used is discussed in Section 4. Section 5 presents a simple example that demonstrates the advantage of the transform approach. Finally, the transform approach is applied to a multi-dimensional problem, namely

† Author for correspondence.

‡ Formerly at the Department of Applied Physics and Mathematics, Columbia University, currently at the Theoretical Division of Los Alamos National Laboratory.

real-time 3D (three-dimensional) object recognition and pose estimation. The paper is concluded with a discussion of a variety of issues related to the proposed scheme.

2. RADIAL-BASIS FUNCTION NETWORKS

At the heart of all neural network schemes is the question of whether a multivariate function can be represented exactly by sums and products of univariate functions. In the case of RBF networks, this representation is formulated using approximation theory and regularization techniques.[7] The first part of this section is a brief overview of RBF and GRBF networks [see reference (1) for details].

Learning a mapping between an input and output space is often viewed as determining a function that performs the mapping. It can be posed as the problem of approximating a continuous multivariate function $f(\mathbf{x})$ by an approximating function $F(\mathbf{W},\mathbf{x})$ that has a fixed set of parameters $\mathbf{W}$. The approximation problem can be stated as follows:

If $f(\mathbf{x})$ is a continuous function defined on $\mathbf{x}$, *and $F(\mathbf{W},\mathbf{x})$ is an approximating function that depends continuously on* $\mathbf{W} \in P$ and $\mathbf{x}$, *then the approximation problem is to define the parameters $\mathbf{W}^*$ such that*:

$$\rho[F(\mathbf{W}^*,\mathbf{x}), f(\mathbf{x})] \leqslant \rho[F(\mathbf{W},\mathbf{x}), f(\mathbf{x})],$$

for all $\mathbf{W}$ *in the set P. $\rho[.,.]$ is a distance function that evaluates a norm between two functions. $\mathbf{W}^*$ are the optimal parameter values of the approximating function.*

When ρ is the L^2 norm, the above corresponds to minimizing the cost functional:

$$H[F(\mathbf{W},\mathbf{x})] = \int_{-\infty}^{+\infty} (f(\mathbf{x}) - F(\mathbf{W},\mathbf{x}))^2 d\mathbf{x}, \quad (1)$$

with respect to $\mathbf{W}$. In the above formulation of the approximation problem, the function $f(\mathbf{x})$ is continuous. However, in the case of learning a smooth mapping from a discrete set of examples, there exists no continuous function $f(\mathbf{x})$. For this reason, the approximation problem is ill-posed for discrete data; the data does not contain sufficient information for a unique mapping.

The approximation problem is made well-posed by introducing *a priori* assumptions about the mapping. Normally the assumptions pertain to the smoothness of the mapping. Regularization techniques are invoked to introduce smoothness constraints into the approximation problem. The resulting cost functional has the form:

$$H[F(\mathbf{W},\mathbf{x})] = \sum_{i=1}^{N} (f(\mathbf{x}_i) - F(\mathbf{W},\mathbf{x}_i))^2 + \lambda \| PF(\mathbf{W},\mathbf{x}) \|^2, \quad (2)$$

where P is a differential operator, $\mathbf{x}_i$ are the N discrete points for which $f(\mathbf{x})$ is known and λ is the regularization parameter that represents the tradeoff between enforcing the smoothness constraint and fitting the known data. Minimizing this functional using variational calculus leads to the Euler–Lagrange equations:

$$P^*PF(\mathbf{W},\mathbf{x}) = \frac{1}{\lambda} \sum_{i=1}^{N} (f(\mathbf{x}_i) - F(\mathbf{W},\mathbf{x}_i))\delta(\mathbf{x} - \mathbf{x}_i), \quad (3)$$

where P^* is the adjoint of the operator P.

The above is a partial differential equation whose solution can be written as the integral transform of the right side with a kernel given by the Green's function of P^*P:

$$F(\mathbf{W},\mathbf{x}) = \frac{1}{\lambda} \sum_{i=1}^{N} (f(\mathbf{x}_i) - F(\mathbf{W},\mathbf{x}_i))G(\mathbf{x};\mathbf{x}_i), \quad (4)$$

where

$$P^*PG(x;y) = \delta(x-y).$$

The Green's functions, $G(\mathbf{x};\mathbf{x}_i)$, serve as the bases for the approximation scheme. When:

$$c_i = (f(\mathbf{x}_i) - F(\mathbf{W},\mathbf{x}_i))/\lambda,$$

the approximating function is:

$$F(\mathbf{W},\mathbf{x}) = \sum_{i=1}^{N} c_i G(\mathbf{x};\mathbf{x}_i). \quad (5)$$

The parameters, $\mathbf{W}$, include the coefficients, c_i, and the centers, $\mathbf{x}_i$, of the Green's functions.

It is apparent from equation (3) the basis functions depend on the operator P^*P. This operator is typically chosen to be both translationally and rotationally invariant. For this reason, the basis functions are rotationally and translationally invariant:

$$G(\mathbf{x};\mathbf{x}_i) = G(\|\mathbf{x} - \mathbf{x}_i\|).$$

These functions are radial basis functions. Equation (5) can be rewritten as:

$$F(\mathbf{W},\mathbf{x}) = \sum_{i=1}^{N} c_i G(\|\mathbf{x} - \mathbf{x}_i\|). \quad (6)$$

The coefficients, c_i, can be calculated by solving the following linear system:

$$\begin{pmatrix} f(\mathbf{x}_i) \\ \vdots \\ f(\mathbf{x}_N) \end{pmatrix} = \begin{pmatrix} G(\mathbf{x}_1;\mathbf{x}_1) & \cdots & G(\mathbf{x}_1;\mathbf{x}_N) \\ \vdots & \cdots & \vdots \\ G(\mathbf{x}_N;\mathbf{x}_1) & \cdots & G(\mathbf{x}_N;\mathbf{x}_N) \end{pmatrix} \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix}.$$

Let:

$$\mathbf{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix}, \mathbf{y} = \begin{pmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_N) \end{pmatrix}, \mathbf{G} = \begin{pmatrix} G(\mathbf{x}_1;\mathbf{x}_1) & \cdots & G(\mathbf{x}_1;\mathbf{x}_N) \\ \vdots & \cdots & \vdots \\ G(\mathbf{x}_N;\mathbf{x}_1) & \cdots & G(\mathbf{x}_N;\mathbf{x}_N) \end{pmatrix}.$$

There exists a solution for $\mathbf{c}$ as long as $\mathbf{G}$ is invertible:

$$\mathbf{c} = \mathbf{G}^{-1}\mathbf{y}. \quad (7)$$

This invertibility of $\mathbf{G}$ restricts the type of basis functions that can be used. If $\mathbf{G}$ is a positive definite matrix then it is invertible. Two theorems by Micchelli[8] exploit this property to impose sufficient conditions for the basis functions. The following are a few basis

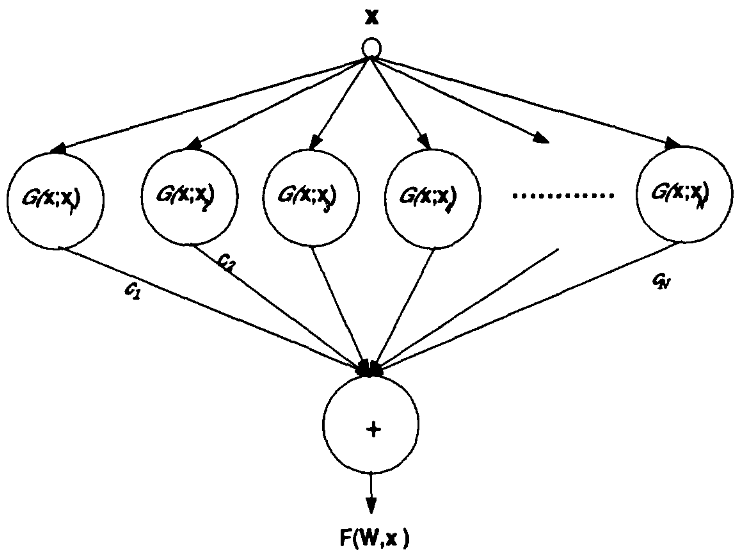

Fig. 1. The RBF network has three layers. The first is the input vector, x. The second layer consists of the radial basis functions $G(\mathbf{x}; \mathbf{x}_l)$ to $G(\mathbf{x}; \mathbf{x}_N)$. The c_i values are the weights between the ith basis function and the output, $F(\mathbf{W}, \mathbf{x})$, which is the third layer.

functions that satisfy Micchelli's condition:[1]

$$G(r) = e^{-r^2/\sigma^2}$$
$$G(r) = \frac{1}{(\sigma^2 + r^2)^\alpha} \quad \alpha > 0$$
$$G(r) = (c^2 + r^2)^\beta \quad 0 < \beta < 1$$
$$G(r) = r \tag{8}$$

where $r = \|\mathbf{x} - \mathbf{x}_i\|$.

Once the type of basis function has been selected, casting the approximating function, equation (6), as a network is straightforward. The RBF network has three layers, each fully connected to the next layer (see Fig. 1). The first layer consists of a single input unit, the vector $\mathbf{x}$. The second layer is composed of the series of multi-dimensional radial basis functions $G(\mathbf{x}; \mathbf{x}_i)$. There exists one basis function for each data point $\mathbf{x}_i$. The third layer is the output, which is a weighted sum of the basis functions.

The above RBF network has as many basis functions as examples. This becomes a problem in most applications, since typically a large number of examples are given. For this reason, RBF networks are generally implemented with fewer basis functions than examples. RBF networks with fewer basis functions than examples are termed generalized radial basis function (GRBF) networks. The approximating function, however, is no longer an exact representation of $f(\mathbf{x})$ and the approximation becomes worse as the number of basis functions is reduced. The approximating function for the GRBF network is:

$$F(\mathbf{W}, \mathbf{x}) = \sum_{j=1}^{n} c_j G(\mathbf{x}; \mathbf{z}_j), \tag{9}$$

where $n < N$ (n is the number of basis functions and N is the number of data points) and $\mathbf{z}_j$ are the centers of the new basis functions (the centers of the basis functions no longer have to be at data points). The center positions and coefficients is equation (9) are computed by minimizing the cost functional:

$$H[F(\mathbf{W}, \mathbf{x})] = \sum_{i=1}^{N} (F(\mathbf{W}, \mathbf{x}_i) - f(\mathbf{x}_i))^2. \tag{10}$$

This cost functional is traditionally minimized in the following *ad hoc* manner. A user provides initial values for the number n of basis functions, their positions $\mathbf{z}_j$ and their spans. Optimization techniques are to used to adjust the center values and the spans. The pseudoinverse is then used to compute the coefficients c_j. If the approximation is not within the desired bound, the user increases the number of basis functions and repeats the procedure.

This process is cumbersome. It also sacrifices the theoretical structure that makes RBF networks appealing. The disadvantage here is that given a data set and a specified error bound, the number of basis functions, as well as their parameters, required to perform the mapping cannot be determined analytically.

3. INTEGRAL WAVELET TRANSFORMS AND RBF NETWORKS

In this section the use of an integral wavelet transform to set the parameters of a GRBF network is described. It shall be shown that, by using the transform, the number of basis functions required for a given mapping and error bound can be determined analytically. In addition, the parameters of the basis functions are directly computed as a result of the transform.

3.1. *The wavelet transform*

The integral wavelet transform (IWT) can be viewed as a generalization of the principle underlying the Fourier transform. The IWT allows us to construct orthonormal or biorthonormal basis functions that are localized in space and decompose a function in terms of these bases. For the case of a 1D function, $f(x)$ ($\mathcal{R}^1 \mapsto \mathcal{R}^1$), the IWT has the basic form:[5]

$$(T_\psi f)(b,a) = \int_{-\infty}^{+\infty} \psi_{b,a}(x) f(x)\,dx,$$

where

$$\psi_{b,a}(x) = |a|^{-1/2} \psi\left(\frac{x-b}{a}\right)$$

are the wavelet bases and $(T_\psi f)(b,a)$ are the transform coefficients. The function $f(x)$ is characterized by the following: (a) The location of a change in $f(x)$ in terms of a position parameter b; (b) the rate of change in $f(x)$ in terms of a span parameter a; (c) the amount of this change in terms of $(T_\psi f)(b,a)$. If the wavelet bases are orthonormal, the function $f(x)$ can be reconstructed from the basis functions using:

$$f(x) = \sum_a \sum_b (T_\psi f)(b,a) |a|^{-1/2} \psi\left(\frac{x-b}{a}\right).$$

The IWT allows us to decompose functions at different resolution levels, from fine to coarse. The accuracy in reconstructing the original function decreases on going to coarser resolution levels. The basis functions at all resolution levels are either orthonormal or biorthonormal to each other. Two functions are involved in a wavelet transform, a scaling function and a wavelet. The a and b parameters of the wavelet are discretized: $a = 1/2^j$ and $b = k/2^j i, j \in Z$, where j is a scale parameter and k is a position parameter. The scaling function is written as:

$$\phi_{j,k}(x) = 2^{j/2} \phi(2^{-j}x - k).$$

These scaling functions need not be orthogonal (in most cases the scaling functions are not orthogonal) and are used to construct the wavelet bases and scaling functions at a coarser resolution level:

$$\psi_{j-1,k} = \sum_k q(k) \phi_{j,k} \qquad (11)$$

$$\phi_{j-1,k} = \sum_k p(k) \phi_{j,k}. \qquad (12)$$

The $q(k)$ values are chosen such that:

$$\langle \psi_{j,k}(x), \phi_{j,k}(x) \rangle = 0.$$

This forces the wavelet bases to be orthonormal across scale. If the $q(k)$ values are selected carefully, the wavelet bases can be made orthonormal across position as well. Alternatively, the bases can be made biorthonormal. The result is a set of orthonormal bases:

$$\langle \psi_{j,k}, \psi_{l,m} \rangle = \delta_{j,l} \delta_{k,m}.$$

An approximation to a function $f(x)$ exists at different resolution levels. At the resolution level j, the approximating function has the form:[5]

$$f_j(x) = \sum_k c_{j+1}(k) \phi(2^{-j}x - k) + \sum_k d_{j+1}(k) \psi(2^{-j}x - k). \qquad (13)$$

From the above, it can be shown that the exact representation of the function $f(x)$ has the form:[5]

$$f(x) = \sum_j \sum_k d_j(k) \psi(2^{-j}x - k), \qquad (14)$$

where

$$d_j(k) = \langle f(k), \psi(2^{-j}x - k) \rangle.$$

The decomposition in equation (14) is equivalent to a weighted sum of scaling functions at the finest resolution level:

$$f_0(x) = \sum_{k=1}^{N} c(k) \phi_0(x - k). \qquad (15)$$

If we use a scaling function that approximates a radial basis function, then equation (15) is identical to the approximating function of equation (5) in Section 2. One group of scaling functions that approximate radial basis functions are B-splines of order greater than one (see Appendix A for details). Our approach is to apply the IWT to the training data of the input–output mapping problem and to use the transform ciefficients and the wavelet bases to construct a GRBF network.

3.2. *Calculating the wavelet coefficients*

The training data given, $x_i \mapsto f(x_i)$, can be considered a discrete signal. In this case, a discrete integral wavelet transform is implemented to calculate the wavelet coefficients. The input space is discretized into 2^J bins, where J typically ranges from 9 to 11. The $f(x_i)$ values are then placed into the appropriate bin by rounding off x_i. The result is a signal of the form $f(k)$, where $k \in Z$. Since the bins are small, we assume that any error introduced from the rounding off is negligible.

The next step involves calculating the transform coefficients, $d_{j,k}$, for the wavelet bases, $\psi_{j,k}$. The fast wavelet algorithm[6] is extended to unevenly sampled data to calculate these coefficients. This algorithm allows one to compute the coefficients recursively from finer levels to coarser levels by convolving the signal at a finer level with two filters. The specific algorithm is as follows:

$$c_{j-1}(k) = [\hat{v} * c_j]_{\downarrow 2}(k) \qquad (16)$$

$$d_{j-1}(k) = [\hat{w} * c_j]_{\downarrow 2}(k), \qquad (17)$$

where $\downarrow 2$ denotes downsampling by two, keeping every other term. The c_j values correspond to an approximation of the signal at the resolution level j. At the finest resolution level c_0 is the signal $f(k)$. The formulae for $\mathring{v}$ and $\mathring{w}$ for the Battle–Lamarié basis constructed from cubic B-splines are given in Appendix B. This algorithm is iterated for $j=0$ to $1-J$, where $j=0$ is the finest resolution level and $j=1-J$ is the coarsest level.

Since little is known about the training data, we assume that $f(k)$ need not be evenly sampled. This brings up the question of how to implement:

$$a(x)=f(x)*g(x), \tag{18}$$

when $f(x)$ is an unevenly sampled discrete signal and $g(x)$ is a continuous function.

Spectral analysis of unevenly sampled data has been explored by astronomers. A popular approach is the Lomb periodogram.[9] The periodogram performs a least-mean-square fit of the data to sines and cosines under the assumption that the error in the fit decreases with the addition of frequency terms. We use it to help define the operation in equation (18). In our implementation, the data was sampled at N_p frequencies, where $N_p=(f_h/2)N, f_h=f_{hi}f_c, f_c=1/\ 2\Delta x_{avg}, f_{hi}=1/2\Delta x_{min}$, Δx_{min} is the smallest distance between two data points and Δx_{avg} is the average distance between data points. The decomposition of $f(x)$ by the periodogram has the form:

$$f(x)=\sum_{m=0}^{N_p} q_m e^{i\omega_0 mx}, \tag{19}$$

where, $q_m=\langle f(x), e^{-i\omega_0 mx}\rangle$, $\omega_0=\pi/T$ and T is the total length of the discrete data. The Lomb periodogram reduces to the discrete Fourier transform when $f(x)$ is evenly sampled.

Using the periodogram we can map $f(x)\mapsto F(\omega)$, where $F(\omega)$ are the values q_m determined by the periodogram. An assumption is then made that there exists a unique continuous function $h(x)$ which has the same spectral properties as $f(x)$, i.e. $H(\omega)=F(\omega)$. The two functions $f(x)$ and $h(x)$ have identical values at all x_i for which $f(x_i)$ is given. This equivalence of the functions in both spatial and frequency domains is a result of the oversampling of $f(x)$ in the periodogram. The continuous function $h(x)$ can be derived:

$$\begin{aligned} h(x) &= \frac{1}{2\pi}\int_{-\infty}^{+\infty} H(\omega)e^{i\omega x}\mathrm{d}\omega \\ &= \frac{1}{2\pi}\int_{-\infty}^{+\infty} F(\omega)e^{i\omega x}\mathrm{d}\omega \\ &= \frac{1}{2\pi}\sum_{m=0}^{N_p} q_m e^{i\omega_0 mx}. \end{aligned} \tag{20}$$

The continuous function $h(x)$ replaces $f(x)$ in the convolution in equation (18) so that the operation $a(x)=h(x)*g(x)$ can be performed.

In the fast wavelet algorithm, the above technique can be used to perform the convolutions in equations (16) and (17). The unevenly sampled $c_j(k)$ is replaced by the continuous $c'_j(x)$ and $C_j(\omega)=C''_j(\omega)$. Since:

$$\mathring{v}(k)*c'_j(k)\leftrightarrow \mathring{V}(\omega)C'_j(\omega),$$

we can calculate a continuous function $c'_{j-1}(x)$:

$$C'_{j-1}(\omega)=\sum_{m=0}^{N_p}\mathring{V}(\omega_m)C'_j(\omega_m)$$

$$c'_{j-1}(x)=\frac{1}{2\pi}\sum_{m=0}^{N_p}\mathring{V}(\omega_m)C'_j(\omega_m)e^{i\omega_0 mx}. \tag{21}$$

This continuous function $c'_{j-1}(x)$ is sampled at the resolution level 2^j and then downsampled by 2 to obtain $c_{j-1}(k)$:

$$c_{j-1}(k)=[c'_{j-1}]_{\downarrow 2}=c'_{j-1}(k/2), \tag{22}$$

where $k=1\text{–}m$ (where m is the maximum index minus the minimum index at which there exits a known signal value). The same procedure is used to calculate $d_{j-1}(k)$.

This variation of the fast wavelet algorithm allows the decomposition of unevenly sampled data. The weakness in this approach is the assumption inherent in the periodogram that the sampling rate exceeds what would be the Nyquist frequency if the data were evenly sampled. A discussion of what approximations are introduced due to this oversampling assumption is beyond the scope of this paper. Alternative approaches to performing wavelet transforms on non-uniform data have been developed by Buhmann and Micchelli[10,11] and Sweldens.[12] We did not use the approach of Buhmann and Micchelli for two reasons. First, extending their approach to multi-dimensional problems results in an enormous number of computations. Secondly, they introduce an extra spline space of radial basis functions called prewavelets to perform the transform. This extra space makes it difficult to construct a network once the parameters are computed. Sweldens construction of second generation wavelets would allow us to perform the wavelet transform on multi-dimensional unevenly sampled data very efficiently. In future research this approach needs to be investigated. At this point, it is not clear if this approach would preserve the smoothness properties of GRBF networks.

The mappings considered so far in this section have been $\mathscr{R}^1\mapsto\mathscr{R}^1$. To extend the transform to multi-dimensional functions, $\mathscr{R}^n\mapsto\mathscr{R}^m$, $\Phi(\mathbf{x})$ and $\Psi(\mathbf{x})$, are introduced for both the scaling and wavelet functions. Both $\Psi(\mathbf{x})$ and $\Phi(\mathbf{x})$ are tensor product splines separable with respect to each dimension x_d:

$$\Phi(\mathbf{x})=\phi(x_1)\phi(x_2)\ldots.\phi(x_n)$$

$$\Psi(\mathbf{x})=\psi(x_1)\psi(x_2)\ldots.\psi(x_n).$$

The decomposition of $f(x)$ becomes:

$$f(\mathbf{x})=\sum_j\sum_{k_1\in K_1(j)}\cdots\sum_{k_n\in K_n(j)} d_{j,\mathbf{k}}(\mathbf{x})\psi(2^{-j}x_1-k_1)\ldots. \psi(2^{-j}x_n-k_n), \tag{23}$$

where $\mathbf{k}$ corresponds to the vector $[k_1k_2\ldots k_n]$ and

$K_d(j)$ is the set of all position parameters at scale j and dimension d. Since $\Psi(\mathbf{x})$ and $\Phi(\mathbf{x})$ are separable, the coefficients $d_{j,\mathbf{k}}$ can be calculated by applying the fast wavelet algorithm across each dimension. The function $\mathbf{f}(\mathbf{x})$ is equivalent to the vector $[f_1(\mathbf{x}),\ldots,f_l(\mathbf{x}),\ldots,f_m(\mathbf{x})]$, where each $f_i(\mathbf{x})$ can be considered a separate signal. The transform is applied to each signal. The decomposition of a function $\mathbf{f}(\mathbf{x})$ has the form:

$$\begin{aligned}
f_1(\mathbf{x}) &= \sum_j \sum_{k_1\in K_1(j)} \cdots \sum_{k_n\in K_n(j)} d_{j,\mathbf{k},1}(\mathbf{x})\psi(2^{-j}x_1-k_1)\ldots\psi(2^{-j}x_n-k_n)\\
f_2(\mathbf{x}) &= \sum_j \sum_{k_1\in K_1(j)} \cdots \sum_{k_n\in K_n(j)} d_{j,\mathbf{k},2}(\mathbf{x})\psi(2^{-j}x_1-k_1)\ldots\psi(2^{-j}x_n-k_n)\\
&\vdots\\
f_m(\mathbf{x}) &= \sum_j \sum_{k_1\in K_1(j)} \cdots \sum_{k_n\in K_n(j)} d_{j,\mathbf{k},m}(\mathbf{x})\psi(2^{-j}x_1-k_1)\ldots\psi(2^{-j}x_n-k_n),
\end{aligned} \tag{24}$$

where $d_{j,\mathbf{k},l}$ is the transform coefficient for the lth output at the $2j$th resolution level for the $\mathbf{k}$th basis. The result of the transform applied to a mapping from $\mathcal{R}^n \mapsto \mathcal{R}^m$ are the coefficients $d_{j,\mathbf{k},l}$. The magnitudes of these coefficients tells us how important the wavelet corresponding to the coefficient is in approximating the function.

3.3 *Using the transform to construct a GRBF network*

The result of applying the transform to a mapping from $\mathcal{R}^n \mapsto \mathcal{R}^m$ is the coefficients $d_{j,\mathbf{k},l}$. Using these coefficients and the error bound specified, a GRBF network is constructed. If the L^2 norm between a function $\mathbf{f}(\mathbf{x})$ and its approximating function $\mathbf{F}(\mathbf{W},\mathbf{x})$ must be less than the error bound, ε, then:

$$\varepsilon > \sum_{i=1}^{N} (\mathbf{f}(\mathbf{x}_i) - \mathbf{F}(\mathbf{W},\mathbf{x}_i))^2, \tag{25}$$

where N is the number of known values of $\mathbf{f}(\mathbf{x})$. Using Parseval's identity:

$$P_t = \sum_{i=0}^{N} \|\mathbf{f}(\mathbf{x}_i)\|^2 = \sum_{l=1}^{m} \sum_{j=0}^{J-1} \sum_{k_1\in K_1(j)} \cdots \sum_{k_n\in K_n(j)} d^2_{j,\mathbf{k},l},$$

where m is the dimensionality of the output, $J-1$ is the coarsest resolution level and P_t is the total energy in the function $\mathbf{f}(\mathbf{x})$. The energy in the approximating function $\mathbf{F}(\mathbf{W},\mathbf{x})$ is:

$$\|\mathbf{F}(\mathbf{W},\mathbf{x})\|^2 = \sum_{l=1}^{m} \sum_{j=0}^{J-1} \sum_{k_1\in K'_1(j)} \cdots \sum_{k_n\in K'_n(j)} d^2_{j,\mathbf{k},l}, \tag{26}$$

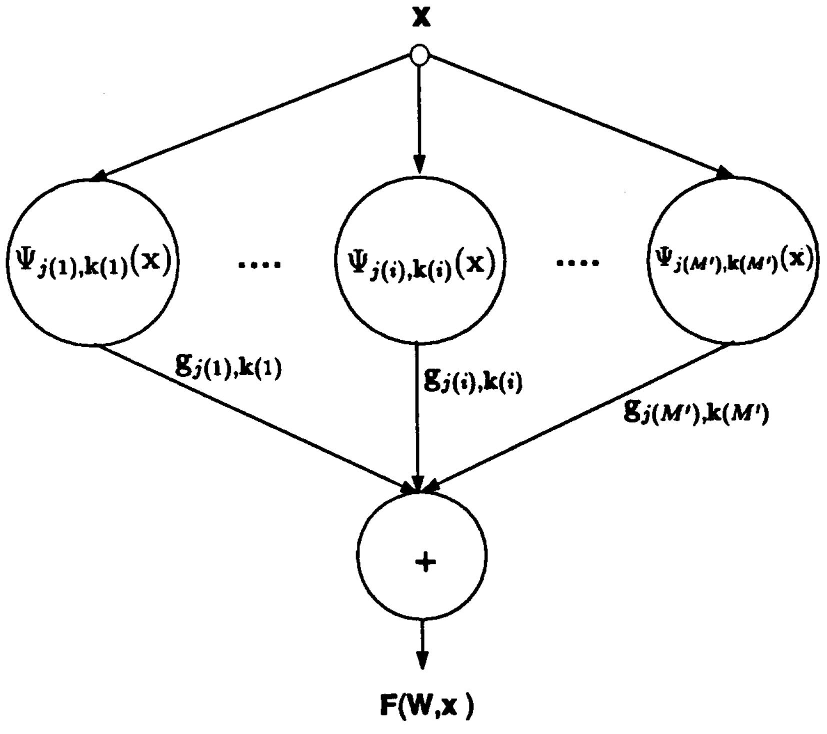

Fig. 2. The GRBF network derived using the wavelet transform. The first layer is the input vector, $\mathbf{x}$. The second layer consists of the wavelet bases $\Psi_{j(l),\mathbf{k}(l)}$ to $\Psi_{j(M'),\mathbf{k}(M')}$. The $\mathbf{g}_{j(i),\mathbf{k}(i)}$ are the weights between the ith wavelet basis and the output, $\mathbf{F}(\mathbf{W},\mathbf{x})$.

where $K'_d(j) \subset K_d(j)$. This is equivalent to setting the coefficients for some of the wavelet bases to 0. Using equation (26), equation (25) can be rewritten as:

$$\varepsilon^2 > 1 - \frac{1}{P_t} \sum_{l=1}^{m} \sum_{j=0}^{J-1} \sum_{k_1 \in K'_1(j)} \cdots \sum_{k_n \in K'(j)} d^2_{j,\mathbf{k},l}. \quad (27)$$

The number of wavelet bases used in the approximation for a given ε is determined by finding $K'_d(j)$ at each resolution level j and each dimension x_d until the above condition is met. One way of doing this is by calculating sums of the coefficients over the output dimensions:

$$s_{j,\mathbf{k}} = \sum_{l=1}^{m} d^2_{j,\mathbf{k},l}.$$

The sums $s_{j,\mathbf{k}}$ are then ordered from 1 to M so that $s_{j(1),\mathbf{k}(1)} \geqslant s_{j(2),\mathbf{k}(2)} \geqslant \cdots s_{j(M),\mathbf{k}(M)}$, where M is the number of wavelet bases in the decomposition of $\mathbf{f}(\mathbf{x})$. Equation (27) can now be expressed as:

$$\varepsilon^2 > 1 - \frac{1}{P_t} \sum_{i=1}^{M'} s_{j(i),\mathbf{k}(i)}, \quad (28)$$

where M' is the smallest integer that satisfies the above condition. It is calculated by adding the $s_{j(i),\mathbf{k}(i)}$ terms until the inequality in equation (28) is satisfied. The resulting approximating function is:

$$\mathbf{F}(\mathbf{W},\mathbf{x}) = \sum_{i=1}^{M'} g_{j(i),\mathbf{k}(i)} \Psi_{j(i),\mathbf{k}(i)}(\mathbf{x}), \quad (29)$$

where

$$g_{j(i),\mathbf{k}(i)} = \begin{pmatrix} d_{j(i),\mathbf{k}(i),l} \\ \vdots \\ d_{j(i),\mathbf{k}(i),m} \end{pmatrix}.$$

The expression in equation (29) can be easily be mapped to a network as shown in Fig. 2. The wavelet bases, $\Psi_{j(i),\mathbf{k}(i)}$, are not radial basis functions. However, each wavelet basis is a weighted sum of scaling functions at a finer level. For this reason, we can consider this network to be a GRBF network.

In this section we demonstrate how the integral wavelet transform can be used to generate a GRBF network. This method replaces the *ad hoc* and cumbersome optimization techniques to set GRBF network parameters in standard approaches. With this approach, the number of basis functions required to satisfy a given error bound is determined analytically. In addition, the parameters of the network are also obtained directly from the transform. The result is a more efficient network with respect to speed and storage memory.

4. COMPUTATIONAL COMPLEXITY AND MEMORY REQUIREMENTS

The two main factors that determine the usefulness of a GRBF network is the time required to perform the mapping and the memory required to store the network. The speed of the mapping depends on the number of computations involved which, in turn, is determined by the dimensionality of the input and output spaces and the number of basis functions in the approximating function. The number of computations required to evaluate the output of the wavelet basis also effects the speed of the mapping. For this reason, the wavelet bases are stored as a look-up table, avoiding the need for numerous calculations. The number of computations for a GRBF network can be expressed as follows:

$$C_{net} = 6n_o n_i N, \quad (30)$$

where C_{net} is the number of additions and multiplications performed by the network, n_i is the dimensionality of the input space, n_o is the dimensionality of the output space and N is the number of basis functions in the network. The complexity is $O(mN)$, where m is the dimensionality of the mapping, $m = n_i \times n_o$.

The second issue is the amount of memory required to store the mapping. This depends again on the dimensionality of the input and output spaces, the number of basis functions and the resolution of the look-up table used for the basis functions. All the basis functions in the network are translations and dilations of an orthonormal or biorthonormal wavelet. This allows us to use a single look-up table and two sets of normalization factors (one set for translation and one set for scaling) for each basis. For each basis function, $n_i + 1$ integers (n_i integers for the position of the basis in each input dimension and an extra integer for the scaling factor) and n_o real numbers (the weights for the basis function) are stored. In addition, the look-up table needs to be stored. The memory required in storing the look-up table is determined by the resolution of look-up table, n_{res}. For a Sparc IPX (where a double is 8 bytes and an integer is 4 bytes) the amount of memory, M_{net}, required to store network parameters is:

$$M_{net} = 4 \times N \times (n_i + 1) + 8 \times N \times n_o + 8 \times n_{res} \text{bytes}. \quad (31)$$

5. A SIMPLE EXAMPLE

In this section, a simple 1D example is used to demonstrate the advantage of using the transform technique over traditional optimization methods. For this demonstration we use the following two functions: $f_1(n) = \sin(2\pi n/64)$ and $f_2(n+1) = 3f_2(n)(1 - f_2(n))$, $f_2(0) = 0.4467$, where $0 \leqslant n \leqslant 63$ and $n \in Z$ [see Figs 3(a) and (b)]. The GRBF networks constructed for these two functions using the transform approach will be referred to as *wavelet-based networks*. For comparison, GRBF networks are also constructed using a traditional algorithm, these are referred to as *conventional networks*. The specific algorithm used to construct the conventional networks is outlined below:

(1) Initialize the number of basis functions, n.

(2) Set the values for the centers of the n basis functions to an arbitrary subset of the N data points.

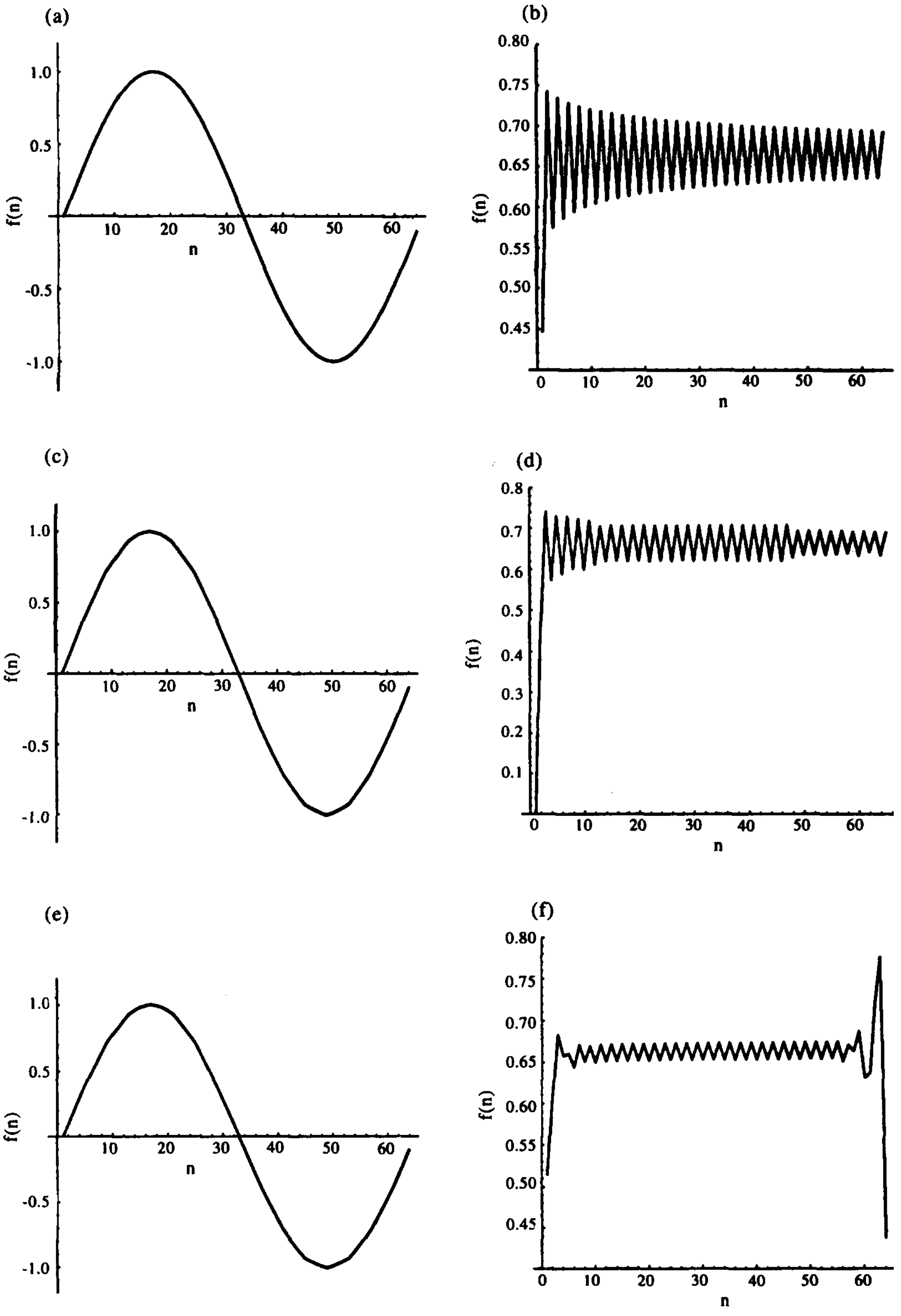

Fig. 3. (a) $f_1(n) = \sin(2\pi n/64)$); (b) $f_2(n+1) = 3f_2(n)(1 - f_2(n))$, $f_2(0) = 0.4467$; (c) reconstruction of f_1 by a wavelet-based network with eight bases; (d) reconstruction of f_2 by a wavelet-based network with 32 bases; (e) reconstruction of f_1 by a conventional network with eight bases; (f) reconstruction of f_2 by a conventional network with 32 bases.

(3) Set the spans of the basis functions to $\Delta x/n$, where Δx is the span of the input space.

(4) Calculate the coefficients, c_i, using the pseudo inverse.

(5) Use conjugate gradient decent to adjust the center values.

(6) Return to (4) until the change in the error functional between two iterations is negligible.

(7) If the error functional is greater than the specified error bound, ε, increment n and return to step (2).

In the above algorithm, the spans of the basis functions are equal and not allowed to vary due to the amount of time required to search both the position and span parameter spaces for minima.

The accuracy of the wavelet-based and conventional networks as a function of the number of bases, for both functions, is shown in Fig. 4. In both cases, the error functional decays faster and monotonically for the wavelet-based networks. In the case of the first function f_1 the improvement is negligible. The performance of the wavelet-based network is dramatically superior for the second function, f_2. The error functional for the conventional network does not even decrease monotonically for this case. The improvement seen in the performance of the wavelet-based network is due to the fact that the problem of local minima in the parameter space is avoided with the use of the transform approach. Figure 4 shows the reconstruction of the two example functions using the two types of networks. Again, the performance of the wavelet-based network is superior to that of the conventional network for the second function. For the first function the results are almost identical.

The parameters of the wavelet-based network are calculated much quicker than those of the conventional network. For the example functions above, the parameters for the wavelet-based network are determined within 9.06 s, independent of the number of basis functions required. In contrast, it can take between 10.04 s and 20 min to set the parameters of the conventional network, depending on the number of

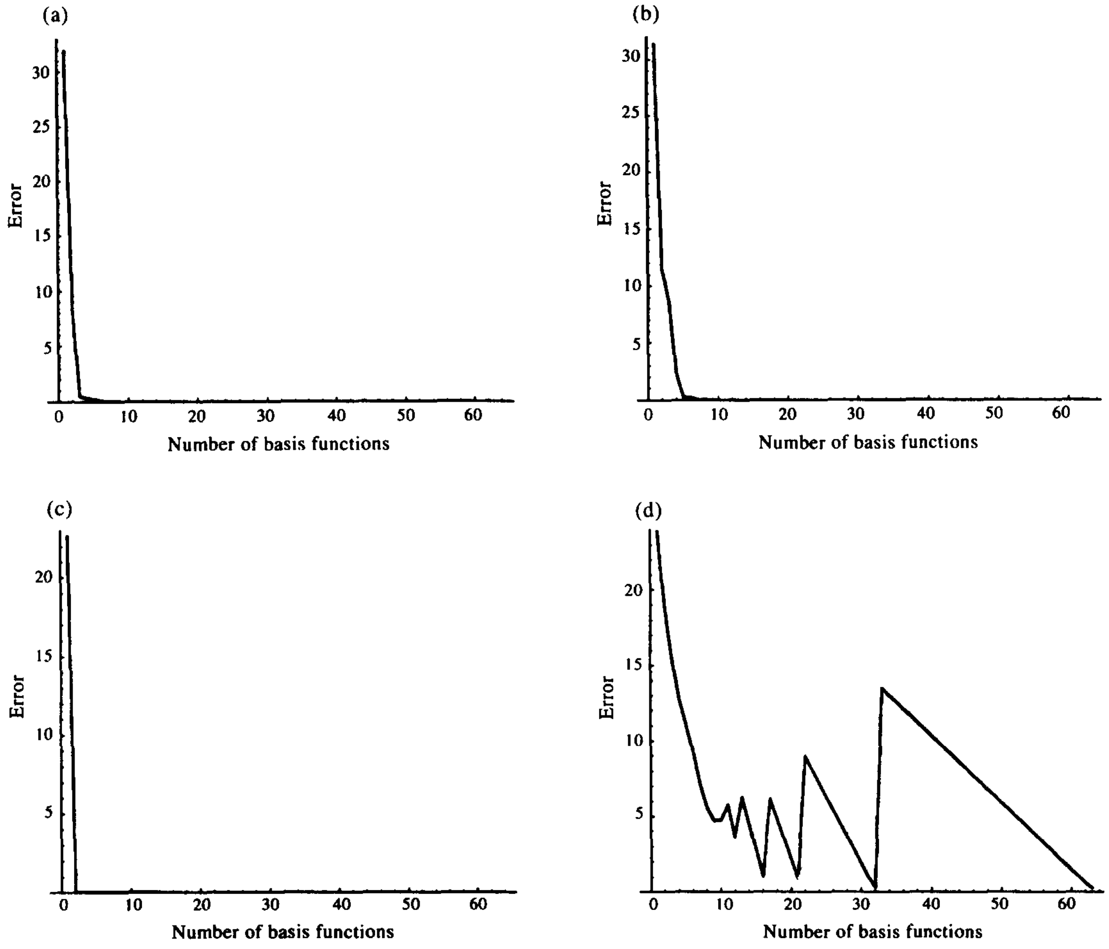

Fig. 4. The decay in the L^2 error in approximation for function $f_1(n) = \sin(2\pi n/64)$ as the number of basis functions increases for (a) the wavelet-based network and (b) the conventional network. The decay in the approximation error for $f_2(n+1) = 3f_2(n)(1 - f_2(n)), f_2(0) = 0.4467$ for (c) the wavelet-based network and (d) the conventional network.

basis functions used. The conjugate gradient decent method has a computational complexity of the order $O(n^2)$, where n is the number of basis functions. The wavelet transform is of the order $O(d \log_2 d)$, where d is the number of known samples of the function. Both networks have the same computational complexity for recognition. However, the wavelet-based network takes less memory and is faster than the conventional network since it generally requires fewer basis functions for any given accuracy. In summary, the wavelet-based network is constructed more accurately and performs faster than the conventional network.

6. A MULTI-DIMENSIONAL PROBLEM

3D object recognition was chosen as a high- dimensional application of GRBF networks. This application involves a network that recognizes an object and estimates its pose in a scene. These networks take as input a compact representation that uses principal component analysis to parameterize object appearance by pose, introduced by Murase and Nayar.(13) A brief overview of this representation is in order.

For each object, a large image set is acquired by varying pose. The eigenvectors of the correlation matrix of this image set, corresponding to the largest eigenvalues, make up the dimensions of a subspace (typically 10–20 dimensions) termed the eigenspace. When such a subspace is computed using image sets of all objects, it is referred to as the universal eigenspace. Each image of an object is projected to the universal eigenspace, by taking the dot product of the image with the eigenvectors. This results in a single point in eigenspace. The projections of all images of an object results in a set of points (corresponding to the different discrete poses), that is referred to as a *discrete manifold*. In reference (13) the discrete manifold is interpolated using biquadratic splines to obtain a continuous manifold that is parameterized by object pose. The manifold is then densely resampled to obtain a large number of manifold points. This large point set represents that object's appearance model. The above process is repeated for all objects of interest to the recognition system.

Given a novel object image, the object region is segmented, normalized in scale and projected to universal eigenspace. The closest manifold determines the identity of the object in the image and the exact position of the new projection on the manifold yields the pose of the object. In reference (13) the closest manifold point is determined either by exhaustive search (which is inefficient in time and memory) or by binary search (which is inefficient in memory).

Our objective is not to interpolate and resample discrete manifolds of the objects, but rather to generate a GRBF network (for each object as shown in Fig. 5) that performs a continuous mapping from a discrete manifold point to a confidence value C_p, which repre-

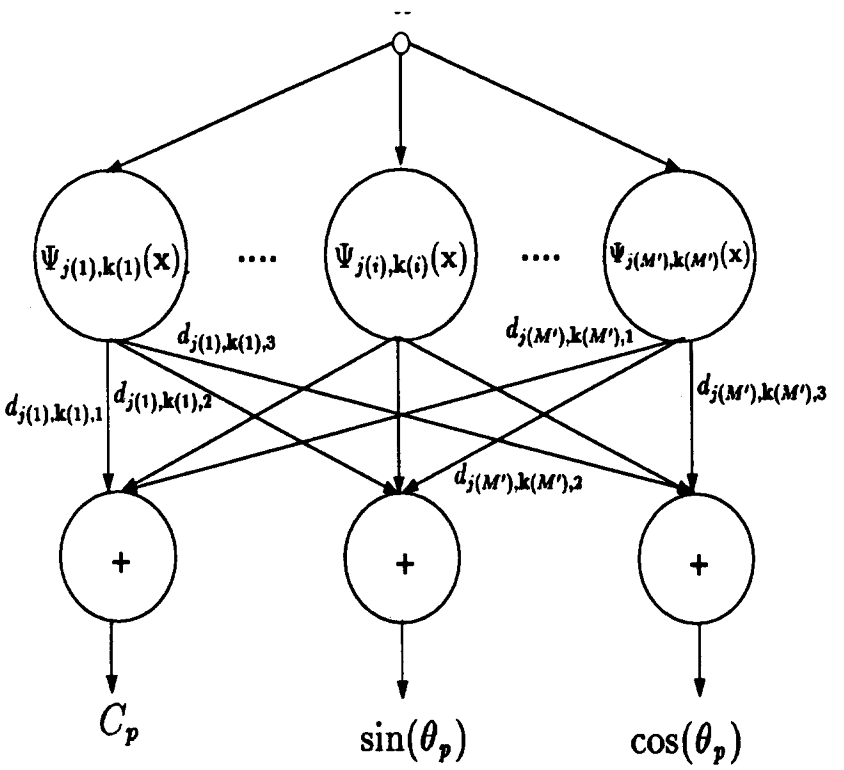

Fig. 5. A wavelet-based network for a particular object, p. The input is a point in eigenspace and the outputs are the object confidence, C_p, and the pose parameters $\sin(\theta_p)$ and $\cos(\theta_p)$.

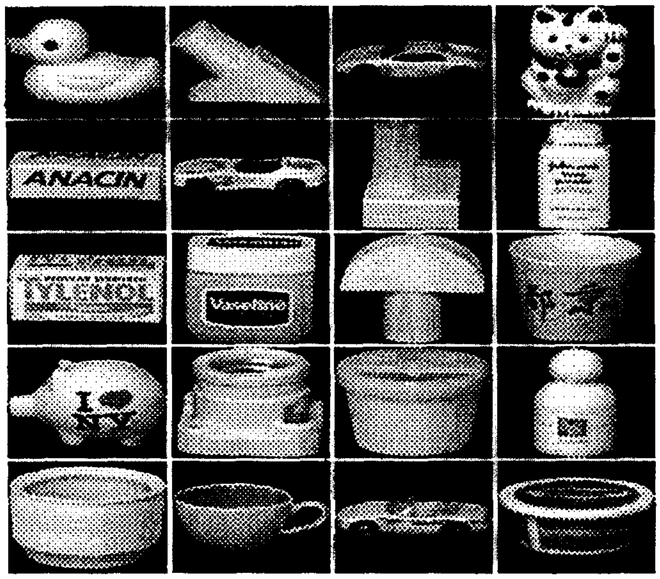

Fig. 6. The 20 objects in the object recognition and pose estimation system (from Murase and Nayar[13]). The image set for each object includes 72 images corresponding to uniformly sampled discrete poses between 0 and 360°. Half of the corresponding eigenspace projections were used as training data to generate GRBF networks and the other half for testing the networks.

sents the likelihood that the novel image is that of the object for which the network has been developed, and the pose θ_p of the object. This network is generated automatically from the discrete manifold of an object using the transform technique developed in this paper. The result is a set of P wavelet-based networks, where P is the total number of objects. During recognition, the input, which is an eigenspace projection, is mapped by each of the networks. The network that produces the highest confidence value reveals the identity of the object and its pose.

Note that the pose of each object has a discontinuity at $\theta_p = 360°$. The size of a GRBF network is clearly related to the continuity of the function it seeks to approximate (this topic will be revisited in Section 7). Therefore, judicious selection of data representation can dramatically reduce the size of the network. In the present application, we benefited by using $\sin(\theta_p)$ and $\cos(\theta_p)$ as outputs, instead of using θ_p. This eliminates the ill-conditioning that results from the discontinuity in θ_p at 360°. During recognition, the network with the highest confidence value C_p is first identified and if this value exceeds a threshold level (i.e. if the projection is close enough to the object's manifold), then pose is computed from $\sin(\theta_p)$ and $\cos(\theta_p)$ using arctan.

In our experiments, wavelet-based networks were constructed for each the 20 objects in the object database (see Fig. 6). The input space, a 15-dimensional eigenspace, was discretized into 1024 boxes in each of its dimensions. Clearly, it is impossible to store and process 1024^{15} entries. Instead, a sparse tensor [an extension of the concept of a sparse matrix[14]] was constructed with only the entries for which $f(\mathbf{x}_i)$ were known. The networks ability to learn and generalize examples presented to it was tested using two data sets. The *training set* includes 36 discrete manifold points (poses) for each object and was used to generate the networks. The *test data* includes a different set of 36 manifold points and was used to test the accuracy of the networks and their ability to generalize to data not seen before.

The most important task of the set of networks is to correctly recognize objects in the test set. In our experiments, every point in both the training and test sets was correctly recognized yielding a 100% recognition rate. We found that for any object v, $0.842 \leqslant C_p \leqslant 1.217$ when $v = p$ and $0.0 \leqslant C_p \leqslant 0.211$ when $v \neq p$, leading to robust object identification.

The networks' accuracy in pose estimation was also studied. For each object, two networks were constructed using the data in the training set. One using the wavelet-based approach the other using the conventional approach. The accuracy of pose estimation is defined as the absolute difference between the known pose and the pose estimated by the network. It is computed for both the test set and the training set and the results are summarized in Fig. 7. It is clear from these results that the wavelet-based network significantly outperforms the conventional network. A comparison of the performance of the two networks is summarized in Table 1.

In multi-dimensional learning problems, it is standard practice to examine the effect of the dimensionality of the input space on the accuracy of the mapping. Four wavelet-based networks corresponding to object number 10 (the jar of Vaseline) were constructed from

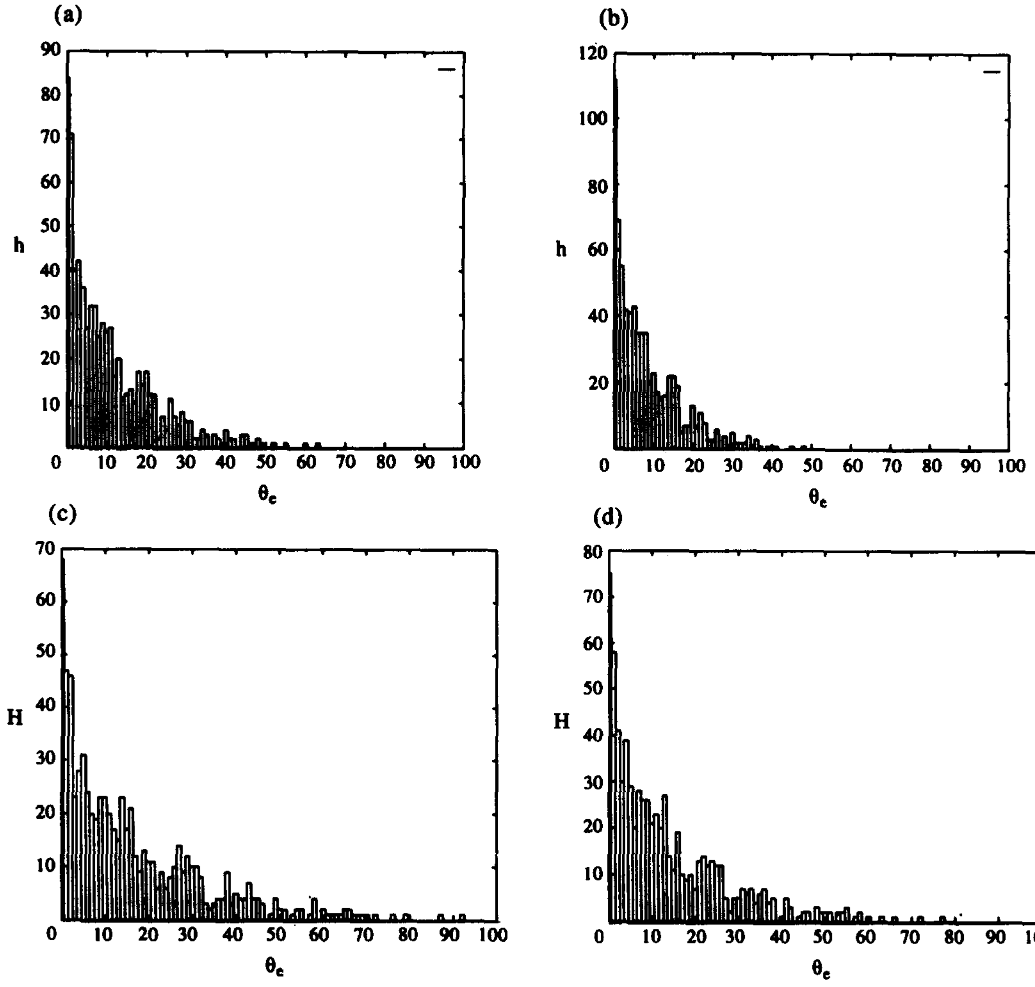

Fig. 7. Histograms of the absolute value of the error, θ_e, between poses estimated by a wavelet-based network and the known pose for (a) the test set and (b) the training set. Histograms for a conventional network for (c) the test set and (d) the training set. Each bar in the histogram corresponds to 0.1°.

Table 1. The performance of GRBF networks for all objects in terms of the time, accuracy and memory involved in recognition and pose estimation

	Time (ms)	Accuracy over test set(°)	Accuracy over training set (°)	Network storage (kbytes)	Number bases
$network_{conven}$	150	1.42°	1.23	54	18–36
$network_{wavelet}$	89	1.13°	0.85	23	11

Table 2. The effect of the dimensionality of the input space on the accuracy, recognition time and learning time for wavelet-based networks

Mapping	Learning time (S)	Recognition time (ms)	Average accuracy over test set (e)	Average accuracy over training set (e)
$\mathscr{R}^{5}\mapsto\mathscr{R}^{3}$	72	~1	5.37	4.21
$\mathscr{R}^{10}\mapsto\mathscr{R}^{3}$	145	~2	4.21	3.11
$\mathscr{R}^{15}\mapsto\mathscr{R}^{3}$	214	~3	1.92	1.66
$\mathscr{R}^{20}\mapsto\mathscr{R}^{3}$	289	~4	1.43	1.32

its training set. The input spaces for these networks were 5, 10, 15 and 20-dimensional, respectively. As before, the output of each network is 3D, including the parameters C_p, $\sin(\theta_p)$ and $\cos(\theta_p)$. For the error bounds specified, each network required 11 wavelet bases. The accuracy of all four networks in addition to time required for learning and recognition are summarized in Table 2. As expected, the average error in pose estimation increases as the dimensionality of the input space decreases. The recognition time is linear with respect to the dimensionality of the input space, as predicted by the complexity analysis in Section 4. The learning time for wavelet-based networks seems to increase as a polynomial function of the input dimensionality, however, further investigation is needed to state anything more definitive.

7. DISCUSSION

We have introduced a novel approach for setting GRBF network parameters using the integral wavelet transform. This approach allows the automatic generation of a GRBF network given an error bound. In addition, the network generated is the smallest network that performs the mapping for the error bound. The advantage of the transform approach over conventional optimization based techniques is demonstrated using two 1D functions as well as the multidimensional problem of real-time recognition and pose estimation of 3D objects. The networks generated by the transform approach outperformed networks generated by conventional approaches. The difference in performance was often significant.

The use of RBF networks is not limited to high-level vision processes such as recognition. These types of networks can be used to predict chaotic time series, recognize and synthesize speech signals and control robot manipulators. Any supervised learning problem that can be formulated from variational and regularization principles is well suited for the automatic network generation approach introduced in this paper. Therefore, the ideas presented here have far-reaching implications. Some of the issues that arose in developing this method that need further investigation are discussed below:

• Continuity conditions of the basis and the mapping: if the continuity conditions of a mapping are known, there should be a way to find the scaling function that gives the best approximation to the mapping. Continuity conditions are normally expressed in terms of the function spaces C^n, where n is the largest derivative that exists for the function. It seems intuitive that a function of the order C^n should be best approximated by scaling functions of the order C^n. For example, it would seem that a mapping that is once differentiable, C^1, would be best approximated using a first-order spline as the scaling function.

The scaling function used in the approximation can be related to the continuity conditions of the mapping from the regularization term in equation (3):

$$P^*PF(\mathbf{W},\mathbf{x}) = \frac{1}{\lambda}\sum_{i=1}^{N}(f(\mathbf{x}) - F(\mathbf{W},\mathbf{x}))\delta(\mathbf{x}-\mathbf{x}_i).$$

The differential operator P in the above equation can be set to the known continuity condition of the mapping. The Green's function solution to the differential equation could then calculated. The scaling function in the wavelet construction can then be chosen to closely approximate the Green's function.

• Representation: one of the key issues that this paper has not addressed is how data must be represented to achieve high network performance. A simple illustration of the importance of representation arose in pose estimation. For pose estimation the network was presented a mapping from a point in the input space (eigenspace) to $\sin(\theta)$ and $\cos(\theta)$ rather than θ. This greatly reduced the number of basis functions required to perform the mapping accurately. Since a function can be represented in a variety of ways, a methodology to evaluate representations is necessary. Using such a methodology, the appropriate representation for a function can be determined. This is helpful because the appropriate representation reduces the size of the network required to perform the mapping.

There exist many transforms, such as conformal mappings, transcendental mappings and other approaches, for representing a function. Suppose a function $f(x)$ can be represented in terms of any two of the above techniques, where $\tilde{f}(x)$ and $\bar{f}(x)$ are the two representations of $f(x)$. One needs a quantative method of judging which representation is better. Approximation spaces such as the Besov space offer possibilities[15] to quantitatively measure representa- tions. For the representation $\tilde{f}(x)$ an error function:

$$\tilde{a}_N(\tilde{f}) = \|\tilde{f}(x) - \tilde{f}'(x)\|^2$$

is defined where $\tilde{f}'(x)$ is a decomposition of $\tilde{f}(x)$ into N orthonormal basis. Given two representations with respective errors $\tilde{a}_N$ and $\bar{a}_N$, the class of functions $f(x)$ can be catalogued for which $\tilde{a}_N$ and $\bar{a}_N$ decays at a set rate as N gets large. A representation has a class α for the set of functions $f(x)$ that satisfy:

$$a_N = O(N^{-\alpha}).$$

The best representation is the one for which a given α contains a larger class of functions. It has been shown that the α-class and Besov spaces are closely related:[16]

$$O(N^{-\alpha/2}) \Leftrightarrow f \in B_2^{\alpha},$$

when the error function is L^2. Here, B_2^{α} is a Besov space. In addition, the relation between Besov spaces and continuity conditions of the class of functions in the space have been explored in approximation theory. It may be useful to examine study results.

• Discrete wavelet transforms on unevenly sampled data: The issue of performing the wavelet transform on unevenly sampled data has been side stepped by using the Lomb periodogram. The assumptions made in using the periodogram and the errors introduced in going between continuous and discrete representations need to be explored. Second generation wavelets(12) offer an efficient and elegant alternative to the Lomb periodogram and future work must explore this possibility.

Acknowledgements—This research was conducted at the Center for Research in Intelligent Systems, Department of Computer Science, Columbia University. It was supported in part by an NSF National Young Investigator Award, a David and Lucile Packard Fellowship and in part by ARPA Contract No. DACA 76-92-C-0007. We thank the anonymous reviewer of this paper for his comments that have helped make the presentation more consistent.

REFERENCES

1. T. Poggio and F. Girosi, Networks for approximation and learning, *Proc. IEEE* **78**, 1481–1497 (1990).
2. P. Baldi, Computing with arrays of bell-shaped and sigmoidal functions, *Advances in Neural Information Processing Systems*, R. P. Lippman, J. E. Moody and D. S. Touretzky (eds), pp. 735–742, Morgan Kaufman San Mateo, CA (1991).
3. T. Poggio and S. Edelman, A network that learns to recognize three-dimensional objects, *Nature* **343**, 263–266 (1990).
4. J. M. Hutchinson, A. W. Lo and T. Poggio, A nonparametric approach to pricing and hedging derivative Securities via learning networks, *J. Finance* **XLIX**, 851–889 (July 1994).
5. C. K. Chui, *An Introduction to Wavelets*. Academic Press, San Diego (1992).
6. S. G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(7), 674–693 (1989).
7. A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill Posed Problems*. W. H. Winston, Washington, D C (1977).
8. C. A. Micchelli, Interpolation of scattered data: distance matrices and conditionally positive definite functions, *Construct. Approx.* **2**, 11–22 (1986).
9. N. R. Lomb, Least-Squares frequency analysis of unequally spaced data, *Astrophys. Space Sci.* **39**, 447–462 (1976).
10. M. D. Buhmann and C. A. Micchelli, Spline prewavelets for non-uniform knots, *Numerische Mathematik* **61**, 455–474 (1992).
11. M. D. Buhmann, Multiquadratic prewavelets on nonequally spaced centres, Technical Report 94–06, Seminar für Angewandte Mathematik Eidgenössische Technische Hochshule (July 1994).
12. W. Sweldens, Lifting scheme: a construction of second-generation wavelets, *SPIE Proc. Wavelet Appl. Sign. Image Process. III*, to be published (1995).
13. H. Murase and S. K. Nayar, Learning and recognition of 3D objects from appearance, *Int. J. Comput. Vis.* 1–24 (January 1995).
14. E. Horowitz and S. Sahni, *Fundamentals of Data Structures*. Computer Science Press Int. Inc., Rockville, MD (1993).
15. R. A. Devore, B. Jawerth and V. A. Popov, Compression of wavelet decompositions, *Am. J. Math.* 123–129 (1992).
16. R. A. Devore, B. Jawerth and B. J. Lucier, Image compression through wavelet transform coding, *IEEE Trans. Inf. Theory* **38**(2), 719–746 (1992).
17. M. Unser, A. Aldroubi and M. Eden, On the asymptotic convergence of β-spline wavelets to Gabor functions, *IEEE Trans. Inf. Theory* **38**, 864–872 (March 1992).
18. M. Unser, A. Aldroubi and M. Eden, A family of polynomial spline wavelet transforms, *Sign. Process.* **30**, 141–162 (1993).
19. M. Unser, A. Aldroubi and M. Eden, The L_2 polynomial spline pyramid, *IEEE Trans. Pattern Anal. Mach. Intell.* **15**(4), 364–378 (April 1993).

APPENDIX A

Orthogonal spline wavelets can be written as:

$$\psi(x/2) = \sum_{k \in Z} q(k)\beta^n(x-k), \tag{32}$$

where

$$q(k) = ([-1^k b^{2n+1} * b^{2n+1}]_{\downarrow 2} * b^{2n+1})^{-1/2} \tag{33}$$

and $b^n(k) = \beta^n(x)|_{x=k}$. This shows that wavelets can be constructed from B-splines. In addition, it has been shown that B-splines asymptotically approach Gaussians for a sufficiently large n:(17)

$$\beta^n(x) \cong \sqrt{\frac{6}{\pi(n+1)}}\exp(-6x^2/n+1). \tag{34}$$

The basis functions in higher dimensions are separable B-splines, which are very similar to a product of Gaussian and, therefore, radial basis functions. (Note that the Gaussian is the only function that is both separable and circular.) Thus, one can conclude that all spline wavelets can be represented as sums of functions that are almost radial basis functions.

APPENDIX B

There exist several ways to construct wavelet bases using B-splines as scaling functions. The wavelet basis used in our

Table 3. The components of the transfer functions $\mathring{v}(k)$ and $\mathring{w}(k)$(18,19)

Function	Frequency response
$B_1^7(f)$	$\frac{1}{5040}(2416 + 1191[e^{2\pi f} + e^{-2\pi f}] + 120[e^{4\pi f} + e^{-4\pi f}] + e^{6\pi f} + e^{-6\pi f})$
$U_2^3(f)$	$\frac{1}{8}(e^{4\pi f} + 4e^{2\pi f} + 6 + 4e^{-2\pi f} + e^{4\pi f})$

work was the Battle-Lamarié orthonormal basis.[6] The scaling functions are cubic B-splines:

$$\beta^3(x) = \sum_{j=0}^{4} \frac{-1^j}{3!} \binom{4}{j} (x+2-j)^3 \mu(x+2-j),$$

where μ is the unit step function.

The filter formulae, $\mathring{v}(k)$ and $\mathring{w}(k)$, that were used in Section 3.2 to calculate the wavelet coefficients, $d_{j,k}$, are given in Tables 3 and 4. These formulae were derived by Unser, Aldroubi and Eden.[18,19]

Table 4. The filter formulae for the transfer functions $\mathring{v}(k)$ and $\mathring{w}(k)$[18,19]

Filter	Frequency response
$v^{\circ}(k)$	$\frac{1}{2}U_2^3(f)\sqrt{\frac{B_1^7(f)}{B_1^7(2f)}}$
$w^{\circ}(k)$	$\frac{1}{2}e^{-2\pi f}U_2^3(f+\frac{1}{2})\sqrt{\frac{B_1^7(f+\frac{1}{2})}{B_1^7(2f)}}$

About the Author—SHAYAN MUKHERJEE is an Associate Staff Scientist at the Theoretical Division and Center for Nonlinear Studies at Los Alamos National Laboratory. He received his MS degree from the Department of Applied Physics and Mathematics at Columbia University in 1994. He received his BSE degree in Electrical Engineering from Princeton University in 1992. His primary research interests are in signal processing, nonlinear wave equations and ill-posed problems relating to modeling and understanding hearing and speech, neural networks and multivariate approximation.

About the Author—SHREE K. NAYAR is an Associate Professor at the Department of Computer Science, Columbia University. He received his PhD degree in Electrical and Computer Engineering from the Robotics Institute at Carnegie Mellon University in 1990. In the summer of 1989, he was a Visiting Scientist at the Production Engineering Research Laboratory at Hitachi Ltd, Yokohama. His primary research interests are in computational vision and robotics with emphasis on physical models for early visual processing, sensors and algorithms for shape recovery, learning and recognition, vision based manipulation and tracking, and the use of machine vision for computer graphics and virtual reality. Dr Nayar has authored and coauthored papers that have received the David Marr Prize at the 1995 International Conference on Computer Vision (ICCV'95) held in Boston, Siemens Outstanding Paper Award at the 1994 IEEE Computer Vision and Pattern Recognition Conference (CVPR'94) held in Seattle, 1994 Annual Pattern Recognition Award from the Pattern Recognition Society, Best Industry Related Paper Award at the 1994 International Conference on Computer Vision (ICPR'94) held in Jerusalem and the David Marr Prize at the 1990 International Conference on Computer Vision (ICCV'90) held in Osaka. He holds several U. S. and international patents for inventions related to machine vision. Dr Nayar was the recipient of the David and Lucile Packard Fellowship for Science and Engineering in 1992 and the National Young Investigator Award from the National Science Foundation in 1993.