Automatic Generation of GRBF Networks for Visual Learning*

Shayan Mukherjee
Dept. Applied Physics
Columbia University
New York, NY 10027

Shree K. Nayar
Dept. Computer Science
Columbia University
New York, NY 10027

Abstract

Learning can often be viewed as the problem of mapping from an input space to an output space. Examples of these mappings are used to construct a continuous function that approximates given data and generalizes for intermediate instances. Generalized Radial Basis Function (GRBF) networks are used to formulate this approximating function. A novel method is introduced to construct an optimal GRBF network for a given mapping and error bound using the integral wavelet transform. Simple one-dimensional examples are used to demonstrate how the optimal network is superior to one constructed using standard ad hoc optimization techniques. The paper concludes with an application of optimal GRBF networks to object recognition and pose estimation. The results of this application are favorable.

1 Introduction

Visual learning is fast emerging as an area of great research interest. Often, learning can be equated to constructing a continuous function that maps a given set of inputs to outputs. A common mechanism for performing such a mapping is a neural network. Poggio and Girosi [12] discuss a particularly interesting class of neural networks called Radial Basis Function (RBF) networks. Unlike multi-layer perceptrons and other traditional neural networks, RBF networks have a rigorous formulation and can approximate any function to any specified degree of precision [1]. These networks have been used for recognition of stick figures [11] and face perception [3].

Although RBF networks have a rigorous formulation, this advantage is lost in most practical implementations. The reason is the assumption that RBF networks consist 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 [12]. In this paper, we introduce an analytic method that sets the network parameters for any given input-output mapping and error bound. The GRBF network constructed using this method can be shown to be the smallest network for the given mapping and error bound.

This result is achieved through a novel construction and application of the integral wavelet transform [4]. Wavelet bases are constructed from radial basis functions. The magnitude of the wavelet coefficients determine the importance to the mapping of the radial basis functions that comprise the wavelet. We use the coefficients and Parseval's theorem to determine the number of bases required and their parameters.

The paper is organized as follows. The formulation of an input-output mapping using a GRBF network is described in section 2. The relation between the integral wavelet transform and a GRBF network is presented in section 3. Section 4 presents a simple example that demonstrates the advantage of the transform approach. Finally, the transform approach is applied to the problem of object recognition and pose estimation. The paper concludes with a discussion of other possible applications of the proposed technique.

2 Radial Basis Function Networks

The strength of RBF networks is that input-output mappings are learned in a mathematically rigorous formalism using approximation theory and regularization techniques [13]. The first part of this section is a brief overview of RBF and GRBF networks (see [12] for details).

Learning the mapping between input and output spaces is equivalent to finding a continuous multivariate function that best approximates the given data and interpolates for intermediary instances. Several functions can approximate and interpolate given data. To derive a unique function a priori assumptions about the smoothness of the mapping are made. These smoothness constraints are embedded in the function using regularization techniques. The approximating function is one that minimizes the following cost functional:

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

Where, $F(\mathbf{W},\mathbf{x})$ is the function that approximates the data and $\mathbf{W}$ are the parameters of the function, P is a differential operator determined by the smoothness assumptions, $\mathbf{x}_i$ are the N points at which $f(\mathbf{x})$ is known, and λ is the regularization parameter that controls the tradeoff between enforcing the smoothness constraint P and fitting the known data. The approximating function that minimizes the above

*This research was supported by a NSF National Young Investigator Award, a David and Lucile Packard Fellowship and ARPA Contract No. DACA 76-92-C-0007.

functional has the form :

$$F(\mathbf{W},\mathbf{x}) = \sum_{i=1}^{N} c_i G(\mathbf{x};\mathbf{x}_i) \qquad (2)$$

where, $G(\mathbf{x};\mathbf{x}_i)$ are radial basis functions and c_i are the coefficients of the functions. Two theorems by Micchelli [8] can be used to determine whether a function can be used as a radial basis function. Thin-plate splines and gaussians are two types of functions that meet these conditions (see [12] for a more extensive list).

Casting the approximating function as a network is straightforward. The RBF network has three layers, each fully connected to the next. The first layer consists of a single input unit, the vector $\mathbf{x}$. The second layer consists of multidimensional RBF's. There exists one RBF centered at each data point $\mathbf{x}_i$. The output unit is a weighted sum of these basis functions. Each weight, c_i, is associated with a connection between a basis and the output unit. So the approximating function in (2) can be implemented as the network shown in Figure (1).

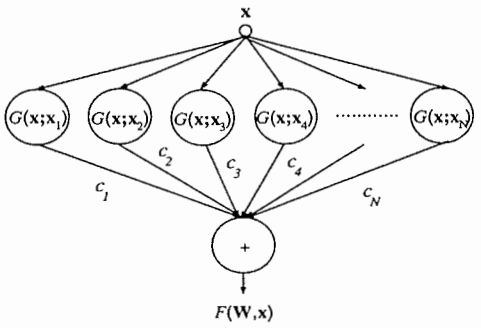

Figure 1: The RBF network has three layers. The first layer is the input vector, $\mathbf{x}$. The second layer consists of the radial basis functions $G(\mathbf{x};\mathbf{x}_1)$ to $G(\mathbf{x};\mathbf{x}_N)$. The c_i are the weights between the i^{th} basis function and the output, $F(\mathbf{W},\mathbf{x})$, which is the third layer.

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 data points. The approximating function, however, is no longer an exact representation of the mapping of interest. Clearly, the representation tends to become worse as the number of basis functions used is reduced. The approximating function now appears as

$$F(\mathbf{W},\mathbf{x}) = \sum_{j=1}^{n} c_j G(\mathbf{x};\mathbf{z}_j) \qquad (3)$$

where $n < N$ and $\mathbf{z}_j$ are the new centers of the basis functions. This type of network is called a Generalized Radial Basis Function (GRBF) Network. The centers and coefficients in (3) are computed by minimizing the following cost functional:

$$H[F(\mathbf{W},\mathbf{x})] = \sum_{i=1}^{N} (f(\mathbf{x}_i) - F(\mathbf{W},\mathbf{x}_i))^2 \qquad (4)$$

This cost functional is almost always 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 then used to adjust the center values and the spans. Then the pseudo-inverse is used to calculate the coefficients, c_j. If the approximation is not within the desired error 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 the RBF approach appealing. The greatest disadvantage 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 GRBF Networks

In this section, we describe how the integral wavelet transform [4] is used to generate a GRBF network. It shall be shown that by using the transform, the smallest number of basis functions required for a given mapping and error bound can be determined analytically. In addition, the parameters of these bases are directly obtained.

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 basis functions that are localized in space and then decompose a function in terms of these bases. For the case of a 1-D function, $f(x)$ ($\mathcal{R}^1 \mapsto \mathcal{R}^1$), the IWT has the basic form [4] :

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

where

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

are the wavelet bases. 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 in the change in $f(x)$ in terms of a span parameter a; (c) The amount of this change in terms of of $(T_\psi f)(b,a)$. In addition, the function $f(x)$ can be reconstructed from the basis functions using:

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

The IWT allows us to decompose functions at different resolution levels, from fine to coarse. The accuracy in reconstructing the original function decreases as one goes to coarser resolution levels. Two functions are involved in

the wavelet transform, a scaling function and the wavelet. The a and b parameters of the wavelet are discretized: $a = \frac{1}{2^j}$ and $b = \frac{k}{2^j}$ $i, j \in Z$ 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 are then used to construct wavelet bases and scaling functions at coarser resolution levels. The wavelet bases are orthonormal across scale and either biorthonomal or orthonormal across position. Using orthonormal bases the following exact representation of a function, $f(x)$, can be obtained [4]:

$$f(x) = \sum_j \sum_k d_j(k)\psi(2^{-j}x - k). \tag{5}$$

where

$$d_j(k) =< f(k), \psi(2^{-j}x - k) > . \tag{6}$$

If we use a scaling function that approximates a radial basis function then (5) is identical to the approximating functions of (2) in section 2. One class of scaling functions that approximate radial basis functions are β-splines.

3.2 Calculating the Wavelet Coefficients

The training data given, $x_i \mapsto f(x_i)$ can be considered a discrete signal. So a Discrete Integral Wavelet Transform [7] is implemented. 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. 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 $\psi_{j,k}$. Mallat's Fast Wavelet algorithm [7] is extended to unevenly sampled data to calculate these coefficients. Mallat's algorithm is pyramidal in nature [7]:

$$c_{j-1}(k) = [\mathring{v} * c_j]_{\downarrow 2}(k) \tag{7}$$

$$d_{j-1}(k) = [\mathring{w} * c_j]_{\downarrow 2}(k) \tag{8}$$

where, $\downarrow 2$ denotes downsampling by two, keep every other term. The c_0 values correspond to the signal at the finest resolution level, i.e. the discrete function $f(k)$. The formulae for $\mathring{v}$ and $\mathring{w}$ for the Battle-Lamarié wavelet basis [7] constructed from cubic β-splines are given in Appendix A. This algorithm is iterated for $j = 0$ to $1 - J$, where 0 is the finest resolution level and $1 - J$ 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:

$$a(x) = f(x) * g(x) \tag{9}$$

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

The Lomb periodogram [6] is used to implement the above convolution. The periodogram performs a Fourier transform on unevenly sampled data, so we can map $f(x) \mapsto F(\omega)$ where $F(\omega)$ are the transform coefficients. A unique continuous function $h(x)$ can then be constructed that has the same spectral properties as $f(x)$, i.e. $H(\omega) = F(\omega)$, and identical spatial properties at all known x_i. The function $h(x)$ has the form :

$$h(x) = \frac{1}{2\pi} \sum_{m=0}^{N_p} F(\omega_o m) e^{i\omega_o m x}. \tag{10}$$

$h(x)$ replaces $f(x)$ in the convolution in (9).

In the fast wavelet algorithm [7], the above technique is used to compute a continuous function $c'_{j-1}(x)$:

$$c'_{j-1}(x) = \frac{1}{2\pi} \sum_{m=0}^{N_p} \mathring{V}\,(m\omega_o) C_j(m\omega_o)\, e^{-i\omega_o m x}. \tag{11}$$

This continuous function $c'_{j-1}(x)$ is sampled at the j^{th} resolution level and then downsampled by 2 to obtain $c_{j-1}(k)$. The same procedure is used to calculate $d_{j-1}(k)$. Approximations due to the oversampling inherent in the periodogram are introduced in this technique. A more rigorous but much more computational intensive method to perform a IWT on unevenly sampled data has been developed by Buhmann and Micchelli [2]. (For more details on our extension of Mallat's algorithm see [9]).

The mappings consider so far have been $\mathcal{R}^1 \mapsto \mathcal{R}^1$. To extend the transform to functions that map $\mathcal{R}^n \mapsto \mathcal{R}^m$, multidimensional functions, $\Phi(\mathbf{x})$ 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 each dimension x_d. The decomposition of $f(\mathbf{x})$ becomes

$$f(\mathbf{x}) = \sum_{j=0}^{1-J} \sum_{k_1(j)=0}^{K_1(j)} .. \sum_{k_n(j)}^{K_n(j)} d_{j,\mathbf{k}}(\mathbf{x})\psi_{j,k_1}(x_1)...\psi_{j,k_n}(x_n) \tag{12}$$

where $\mathbf{k}$ corresponds to the vector $[k_1 k_2 ... k_n]$ and $K_d(j)$ are the number of basis functions at the resolution level j in the dimension x_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 to each dimension. Often we have multiple functions as the output rather than a scalar. A multiple function $\mathbf{f}(\mathbf{x})$ is equivalent to a vector of functions $[f_1(\mathbf{x}), ..., f_l(\mathbf{x}), ...f_m(\mathbf{x})]$ where $f_i(\mathbf{x})$ has the same form as (12) except $d_{j,\mathbf{k}}$ is replaced with $d_{j,\mathbf{k},i}$, the transform coefficient for the i^{th} output at the j^{th} resolution level for the $\mathbf{k}^{th}$ basis. The result of the IWT applied to a mapping from $\mathcal{R}^n \mapsto \mathcal{R}^m$ are the wavelet coefficients $d_{j,\mathbf{k},i}$.

3.3 From Wavelet Coefficients to a GRBF Network

Using the coefficients, $d_{j,\mathbf{k},i}$, and a specified approximation error bound, a GRBF network is constructed. Assume that the L^2 norm between a function $\mathbf{f}(\mathbf{x})$ and the approximating function $\mathbf{F}(\mathbf{W}, \mathbf{x})$ must be less than the error bound, ϵ :

$$\epsilon > \sum_{i=1}^{N} ||(\mathbf{f}(\mathbf{x}_i) - \mathbf{F}(\mathbf{W}, \mathbf{x}_i))||^2 \tag{13}$$

where, N is the number of known examples of $\mathbf{f(x)}$. Since the wavelet bases are orthonormal the energy in the approximating function is

$$||\mathbf{F(W,x)}||^2 = \sum_{l=1}^{m} \sum_{j=0}^{1-J} \sum_{k_1=0}^{K_1'(j)} \cdots \sum_{k_n=0}^{K_n'(j)} d_{j,\mathbf{k},l}^2 \qquad (14)$$

where $K_d'(J) < K_d(j)$. Hence, not all the wavelet bases are used in the approximating function. Using (14), equation (13) can be written as :

$$\epsilon^2 > 1 - \frac{1}{P_t} \sum_{l=1}^{m} \sum_{j=0}^{J-1} \sum_{k_1=0}^{K_1'(j)} \cdots \sum_{k_n=0}^{K_n'(j)} d_{j,\mathbf{k},l}. \qquad (15)$$

where P_t is the total energy in the signal $\mathbf{f(x)}$.

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

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

These sums $s_{j,\mathbf{k}}$ are then ordered from 1 to M so that $s_{j(1),\mathbf{k}(1)} \geq s_{j(2),\mathbf{k}(2)} \geq \cdots s_{j(M),\mathbf{k}(M)}$ where M is the number of wavelet bases in the decomposition of $\mathbf{f(x)}$. Equation (15) can now be expressed as follows:

$$\epsilon^2 \geq 1 - \frac{1}{P_t} \sum_{i=1}^{M'} s_{j(i),\mathbf{k}(i)} \qquad (16)$$

where M' is the smallest integer that satisfies the above condition. The resulting approximating function is

$$\mathbf{F(W,x)} = \sum_{i=1}^{M'} \mathbf{g}_{j(i),\mathbf{k}(i)} \, \Psi_{j(i),\mathbf{k}(i)}(\mathbf{x}) \qquad (17)$$

where $\mathbf{g}_{j(i),\mathbf{k}(i)}$ is a vector composed of the wavelet coefficients $d_{j(i),\mathbf{k}(i),l}$:

$$\mathbf{g}_{j(i),\mathbf{k}(i)} = \begin{pmatrix} d_{j(i),\mathbf{k}(i),1} \\ \vdots \\ d_{j(i),\mathbf{k}(i),m} \end{pmatrix}.$$

The expression in (17) can easily be mapped to a GRBF network as shown in Figure 2. The wavelet bases, $\Psi_{j(i),\mathbf{k}(i)}(\mathbf{x})$, are a linear sum of scaling functions that approximate RBF's asymptotically. For this reason, thes networks are GRBF network.

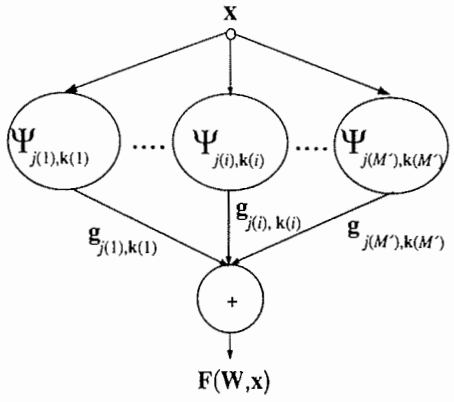

Figure 2: The optimal GRBF network derived using the wavelet transform. The first layer is the input vector, x. The second layer consists of the wavelet bases $\Psi_{j(1),\mathbf{k}(1)}$ to $\Psi_{j(M'),\mathbf{k}(M')}$. $\mathbf{g}_{j(i),\mathbf{k}(i)}$ is the weight between the i^{th} wavelet basis and the output, $\mathbf{F(W,x)}$.

4 A Simple Example

Later in this paper a network is constructed that recognizes objects and estimates their pose in a scene. Here, a simple 1-dimensional example is presented 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 $0 \leq n \leq 63$ and $n \in Z$ (see Figure 3(a) and (b)). The GRBF networks constructed for these two functions using the transform approach will be referred to as *optimal networks*, henceforth. For a comparison, GRBF networks are also constructed using a traditional algorithm, and are referred to as *conventional networks*. The specific algorithm used is outlined below :
(a) Initialize the number of basis functions, n. (b) Set the values of these bases to a subset of the N data points, and set the spans to $\frac{\Delta x}{n}$ where Δx is the span of the input space. (c) Calculate the coefficients using the pseudo-inverse. (d) Use conjugate gradient decent to adjust the centers of the basis functions. (e) Return to (c) until the difference in the error functional in two iterations is negligible. (f) If the error functional is greater than the specified bound, increment n and return to (b).

The accuracy of the optimal and conventional networks as a function of the number of bases, for both functions, is shown in Figure (4). In both cases, the error functional decays faster and monotonically for the optimal network. In the case of the first function f_1 the improvement is negligible. The performance of the optimal network is dramatically superior for the second function f_2, since the optimal network is not susceptible to the problem of local minima in the parameter space.

The parameters of the optimal network are computed

much quicker than those of the conventional network. For the example functions above, the parameters for the optimal network are determined within 9.06sec on a Sun Sprac workstation. In contrast, it can take up to 10.04sec and 20min to set the parameters of the conventional network, depending on the number of basis functions required. Figure (3) shows the reconstruction of the two example functions using the two types of networks. The performance of the optimal network is superior to that of the conventional network for the second function. For the first function the optimal network is slightly more accurate. In summary, the optimal network is constructed more accurately and faster than the conventional network.

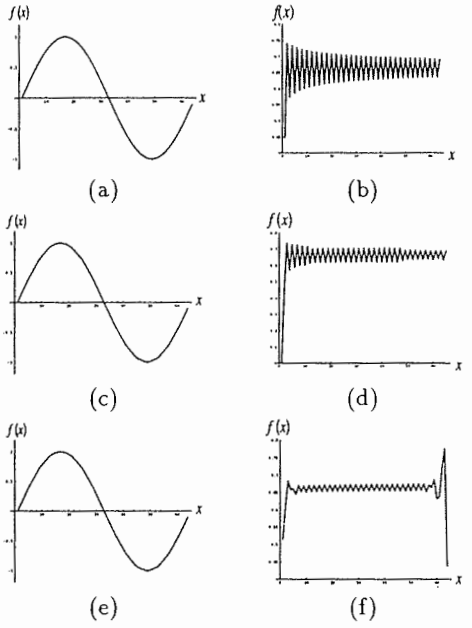

Figure 3: (a) f_1. (b) f_2. (c) Reconstruction of f_1 by an optimal network with 8 bases. (d) Reconstruction of f_2 by an optimal network with 32 bases. (e) Reconstruction of f_1 by a conventional network with 8 bases. (f) Reconstruction of $f_2$the by a conventional network with 32 bases.

5 Learning and Recognition of 3-Dimensional Objects

Object recognition and pose estimation was chosen as a high-dimensional application of GRBF networks. The inputs and outputs that are used to construct the network are those used by the system developed by Murase and Nayar [10] that creates a compact representation of object appearance parameterized by pose. Murase and Nayar construct a subspace (typically of 10-20 dimensions) called the eigenspace from a large image set of several objects in various poses. Every image of an object is mapped to a point in the eigenspace. The set of points in the eigenspace that correspond to all the images of an object is referred to as a *discrete manifold*. Given a novel object image, the object

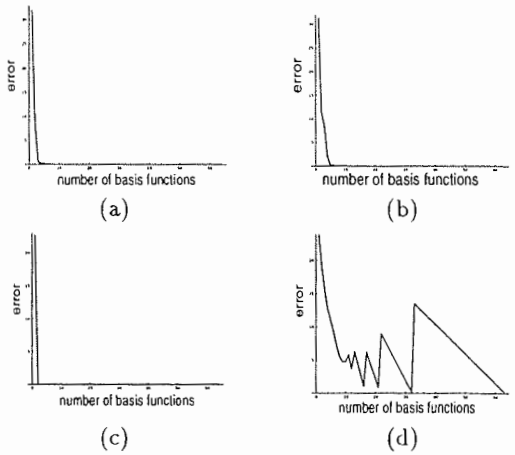

Figure 4: The decay in the L^2 error in approximating f_1 as the number of bases increases for (a) the optimal network and (b) the conventional network. The decay in the error in approximating f_2 for (c) the optimal network and (d) the conventional network.

region is segmented, normalized in brightness and scale and mapped to the 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.

We develop a GRBF network (for each of the 20 objects shown in Figure 6) that uses the discrete manifold of an object to construct a *continuous* mapping from a point in the eigenspace to a confidence value C_p that represents 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 using the transform technique developed in this paper. The result is a set of P optimal networks where P is the total number of objects. During recognition, the input projection in eigenspace is mapped by each of the networks and 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 $\theta_p = 360^o$. The size of any GRBF network is clearly dependent on the continuity of the function it seeks to approximate. Therefore, judicious selection of data representation can dramatically reduce network size. 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. 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 θ_p is computed from $\sin(\theta_p)$ and $\cos(\theta_p)$.

In our experiments, an optimal network was constructed for each of the 20 objects used by Nayar and Murase [10] (see Figure 5). The input space, a 15-dimensional eigenspace, was discretized into 1024 boxes in each of its dimensions.

Figure 5: The 20 objects used in the object recognition and pose estimation experiments (from [10]).

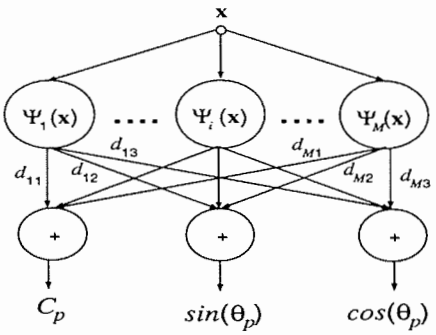

Figure 6: An optimal recognition network for an object, p. The input is a point in eigenspace and the outputs are the confidence C_p, and pose $\sin(\theta_p)$ and $\cos(\theta_p)$.

Clearly, it is impossible to store and process 1024^{15} entries. Instead, a sparse tensor (an extension of the concept of a sparse matrix [5]) 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 set* includes a different set of 36 manifold points and was used to test the accuracy of the networks in generalizing to data not seen before.

The most important task of the set of networks is to correctly recognize objects in both sets. 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, $0.842 \leq C_p \leq 1.217$ when $v = p$ and $0.0 \leq C_p \leq 0.211$ when $v \neq p$, leading to robust object identification.

The networks' accuracy in pose estimation was also studied. For each object, two networks (network_1 and network_2) were constructed using the data in the training set by varying the error bound, ϵ. The error bound for the networks was set such that average absolute pose error for network_1 was 0.5° and 1.5° for network_2, for each object. 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 Figure 7. As expected, the average errors in pose estimation are greater for the test set than the training set. A comparison of the performance of the two networks is summarized in Figure 8.

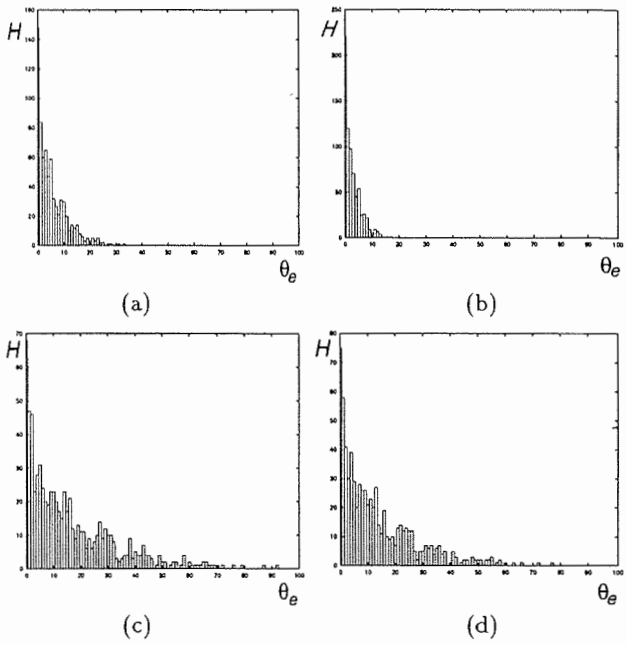

Figure 7: Histograms of the absolute value of the error, θ_e, between poses estimated by network_1 and the known poses, for (a) the test set and (b) the training set. Histograms for network_2 for (c) the test set and (d) the training set. Each bar in the histograms corresponds to 0.1°.

	Time	Average Accuracy (test set)	Average Accuracy (training set)	Memory
Network_1	211msec	0.58°	0.31°	40 kbytes
Network_2	58msec	1.63°	1.36°	13 kbytes

Figure 8: The performance of optimal networks for all objects in terms of the time, accuracy, and memory involved in recognition is summarized above.

6 Conclusion

We have introduced a novel approach to setting RBF network parameters using the integral wavelet transform. This approach allows us to generate optimal RBF networks

given an error bound. 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 1-dimensional functions as well as the multidimensional problem of 3-dimensional object recognition and pose estimation. The networks generated by the transform approach outperformed the other networks. The difference in performance was often dramatic.

The use of optimal RBF networks is not limited to high-level vision processes like recognition. This type of network can be used for edge detection, a low-level vision process. The network can be taught edges of varying angles and sharpness and then find them in an image. For any vision problem that can be formulated from variational and regularization principles, the optimal RBF network should offer an efficient solution. A few problems that can be stated in these terms are shape from shading, both area based and contour based optical flow, and spatiotemporal approximation. Therefore, the ideas presented in this paper have far reaching implications.

7 Appendix A

The filter formulae, $\overset{\circ}{v}(k)$ and $\overset{\circ}{w}(k)$, that were used in section 3.2 to calculate the wavelet coefficients, $d_{j,k}$, are given below. These formulae were calculated by Unser, Aldroubi, and Eden [15] [14].

$$\overset{\circ}{v}(k) \quad \mapsto \quad \tfrac{1}{2} U_2^3(f) \sqrt{\frac{B_1^7(f)}{B_1^7(2f)}}$$

$$\overset{\circ}{w}(k) \quad \mapsto \quad \tfrac{1}{2} e^{-2\pi f} U_2^3(f+\tfrac{1}{2}) \sqrt{\frac{B_1^7(f+\frac{1}{2})}{B_1^7(2f)}}$$

$$B_1^7(z) = \frac{1}{5040}(2416 + 1191[z + z^{-1}] + 120[z^2 + z^{-2}] + z^3 + z^{-3})$$

$$U_2^3(z) = \frac{1}{8}(z^2 + 4z + 6 + 4z^{-1} + z^2)$$

The above filters are iterated across each dimension to yeild, $d_{j,\mathbf{k}}$.

References

[1] P. Baldi, "Computing with Arrays of Bell-Shaped and Sigmoidal Functions," *Advances in Neural Inf. Proc. Systems,* R. P. Lippman, J. E. Moody, and D. S. Touretzky (eds.), Morgan Kaufman, pp. 735-742, 1991.

[2] M. D. Buhmann and C. A. Micchelli, "Spline Prewavelets for Non-uniform Knots," *Numerische Mathematik,* Vol. 61, pp. 455-474, 1992.

[3] R. Brunelli and T. Poggio, "Caricatural Effects in Automated Face Perception," *Biological Cybernetics,* Vol. 69, pp. 235-241, 1993.

[4] C. K. Chui, *An Introduction to Wavelets,* Academic Press, San Diego, 1992.

[5] E. Horowitz and S. Sahni *Fundamentals of Data Structures,* Computer Science Press Int. Inc., 1993.

[6] N. R. Lomb, "Least-Squares Frequency Analysis of Unequally Spaced Data," *Astrophysics and Space Science,* Vol. 39, pp. 447-462, 1976.

[7] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 11, No. 7, pp. 674-693, 1989.

[8] C. A. Micchelli, "Interpolation of Scattered Data: Distance Matrices and Conditionally Positive Definite Functions," *Constructive Approximation,* Vol. 2, pp. 11-22, 1986.

[9] S. Mukherjee and S. K. Nayar, "Automatic Generation of RBF Networks," Technical Report CUCS-001-95, Department of Computer Science, Columbia University, December, 1994.

[10] H. Murase and S. K. Nayar, "Learning and Recognition of 3D Objects from Appearance," *International Journal of Computer Vision,* pp. 1-24, Jan 1995.

[11] T. Poggio and S. Edelman, "A Network That Learns to Recognize Three-dimensional Objects," *Nature,* Vol. 343, pp. 263-266, 1990.

[12] T. Poggio and F. Girosi, "Networks for Approximation and Learning," *Proc. of IEEE,* Vol. 78, pp. 1481-1497, 1990.

[13] A. N. Tikhonov and V. Y. Arsenin *Solutions of Ill Posed Problems* W.H. Winston, Washington D.C., 1977.

[14] M. Unser, A. Aldroubi, M. Eden, "The L_2 Polynomial Spline Pyramid," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 15, No. 4, pp. 364-378, April 1993.

[15] M. Unser, A. Aldroubi, and M. Eden, "A Family of Polynomial Spline Wavelet Transforms," *Signal Processing,* Vol. 30, pp. 141-162, 1993.