A Self-Organizing Auto-Associative Network for the Generalized Physical Design of Microstrip Patches

Bonny Banerjee, *Student Member, IEEE*

***Abstract*—The current work deals with the efficient physical design of patch antennas given the desired parameters like resonant frequency f_r, feed point position a_f, substrate thickness h, relative permittivity ε_r, input impedance $Z\ (= R + jX)$, and efficiency η. Based loosely on the analogy of perception of the human brain, a neurocomputing network has been designed, consisting of two distinct phases, namely, the training phase and the application phase. The training phase accepts as input the exhaustive set of the said parameters for patches of different shapes and sizes and determines the optimized processors (processors that adequately define the information topology of the input data set) from the exhaustive training instances using a set of information extracting self-organizing neural networks. The outputs of the training phase are n sets of processors, n being the number of different shapes of patches taken into consideration. The application phase determines the shape and size of a microstrip antenna when its desired parameters are presented to the network as the external input. This is achieved by comparing the external input with each set of processors, hence determining the cost due to each comparison. A cost matrix is thus formed which when passed through an optimization network gives the best match and hence the shape and shape determining attributes of the patch whose parameters had been passed as external input.**

***Index Terms*—Hopfield-like optimization network, information extracting self-organizing neural network (IESONN), patch antenna.**

I. Introduction

RECENTLY, nonlinear neural optimization networks have been used for the analysis of microstrip patch antennas [1]–[3]. Microstrip patches have become very popular due to their many advantages such as small size, low cost and wide usage in conformal structures. The analyses of microstrip antennas are complicated and exhaustive. In this work, an attempt has been made to obtain the best-suited patch shape and size corresponding to the specified parameters of the patch for a particular application. The analyses of the patch antennas have been done by cavity model [4], [5] to generate the data for the training of the nets. After proper training, the optimization network completely bypasses the repeated analysis of the patch antennas.

This work, in a loose sense, uses the analogy of perception of the human brain to solve an otherwise very difficult problem of ascertaining the shape along with the measure of the shape determining attributes (SDA) of a patch antenna when the desired parameters are given. SDA for a particular shape of a patch antenna may be defined as the attributes necessary and sufficient for specifying the shape. However, these attributes might not be unique. As for example, radius is the SDA of a circular patch antenna while any one side is the SDA for a square patch antenna. Equivalently, any one diagonal might also be considered as the SDA for a square patch antenna. In order to use the stored knowledge to face an unknown situation decisively, the neural nets have been resorted to. The proposed network uses two distinct phases—the training phase and the application phase (vide Fig. 1). The training phase utilizes a set of information extracting self-organizing neural networks (IESONN) [6] to extract and store the relevant information from the exhaustive set of inputs consisting of the different parameters of patches of all possible shapes and sizes while the application phase uses a modified Hopfield-like optimization network [7], [8] to decide the best suitable match for the unknown input using the knowledge stored by the IESONNs.

Manuscript received October 25, 2001; revised January 8, 2002.

The author is with the Department of Electrical Engineering and the Laboratory for Artificial Intelligence Research (LAIR), The Ohio State University, Columbus, OH 43210 USA (e-mail: banerjee@cis.ohio-state.edu).

Digital Object Identifier 10.1109/TAP.2003.812266

II. Training Phase

This section deals with the phase responsible for extracting meaningful information from a huge multidimensional data set. For the purpose, the IESONN [6] has been resorted to. The IESONN is a variant of the Kohonen's self-organizing feature map (SOFM) [9], [10]. The SOFM has a fixed neighborhood topology as a result of which the topology of the input pattern cannot be completely adapted [10], [11]. Hence, it is necessary to implement a dynamic change in the network topology. Though, several variants of the SOFM were reported in different contexts [11]–[14], yet in almost all of them, the weights specify the clusters or vector centers of the set of input vectors such that the point density function of the vector centers tends to approximate the probability density function of the input vectors [9]. As a result, these algorithms do work well to explore structures in cases of multidimensional data sets only when the data points are uniformly distributed. The IESONN will be considered here as it overcomes this drawback and has properties most beneficial for the purpose of the present work.

The IESONN is initialized with a very small number of noninterconnected processors, the weight corresponding to each of which assumes random initial values. Each feature vector, presented to the IESONN, is associated with an input vector from the m dimensional input space, and an output vector from the p dimensional output space. The weight vectors of the processors, having exactly the same input/output dimensions as the features, are updated iteratively on the basis of the feature vectors in S, $S = (P_1, P_2, \ldots, P_N)$ being the set of feature vectors defining the neighborhood of a processor initially. It might be noted that S is initialized to contain the entire set of feature vectors but as time proceeds, S will dynamically shrink to define a specific

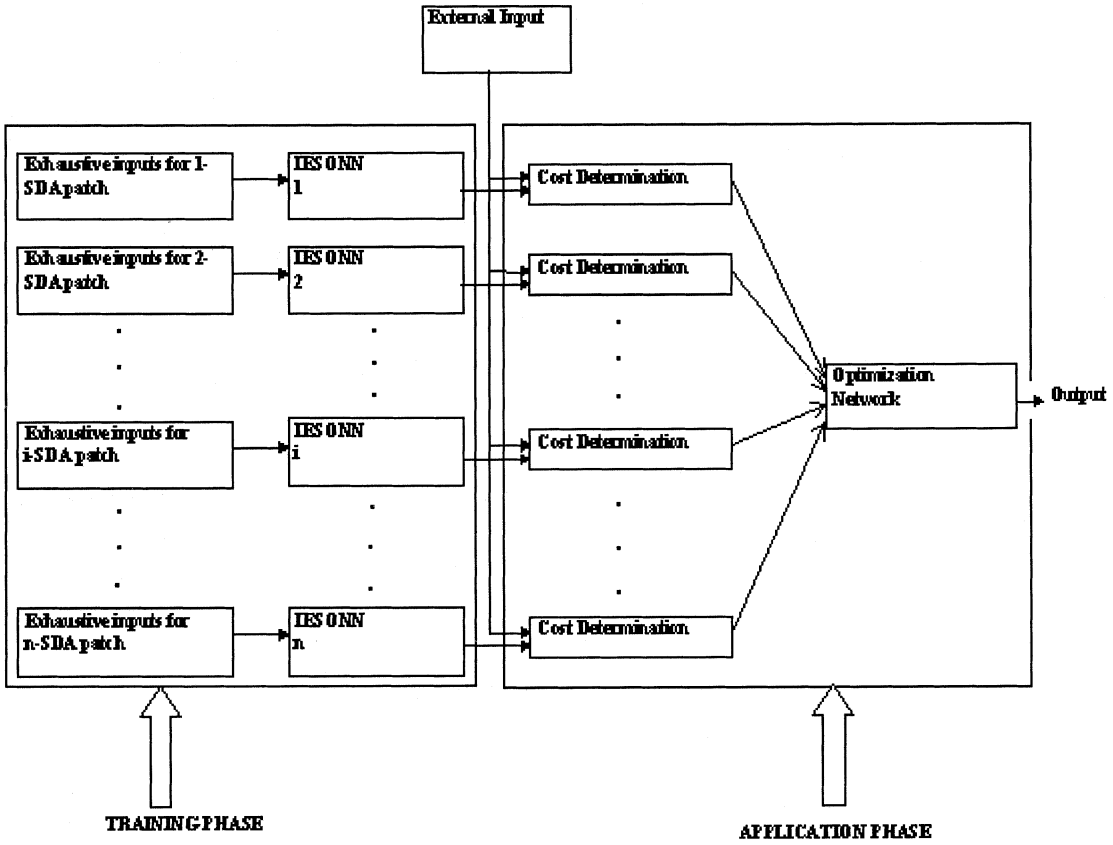

Fig. 1. Block diagram showing the different units of the training phase and the application phase of the overall system. For example, for a circle, the SDA denotes its radius.

neighborhood unique for each processor. On convergence, a set of processors whose weight vectors efficiently represent the informative features of the exhaustive input set is obtained. From each IESONN in the memory of the proposed network (vide Fig. 1), we obtain a set of such weight vectors.

In IESONN model, initially the topology is completely data driven. The net grows in size by means of a certain processor evolution mechanism [9], given by

$$W_p(t+1) = W_p(t) + \alpha(t)\left[P_j - W_p(t)\right], \qquad 0 < \alpha(t) < 1 \tag{1}$$

where $\alpha(t)$ is the gain term which decreases with t, $(\pi_1, \pi_2, \ldots, \pi_n)$ denote the set of processors and $W_p(t)$ is the weight vector for the processor π_p at the time t. At time instant t, P_j is presented to the net. All the processors compete and two weight vectors $W_k(t)$ and $W_l(t)$ are selected such that

$$\|W_k(t) - P_j\| = \min_i \|W_i(t) - P_j\| \tag{2}$$

$$\|W_l(t) - P_j\| = \min_{(i \neq k)} \|W_i(t) - P_j\|. \tag{3}$$

Thereafter P_j modifies the weight vector $W_k(t)$ according to (1). However, $W_l(t)$ is also modified according to (1) but with a lesser gain if and only if it lies within the neighborhood defined by S at that time instant t. In this process the modification of the weights is continued, the weights tend to approximate the information content of the feature set in an orderly fashion. The limiting weight vectors define the ordering. One presentation each of all the vectors makes one *sweep* consisting of N iterations. After one sweep is completed, the iterative process for the next sweep starts again from P_1 through P_N. Several sweeps make one *phase*. One phase is completed when the weight vectors of the current set of processors converge, that is, when

$$\|W_i(t) - W_i(t')\| < \varepsilon, \forall\, i \tag{4}$$

where t and t' are the iteration numbers at the end of two consecutive sweeps and ε is a predetermined small positive quantity that decreases with each phase exponentially.

After the completion of a phase, connectivity is assigned to the processors. For each pair of weight vectors, say W_k and W_l, the number of feature vectors is calculated for which W_k is the nearest and W_l is the second nearest. Thus the entire space spanned by the feature vectors is partitioned with respect to the weight vectors and the cardinality of the partitions is stored in the form of a matrix, say N. If $N(k,l) > \theta, \theta$ being a predefined threshold, then a link between the processors W_k and W_l is established.

After the completion of a particular phase, a new phase starts with the introduction of a new processor. The eigen values of the correlation matrix of each partition is computed and a new processor is introduced in that partition which has the minimum principal eigen value among all the partitions. The correlation matrix provides a measure of the correlation among the different dimensions of all the elements in the partition. It might be inferred that better the correlation is, lesser is the information content in the partition [6]. This process continues until there are enough processors to represent the informative structure of the multidimensional data set under consideration. In [6], Banerjee provides a detailed discussion on the various issues related to the

IESONN and how information might be interpreted for practical purposes like in the present problem.

III. Application Phase

The application phase is a decision making phase, the decision being achieved by deploying a continuous Hopfield-like optimization network [8]. An optimization network [7], [15]–[19] can be used to solve discrete combinatorial optimization problems like the Travelling Salesman Problem (TSP). The current phase utilizes an altogether new approach for determining the best suitable match using parallel computation. Solutions of this form are unique and at the same time elegant for two specific reasons. First, the cost matrix, formed for the purpose of parallel computation, has an immense information storing capability. In fact all the information available about the parameters of the patches participating in the training phase is contained in the matrix which thus helps in efficient interpretation. Secondly, once the cost matrix is formed, a parallel computation algorithm based on the lines of Hopfield and Tank's optimization network [7], [8], [15] allows fast and easy interpretation of the stored information. So, the current section focuses on two main aspects—information storage in a cost matrix and interpretation of the stored information using an optimization network.

The correct decision is achieved by matching the input parameters of the external input with each set of weight vectors obtained from the IESONNs and associating a cost with each, reflecting the degree of each match. These costs form a two-dimensional (2-D) cost matrix, which is presented to the input of the optimization network. The energy function of the network is designed such that on convergence, a 2-D matrix having exactly one element as unity and all others zero is obtained, the unity element giving the best match out of all the matches. Such a matrix is obtained by minimizing the energy function E_P of the optimization network, where E_P is specifically designed to meet the requirements of the problem as follows:

$$\begin{aligned} E_P =& \frac{\mu_1}{2}\sum_{X=1}^{m}\left(\sum_{j=1}^{n}V_{\mathrm{Xj}}\right)\left(\sum_{j=1}^{n}V_{Xj}-1\right) \\ &+\frac{\mu_2}{2}\sum_{j=1}^{n}\left(\sum_{X=1}^{m}V_{\mathrm{Xj}}\right)\left(\sum_{X=1}^{m}V_{\mathrm{Xj}}-1\right) \\ &+\frac{\mu_3}{2}\sum_{X=1}^{m}\sum_{j=1}^{n}V_{\mathrm{Xj}}(1-V_{\mathrm{Xj}}) \\ &+\mu_4\sum_{X=1}^{m}\sum_{j=1}^{n}C_{\mathrm{Xj}}V_{\mathrm{Xj}} \\ &+\frac{\mu_5}{2}\left(\sum_{X=1}^{m}\sum_{j=1}^{n}V_{\mathrm{Xj}}-1\right)^2 \end{aligned} \tag{5}$$

see (6) at the bottom of the page. C_{Xj} is the cost due to matching the external input with the Xth output of the IESONN corresponding to the jth shape and μ_i are constants to be determined, $\forall\, i$.

The first two terms in (5) have minima if the sum over all outputs equals either 1 or 0 for only one row and one column respectively. The third term has minima if all V_{Xj} are either 1 or 0 and, together with the first two terms, it enforces the constraints. The fourth term in (5) is simply the overall cost of a particular solution given the constraints are met. The last term further reinforces the above constraints by ensuring that there exists only a single 1 in the matrix at convergence. Furthermore, it is common to use constant factors μ_1, μ_2, μ_3, μ_4, and μ_5 as additional parameters in (5). These parameters have the effect of "weighing" the constraints and the cost-function and allow a fine-tuning of the performance [16].

For mapping (5) into an optimization network, the values for the connections and the external inputs are to be derived. First, extending the notation of the Liapunov function to two dimensions, we obtain

$$E=-\frac{1}{2}\sum_{X=1}^{m}\sum_{j=1}^{n}\sum_{Y=1}^{m}\sum_{k=1}^{n}T_{\mathrm{Xj,Yk}}V_{\mathrm{Xj}}V_{\mathrm{Yk}}-\sum_{X=1}^{m}\sum_{j=1}^{n}V_{\mathrm{Xj}}I_{\mathrm{Xj}}. \tag{7}$$

By comparing (7) and (5) it follows after some algebraic transformations that $E = E_P$ if

$$T_{Xj,Yk} = -\mu_1\delta_{\mathrm{XY}}-\mu_2\delta_{\mathrm{jk}}+\mu_3\delta_{\mathrm{XY}}\delta_{\mathrm{jk}}-\mu_5 \tag{8}$$

$$I_{Xj} = \frac{\mu_1}{2}+\frac{\mu_2}{2}-\frac{\mu_3}{2}-\mu_4C_{Xj}+\mu_5 \tag{9}$$

where δ is the Kronecker delta defined by

$$\delta_{\mathrm{ab}}=\begin{cases}1, & \text{if } a=b\\ 0, & \text{otherwise.}\end{cases} \tag{10}$$

Fig. 2 sketches the resulting 2-D network architecture as a directed graph. With the specific values from (8) and (9), the equations of motion for the network may be given as follows:

$$\begin{aligned} \varsigma_{\mathrm{Xj}}\frac{du_{\mathrm{Xj}}}{dt} =& -\frac{u_{\mathrm{Xj}}}{R_{\mathrm{Xj}}}-\mu_1\sum_{k=1}^{n}V_{\mathrm{Xk}}-\mu_2\sum_{Y=1}^{m}V_{\mathrm{Yj}} \\ &+\mu_3V_{\mathrm{Xj}}-\mu_5\sum_{X=1}^{m}\sum_{j=1}^{n}V_{\mathrm{Xj}}+\frac{\mu_1}{2} \\ &+\frac{\mu_2}{2}-\frac{\mu_3}{2}-\mu_4C_{\mathrm{Xj}}+\mu_5. \end{aligned} \tag{11}$$

The cost function designed for the matching purpose may be given by

$$\begin{aligned} C_{\mathrm{pq}} =& \lambda_1(m_{\mathrm{pq}}-m)^2+\lambda_2(n_{\mathrm{pq}}-n)^2+\lambda_3(f_{\mathrm{rpq}}-f_r)^2 \\ &+\lambda_4(\varepsilon_{\mathrm{rpq}}-\varepsilon_r)^2+\lambda_5(R_{\mathrm{pq}}-R)^2 \\ &+\lambda_6(X_{\mathrm{pq}}-X)^2+\lambda_7(a_{\mathrm{fpq}}-a_f)^2 \\ &+\lambda_8(h_{\mathrm{pq}}-h)^2+\lambda_9(\eta_{\mathrm{pq}}-\eta)^2 \end{aligned} \tag{12}$$

$$V_{\mathrm{Xj}}=\begin{cases}1, & \text{if the external input matches with the } X\text{th output of the IESONN corresponding to the } j\text{th shape}\\ 0, & \text{otherwise.}\end{cases} \tag{6}$$

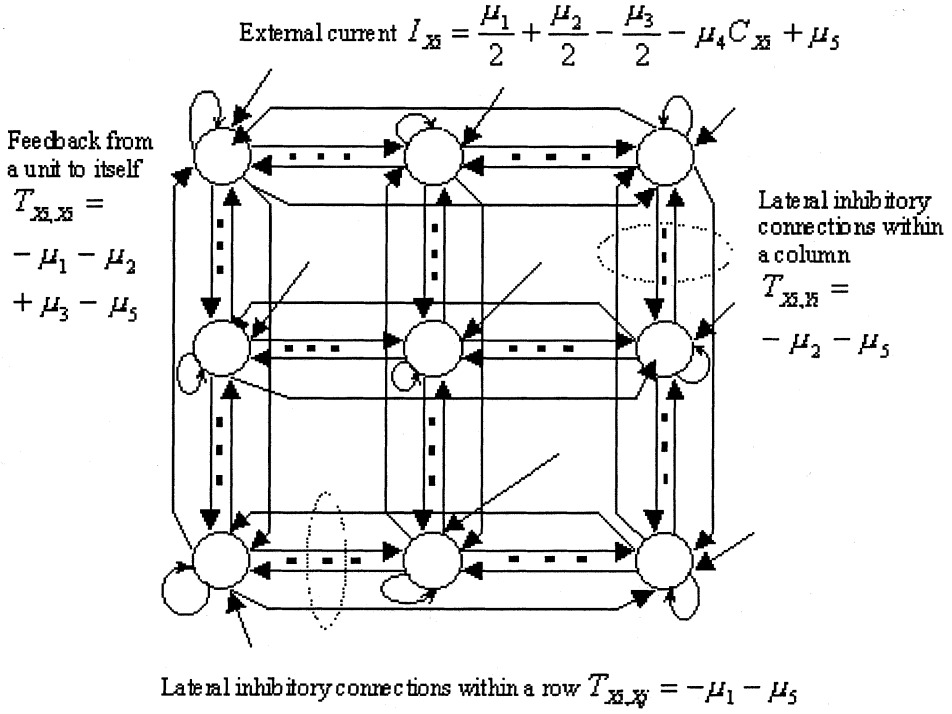

Fig. 2. Schematic architecture of a 2-D optimization neural network with the connectivity required to achieve the parallel computation in the application phase of the solution [23].

where (m, n) is the mode number, f_r is the resonant frequency, ε_r is the relative permittivity, $Z(= R + jX)$ is the input impedance, a_f is the feed point position, h is the substrate thickness while η denotes the efficiency of the antenna under consideration as the external input. The subscript "pq" refers to the pth output of the IESONN corresponding to the qth shape. $\lambda_i \forall i$, are constants to be determined and are dependent on the nature of influence of a particular parameter on the physical shape of the antenna.

Now this cost function is applied to evaluate the cost matrix for storing the information obtained from the IESONNs after the training phase is over. The cost matrix that is to be formed here will, in general, not be a square matrix. In fact one will be unable to predict the number of rows of the matrix beforehand as it will be formed dynamically after the training phase. The columns of the cost matrix contain information about the different shapes while the rows contain those of sizes. Table II shows an example of a typical cost matrix.

Once the cost matrix has been formed, the task of efficient interpretation of the information stored therein is achieved with the help of the optimization network, the initial concept of which was originated by Hopfield and Tank [7] and later modified by many [16]–[19]. The reasons for the use of such a network are manifold. First, it allows parallel computation and, hence, reduces considerable time in case of large cost matrices i.e., it helps to provide the required interpretation in real time. Second, the proposed model will be adaptive to changes in costs and variation in the number of patch sizes and shapes. Finally, it is highly fault-tolerant.

An advantage of the proposed model is that it maps the data (here defined by C) into the biases rather than into the neural interconnections [16], [17]. This is due to the fact [18], [20], [21] that the data terms are associated with linear rather than quadratic expressions in the energy function, given by (5). One advantage of the proposed representation scheme is flexibility reflected by the fact that costs $C_{\rm pq}$'s and the network topology information can be changed through the biases. Since, in this case, we do not have to modify the internal parameters of the neural net architecture, then the proposed algorithm can adapt rapidly to changes in network topology and costs. This will make the algorithm very attractive to operate in real time. Another advantage is that the interconnection strengths do not depend on a particular training or input pattern. Hence, the neural network can find the best solution by properly choosing the input biases as given in (8) and (9). In addition the connection matrix of the proposed algorithm is simple and requires minimal connections.

IV. Results

Given the initial neurons' input voltages $u_{\rm Xj}$'s at time $t = 0$, the time evolution of the state of the optimization network is simulated by numerically solving (11). This corresponds to solving $m.n$ nonlinear differential equations, where the variables are the neurons' output voltages $V_{\rm Xj}$'s. To achieve this, the Predictor-Connector method was chosen since it is sufficiently accurate and easy to implement. Accordingly, the simulation consisted of observing and updating the neuron output voltages at incremental time steps δt. In addition, the time constant τ of each neuron is set to 1 without any loss of generality and for simplicity, it is assumed that $\lambda_{\rm Xj} = \lambda$ all independent of the subscript (X, j) [17]. Simulation has shown that a good value of δt is 10^{-5}. Reducing this tolerably small value increases the simulation time without improving

TABLE I
EXTERNAL INPUTS TO THE NEURAL NETWORK AND THEIR OUTCOMES, COMPARED WITH THE STANDARD RESULTS GIVEN IN [5]

Inputs to the Neural Network							Outputs from the Neural Network		Patch Shape and Size(cm) as in [5] corresponding to the given inputs
Mode No. (m,n)	f_r (MHz)	a_f (cm)	ε_r	h (cm)	η (%)	R+jX (Ω)	Shape	Measure of shape-determining attributes (cm)	
(1,0)	810	0.0	2.62	0.159	65	6.0 + j0.3	Square	11.3	Square, 11.3cm
(0,1)	813	8.38	2.62	0.159	70	3.0 + j0.0	Rectangle	11.39, 7.58	Rectangle, 11.4, 7.6
(1,1)	805	0.0	2.62	0.159	60	6.0 + j0.0	Circle	6.72	Circle, 6.75
(1,0)	805	3.84	2.62	0.159	40	10.0 + j0.0	Equilateral Triangle	15.358	Equilateral Triangle, 15.36

the results. Another important parameter in the simulation is the neuron's initial input voltage u_{Xj}'s. Since the neural network should have no *apriori* favor for a particular code, all the u_{Xj}'s should be set to zero. However, some initial random noise $-0.0002 \leq \delta u_{\mathrm{Xj}} \leq +0.0002$ will help to break the symmetry caused by symmetric network topology [17]. The simulation is stopped when the system reaches a stable final state. This is assumed to occur when all neuron output voltages do not change by more than a threshold value of $V_{\mathrm{th}} = 10^{-5}$ from one update to the next. At the stable state each neuron is either ON ($V_{\mathrm{Xi}} > 0.5$) or OFF ($V_{\mathrm{Xi}} < 0.5$). One major issue related to the efficiency of the Hopfield model in solving combinatorial optimization problems is the lack of rigorous guidelines in selecting appropriate values of the energy function coefficients [16], [17]. A trial and error approach, along with intuition, has been extensively applied throughout the literature.

Experimentally, it has been found that there is a compromise between choosing a small or a large value of the neural transfer parameter λ. While a large λ gives rise to a fast neural response for which the solution is not always a global minimum, a small λ yields a slower response which can guarantee an optimum solution. In order to allow the neurons' dynamics to wander freely in their state space [17] in the search for the global minima, λ is chosen to be 0.25. The simulated neural algorithm is run quite a few times using different patch shapes and sizes.

Simulations had been performed using circular, semi-circular, elliptical, triangular and rectangular patches. Data have been generated for these patches over a range of $1 < \varepsilon_r < 15$ and $0.02 < h < 0.3$. After training the IESONNs, some of the external inputs presented to the network along with the corresponding outputs from the network are shown in Table I. Unless otherwise stated, in all the observations, $\mathrm{TE_{mn}}$ or $\mathrm{TM_{mn}}$ modes have been used with f_r in megahertz, R in ohm, X in ohm, a_f in centimeters, h in centimeters and measure of the SDA in centimeters.

As for example, in the first observation $m = 1$, $n = 0$, $f_r = 810$ MHz, $\varepsilon_r = 2.62$, $R = 6.0\ \Omega$, $X = 0.3\ \Omega$, $a_f = 0.0$ cm, and $h = 0.159$ cm. At convergence, the output of the optimization net indicated that a square patch antenna having each side length equal to 11.3 cm best suits the external input parameters. This exactly matches with Lo's report [5]. Thus the proposed network successfully interprets the shape and measure of the SDAs from the stored information. It may be noted that the observations shown in Table I were not included in the exhaustive set of inputs presented to the IESONNs at the start of the training phase. The number of iterations required for the application phase to converge varies between 10 and 50. It generally depends on the dimension of the cost matrix as well as on the elements of the cost matrix. Larger the dimensions, more iterations are required. Also, if the relative differences of the elements of the cost matrix are lesser, number of iterations increases. However, the network always converged within two seconds in a Pentium III 733 MHz PC. Hence, the network is suitable for real time applications. Thus the proposed network works similar to human perception with respect to interpretation from previously stored knowledge. Again, the elegance of this network lies in the fact that it bypasses all complex calculations in determining the shapes and measure of the SDAs in real time. Moreover, a network to determine the shape of a patch antenna that would be suitable for the given set of parameters is perhaps the first of its kind.

TABLE II
TYPICAL COST MATRIX OBTAINED BY APPLYING (12). THE ELEGANCE OF THE COST FUNCTION C_{Xj} LIES IN ITS IMMENSE INFORMATION EXTRACTING CAPABILITY. APPARENTLY, THE COST MATRIX CONTAINS ALL THE INFORMATION THAT HAS BEEN EXTRACTED BY THE IESONNS FROM THE EXHAUSTIVE SET OF INPUTS PROVIDED TO THE TRAINING PHASE

10.9	30.1	24.6	36.4	60.8	64.0	20.1
42.6	20.0	30.3	60.1	36.6	30.5	40.9
40.2	72.3	60.2	3.2	12.7	45.9	60.1
18.9	24.7	30.2	20.5	18.1	10.0	36.2

V. DISCUSSION

The results show that the performance of the proposed neural architecture is efficient with respect to accuracy as well as speed. In case of the problem under consideration, the neural network has been properly designed (through careful choice of the weighing coefficients) to guarantee convergence to valid solutions which are often global minima. This is also supported

by earlier results where it was found that neural networks with linear cost functions perform much better than those with quadratic cost functions. Further, although it is intended to solve a problem which is by far simpler than the nondeterministic polynomial time (NP) complete TSP, the Hopfield energy function inherits a key problem, namely the growing tendency to converge toward an invalid final stable state as the network size gets larger [16], [22]. But fortunately, here, this problem was not encountered because our constraints were much loosely bound. However, the number of iterations required for valid convergence was considerable until an adaptive nature of the increment δt was resorted to [19]. δt was initialized at 0.0005 and gradually decreased to provide time for finer convergence using the following rule:

$$\delta t = \frac{\delta t}{1 + \frac{\text{iterations}}{100}}. \tag{13}$$

As a result of this adaptive increment, our algorithm converged within 50 or lesser iterations with the stable states occurring at $V_{\text{Xi}} = 0.9999$ or, 0.0000.

VI. Conclusion

Thus an elegant solution to the problem of generalized physical design of microstrip patch antennas is proposed, exploiting the synergism of a set of IESONNs and a modified Hopfield type optimization network. The proposed model combines many features, such as a very good convergence and scaling properties, a relatively low programming complexity, an ability to operate in real time and to adapt to changes in network topology and costs. From the simulation results, obtained with different degrees of problem complexity, it can be concluded that the proposed model is both efficient and effective in determining the shape and the measure of the SDA corresponding to the unknown input.

Acknowledgment

This project was initiated at the Department of Electronics and Telecommunication Engineering, Jadavpur University, Calcutta, India and was successfully completed as a part of independent studies in the Department of Electrical Engineering, The Ohio State University, Columbus, OH. The author gratefully acknowledges the guidance and support rendered by Dr. I. S. Misra in making this project a reality. The author would like to thank Dr. Glisson and the anonymous referees for their constructive comments. Thanks are also due to S. Martinez for her timely responses.

References

[1] L. Vegni and A. Toscano, "Analysis of microstrip antennas using neural networks," *IEEE Trans. Magn.*, vol. 33, pp. 1414–1419, Mar. 1997.

[2] A. Patnaik *et al.*, "An artificial neural network model for determining effective dielectric constant of microstrip line," *IEEE Trans. Antennas Propagat.*, vol. 45, p. 1697, Nov. 1997.

[3] B. Banerjee and I. S. Misra, "A neuro-computing model for the design of patch antennas," in *Proc. Int. Conf. Communications, Computers and Devices (ICCCD)*, Kharagpur, India, 2000, pp. 340–343.

[4] J. R. James and P. S. Hall, *Handbook of Microstrip Antennas*, UK: Peter-Peregrinus Ltd., 1989, vol. 1.

[5] Y. T. Lo *et al.*, "Report on Study of Microstrip Antennas, Microstrip Phased Arrays and Microstrip Feed Network," Elect. Eng. Dept., Univ. Illinois, Urbana, IL, RADC-TR-77-406.

[6] B. Banerjee, "An information extracting self-organizing neural network," presented at the IEEE Annu. Conv. Exhibition, India, Dec. 2002.

[7] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," in *Proc. Natl. Acad. Sci. USA, Biophys.*, vol. 81, May 1984, pp. 3088–3092.

[8] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems," *Biol. Cybern. 52*, pp. 141–152, 1985.

[9] T. Kohonen, *Self-Organization and Associative Memory*. New York: Springer-Verlag, 1989.

[10] J. A. Kangas, T. Kohonen, and J. Laaksonen, "Variants of self-organizing maps," *IEEE Trans. Neural Networks*, vol. 1, pp. 93–99, 1990.

[11] D. Choi and S. Park, "Self-creating and organizing neural networks," *IEEE Trans. Neural Netw.*, vol. 5, pp. 561–575, Aug. 1994.

[12] A. Datta, T. Pal, and S. K. Parui, "A modified self-organizing neural net for shape extraction," *Neurocomput.*, vol. 14, pp. 3–14, 1997.

[13] B. Fritzke, "Let it grow—Self-organizing feature maps with problem dependent cell structure," in *Proc. Int. Conf. Artificial Neural Networks*, vol. 1, North-Holland, Amsterdam, 1991.

[14] M. Sabourin and A. Mitiche, "Modeling and classification of shape using a kohonen associative memory with selective multiresolution," *Neural Netw.*, vol. 6, pp. 275–283, 1993.

[15] D. W. Tank and J. J. Hopfield, "Simple 'neural' optimization networks: an A/D converter, signal decision circuit and a linear programming circuit," *IEEE Trans. Circuits Syst. II*, vol. CAS-33, pp. 533–541, May 1986.

[16] P. W. Protzel, D. L. Palumbo, and M. K. Arras, "Performance and fault-tolerance of neural networks for optimization," *IEEE Trans. Neural Netw.*, vol. 4, July 1993.

[17] M. K. M. Ali and F. Kamoun, "Neural networks for shortest path computation and routing in computer networks," *IEEE Trans. Neural Netw.*, vol. 4, Nov. 1993.

[18] R. D. Brandt, Y. Wang, A. J. Laub, and S. K. Mitra, "Alternative networks for solving the travelling salesman problem and the list matching problem," in *Proc. IEEE Int. Conf. Neural Networks*, San Diego, CA, July 1988, pp. II-333–340.

[19] B. Banerjee and P. Mukhopadhyay, "Recognition of partially occluded and/or distorted shapes in a cluttered scene," B.E. thesis, Jadavpur Univ., Calcutta, India, 2000.

[20] P. W. Protzel, "Comparative performance measure for neural networks solving optimization problems," in *Proc. Int. Joint Conf. Neural Networks*, 1990, pp. II-523–626.

[21] A. Moopenn, T. Duong, and A. P. Thakoor, "Digital-analog hybrid synapse chips for electronic neural networks," in *Advances in Neural Information Processing Systems*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990, vol. 2, pp. 769–776.

[22] S. Hedge, J. Sweet, and W. Levy, "Determination of parameters in a hopfield/tank computational network," in *Proc. IEEE Int. Conf. Neural Networks*, San Diego, CA, July 1988, pp. II-291–298.

[23] J. A. Anderson, *An Introduction to Neural Networks*. Cambridge, MA: MIT Press, 1995.

Bonny Banerjee (S'00) received the B.E. degree in electronics and telecommunication engineering with honors from Jadavpur University, Calcutta, India, in 2000 and the M.S. degree from the Department of Electrical Engineering, The Ohio State University, Columbus, in 2002.

Since September 2001, he has been a Graduate Research Associate in the Laboratory for Artificial Intelligence Research (LAIR), Department of Computer and Information Science at The Ohio State University. His current research interests include neural networks, artificial intelligence and computational aspects of cognition.

Mr. Banerjee received the National Scholarship in 1996 and 1994, the M. V. Chauhan Merit Certificate by the IEEE India Council in 1999, and the J. N. Tata Scholarship and Jamsetji Tata Gift Award from 2000 to 2001. He is a student member of the IEEE and the IEEE Computer Society.