'89 KACC 1989. 10. 27~28

A Composite Neural Network Architecture for Inverse Kinematic Robot Control Based on a Kohonen-BP Algorithm *

°Jae-Myeong Song and Se-Young Oh

Department of Electrical Engineering
Pohang Institute of Science and Technology

A new neural network architecture has been developed that combines a modified version of the Kohonen's self-organizing feature map and the backpropagation network. This network is used to solve the inverse kinematic problem of a simulated 2-link robot manipulator. The results indicate this architecture demonstrates better accuracy and convergence properties than the backpropagation network.

1 Introduction

In general, control makes a given plant perform a certain desired task. Control problem is essentialy a question of finding a nonlinear mapping between the sensory space and the acuator command space. Neural networks have been drawing a great deal of attention recently as a great way to realize this nonlinear mapping. Instead of the conventional way of mathematical modeling, neural nets can implicitly learn this mapping through adaptation and learning due to emergent properties of neurons.

Many papers have come out during the past two years that apply neural nets to inverse kinematic robot control[1-4]. This neural net control (neurocontrol) has so many great potentials that it commands great promises for the future applications to robotics and automation. Its features include:

a) Due to Massive Parallelism of Neural Nets

1. Real-Time Control
2. Fault-Tolerance
3. Robustness against Noise

b) Due to the Learning Capability of Neural Nets

1. No Need for the Model of the Plant or its Environment
2. Adaptation to Changing Environment
3. Easy Inclusion of Sensor Data Fusion
4. Constant Performence Improvement thrpugh Learning
5. No Need for the Knowledge on Difficult Control
6. Simple Programming

Up to now, the backpropagation (BP) network[5,6] has been the mainstay for neural nets for control. Although any continuos mapping can be approximately realized by the BP network[7], the BP network has some severe disadvantages:

1) Painfully slow training

2) No guarantee of convergence

3) Accuracy problem.

Much research has been done to overcome these limitations[8-12]. However, it seems that part of the BP's accuracy and convergence problem may come from the fact that BP attempts to fit a desired nonlinear mapping in the entire input space with the same set of neurons. This may be pushing the BP's flexibility to the limit.

In this paper, we introduce a composite neural network architecture to alleviate some of the above disadvantages of the BP. Somewhat like the divide-and-conquer method, this network uses a modified version of Kohonen's self-organizing feature map (SOFM)[13] to partition the input space and then the subsequent BP network acts on the local weights as selected by the SOFM. It essentially realizes a piecewise linear mapping in a small partitioned region. It has been named the Kohonen-BP (K-B) network.

2 The Proposed K-B Network Architecture

As described above, the K-B net consists of two successive layers, the SOFM and the BP net.

*This research was supported by a grant form the Korean Science Foundation and also in part by a grant from the Research Institute of Industrial Science & Technology.

2.1 Kohonen's Feature Map

Figure 1 shows the SOFM architecture. The j-th output node computes its input as follows:

$$x_j = \sum_i I_i W_{ji} \quad (1)$$

Where I_i is the i-th component of the input pattern vector and W_{ji} is the weight from the i-th input node to the j-th output node. Since all input pattern vectors and weights of each output node is normalized to unity, x_j is the dot product between the input pattern vector and the weight vector, indicating a similarity measure.

There are lateral interactions between the output nodes, as shown in Figure 2. Competition occurs through these lateral inhibitions and the node whose weight is most similar to the input pattern becomes the winner of this competition according to Eq. (2). The winning node has an output of 1 and all the others have outputs of 0.

$$O_j(t) = f(x_j + \sum_k \gamma_k O_{j+k}(t-1)) \quad (2)$$

where f(x) is a sigmoidal function. If the winner is the j-th node, its weight is moved closer to the input pattern according to the delta rule as follows:

$$W_{ji}(t+1) = W_{ji}(t) + \alpha(t)\{I_i(t) - W_{ji}(t)\} \quad (3)$$

where $\alpha(t)$ is a learning rate that monotonically decreases as the time increases to ensure the convergence.

If the neighboring nodes for the winner are also updated simultaneously, the training becomes faster and the network performs a nonlinear mapping from a high dimensional input space to a low dimensional space (1-D or 2-D). This network is also called 'topology conserving map' because the relative distances of input patterns are conserved by the transformation. These results are from Kohonen[14].

2.2 Modified SOFM Architecture

For our purpose, Kohonen's SOFM is modified as follows:

a) In the SOFM, the number of the output nodes is equal to the desired number of partitioned regions. Since our intention is not to partition the input space itself but to realize a piecewise linear approximation of a nonlinear mapping in small regions, it suffices to partition the input space along each coordinate axis and train the subsequent BP network for a desired mapping. For example, if we want to partition the 2 dimensional input space into M × N regions, the normal SOFM requires M × N output nodes. But our modified SOFM needs only M + N output nodes for mapping.

With this approach, the overall network becomes simpler and we can easily construct an efficient network which fits into the dynamic range of each coordinate dimension by adjusting the number of output nodes in each slab (coordinate). This modified SOFM is obtained by replacing the similarity measure used in the original SOFM Eq. (1) by Eq. (4).

$$x_{ij} = exp(-|I_i - W_{ji}|/a) \quad (4)$$

where x_{ij} is the input of the j-th output node in the i-th slab.

b) Since we cannot distinguish between similar inputs at the neural outputs in the SOFM, we have assigned the similarity measure as computed by Eq. (4) as the outputs of the winner and its neighbors.

2.3 BP Network for the Mapping Realization

After partitioning the input space, the remaining task is the realization of a desired mapping. The BP network can do this task. However, since the controller outputs should be continuous, the output neurons are linear with the transfer function $y = ax + b$. But the conventional BP network proposed by Rumelhart et al. which has a sigmoidal output function can suitably be applied to binary function [5,6].

Since we do not know the dynamic range of the control output, beforehand, a linear output nodes will be appropriate. During some simulation, we found that the function $y = x$ is inadequate because the interaction between the hidden nodes and the output nodes. So, we used the function $y = ax + b$.

The overall architecture of the K-B network is as shown in Figure 3.

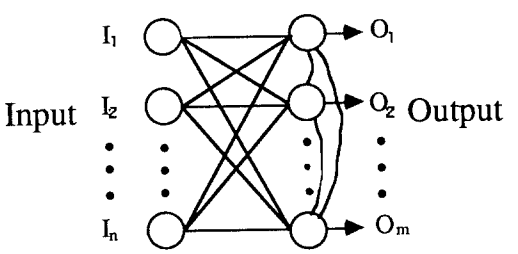


Fig.1 Architecture of the Self-Organizing Feature Map

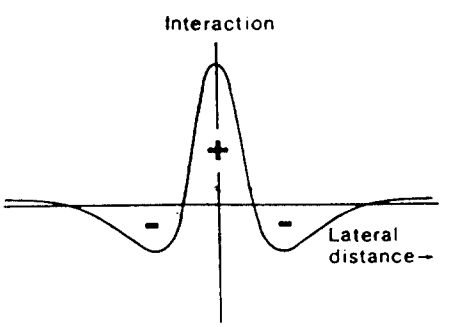


Fig.2 The "Mexican-hat function" of lateral interation

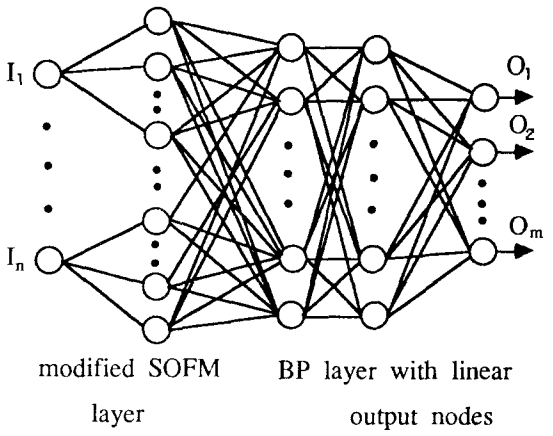


Fig.3 Overall architecture of the K-B network

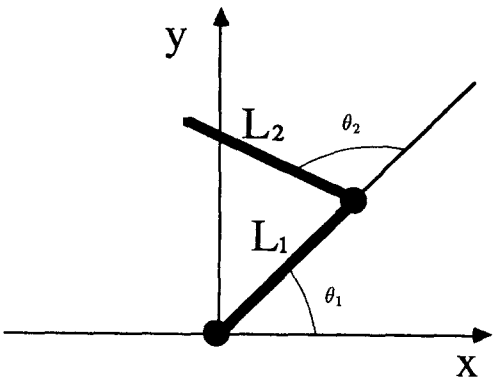


Fig.4 Simulated 2-link robot manipulator (where $L_1 = L_2 = 30$cm)

3 Simulation Results

The proposed K-B network has been applied to the solution of the inverse kinematic problem for a 2-link robot manipulator as shown in Figure 4. The input to the network is the desired Cartesian position (x, y) and the output is the corresponding joint angles (θ_1, θ_2) to reach that position. $L_1 = L_2 = 30cm$, and the arm's workspace is restricted within $0 < \theta_1 < \pi, 0 < \theta_2 < \pi$, and $y > 0$.

The simulation program was written in MicroSoft C on an IBM PC/AT. 40 points along a square within the workspace were used to train the K-B network as shown in Figure 5. After training, the network tried to reach the same set of points but with error shown in Figure 5. Notice that the network has no initial knowledge on the arm's kinematic parameters. The K-B network used has 2 input nodes, 15 nodes for the x-slab, 15 nodes for the y-slab, and 2 output nodes for the BP part. It has 34 nodes in total and the output nodes are linear with $y = 0.1x + 0.5$.

The performance of this network was compared to that of the conventional BP network in Figure 6. The curve labeled (a) corresponds to the K-B net and (b) corresponds to the BP net. Both nets have 2 input nodes, 30 hidden nodes, and 2 output nodes. The curve (c) shows the case where the number of nodes is equal to that of Guez et al. for a similar task[1]. The figure shows that the K-B net outperforms the BP network both in accuracy and convergence.

In order to test the generalization capability, a test set was given outside the training set. Figure 7 shows the outputs of the network for 121 points chosen equally spaced from inside the square. Initally, due to random weights, the arm moved to erratic points (Fig. 7a). Figures 7b, 7c, 7d, 7f and 7e show the results after 10, 20, 40, 100, and 200 iterations. It is almost perfect after 200 iterations. Furthermore, we find that there is little difference in accuracy between the trained points and the test points.

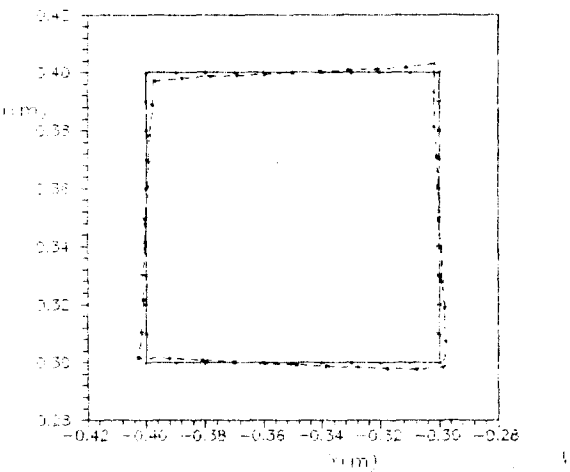


Fig.5 Performances of the K-B net after learning (solid lines:desired points, dashed line:actually reached points)

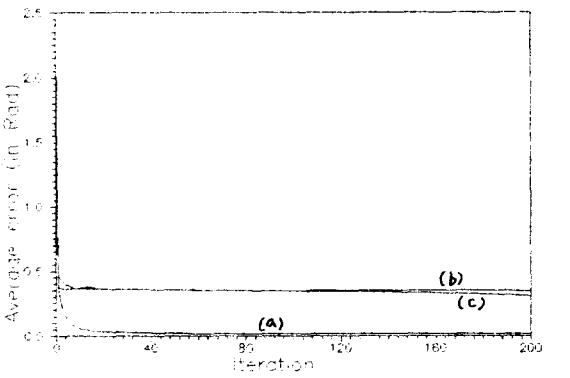
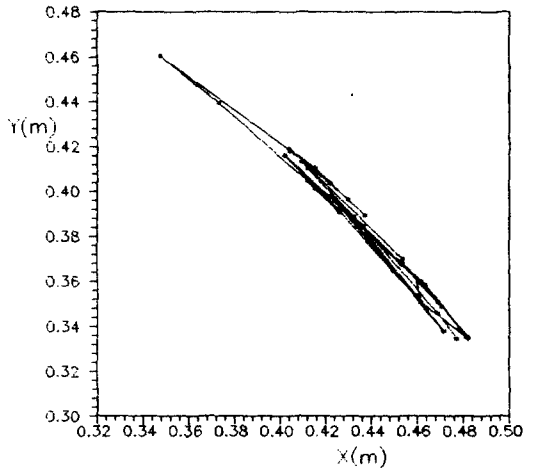
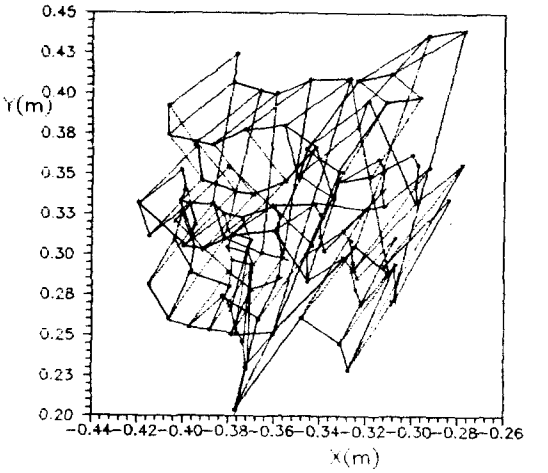


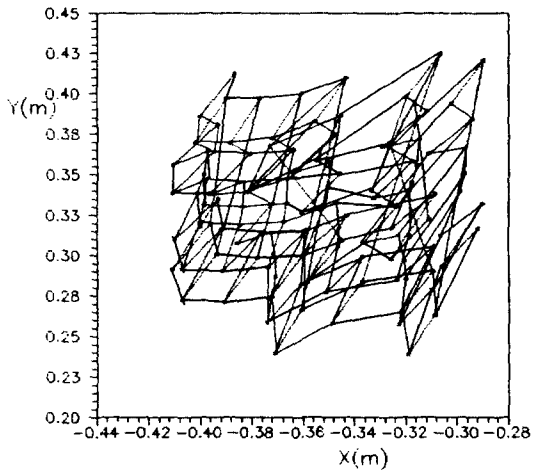
Fig.6 A comparison of the convergent speed
(a) K-B : when the number of nodes in x-slab = the number of nodes in y-slab = 15
(b) BP : when the number of hidden nodes is 30
(c) BP : 1st hidden nodes = 2nd hidden nodes = 10



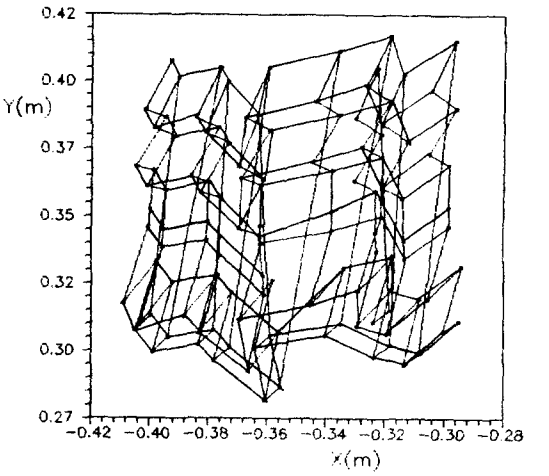
(a) when no learning occured at all



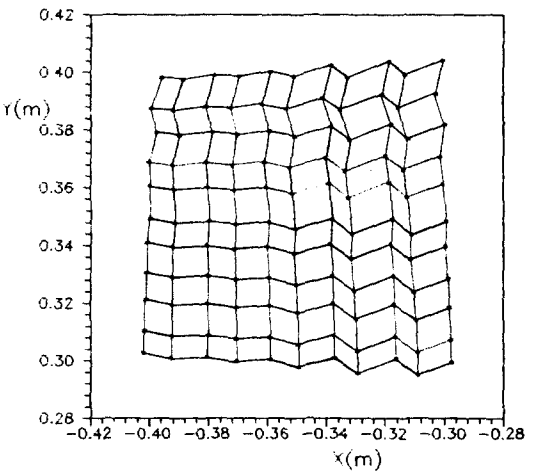
(b) after 10 iteration of learning



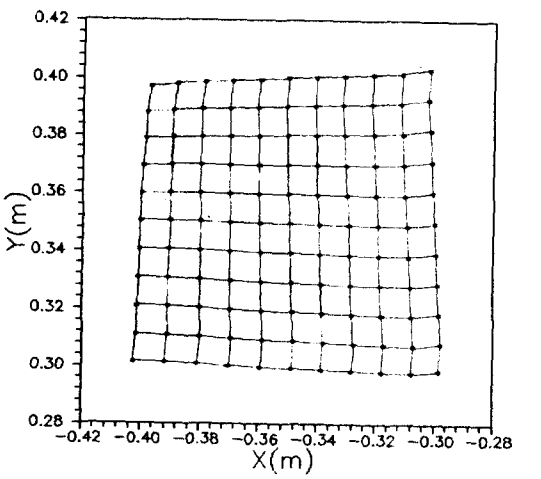
(c) after 20 iteration of learning



(d) after 40 iteration of learning



(e) after 100 iteration of learning



(f) after 200 iteration of learning

Fig.7 Improvement of the performance as learning is proceeded.

Finally, the network has been tested in the entire workspace as shown in Figure 8. Figure shows the performance comparison between the K-B net and the conventional BP net. 130 points were selected to train the neural nets. The number of nodes in the x-slab was 30 and that of the y-slab was 20. The K-B net again outperforms the BP net.

4 Conclusion

In this paper, we proposed a composite neural network controller that combines a modified version of Kohonen's self-organizing feature map and the backpropagation network. This new network allows efficient mapping and learning by confining the role of the BP in a small region. Namely, the BP can work better in these small regions. Simulation results shows that it converges faster with better accuracy than the conventional BP net. In addition, it was found that the BP network portion need not even be multilayered and a single layer network with delta rule learning suffices.

A qualitative explanation of this improvement follows. It is a significant feature of the BP net that it can learn an internal representation of the training inputs. However, Long training time and not necessarily optimal internal representation are two major limitations of a BP net. The K-B net attempts to find a suboptimal representation very quickly by using a modified SOFM.

The functional link net of Klassen et al.[8] uses high order terms as optimal (or suboptimal) internal representations. But, for a complex task, selection of the requred higher order terms is a difficult problem.

As future work, a more efficient algorithm to partition the input pattern space is needed. Furthermore, once the error gets small, convergence slows down like the BP network due to the gradient descent algorithm. A network with faster convergence should be investigated.

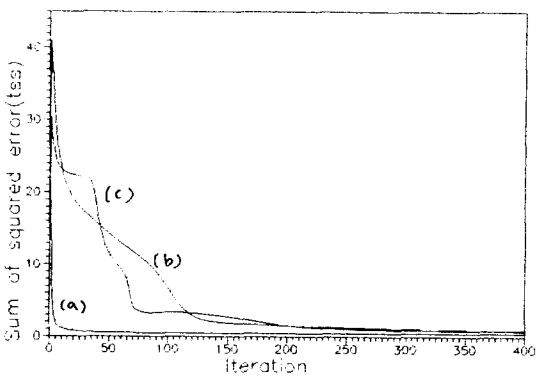


Fig.8 A comparison of the convergent speed (when the training patterns are sampled from the entire work space of the manipulator shown in figure 4)
(a) K-B : number of nodes in x-slab = 30
number of nodes in y-slab = 20
(b) BP : hidden nodes = 50
(c) BP : 1st hidden nodes = 2nd hidden nodes = 10

References

[1] A. Guez and Z. Ahmad, "*Solution to the inverse kinematics problem in Robotics by Neural Networks*", Proceedings of the IEEE 2nd International Conference on Neural Networks, Vol.II, June 1988.

[2] D. Psaltis, A. Sideris, and A. A. Yamamura, "*A Multilayered Neural Network Controller* ", IEEE Control System Magazine, April 1988.

[3] G. Josin, D. Charney, and D. White, "*Robot Control Using Neural Networks* ", Proceedings of the International Neural Network Society 1st Annual Meeting, September 1988.

[4] R. K. Elsley, "*A Learning Architecture for Control Based on Back Propagation Neural Networks* ", Proceedings of the International Neural Network Society 1st Annual Meeting, September 1988.

[5] D. E. Rumelhart, James L. McClelland , and the PDP Research Group, "*Parallel Distributed Processing.* vol.1", MIT Press, 1986.

[6] D. E. Rumelhart and J. L. McClelland, "*Explorations in Parallel Distributed Processing*", MIT Press, 1988.

[7] Ken-Ichi Funahashi, "*On the Approximate Realization of Continuous Mappings by Neural Networks* ", Neural Networks, Vol. 2, 1989.

[8] M. S. Klassen and Y. H. Pao, "*Characteristics of the functional link net: A higher order delta rule net*", Proceedings of the IEEE 2nd International Conference on Neural Networks, Vol.II, June 1988.

[9] D. R. Hush and J. M. Salas, "*Improving the learning rate of back-propagation with the gradient reuse algorithm* ", Proceedings of the IEEE 2nd International Conference on Neural Networks, vol.I, June 1988.

[10] R. A. Jacobs, "*Increased rates of convergence through learning rate adaption* ", Neural Networks, Vol.1, 1988.

[11] T. Ash, "*Dynamic node creation in backpropagation networks* ", ICS Report 8901, University of California, San Diego, February 1989.

[12] M. Hagiwara and M. Nakagawa, "*Supervised learning with artificial selection* ", International Joint Conference on Neural Networks, Poster session, June 1989.

[13] T. Kohonen, "*Self-Organization and Associative Memory*" 2nd Ed., Springer-Verlag, 1988.

[14] T. Kohonen, "*Analysis of a Simple Self-Organizing Process*", Biological Cybernetics, Vol.44, 1982