01 Jan 1992

Planning and Control of a Robotic Manipulator Using Neural Networks

Xavier J. R. Avula
Missouri University of Science and Technology, avula@mst.edu

Heng Ma

Anil Malkani

Jay-Shinn Tsai

et. al. For a complete list of authors, see https://scholarsmine.mst.edu/che_bioeng_facwork/353

Recommended Citation

X. J. Avula et al., "Planning and Control of a Robotic Manipulator Using Neural Networks," *Proceedings of the First IEEE Conference on Control Applications, 1992*, Institute of Electrical and Electronics Engineers (IEEE), Jan 1992.
The definitive version is available at https://doi.org/10.1109/CCA.1992.269858

PLANNING AND CONTROL OF A ROBOTIC MANIPULATOR USING NEURAL NETWORKS

Luis Carlos Rabelo[1] Xavier J. R. Avula[2] Heng Ma[1]
Anil Malkani[3] Jay-Shinn Tsai[1]

[1]Department of Industrial and Systems Engineering, Ohio University, Athens, Ohio 45701
[2]Department of Mechanical and Aerospace Engineering and Eng. Mechanics University of Missouri-Rolla, Rolla, Missouri 65401
[3]Department of Electrical and Computer Engineering, Ohio University Athens, Ohio 45701

This paper presents an architecture which utilizes two artificial neural systems for planning and control of a robotic arm. The first neural network system participates in the trajectory planning and the motion decision making process. The second neural network system provides the correct sequence of control actions with a high accuracy due to the utilization of an unsupervised/supervised neural network scheme. The utilization of a hybrid hierarchical/distributed organization, supervised/unsupervised learning models, and forward modelling yielded an architecture with capabilities of high level functionality.

INTRODUCTION

The work presented here develops an architecture which can yield a strategy for dynamic decision-making that allows the robot end-effector to reach it's goal using apriori and on-line contextual information. The neural network systems cooperate with the entire architecture to achieve the necessary flexibility to adapt to unforeseen changes in the robot workspace and interactions with other systems.

This architecture utilized an emulator to enhance the supervised learning strategy. This idea has been proposed by Werbos [24,25], Jordan [7,8], Jordan and Rumelhart [9], Kawato [10] and Nguyen and Widrow [12,13]. As opposed to inverse modelling, forward modelling emphasizes on the utilization of a robot model, and after using that model to train the controller. With this mechanism, forward modelling overcomes the many-to-one mapping from actions to sensations problems and provides an effective sensitivity analysis capability [9]. In addition, forward modelling could be easily modified using the emulator/controller structure as a backbone for more complex control structures [21].

SYSTEM MODEL

The computer simulated robotic manipulator model consists of a two dimension version of an arm with two jointed links of equal length (Fig.1). The workspace is constrained by the combined lengths of the two members (200 cm) in the simulation and by a maximum rotational displacement limit of 170 degrees at the elbow joint. The robotic manipulator's dimensions can also change due to the effects of temperature induced link expansion or contraction (e.g., deformations up to 15% with an increment of temperature of 100°C are possible).

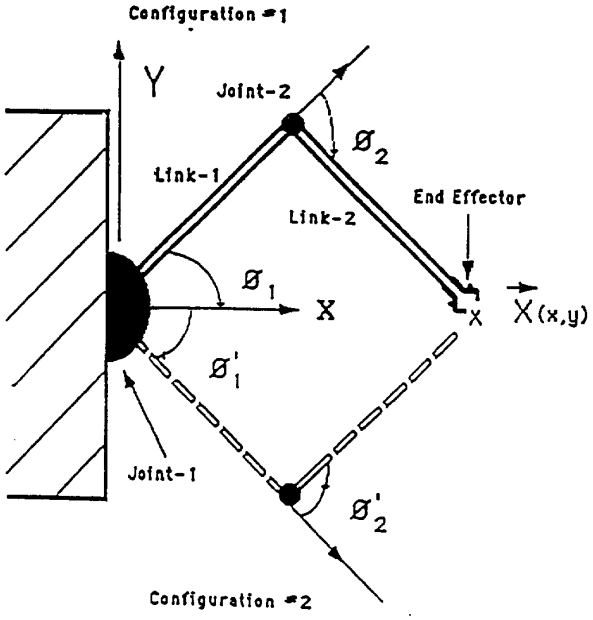


Figure 1. Arm Configurations

NEURAL NETWORKS

Two different neural network systems are presented associated with the prototype of a scheme which uses the integration of neural networks and knowledge-based systems for robotics motion control. These neural network systems participate in the tasks of the motion analysis process at the higher hierarchical level and the process of control-emulation at the lower level.

The neural network system at the higher hierarchical level supports the decision making by providing the motion analysis process with an initial hypothesis. This initiative does not preclude adaptive changes during the course of motion due to unexpected changes. The simplicity of the model utilized permits decomposition of the initial stage into two distinct tasks, each with its own associated neural network arrangement. The first part, consisting of the preliminary motion feasibility analysis, uses a restrictive coulomb energy

(RCE) network to delineate the robot-arm workspace and evaluate whether the end-effector could reach the proposed location. If the goal is feasible, a second system of neural networks is employed to analyzes and select a configuration in accordance with the initial position. This second network makes an initial proposition to the motion decision making mechanism which could use the cooperation of other knowledge sources (e.g., knowledge bases, algorithms, procedures) with more contextual information leading to the final decision.

The process of control-emulation is implemented at the lower part of the hierarchy. The problem to be solved is to provide the correct sequence of control actions to incite the robot arm to go from an initial position to a target position. This "correct sequence" of control actions is decided by a plan generated by the high-level planner which manages and coordinates information concerning the task to be performed, and updates the system knowledge. The forward model is :aught by the high-level planner using simulated sensor 'eedback and implemented in a backpropagation network. Then, the emulator network of the robot arm dynamics is leveloped and the controller implementation is started. The :ontroller is implemented using a backpropagation network which is driven by the high-level planner which takes the lecision on the kind of trajectory to be followed (i.e., linear, :ircular, linear/circular). The frequency of these trajectory :hanges is totally handled by the high-level planner according o sensory information, goals, and optimization factors. In ıddition, the controller receives input from the emulator at a ıigher frequency.

RCE Mapping of a Two Dimensional Robotic Arm Workspace

In the present study, an RCE network was used to accomplish this workspace filter. In an RCE network [20], all examples of a pattern category (e.g., IN, OUT) define a set of points in the feature space that can be characterized as a region (or a set of regions) having some arbitrary shape. This feature of RCE networks make them appropriate to define and learn complex workspaces (i.e., several degrees of freedom and links of arbitrary geometric shapes). In addition, RCE networks are appropriate for real-time learning of complicated non-linear class regions, and capable of probability estimations to handle uncertainty (Fig.2).

The RCE network used in this study was of the most liberal type since it was desired to prompt it to decide the categorization of the X-Y pair (it is also possible to use polar coordinates if required) even at relatively high uncertainty levels due to the fact that a response was needed. The nearest neighbor approximation was used whenever an input vector fell outside the influence region of any hidden layer cell. A training data file consisting of 4500 points and a testing file of 499 points were utilized. The final network had 426 units.

This approach has been utilized for more complex robotic configuration . Figures 3 and 4 show a robotic arm of 4 degrees of freedom with a 3D workspace. An RCE network was capable to describe it with a high accuracy after being trained with 10000 points with a reasonable computational cost. However, the architecture consisted of 1141 hidden units!.

Mapping with Backpropagation

The neural networks for configuration selection, arm emulator, and neurocontroller were developed using the

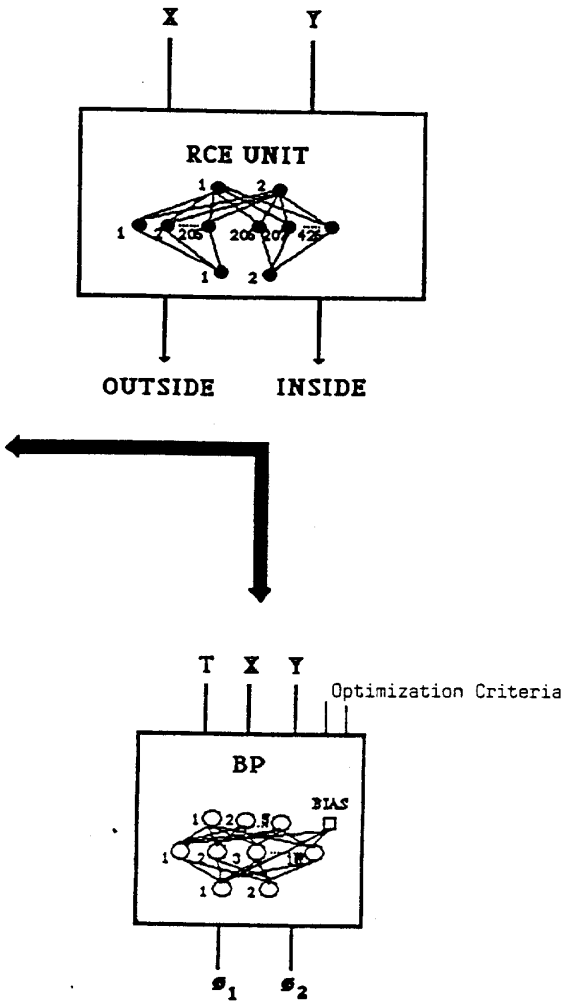


Figure 2. Artificial Neural System I

standard backpropagation paradigm. The training process using backpropagation is a difficult problem [1,6,11,23]. It is needed to find an appropriate architecture (e.g. number of hidden units, number of hidden layers, etc.), adequate size and quality of training data, satisfactory initialization (e.g., initial weights), learning parameter values (e.g., learning rates), and to avoid over-training effects (performance degradation due to prolonged training).

In this research the following approaches were utilized:

1. To help to find an appropriate architecture (i.e., number of hidden units) an interactive addition of nodes was performed as proposed by the Dynamic Node Creation (DNC) [1].

2. To speed up the convergence behavior, the selection of parameters such as learning rates and the utilization of first and second order momentum factors are emphasized [15]. The learning rule utilized consisted of a weight update using momentum (ßx) factors with the exception that each weight had its own "adaptive" learning rate parameter

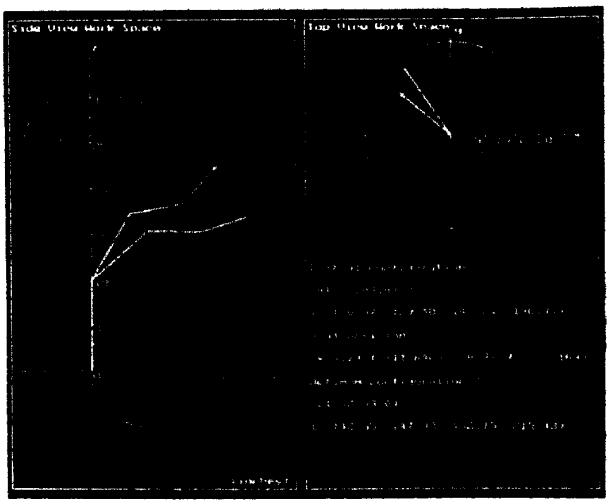


Figure 3. Initial and Final Configurations of a Robotic Arm with Four Degrees of Freedom

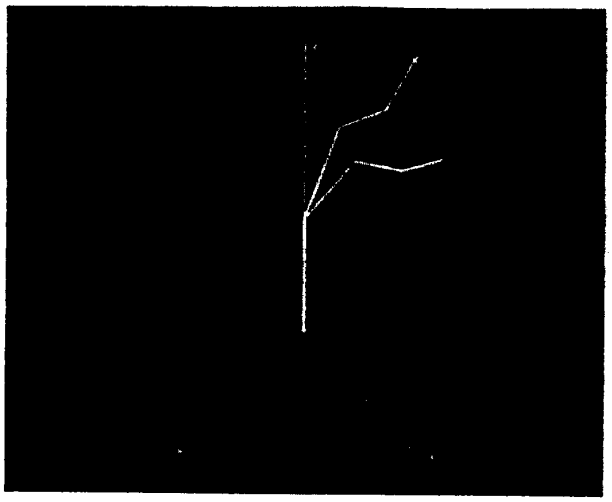


Figure 4. 3D View of a Robotic Arm with Four Degrees of Freedom

(μ) [6]. The "adaptive" learning rate strategy increments the $\mu(s)$ by a small constant if the current partial derivative of the objective function ($E = 1/2\Sigma(\text{Target} - \text{Output})^2$) with respect to the weight (w) and the exponential average of the previous derivatives have the same sign, otherwise μ will be decremented by a proportion to its value. The updating equation of the weights is defined by using w_{ij} as the weight value located between nodes i and j, t is the present iteration, Δw is the weight increment which is equal to the product of the μ and the partial derivative of the objective function with respect to the weight ($\partial E/\partial w_{ij}$),

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij}(t) + \beta 1\, \Delta w_{ij}(t-1) + \beta 2\, \Delta w_{ij}(t-2).$$

3. To reinforce learning, Combined Subset Training (CST) was utilized [22]. CST combines old and new training sets. First, a random subset to train the network is selected. When the network has learned it fairly well, a new subset (of the same size as the previous one) is added to the first training set, and the network is trained with the combined set.

4. In this research, three different types of data sets were utilized: a training set, a validation set, and a testing set. The training set was utilized for determining the values of the weights. The validation set (a set with unseen data to make on-line tests about network performance) was utilized to avoid over-training effects [11,23]. The testing set (as expressed by Weigend et al. [23] "It is strictly set apart and never used in training.") was utilized to estimate the expected performance of the network.

Robot Arm Coordinates Transformation and Configuration Selection

The coordinates transformation for the robot used in this research is not a one-to-one mapping. Two configurations, with a positive and a negative elbow rotation, are valid if no arbitrary constraints are placed on the solution space. Consequently, it is important to develop a methodology for choosing one which satisfies the motion and contextual constraints. The system used to produce this transformation is based on a neural network (Fig. 2). The mapping assigns a three dimensional input vector consisting on the X and Y location of the end-effector as well as the temperature, T, (X, Y, T) to a two dimensional vector containing the required shoulder and elbow angular position ($q1, q2$).

The final architecture for the neural network had 18 hidden units. When the final architecture of 18 hidden units was achieved CST was applied up to 1600 data samples. It is interesting to note, that the next set of data to be added to the training file was based on the minimum output errors achieved before. Therefore, each data set added as defined by CST concentrated in those points where the network has had some lower performance.

Several neural networks have been developed for the robotic configuration depicted in Figures 3 and 4 for different optimization criteria (e.g., minimization of total displacement, minimization of the maximum displacement of the angles). The neural networks have been able to produce reasonable solutions. However, the number of training samples required and the final error (e.g., around 1^o) required the utilization of other optimization procedures as a second stage (e.g,., modified Newton Raphson, Genetic Algorithms). Table 1 shows the precision of a genetic algorithm [14] using "real" strings, a population of 16, and a reduced number of generations (around 20) as a second stage of a neural networks for the optimized inverted kinematics problem. It is possible to see the high accuracy of 0.0008 inches achieved in a reduced computational time !.

Arm Emulator

The arm emulator is needed in order to identify the arm dynamic behavior. The procedures, in order to develop the emulator, should be encoded in the high-level planner. The high-level planner could examine the arm responses in the cartesian plane with different motor actions. This process will be implemented repetitively until an efficient emulator is developed.

To develop a high accuraccy emulator, several neural networks are developed using the techniques previously

Table 1. Genetic Algorithm Trial

Training Table (Press ESC to DOS in processing !)						
Theta 2	Theta 3	Theta 4	Error	Displacement	Fitness	Rank
139.99	166.73	117.26	0.0007	40.93	0.1867	1
139.96	166.64	117.29	0.0261	40.90	0.2061	4
136.12	166.49	117.38	0.0081	41.06	0.2201	8
136.06	166.63	117.49	0.0192	41.00	0.2192	7
136.11	166.73	117.39	0.0446	41.05	0.2546	13
135.83	166.93	117.14	0.0233	41.13	0.2492	12
136.17	166.67	117.33	0.0493	41.11	0.2705	15
135.96	166.49	117.38	0.0430	40.90	0.2230	10
139.95	166.73	117.16	0.0203	40.93	0.2063	6
136.08	166.73	117.31	0.0064	40.94	0.1944	2
136.13	166.56	117.31	0.0180	41.07	0.2320	11
136.20	166.56	117.38	0.0422	41.14	0.2708	14
139.99	166.67	117.29	0.0090	40.93	0.1930	3
139.96	166.73	117.13	0.0281	40.93	0.2061	5
136.05	166.49	117.01	0.0233	40.99	0.2203	9
136.17	166.70	117.43	0.0603	41.11	0.2822	16

mentioned. The utilization of several neural networks for different points [5] increased the accuracy of previous developments [16,17,18,19]. These neural networks have 5 inputs (Fig. 5) identifying the current X, Y cartesian positions, the elbow and shoulder angle increments (which could define the width of the motors drive pulses in our discrete-time model), and the temperature. The neural networks have two outputs which correspond to the X, Y positions (i.e., next state) after the movement has been performed. To cluster the space in several groups, a Fuzzy ART [4] (unsupervised) neural network was utilized. These clusters gave us the division of the workspace that was mapped on to backpropagation networks. The division of the workspace was done using vigilance factor in the interval [0.6,0.8] (Fig. 6).

The Neurocontroller

The neurocontroller has the functions to provide a sequence of orders in order to drive the arm from an initial position to a target position. The trajectory could be defined as linear or circular. The high-level planner decides according to sensory feedback or emulator responses to modify the plans given to the neurocontroller. To define the type of trajectory the coefficients of a line ($Bx + C$) or circle ($Ax^2 + Bx + C$) are provided to the neurocontroller. The neurocontroller taking into consideration the present state vector (X, Y, T) and the plan (A, B, C) generates the discrete increments for the elbow and shoulder angles. Then the "new" present state vector is used repeatedly till the targeted position is achieved (Fig.5).

Several neural networks are developed using backpropagation and the techiques mentioned above to implement the neurocontroller. The Fuzzy ART network utilized for the emulator is utilized to select the respective backpropagation neural network. These neural networks have 6 inputs corresponding to X, Y, T, A, B, and C. The outputs correspond to the increments of the shoulder and elbow angles. The neurocontroller networks showed a better accuracy and a more continuous path definition than previous developments [16,17,18,19].

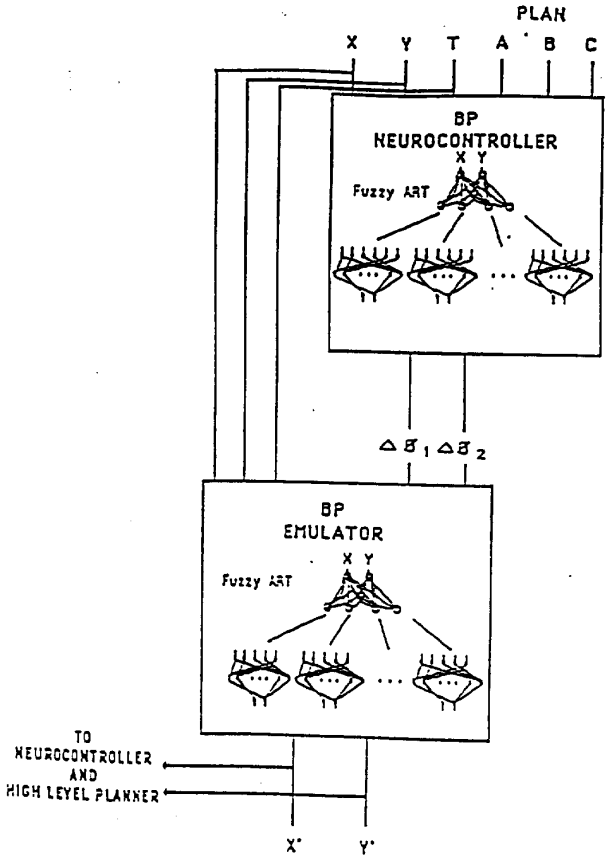
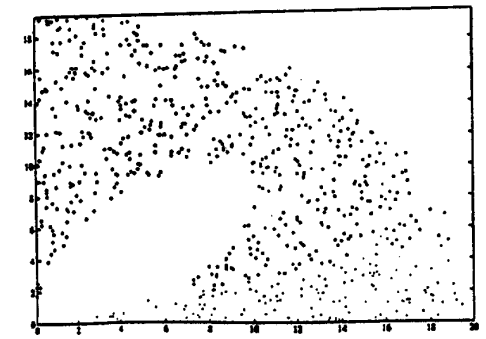


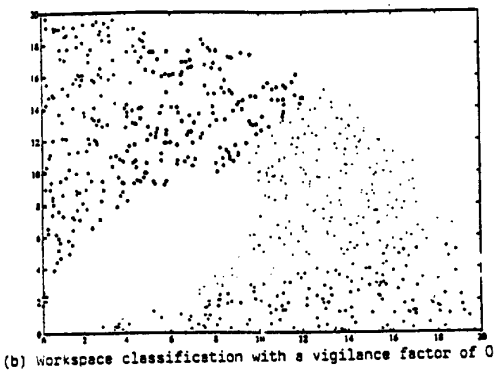
Figure 5. Artificial Neural System II

ON-GOING RESEARCH

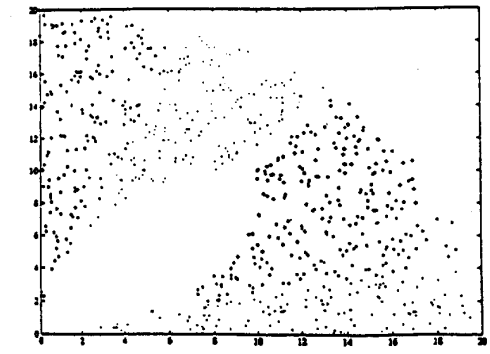
Neural networks have capabilities to learn, to perform massively parallel processing, and to adapt to complex environmental changes. The hierarchical structure introduced here significantly enhances the system capabilities because it takes into consideration not only the knowledge about the manipulator but also about the controller. On-going developments are concentrating using arm models with more than two degrees of freedom and three dimensional work envelopes.



(a) Workspace classification with a vigilance factor of 0.6



(b) Workspace classification with a vigilance factor of 0.7



(c) Workspace classification with a vigilance factor of 0.8

Figure 6. Workspace Category Classification Using Fuzzy ART

REFERENCES

[1] T. Ash, "Dynamic Node Creation," Technical Report, ICS 8901, University of California-San Diego,1989.

[2] A. Barto, R. Sutton, C. Anderson, "Neuronlike Adaptive Elements that Can Solve Difficult Learning Control Problems," IEEE Transactions on Systems, Man, and Cybernetics, SMC-13, pp 834-846.

[3] Y. Chauvin, "Dynamic Behavior of Constrained Back-Propagation Networks," Advances in Neural Information Processing Systems 2, edited by David Touretzky, Morgan Kaufmann Publishers, 1990, pp 642.

[4] G. Carpenter , S Grossberg and D. Rosen (1991), "FUZZY ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system" , CAS/CNS-TR-91-015.

[5] D. DeMers, K. Kreutz-Delgado, "Learning Global Direct Inverse Kinematics," Poster Presentation 1991 Conference on Neural Information Processing Systems - Natural and Synthetic, December 2-5, 1991.

[6] R. Jacobs, "Increased Rates of Convergence Through Learning Rate Adaptation," Neural Networks, Vol.1 No.3, 1988, pp.295-307.

[7] M. Jordan, "Supervised Learning and Systems with Excess Degrees of Freedom," Proceedings of the 1988 Summer School on Connectionist Models ed. Touretzy, pub. Morgan Kaufman, pp 62-75.

[8] M. Jordan, "Generic Constraints on Underspecified Target Trajectories," IJCNN Conference Proceedings, Vol. 1, June 1989, pp I217-225.

[9] M. Jordan, D. Rumelhart, "Forward Models: Supervised Learning with a Distal Teacher," Occasional Paper # 40, 1991, Center for Cognitive Science, Massachusetts Institute of Technology.

[10] M. Kawato, "Computational Schemes and Neural Network Models for Formation and Control of Multijoint Arm Trajectory," NEURAL NETWORKS FOR CONTROL, T. Miller, R. Sutton, and P. Werbos (Eds.), MIT Press, 1990, pp 197-228.

[11] N. Morgan, H. Bourlard, "Generalization and Parameter Estimation in Feedforward Nets: Some Experiments," International Computer Science Institute, TR-89-017, 1989.

[12] D. Nguyen, B. Widrow, "The Truck Backer-Upper: An Example of Self-Learning in Neural Networks," Proceedings of IJCNN, Washington D.C., 1989, pp II357-II363.

[13] D. Nguyen, B. Widrow, "Neural Networks for Self-Learning Control Systems," IEEE Control Systems Magazine, Vol. 10 No. 3, 1990, pp 18 -23.

[14] J. Parker, D. Goldberg, A. Khoogar, "Inverse Kinematics of Redundant Robots using Genetic Algorithms," Proceedings of the 1989 IEEE International Conference on Robotics and Automation, 1989, pp 271-276.

[15] B. Pearlmutter, "Asymptotic Convergence of Gradient Descent with Second Order Momentum," Poster Presentation 1991 Conference on Neural Information Processing Systems - Natural and Synthetic, December 2-5, 1991.

[16] L. Rabelo, X. Avula, " Adaptive Control of a Manipulator Using Artificial Neural Networks," International Journal of Production Research, Taylor and Francis, February 1992.

[17] L. Rabelo, X. Avula, " Hierarchical Neurocontroller Architecture for Robotic Manipulation ," IEEE Control Systems Magazine, April 92, Special issue of Neural Networks in Control.

[18] L. Rabelo, X. Avula, " Intelligent Control of a Robotic Arm Using Hierarchical Neural Network Systems," Proceedings of the International Joint Conference on Neural Networks, Seattle, 1991, pp II747-II751.

[19] L. Rabelo, X. Avula, " Hierarchical Neurocontroller Architecture for Intelligent Robotic Manipulation ," Proceedings of the IEEE Conference on Robotics and Automation, Sacramento, California, 1991
[20] D. Reilly, L. Cooper, C. Elbaum, "A Neural Model for Category Learning," Biological Cybernetics, Vol. 45, 1982, pp 35-41.
[21] S. Thrun, K. Moeller, "Active Exploration in Dynamic Environments," Oral presentation 1991 Conference on Neural Information Processing Systems - Natural and Synthetic, December 2-5, 1991.
[22] F. Tsung, G. Cottrell, "Sequential Adder Using Recurrent Networks," IJCNN Conference Proceedings, Vol.2, June 1989, pp II133-II139.
[23] A. Weigend, A., D. Rumelhart, B. Huberman, "Back-Propagation, Weight Elimination and Time Series Prediction," CONNECTIONIST MODELS: Proceedings of the 1990 Summer School, D. Touretzky, J. Elman, T. Sejnowski, and G. Hinton (Eds.), Morgan Kaufmann Publishers, 1990, pp 105-116.
[24] P. Werbos, "Building and Understanding Adaptive Systems: A Statistical/Numerical Approach to Factory Automation and Brain Research," IEEE Transactions on Systems, Man, and Cybernetics, 1987, 17, pp 7-20.
[25] P. Werbos, "Consistency of HDP Applied to a Simple Reinforcement Learning Problem, " Neural Networks, vol. 3, No. 2, 1990, pp 179-190.