Connectionist Learning Control at GTE Laboratories

Judy A. Franklin, Richard S. Sutton, Charles W. Anderson
Oliver G. Selfridge, Daniel B. Schwartz

GTE Laboratories Incorporated
Waltham, MA 02254

Abstract

At GTE Laboratories, we are advancing the theory of connectionist learning architectures for real-time control while exploring their relationships to animal learning models, applications in manufacturing quality control, and VLSI implementations. We seek connectionist-network architectures with improved convergence rate and scaling properties, as assessed on simulated and actual control problems.

Our primary focus is on extensions to *reinforcement learning*. These include adaptive critics, feature/representation adaptation in multilayer networks, hybrid connectionist/conventional controllers, and modular networks for hierarchical control. We are also extending methods for system identification, or *model learning*, to include internal models learned using temporal-differences. We propose the integration of reinforcement and model learning based on their relationships to dynamic programming. We are working to resolve how connectionist systems should serve as a total systems concept or as tools in a larger architecture.

1 Introduction

Adaptive and robust control techniques have made great progress over the last two decades. Model reference adaptive control techniques have been developed that are globally stable for the control of certain kinds of systems in the presence of noise. Adaptive control can be successfully applied to identify and control systems of certain structures, when the structure is known a priori. For general nonlinear systems, or systems whose structure is unknown or complex, other methods based on learning or artificial intelligence must be incorporated into the design of the controller. Most adaptive control methods require a continuous target or reference signal, whereas many tasks involve goals that are not conveniently expressed as reference signals. Adaptive control methods also have difficulty using sensory information whose relationship to system state is not known a priori. Finally, no general adaptive control method exists for problems with unknown delays between control actions and resultant feedbacks.

By a *learning controller* we mean a controller that adapts its input-output behavior on-line and in real time, based on its experience. Adaptive controllers also fit this definition, but, as that term is currently used, they are restricted to particular structural forms. We are studying learning controllers composed of connectionist or neural networks [14, 23, 43, 45]. Our methodology emphasizes empirical comparison of alternative learning control architectures on specific simulated and actual control problems (e.g., see [6]).

Connectionist networks are a promising technology for real-time adaptive and learning controllers because of their capabilities for on-line learning with minimal external intervention, for real-time response, for utilizing delayed feedback, for learning internal models of the external world, for robustness and noise-tolerance, and for learning nonlinear mappings. Connectionist networks are also promising because of their potential for parallel implementation, and because of their similarities with biological information processing systems.

The authors would like to acknowledge the contributions of this research of Andrew Barto, Ronald Williams, Vijay Gullapalli, Vijay Samalam, and John Vittal.

2 Overview of Our Approach

The most direct connectionist-network approach is that of the *trainable controller* [25, 58]. Here, the control law is implemented by a connectionist network trained using *supervised learning* techniques, using examples of desired input-output behavior provided by a *teacher*, typically a human expert or an existing controller. After training, the connectionist network may respond faster or be cheaper to operate than the teacher, but it cannot learn a better control law than that implemented by the teacher. Thus, trainable controllers cannot be used on control problems for which we do not already know the solution.

At GTE, we are extending another approach to connectionist learning control, known as *reinforcement learning* [7, 8, 9, 10, 11, 17, 24, 29, 33, 36, 48, 60, 61]. In reinforcement learning, the control law is learned by trial and error, that is, by trying a variety of actions and correlating them with subsequent performance feedback, called *reinforcement*. The key characteristic of reinforcement feedback is that it only evaluates the actions that were taken; it does not indicate which action was correct, as the teacher does for a trainable controller.

Reinforcement feedback is the minimum that can be required by way of external supervision. If a controller is to improve through learning, it must at least be able to detect variations in the quality of its control. In practice, reinforcement feedback can usually be obtained through simple measurements, and a knowledgable teacher is not required. For this reason, reinforcement learning can proceed on-line and can find new, better ways of behaving that were not anticipated or previously known. We are using this ability to refine conventional controllers that are designed from an inexact model of the system (Section 5).

Reinforcement learning has been shown capable of learning the correct control law when delays exist between actions and their consequences. *Temporal-difference* and *adaptive heuristic critic* methods have been developed [9, 48, 52, 56, 57] that enable reinforcement learning to accommodate substantial delays between actions and reinforcement. Temporal-difference methods can address notions of long-term goals and subgoals, and of using an evaluation function such as a Lyapunov function to learn control actions.

The second major approach to learning control that we are extending at GTE Laboratories is using connectionist networks for *model learning*. By model learning we mean any approach to learning control in which the controller does not directly learn a control law, but rather learns an *internal model* of the system, from which the control law is determined [27, 28, 30, 35, 39, 41, 59]. A learned system model permits more flexible behavior than a single learned control law because a model can contain more knowledge about the system. While a control law merely specifies what to do in each situation, a system model can specify how the system behaves in a range of possible situations. Another advantage of the model-learning approach is that it is related to conventional adaptive control methods, which may enable better understanding and transfer of theory.

On the other hand, model-learning approaches have some of the same problems as conventional control methods. First, there is the problem of determining a control law once the model is known. For nonlinear, non-invertible models such as those typically formed by multilayer connectionist networks, this remains problematic. Although gradient methods exist for such models [27, 38, 41], no way is known to accommodate delays between actions and corresponding feedbacks. Furthermore, to learn a good model, the system must be *explored*—its behavior must be observed under a wide variety of situations in response to a wide variety of actions. How is this exploration to be ensured on a continuing basis? And how is the need for exploration to be balanced against the need for good control, which by definition implies a reduction in the variety of actions? This problem has so far prevented model-learning approaches from meeting their expectations in terms of rapid learning. Third, model-learning approaches, like conventional adaptive control approaches, have generally required the environment to provide detailed reference signals specifying desired system outputs.

In our view, model learning and reinforcement learning are not so much alternative approaches as they are different, complementary parts of a solution to the overall problem of learning control. We are researching ways to extend both reinforcement-learning and model-learning approaches, and, moreover, to integrate them into an architecture that combines their strengths. This and the other major components of our approach—as listed in the abstract—are discussed in the remaining sections of the paper.

3 Internal Models based on Temporal-Difference Learning

The classical approach to modeling has been to describe a system by a set of differential or difference equations in terms of its input, output, and internal state. This approach is limited to capturing temporal relationships acting at a single time scale.

Anytime we choose actions to influence our long term future, we must make long term predictions. In principle, these can be formed by combining many smaller short term predictions. However, that approach is very brittle: if one step in the chain is missing, the long-term prediction cannot be made. It also is computationally intensive, because it requires a detailed and accurate level of modeling. Temporal-difference methods provide a different approach to learning models of dynamic systems. Rather than relying entirely on their local structure—the relationship between inputs, actions and state at an instant in time—as in conventional models, one instead allows longer term relationships to be stated and combined. Sutton [52] has argued that for all predictions other than immediate, next step predictions, temporal-difference methods have substantial advantages in terms of implementation and speed of learning.

Barto, Sutton, and Anderson [9] used an adaptive heuristic critic (AHC) that was an early version of the temporal difference system. The AHC learned a model of the *reinforcement process* that contained delays (see Section 7 for more discussion). The AHC algorithm as described by Sutton [48] is based on the class of linear models, and the simulation of Anderson [5], which uses the back propagation method, is based on nonlinear layered networks.

We are extending the current class of temporal-difference methods into a full model, enabling the prediction of not just reinforcement, but of all the observable states of a system. These will be long term predictions combined using a recurrent network architecture. Sutton, Barto, and Pinette [47, 49] proposed the basic architecture that we are extending. The major extension will be to combine multiple time scales. For example, to predict events on a manufacturing line one clearly has to model at a variety of time scales—months, weeks, days, hours, minutes, and even seconds.

4 Learning Internal Data Representations

One of the most important design aspects of a connectionist network architecture is the preprocessing: the set of features used to internally *represent* data from the network's environment. The representation used for a particular problem can make the learning task faced by the network extremely simple or prohibitively complex. In real modeling and control problems, there is insufficient information a priori to determine which is the "right" representation. Therefore, it is essential that a connectionist network be able to enhance its internal data representation with experience, tailoring it to specific tasks. This enhancement is important in reinforcement learning and model learning as well as in the passing of data between levels of a hierarchical controller.

A network's internal data representation is defined by the set of features encoded by the network's units. The most common method today for learning new features in layered networks is the error back-propagation algorithm [40], which performs a gradient-descent search of an error function with respect to the weights of a network. Gradient descent is known to scale poorly with the complexity and size of the function being searched, due to the presence of local minima and regions of small gradient. In fact, for the learning of representations, gradient descent can be very inefficient [50].

An alternative to gradient-based search is the *generate-and-test* approach to searching for useful features (see [3, 4]). In this approach, new features are generated and tested by adding them to a network while the network continues to learn. Gradient-descent and generate-and-test methods are complementary: when close to an optimum, gradient descent can be quite efficient, while generate-and-test has advantages when far from an optimum or on a hilly surface.

Novel algorithms that are a combination of gradient-descent and generate-and-test methods are being developed at GTE Labs. Several critical issues must be addressed in this work. Measures must be found that indicate the usefulness of units in a network. New features should only be generated in unused units. Algorithms that shift emphasis continuously between gradient-descent and generate-and-test should result in adaptations inversely proportional to the usefulness of a unit. Another difficulty is that unused units can remain unused for a long time if they are adapting according to a gradient technique. Techniques for quickly changing the weights of unused units are needed that drive the weights towards values representing potentially useful features. Another question being addressed concerns the possible output functions employed by the units. Some functions are more amenable than others to the generate-and-test methods.

5 Hybrid Connectionist/Conventional Controllers

The design of a conventional controller is very difficult if the system is complex or possesses unknown nonlinearities. In these cases, implementation engineers will generally follow an iterative design strategy. A controller is designed and implemented, and its performance is evaluated by the engineer. Performance is usually better than that of the uncontrolled system, but still not satisfactory in all cases. Rather than redo the whole design, a new controller is often developed for the cases where the first controller is unsatisfactory. The output of the new controller is then, for example, added to that of the original controller, refining and improving its control. This overall process of changing and improving the original controller may be repeated many times.

We have explored a way to automate this iterative implementation strategy using hybrid connectionist/conventional controllers. The first controller is a conventional one, designed by pole placement methods. The refining controller is a reinforcement-learning connectionist network. This approach is very different from current adaptive and self-tuning PID controllers, in which the adaptation is parametric in the system model or in the controller gains. We have used this hybrid approach to refine the control of a second order system. The reinforcement-learning network successfully learned a control signal that compensated for the effect of an unknown additive nonlinearity [18, 19, 20, 22]. Another important feature of the work in [21, 22] is the use of a real-valued output reinforcement learning algorithm developed by Gullapalli [26]. This algorithm enables a very general control signal to be learned that can vary widely over state space.

The issue of internal or input space representation discussed in Section 4 is an important one for refinement learning control. In earlier work [18, 19, 21], the input space was quantized into control situations [32] or boxes [34] for the learning controller. A new input space representation for this problem has been studied empirically by Franklin [22]. It consists of functions that act as *receptive fields* [37] and have the shape of multivariate gaussian probability density functions. They are adapted by adjusting their positions and

by widening or narrowing their widths. In our implementations these functions were the first layer in the learning network.

We continue to be concerned with refining controlled processes. Refinement techniques are especially important for improving already existing controllers that cannot be completely redesigned. These techniques will also be useful in learning to control systems that must be under some sort of initial control. In these cases, it may be either dangerous for the system to be subjected to arbitrary control signals or it may be impossible to learn to control it without a base controller. We are also interested in exploring refinement at a more general level, for example to improve the choice of subgoals of a system given an overall goal and a set of initial subgoals.

6 Modular Networks for Hierarchical Control

The solution of complex control problems must involve multiple levels of control (e.g., see [1, 2]). We can make the analogy of a corporation. Each officer or employee may be thought of as expressing a purpose. All the purposes are subpurposes of the over-riding corporate purposes, one of which, ostensibly the over-riding one, is to make a profit. The fact that subpurposes may conflict with each other is a desirable feature of this situation. In machine learning, it may be imagined that every control loop must be run with respect to the overall goal of the system. But clearly, the particular tactics of a janitor in cleaning the floor cannot be guided by the current price of a share of stock. That argument applies especially to systems where the preprocessing for the connectionist learning network must itself be subject to continual tuning. Not only will a realistic system have a hierarchy of purposes or goals for its separate levels, but it will need to respond to several different and changing purposes at each level. A driver must not only keep to the right, but also avoid pedestrians, move to the left to pass stationary vehicles, stop at traffic lights and so on. Connectionist systems may be able to balance purposes in a natural way. One difficulty in adapting a hierarchical structure of control modules is in deciding at which level a particular behavior should be modified.

We have explored the use of adaptive strategies for playing simple two-person zero-sum games [46]. The adaptation is different from that usually considered in adaptive control; it is a form of test and gradient descent. One of the more general conclusions that can be drawn from this work is that adaptive learning from experience seems to be far richer a topic and more complicated a process than would be imagined. There are many lessons here for studies in learning from experience in more complicated domains. The general adaptive element here includes 1) an action cycle, which exercises some control; 2) a testing cycle, which informs the strategy, through 3) its evaluation function, which way and how much to alter the parameter of its strategy. Such an element may itself be controlled with adaptive loops, and these hierarchical systems can exhibit surprisingly subtle and powerful behavior. In this work [46], Selfridge explores the use of such adaptive procedures on variables that can attain some range of real values. In these experiments, it turns out that convergence to minimax is the exception, and behavior in most cases tends to some kind of a joint limit cycle. The result is that such competitive systems usually do not converge to any kind of stability. An interesting application of this is to distributed or hierarchical control systems where different control units may be considered to be competing for some resource.

We are exploring ways in which reinforcement learning might be used for modifying connectionist network modules interconnected in a hierarchical structure. We will begin by using the known and tested capabilities of connectionist networks as units or modules that control other similar modules. We are also investigating ways of tuning individual controllers by connectionist networks as a continuing process, either in a training mode, or by reinforcement learning. We expect that good performance on simpler subtasks

will facilitate performance on harder ones [44]. One possible technique for the development of a learning hierarchy is to study the presentation of tasks in an increasing order of difficulty. It must be realized that the journey towards hierarchical control has hardly begun, and that the road is likely to be a rocky one. For example, there is no reason to believe that the control variables that provide the rewards at one level are suitable for another. Indeed, the entire structuring of rewards and changes must be fashioned to satisfy a number of differing goals.

7 Integrating Model Learning and Reinforcement Learning through Dynamic Programming

In our view, model learning and reinforcement learning are not so much alternative approaches as they are different, complementary parts of the overall problem of learning control. Reinforcement learning is good at quickly learning simple control laws, at handling delayed feedback, and at learning without teachers or reference signals. Model learning is potentially faster and much more flexible and powerful in responding to changes in the plant. We propose to research both kinds of learning in connectionist controllers, and, moreover, to explore their integration in a single controller that takes full advantages of each of their capabilities.

Our approach to integrating reinforcement learning and model learning is based on the conventional control method of *dynamic programming* [16]. There is not room here to fully explain the proposed integration, but the key idea is to think of reinforcement learning as dynamic programming where the actual system acts as its own model. The interaction between the reinforcement learning algorithm and the real system is very similar to that between certain dynamic programming algorithms and the system model. Dynamic programming involves a process that, like a temporal-difference critic, gradually builds up an evaluation function over states reflecting the long-term consequences for reward of being in that state. One dynamic programming procedure, *policy improvement*, also gradually builds up estimates of the optimal control law using the developing evaluation function, just as the reinforcement learning process gradually learns a control law using the critic's evaluations. Of course, there are substantial differences as well. Policy improvement alternates between updating the control law for all states and updating the evaluation function for all states, whereas reinforcement learning does a control law update and an evaluation function update for each state that is visited. In addition, when the dynamic-programming process updates a state, it normally considers all possible actions and outcomes from that state, whereas reinforcement learning considers only the action actually selected and the outcome that actually occurs. This reflects a major limitation of using the actual system as model - actions cannot be retracted, and only samples from the probability distributions are known, not the actual distributions that would be available from an explicit model; the actual system acts only as a *Monte Carlo* model.

Viewing reinforcement learning as a dynamic programming process suggests a conceptually clear way of combining it with model learning. Three connectionist networks are used, one to implement the model, one to implement the control law, and one to implement the evaluation function or critic. The model network learns in the usual way to form a forward model—given a state and an action it predicts the next state and reward. The control-law and critic networks learn according to a reinforcement learning algorithm working in two ways. One way is to do dynamic programming on the model to gradually improve the control law and evaluation function (critic) on the assumption that the current model is correct. This should give us all of the benefits of the model-learning approach. The second way is to do dynamic programming on the actual process as model; this should be the same as conventional reinforcement learning. That is, we view these two approaches as the same dynamic-programming/reinforcement-learning process operating on two

different structures—the model and the actual system. This integration has the following advantages:

- Delayed effects of actions on rewards are taken into account, both in reinforcement learning and in model use;
- Model use, a computationally intensive search process, can be done incrementally and as time allows: if rapid response is at a premium, actions can be read directly from the control law relying on reinforcement learning, whereas, if time is available, the model can be used to improve the control law before using it;
- Model learning can take full advantage of all feedback provided by the environment;
- The model is not required to be either invertible or differentiable in order to be used, although gradient methods can be used to speed the search if the model is differentiable.

The current status of this work is described in [12, 14, 52, 53].

8 Relationships to Animal Learning

Learning is of course also studied by life-scientists. It is appropriate to ask about the nature of the correspondence between the behavior of animals in conditioning experiments and the mathematical theories and computational procedures developed for "synthetic" learning. We are exploring in detail a model of classical conditioning in animals that is based on temporal-difference learning (as in adaptive critics) [46, 52] and also broader relationships between instrumental animal learning and reinforcement learning [12, 14, 50]. We are beginning to build a framework that we hope will lead to increased understanding of animal behavior as well as novel computational procedures for practical tasks.

9 VLSI Implementations

In many control applications, the computations must be performed in real time, suggesting the need for special purpose hardware. Connectionist learning systems are particularly intriguing in this regard since in principle they can be mapped directly onto analog VLSI. Because learning can compenstate for defects and imprecision in the circuits, these designs can be considerably simpler than more traditional analog circuits. However, in practice most existing connectionist learning algorithms do not map well onto VLSI due their reliance upon high precision floating point arithmetic. Rather than trying to implement known algorithms and architectures in VLSI, we are concentrating on developing a set of analog computational primitives from which VLSI connectionist network learning systems can be built. The full potential of VLSI connectionist networks can be realized only by developing structures within the constraints imposed by device physics and fabrication technology [30, 41]. To encourage others to work within these constraints, we are developing C++ behavioral models of our circuits that can be used in connectionist network simulations to develop VLSI compatible algorithms and architectures.

10 Application: Monitoring Manufacturing Processes

One application of connectionist learning being pursued at GTE Labs involves using neural networks to analyze quality control on a fluorescent lamp manufacturing line. All manufacturing processes are subject to incompletely understood changes due to variations in raw materials, environmental factors such as

weather, wearing and aging of the machinery, and changes in operators. By using past experience to find correlations between approximately 100 sensory measurements, we will determine which process variables most affect quality. These correlations are then turned over to engineers, who are responsible for making all actual changes to the running of the plant at this time.

Our approach is to compare conventional (batch) and connectionist-learning (incremental) techniques. Conventional computer-integrated manufacturing (CIM) approaches suffer from several problems that can potentially be solved by learning approaches. We have shown in simulation that connectionist learning networks can monitor manufacturing processes to determine causal relationships with an accuracy competitive with that of conventional statistical techniques [19]. Moreover, the network operates on-line, in real-time, and with substantial savings in computational complexity as compared to conventional CIM techniques. The network requires processing per time step that is $O(n)$, where n is the number of sensors, whereas the total processing required by a conventional method such as linear regression is approximately $O(n^3)$, and even the most incremental implementation of linear regression requires at least $O(n^2)$ processing per time step. For hundreds of sensors, this and other computational advantages of the learning network have a tremendous effect; the network can be implemented on a much smaller computer, or it can be used with many more sensors, or more frequently sampled sensors. The reduced computational complexity of the connectionist network approach also allows more freedom in the choice of the model used to predict outcomes and correlations.

A connectionist network is currently installed and learning at one fluorescent-lamp manufacturing plant, but it is too early yet to assess its performance.

REFERENCES

[1] Albus, J. S., *Brains, Behavior, and Robotics*, 1981, BYTE Publications, Peterborough, New Hampshire.

[2] Albus, J. S. , "Hierarchical Control of Intelligent Machines Applied to Space Station Telerobotics," *Proceedings of the IEEE International Symposium on Intelligent Control 1987*, Meystel, A. & Luh, J. Y. S., eds., January 1987, Philadelphia, Pennsylvania.

[3] Anderson, C. W., *Learning and Problem Solving with Multilayered Connectionist Systems*, Doctoral Dissertation, Department of Computer and Information Science, 1986, University of Massachusetts, Amherst, Massachusetts.

[4] Anderson, C. W., "Strategy Learning with Multilayer Connectionist Representations," *Proceedings of the Fourth International Workshop on Machine Learning*, 1987, Irvine, California.

[5] Anderson, C. W., "Tower of Hanoi with Connectionist Networks: Learning New Features," *Proceedings of the Sixth International Workshop on Machine Learning*, June 1989, Ithaca, New York.

[6] Anderson, C. W. & Miller, W. T. III, "A Set of Challenging Control Problems," in *Neural Networks for Control*, Miller, W. T., III, Sutton, R. S., Werbos, P. J., eds., MIT Press, in press.

[7] Barto, A. G. & Sutton, R. S., "Landmark Learning: an Illustration of Associative Search," *Biological Cybernetics*, *42*, 1981, pp. 1–8.

[8] Barto, A. G., Sutton, R. S., & Brouwer, P. S., "Associative Search Network: A Reinforcement Learning Associative Memory," *Biological Cybernetics*, *40*, 1981, pp. 201–211.

[9] Barto, A. G., Sutton, R. S., & Anderson, C. W., "Neuronlike Elements that can Solve Difficult Learning Control Problems," *IEEE Trans. on Systems, Man, and Cybernetics*, *13*, 1983, pp. 835–846.

[10] Barto, A. G., "Learning by Statistical Cooperation of Self-interested Neuron-like Computing Elements," *Human Neurobiology*, 4, 1985, pp. 229–256.

[11] Barto, A. G. & Anandan, P., "Pattern Recognizing Stochastic Learning Automata," *IEEE Transactions on Systems, Man, and Cybernetics*, *15*, 1985, pp. 360–375.

[12] Barto, A. G., Sutton, R. S., & Watkins, C., "Sequential Decision Problems and Neural Networks," *Proceedings of the 1989 IEEE Conference on Neural Information Processing Systems*, November 1989, Snowbird, Utah.

[13] Barto, A. G., "Connectionist Learning for Control: An Overview," in *Neural Networks for Control*, Miller, W. T., III, Sutton, R. S., Werbos, P. J., eds., MIT Press, in press.

[14] Barto, A. G., Sutton, R. S., & Watkins, C., "Learning and Sequential Decision Making," in *Learning and Computational Neuroscience*, Gabriel, M. & Moore, J. W., eds., MIT Press, in press.

[15] Bellman, R. E., *Dynamic Programming*, 1957, Princeton University Press, Princeton, New Jersey.

[16] Farley, B. G., and Clark, W. A., Simulation of Self-organizing Systems by Digital Computer. *I.R.E. Transactions on Information Theory*, 4, 1954, pp. 76–84.

[17] Franklin, J. A., "Learning Control in a Robotic System," *The Proceedings of the 1987 IEEE International Conference on Systems, Man, and Cybernetics*, October 1987, Alexandria, Virginia, pp. 466-470.

[18] Franklin, J. A., *Compliance and Learning: Control Skills for a Robot Operating in an Uncertain World*, Ph.D. Dissertation, February 1988, Electrical and Computer Engineering Department, University of Massachusetts, Amherst, Massachusetts.

[19] Franklin, J. A., Sutton, R. S., & Anderson, C. A., "Application of Connectionist Learning Methods to Manufacturing Process Monitoring," *Proceedings of the Third IEEE International Symposium on Intelligent Control*, August 1988, Arlington, Virginia, pp. 709-712.

[20] Franklin, J. A., "Refinement of Robot Motor Skills Through Reinforcement Learning," *Proceedings of the* 27^{th} *IEEE Conference on Decision and Control*, December 1988, Austin, Texas, pp. 1096-1101.

[21] Franklin, J. A., "Input Space Representation for Refinement Learning Control," *Proceedings of the Fourth IEEE International Symposium on Intelligent Control*, September, 1989, Albany, New York, pp. 115-122.

[22] Franklin, J. A., "Historical Perspective and State of the Art in Connectionist Learning Control," *Proceedings of the* 28^{th} *IEEE Conference on Decision and Control*, December, 1989, Tampa, Florida.

[23] Fu, K. S., "Learning Control Systems—Review and Outlook," *IEEE Trans. on Automatic Control*, Vol. AC-15, No. 2, April, 1970.

[24] Guez, A. & Selinsky, J., "A Trainable Neuromorphic Controller," *J. of Robotic Systems*, vol. 5, no. 4, pp. 363–388, August, 1988.

[25] Gullapalli, V., "A Stochastic Algorithm for Learning Real-Valued Functions via Reinforcement Feedback," COINS Technical Report 88-91, 1988, Computer and Information Science Department, University of Massachusetts, Amherst, Massachusetts.

[26] Jordan, M. I, "Sequential Dependencies and Systems with Excess Degrees of Freedom," COINS Technical Report 88-27, 1988, Computer and Information Sciences, University of Massachusetts, Amherst, Massachusetts.

[27] Kawato, M., Uno, Y., Isobe, M., & Suzuki, R.,"Hierarchical Connectionist Network Model for Voluntary Movement with Application to Robotics," *IEEE Control Systems Magazine*, Vol. 8, No. 2, April 1988.

[28] Klopf, A. H., "Brain Function and Adaptive Systems—A Heterostatic Theory," Air Force Cambridge Research Laboratories Research Report, AFCRL-72-0164, Bedford, Massachusetts (A summary appears in *Proceedings of the IEEE International Conference on Systems, Man, Cybernetics*, 1974, Dallas, Texas, 1974).

[29] Kuperstein, M., "Adaptive Visual-motor Coordination in Multijoint Robots Using Parallel Architecture," *Proceedings of the IEEE Conference on Robotics and Automation*, 1987, pp. 1595–1602.

[30] Mead, C., *Analog VLSI and Neural Systems*, 1989, Addison-Wesley.

[31] Mendel, J., and Zapalac, J., "The Application of Techniques of Artificial Intelligence to Control System Design," *Advances in Control Systems Theory and Applications*, Vol. 6, 1968, C. Leondes, ed., Academic Press, New York.

[32] Mendel, J. M. & McLaren, R. W., "Reinforcement Learning Control and Pattern Recognition Systems," in *Adaptive, Learning and Pattern Recognition Systems: Theory and Applications*, Mendel, J. M. & Fu, K. S., eds., 1970, Academic Press, New York.

[33] Michie, D., and Chambers, R., ""Boxes" as a Model of Pattern-Formation," *Towards a Theoretical Biology; 1 Prolegomena*, C.H. Waddington, ed., 1968, Edinburgh University Press, Edinburgh.

[34] Miller, W. T. III, "Sensor-based Control of Robotic Manipulators Using a General Learning Algorithm," *IEEE Journal of Robotics and Automation*, Vol RA-3, No. 2, April, 1987.

[35] Minsky, M. L., *Theory of Connectionist-analog Reinforcement Systems and its Application to the Brain-model Problem*, Princeton University Ph.D. Dissertation, 1954.

[36] Moody, J. & Darken, C., "Learning with Localized Receptive Fields," *Proceedings of the 1988 Connectionist Summer School*, 1988, Morgan Kaufmann.

[37] Munro, P., "A Dual Back-propagation Scheme for Scalar Reward Learning," *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, 1987, Amherst, Massachusetts, pp. 165–176.

[38] Psaltis D., Sideris, A. & Yamamura, A. A., "A Multilayered Connectionist Network Controller," *IEEE Control Systems Magazine*, Vol. 8, No. 2, April, 1988, pp. 17–21.

[39] Rumelhart, D. E., Hinton, G. E., & Williams, R. J., "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, ed. by Rumelhart, D. E., McClelland, J. L., and the PDP research group., 1986, Cambridge, Massachusetts, Bradford Books.

[40] Rumelhart, D. E. & Jordan, M. I., "Supervised Learning with a Distal Teacher," 1989, Paper in preparation.

[41] Schwartz, D. S., Howard, R. E., & Hubbard, W. E., "A Programmable Analog Neural Network Chip," *IEEE Journal of Solid-State Circuits*, 1989, SC-24, pp. 313.

[42] Selfridge, O. G., Rissland, E. L., & Arbib, M. A., *Adaptive Control of Ill-Defined Systems*, 1984, NATO Conference Series, Plenum Press, New York.

[43] Selfridge, O. G., Sutton, R. S., & Barto, A. G., "Training and Tracking in Robotics," *Proceedings of IJCAI-85*, International Joint Conference on Artificial Intelligence, 1985, pp. 670–672.

[44] Selfridge, O. G., Sutton, R. S., & Anderson, C. W., "Selected Bibliography on Connectionism," *Evolution, Learning, and Cognition*, Y. C. Lee, ed., 1988, World Scientific Publishing, Singapore.

[45] Selfridge, O. G., "Adaptive Strategies of Learning; A Study of Two-Person Zero-Sum Competition," *Proceedings of the Sixth International Workshop on Machine Learning*, June, 1989, Ithaca, New York.

[46] Sutton, R. S. & Barto, A. G., "Toward a Modern Theory of Adaptive Networks: Expectation and Prediction," *Psychological Review, 88*, 1981, pp. 135–171.

[47] Sutton, R. S., *Temporal Credit Assignment in Reinforcement Learning*, Doctoral Dissertation, 1984, Computer and Information Science Department, University of Massachusetts, Amherst, Massachusetts.

[48] Sutton, R. S. & Pinette, B., "The Learning of World Models by Connectionist Networks," *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, 1985, Seattle, Washington, Lawrence Erlbaum, pp. 355–378.

[49] Sutton, R. S., "Two Problems with Backpropagation and Other Steepest-descent Learning Procedures for Networks," Poster Session at Eighth Annual Meeting of the Cognitive Science Society, 1986, Amherst, Massachusetts.

[50] Sutton, R. S. & Barto, A. G., "A Temporal-difference Model of Classical Conditioning," *Proceedings of the Ninth Conference of the Cognitive Science Society*, 1987, pp. 355–378.

[51] Sutton, R. S., "Learning to Predict by the Methods of Temporal Differences," *Machine Learning*, Vol 3, 1988, pp. 9-44.

[52] Sutton, R. S. & Barto, A. G. "Time-Derivative Model of Pavlovian Reinforcement," in *Learning and Computational Neuroscience*, Gabriel, M. & Moore, J. W., eds., MIT Press, in press.

[53] Watkins, C. J. C. H., *Learning with Delayed Feedback*, Ph. D. thesis, 1989, Cambridge University.

[54] Werbos, P. J., "Building and Understanding Adaptive Systems: A Statistical/Numerical Approach to Factory Automation and Brain Research," *IEEE Transactions on Systems, Man, and Cybernetics*, 1987, Vol. 17, pp. 7-20

[55] Werbos, P. J., "A Menu of Designs for Reinforcement Learning Over Time," in *Neural Networks for Control*, Miller, W. T., III, Sutton, R. S., Werbos, P. J., eds., MIT Press, in press.

[56] Widrow, B. & Smith, F. W., "Pattern-Recognizing Control Systems," *1963 Computer and Information Sciences (COINS) Symposium Proceedings*, Washington, D.C., Spartan.

[57] Widrow, B. & Stearns, S. D., *Adaptive signal processing*, Englewood Cliffs, New Jersey, Prentice-Hall, 1985.

[58] Williams, R. J., "On the Use of Backpropagation in Associative Reinforcement Learning," *Proceedings of the Second Annual International Conference on Connectionist Networks*, San Diego, CA, Vol. I, July, 1988, pp. 263-270.

[59] Williams, R. J., "Toward a Theory of Reinforcement-learning Connectionist Systems," Technical Report NU-CCS-88-3, College of Computer Science, Northeastern University, Boston, Massachusetts, 1988.