Beyond Adaptive Critic- Creative Learning for Intelligent Mobile Robots

[*]Xiaoqun Liao, Ming Cao, and Ernest Hall

Center for Robotics Research
University of Cincinnati

Abstract

Intelligent industrial and mobile robots may be considered proven technology in structured environments. Teach programming and supervised learning methods permit solutions to a variety of applications. However, we believe that to extend the operation of these machines to more unstructured environments requires a new learning method. Both unsupervised learning and reinforcement learning are potential candidates for these new tasks. The adaptive critic method has been shown to provide useful approximations or even optimal control policies to non-linear systems. The purpose of this paper is to explore the use of new learning methods that goes beyond the adaptive critic method for unstructured environments.

The adaptive critic is a form of reinforcement learning. A critic element provides only high level grading corrections to a cognition module that controls the action module. In the proposed system the critic's grades are modeled and forecasted, so that an anticipated set of sub-grades are available to the cognition model. The forecasting grades are interpolated and are available on the time scale needed by the action model.

The success of the system is highly dependent on the accuracy of the forecasted grades and adaptability of the action module. Examples from the guidance of a mobile robot are provided to illustrate the method for simple line following and for the more complex navigation and control in an unstructured environment.

The theory presented that is beyond the adaptive critic may be called creative theory. Creative theory is a form of learning that models the highest level of human learning- imagination. The application of the creative theory appears to not only be to mobile robots but also to many other forms of human endeavor such as educational learning and business forecasting. Reinforcement learning such as the adaptive critic may be applied to known problems to aid in the discovery of their solutions. The significance of creative theory is that it permits the discovery of the unknown problems, ones that are not yet recognized but may be critical to survival or success.

Keywords: Reinforcement learning systems, adaptive critic, intelligent robots, heuristic dynamic programming (HDP), dual heuristic programming (DHP), globalized DHP (GDHP).

1. Introduction

Intelligence is the most outstanding human characteristic; however, it is still not totally understood and therefore has many varying definitions, implied meanings, and levels of sophistication. Studies in Artificial Intelligence (AI) attempt to implement the capacity of learning or understanding with a mathematical or computer algorithm. Research in Machine Intelligence (MI) is directed toward designing new, useful, adaptive machines.

Current researchers are attempting to develop *intelligent robots*. Hall[1] defines an intelligent robot as one that responds to changes to its environment through sensors connected to a controller. Much of the research in robotics has been concerned with vision and tactile sensing. *Artificial intelligence*, or AI, programs using heuristic methods have

[*] Xiaoqun Liao: Liaoxn@email.uc.edu, Ming Cao: Caomg@email.uc.edu, Ernie L. Hall: Ernie.Hall@uc.edu, The center for Robotics Research, University of Cincinnati, Cincinnati, OH 45221-0072

somewhat solved the problem of adapting, reasoning, and responding to changes in the robot's environment. For example, one of the most important considerations in using a robot in a workplace is human safety. A robot equipped with sensory devices that detect the presence of an obstacle or a human worker within its workspace could automatically shut itself down in order to prevent any harm to itself or the human worker.

The learning of locomotion in an unknown environment is extremely difficult to achieve by formal logic programming. However, typical robot applications in manufacturing assembly tasks would require locating components and placing them in random positions. Fortunately, Kohonen[2] suggests that a higher degree of learning is possible with the use of *neural computers*. The intelligent robot is supposed to plan its action in the natural environment, while at the same time performing non-programmed tasks. Learning has not yet been applied to industrial robots to any major extent. This limits the application of intelligent robots.

The purpose of this paper is to present a new theory of learning called creative learning. This theory is beyond the adaptive controller in that the reinforcement comes from the learning machine rather than from an external critic. Such an approach offers potential solutions to problems in which the objective criteria is unknown or yet to be discovered.

A brief review of robot control strategies is present in Section 2. The intelligent robot controller is presented in Section 3. Adaptive critic and creative theory are described in Section 4. Results and conclusions are given in Section 5**.**

2. Robot Control Strategies

One popular robot control scheme is *computed-torque control* or *inverse-dynamics control*. Most robot control schemes found in robust, adaptive, or learning control strategies can be considered special cases of computed-torque control. These techniques involve the decomposition of the control design problem into two parts [3]:

1. A primary controller, a feedforward (inner-loop) design to track the desired trajectory under ideal conditions.
2. A secondary controller, a feedback (outer-loop) designs to compensate for undesirable deviations (disturbances) of the motion from the desired trajectory based on a linearized model.

The primary controller compensates for the nonlinear dynamic effects and attempts to cancel the nonlinear terms in the dynamic model. However, since the parameters in the dynamic model of the robot are not usually exact, undesired motion errors are expected. The secondary controller can correct these errors. Figure 1 represents the decomposition of the robot controller showing the primary and secondary controllers.

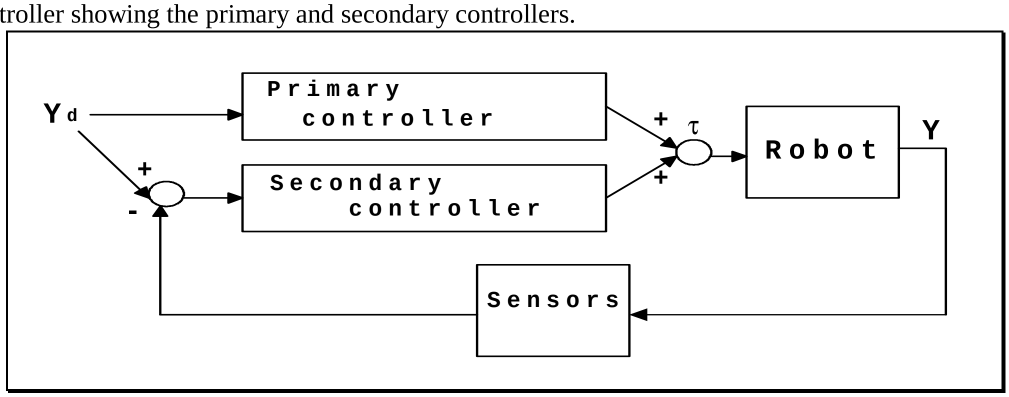

Figure 1. Controller decomposition in primary and secondary controllers

The human brain has been the model information-processing device for many researchers in the design of intelligent computers, or *neural* computers. Psaltis, *et al.* [4] describes the neural computer as a large interconnected mass of simple processing elements, or artificial neurons. The functionality of this mass, called the *artificial neural network,* is determined by modifying the strengths of the connections during the learning phase.

Researchers interested in neural computers have been successful in computationally intensive areas such as pattern recognition and image interpretation problems. These problems generally involve the static mapping of input vectors into corresponding output classes using a feedforward neural network. The feedforward neural network is specialized for the static mapping problems. In the robot control problem, nonlinear dynamic properties need to be dealt with and a different type of neural network structure must be used. Recurrent neural networks have the dynamic properties, such as feedback architecture, needed for the appropriate design of such robot controllers.

Artificial Neural Networks

ANNs are highly parallel, adaptive and fault tolerant dynamical systems modeled like their biological counterparts. The phrases "neural networks" or "neural nets" are also used interchangeably in the literature, which refer to neurophysiology, the study of how the brain and its nervous system work. ANNs are specified by the following definitions [5]:

Topology

This describes the networked architecture of a set of neurons. The sets of neurons are organized into layers, which are then classified as either *feedforward networks* or *recurrent networks.* In feedforward layers, each output in a layer is connected to an input in the next layer. In a recurrent ANN, each neuron can receive as its input a weighted output from other layers in the network, possibly including itself. Figure 2 illustrates three simple representations of the ANN topologies.

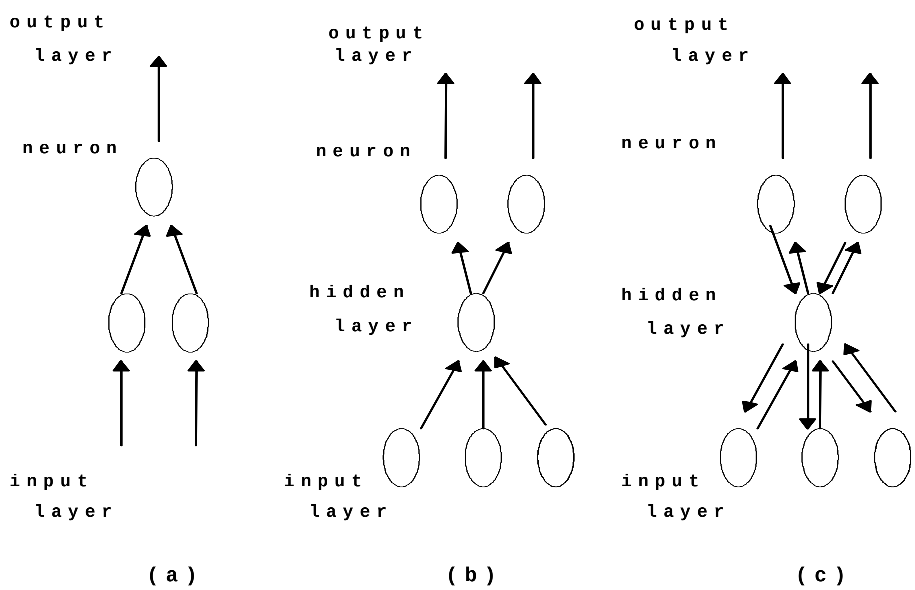

Figure 2 - ANN topologies: (a) single-layer feedforward (b) multilayer feedforward; (c) multilayer recurrent

Neuron

In ANNs, a neuron is a computational element that defines the characteristics of input/output relationships. A simple neuron is shown in Figure 3, which sums N weighted inputs (called activation) and passes the result through a nonlinear transfer function to determine the neuron output. Two nonlinear functions that are often used to mimic biological neurons are the unit step function and the linear transfer-function. A very common formula for determining a neuron's output is through the use of sigmoidal (squashing) functions:

$$g(x) = (1 + e^{-kx}) \quad \text{range of (0,1)} \qquad \textbf{(1a)}$$

$$g(x) = \tanh(kx) \quad \text{range of (-1,1)} \qquad \textbf{(1b)}$$

For various values of the slope parameter, k, these functions are continuous and have derivatives at all points.

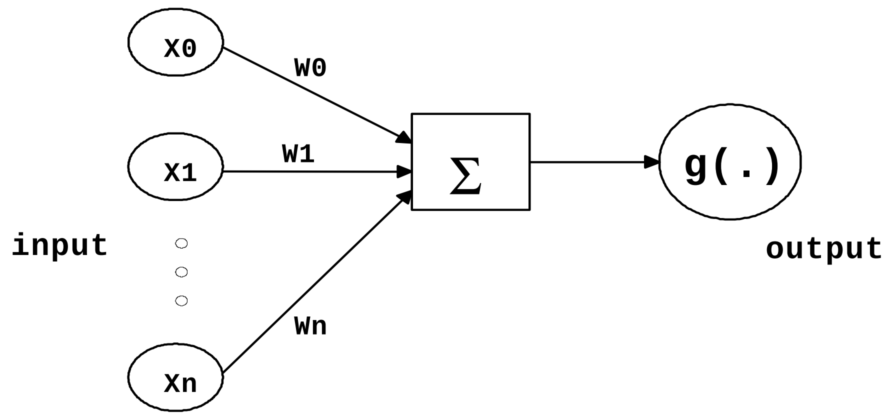

Figure 3 - McCulloch and Pitts neuron

Learning Rules

Given a set of input/output patterns, ANNs can learn to classify these patterns by optimizing the weights connecting the nodes (neuron) of the networks. The learning algorithms for weight adaptation can be described as either supervised or unsupervised learning or reinforcement learning. In supervised learning, the desired output of the neuron is known, perhaps by providing training samples. During supervised training, the network compares its actual response, which is the result of the transfer function described above, with the training example. It then adjusts its weight in order to minimize the error between the desired and its actual output. In unsupervised training, where there are no teaching examples, built-in rules are used for self-modification, in order to adapt the synaptic weights in response to the inputs to extract features from the neuron. Kohonen's self-organizing map is an example of unsupervised learning [6]. Reinforcement learning is also named as adaptive critic learning, which is addressed in next section.

One of the first models of an artificial neuron, was introduced in 1943 by McCulloch and Pitts and is shown in Figure 3. It proved that a synchronous network of neurons (M-P network) is capable of performing the simple logical tasks (computations) that are expected of a digital computer. In 1958, Rosenblatt introduced the "perceptron", in which he showed how an M-P network with adjustable weights can be trained to classify sets of patterns. His work was based on Hebb's model of adaptive learning in the human brain [7], in which he stated the neuron's interconnecting weights change continuously as it learns [8].

In 1960, Bernard Widrow introduced its ADALINE (**ADA**ptive **LIN**ear element), a single-layer perceptron, and later extended it to what is known as MADALINE, multilayer ADALINE [9]. In MADALINE, Widrow introduced the steepest descend method to stimulate learning in the network. His variation of learning is referred to as the Widrow-Hoff rule or delta rule.

In 1969, Minsky and Papert [10] reported on the theoretical limitations of the single layer M-P network, by showing the inability of the network to classify the exclusive-or (XOR) logical problem. They left the impression that neural network research is a farce, and went on to establish the "artificial intelligence" laboratory at MIT. Hence, the research activity related to ANNs was largely dormant until the early 1980s when the work by Hopfield, an established physicist, on neural networks rekindled the enthusiasm for this field. Hopfield's autoassociative neural network (a form of recurrent neural network) solved the classic hard optimization problem (traveling salesman) [11].

Other contributors to the field, Steven Grossberg and Teuvo Kohonon, continued their research during the seventies and early eighties. During these "quiet years", Steven Grossberg [12, 13] worked on the mathematical development necessary to overcome one of the limitations reported by Minsky and Papert [10]. Teuvo Kohonon [14] developed the unsupervised training method, the self-organizing map. Later, Bart Kosko [15] developed bi-directional associative memory (BAM) based on the works of Hopfield and Grossberg. Robert Hecht-Nielson [16] pioneered the work on neurocomputing.

It wasn't until 1986 that the two-volume book, by McClleland and Rumelhart, titled *Parallel Distributed Processing* (PDP), exploded the field of artificial neural networks [16]. In this book a new training algorithm, called *The Backpropagation* method (BP), the gradient search technique was used to train a multilayer perceptron to learn the XOR mapping problem described by Minsky and Papert [17]. Since then, ANNs have been studied for both design procedures and training rules (supervised and unsupervised). An excellent collection of theoretical and conceptual papers on neural networks can be found in books edited by Vemuri[8], and Lau [18]. Interested readers can also refer to a survey of neural networks book by Chapnick[19] categorized by: theory, hardware and software, and how-to books.

The multilayer feedforward networks, using the BP method, represent a versatile nonlinear map of a set of input vectors to a set of desired output vectors on the spatial context (space). During the learning process, an input vector is presented to the network and propagates forward from input layers to output layers to determine the output signal. The output signal vector is then compared with the desired output vector, resulting in an error signal. This error signal is backpropagated through the network in order to adjust the network's connecting strengths (weights). Learning stops when the error vector has reached an acceptable minimum [5].

Many studies have been undertaken in order to apply both the flexibility and the learning ability of backpropagation to robot control on an experimental scale [20 21 22]. In a recent study, an ANN utilizing an adaptive step size algorithm based on random search techniques, improved the convergence speed of the BP method for solving the inverse kinematic problem for a two-link robot [23]. The robot control problem is a dynamic problem, while the BP method only provides a static mapping of the input vectors into output classes, which limits its benefits. In addition, like any other numerical method, this novel learning method has limitations, like a slow convergence rate, and a local minimum. Attempts to improve the learning rate of BP have resulted in many new approaches [24 25]. It is necessary to note that the most important behavior of the feedforward networks using the BP method is its classification ability or the generalization to fresh data rather than temporal utilization of past experiences.

A *recurrent network* is a multilayer network in which the activity of the neurons flows both from input layer to output layer (feedforward), and also from the output layer back to the input layer (feedback), in the course of learning [16]. In a recurrent network, each activity of the training set (input pattern) passes through the network more than once before it generates an output pattern, whereas in standard BP only the error flows backward, not the activity. This network architecture can base its response to problems on both spatial (space) and temporal (time) contexts [26 27]. Therefore, it has potential to model time-dependent processes such as robotic applications.

It is evident that a recurrent network will require a more substantial memory in simulation (more connections) than a standard BP. Recurrent network computing is a complex method, with a great deal of record keeping of errors and activities at each time phase. However, preliminary results indicate that they have the ability to learn extremely complex temporal patterns where data is unquantified with very little preprocessing, i.e. stock market prediction and Fourier transforms relationships [28]. In feedforward networks where the training process has no memory, each input is independent of the previous input. It is advantageous, especially in repetitive dynamical systems, to focus on the properties of the recurrent networks to design better robot controllers.

3. Robot Neural Controller

In order to design intelligent robot controllers, one must also provide the robot with a means of responding to problems in both a temporal and spatial time context. It is the goal of the robot researcher to design a neural learning controller to utilize the available data from the repetition in robot operation. The neural learning controller, based on the recurrent network architecture, has the time-variant feature that once a trajectory is learned, it should learn a second one in a shorter time.

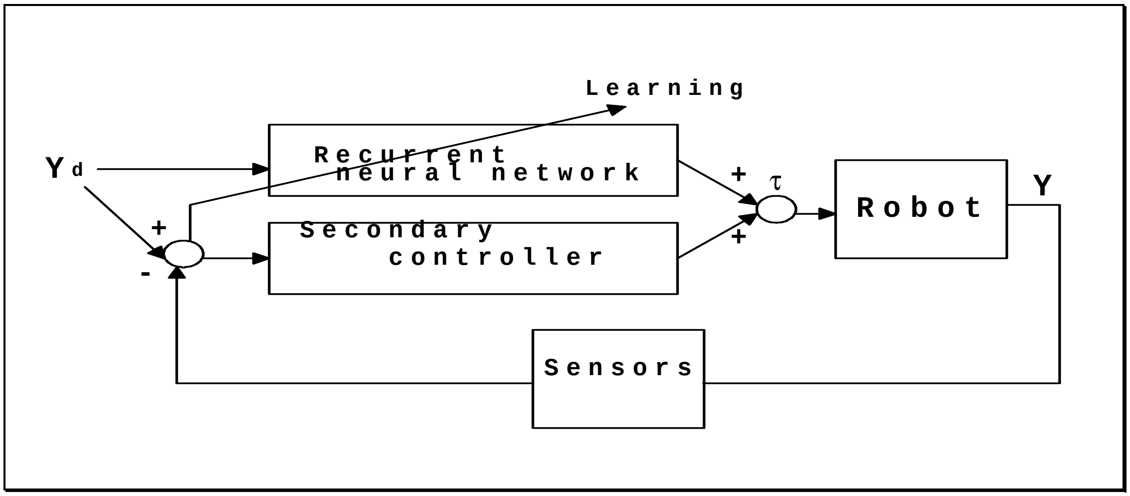

Figure 4. Recurrent neural learning controller

In Figure 4, the time-variant, recurrent network will provide the learning block, or primary controller, for the inverse dynamic equations discussed above. The network compares the desired trajectories: η_d, $\dot{\eta}_d$, $\ddot{\eta}_d$, with continuous paired values for the three-axis robot η, $\dot{\eta}$, $\ddot{\eta}$: τ, at every instant in a sampling period. The new trajectory parameters are then combined with the error signal from the secondary controller (feedback controller) for actuating the robot manipulator arm.

Neural networks can be applied either as a system identification model or as a control for the robot controller described in Figure 4. ANNs can be used to obtain the system model identification that can be used to design the appropriate controller. Once the real system model is available, they can also be used directly in design of the controller[29]. Neural network approaches to robot control are discussed in general by Psaltis *et al.* [4], and Yabuta and Yamada [30]. These approaches can be classified as:

1. **Supervised control**, a trainable controller that, unlike the old teaching pendant, allows responsiveness to sensory inputs. A trainable neuromorphic controller reported by Guez and Selinsky [31] provides an example of a fast, real-time and robust controller.
2. **Direct inverse control** is trained for the inverse dynamic of the robot. Kung and Hwang [32] used two networks on-line in their design of the controller.
3. **Neural adaptive control**, neural nets combined with adaptive controllers result in greater robustness and the ability to handle nonlinearity. Chen [33] reported the use of the BP method for a nonlinear self-tuning adaptive controller.
4. **Backpropagation of utility** involves information flowing backward through time. Werbos's back-propagation through time is an example of such a technique [34].
5. **Adaptive critic method** uses a critic evaluating robot performance during training. This is a very complex method that requires more testing [35].

In the direct inverse control approach, the recurrent neural network will learn the inverse dynamic of the robot in order to improve the controller performance. In such a system, the neural network model replaces the primary controller (see Figure 1). In this approach, a secondary feedback controller will be used to teach the network initially. As learning takes place, the neural network takes full control of the system. Kawato and his research group were successful using this approach in trajectory control of a three degree-of-freedom robot [36 37]. Their approach is known as feedback-error-learning control. However, their neural network structure was simply the linear collection of all nonlinear dynamic terms, or subsystems, in the dynamic motion equation. Learning was used purely for estimating the subsystems. As the degrees of freedom increase, the network size needs to increase in the order of n^4. For example, for six degrees-of-freedom, 942 subsystems are needed, compared with 43 for a robot with three degrees-of-freedom. However, due to the parallel processing capability of the neural network, the implementation of Kawato's method is still an attractive method.

Goldberg and Pearlmutter [26] have demonstrated the utility of the feedback-error-learning approach for the motion control of the first two joints of the CMU DDArm II, using temporal windows of measured positions as input to the network; the output of the network is the torque vector. Newton and Xu [38] used this approach to control a flexible space robot manipulator (SM^2) in real-time. The trajectory tracking error was reduced by 85% when compared to conventional PID control scheme. More recently, Lewis *et al.* [39] developed an on-line neural controller, based on the robot passivity properties (that the system cannot become unstable if the robot cannot create energy), using a similar approach with good tracking results. The feasibility and performance of the feedback-error-learning control with global asymptotic stability has also been reported [40 41]. The design of a compact and generic *recurrent network* has shown promising results in replacing the need for custom subsytems-type design such as the one by Kawato's group[42]. The proposed controller performs based on the systematic design approach and the recurrent network's time-variant feature.

4. Adaptive Critic and Creative Learning

Werbos[43] summarized recent accomplishments in neurocontrol as a "brain-like" intelligent system. It should contain at least three major general-purpose adaptive components: (1) an Action or Motor system, (2) an "Emotional" or "Evaluation" system or "Critic" and (3) an "Expectations" or "System Identification" component.

"Critic" served as a model or emulator of the external environment or the plant to be controlled, solving optimal control problem over time classified as adaptive critic designs (ACD) [44]. ACD is a large family of designs which learn to perform utility maximization over time. According to modern control theory, dynamic programming is the only exact and efficient method for utility maximization or optimization over future time. In dynamic programming, normally the user provides the function $\mathbf{U}(\underline{\mathbf{X}}(t), \underline{\mathbf{u}}(t))$, an interest rate **r,** and a stochastic model. Then the analyst tries to solve for another function $\mathbf{J}(\underline{\mathbf{X}}(t))$, so as to satisfy some form of Bellman equation, the equation that underlies dynamic programming [45]:

$$J(\underline{\mathbf{X}}(t)) = \max_{\underline{\mathbf{u}}(t)} \left(U(\underline{\mathbf{X}}(t), \underline{\mathbf{u}}(t)) + \langle J(\underline{\mathbf{X}}(t+1))\rangle/(1+r) \right), \qquad \textbf{(2)}$$

where "<>" denotes expected value.
In principle, any problem in decision or control can be classified as an optimization problem. Many ACDs solve the problem by approximating the function **J**. They try to find the best possible function $\mathbf{J}(\underline{\mathbf{X}},\mathbf{W})$, defined by a set of parameters or weights **W.** The nonlinear function approximator **J** is called a "Critic"
If the weights **W** are adapted or iteratively solved for, in real time learning or offline iteration, we call the Critic as Adaptive Critic[45]. An ACD is any system which includes an adapted Critic component; a Critic, in turn, is a neural net or other nonlinear function approximator which is trained to converge to the function **J(X)**.

The adaptive critic approach, like the neurocontrol in general, is a complex field of study with its own "ladder" of design from the simplest and most limited all the way up to the brain itself with five levels. The simplest level is the original Widrow design [46]. He shaped the term "Critic". Level one is the Barto-Sutton-Anderson design, which uses a global reward system to train an Action network and "TD" methods to adapt the Critic. Level two is called "Action-Dependent Adaptive Critic" (ADAC). In ADAC, the Critic sends derivative signals back to the Action network, so that backpropagation can be used to adapt the Action network [47]. "Brain-like control", represents levels 3 and above. Level 3 is to use heuristic dynamic programming (HDP) to adapt a Critic, and backpropagate through a Model to adapt the Action network. Levels 4 and 5 respectively use more powerful techniques to adapt the Critic – Dual Heuristic Programming (DHP) and Globalized DHP (GDHP). The specific discussion on HDP, DHP and GDHP is followed in the next section[43].

Heuristic Dynamic Programming (HDP)

HDP and its ACD form have a critic network that estimates the function J (cost-to-go or strategic utility function) in the Bellman equation of dynamic programming, presented as follows [48]:

$$J(t) = \sum_{k=0}^{\infty} \gamma^k U(t+k) \quad \textbf{(3)}$$

where γ is a discount factor for finite horizon problems (0<γ<1), and U(.) is the utility function or local cost. The critic network tries to minimize the following error measure over time:

$$\| E_1 \| = \sum_t E_1^2(t) \quad \textbf{(4)}$$

$$E_1(t) = J[Y(t)] - \gamma J[Y(t+1)] - U(t) \quad \textbf{(5)}$$

where Y(t) stands for either a vector R (t) of observables of the plant or a concatenation of R(t) and a control (action) vector A(t). The configuration for training the critic according to (5) is shown Figure 5.

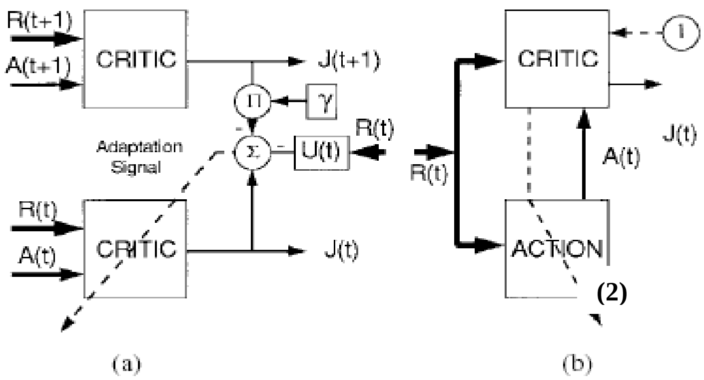

Figure 5 (a) Critic Adaptation in ADHDP/HDP (b) Action adaptaion (Detailed explanation in Prokhorov [48])

Dual Heuristic Programming (DHP)

DHP and its ACD form have a critic network that estimates the derivatives of J with respect to the vector Y. The critic network learns minimization of the following error measure over time:

$$\| E_2 \| = \sum_t E_2^T(t) E_2(t) \quad \textbf{(6)}$$

where

$$E_2(t) = \frac{\partial J[Y(t)]}{\partial Y(t)} - \gamma \frac{\partial J[Y(t+1)]}{\partial Y(t)} - \frac{\partial U(t)}{\partial Y(t)} \quad \textbf{(7)}$$

The critic network’s training is more complicated than in HDP since all the relevant pathways of backpropagation is taken into account as shown Figure 6.

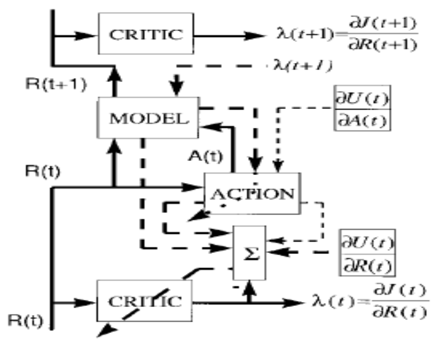

Figure 6 Adaptation in DHP. (Detailed explanation in Prokhorov [48])

Globalized Dual Heuristic Programming (GDHP)

GHDP minimizes the error with respect to both J and its derivatives. Werbos [HIC] first proposed the idea how to do GDHP. Training the critic network in GDHP utilizes an error measure which is a combination of the error measures of HDP and DHP (4) and (6). This results in the following LMS update rule for the critic's weights [Prokhorov 1997]:

$$\Delta W_c = -\eta_1 [J(t) - \gamma J(t+1) - U(t)] \frac{\partial J(t)}{\partial W_c} - \eta_2 \sum_{j=1}^{n} E_{2j} \frac{\partial^2 J(t)}{\partial R_j(t) \partial W_c} \tag{8}$$

where $E_{2j}$is given by DHP training, η_1 and η_2 are positive learning rates.

The major source of the additional complexity in GDHP is the necessity of computing second-order derivatives $\partial^2 J(t)/\partial R(t)\partial Wc$. To get the adaptation signal-2 (Figure 7), we first need to create a network dual to our critic network. The dual network inputs the output J and states of all hidden neurons of the critic. Its output, $\partial J(t)/\partial R(t)$, is exactly the critic's output to its input R(t) while performing backpropagation. Prokhorov stated that the group first successfully implemented a GDHP design with critic's training based on deriving explicit formulas for finding $\partial^2 J(t)/\partial R(t)\partial Wc$.

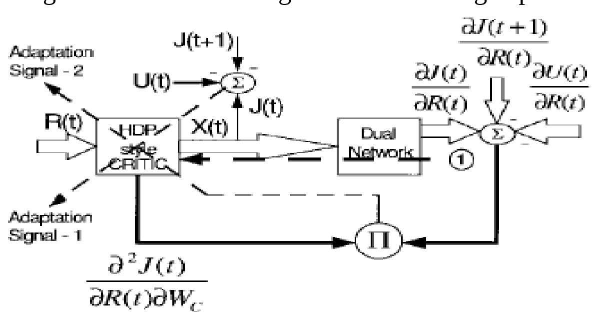

Figure 7 Adaptation in general GDHP design. (Detailed explanation in Prokhorov[48])

Creative Learning

The previous discussion is a summary of three of the most popular adaptive critic methods for adaptive critic design. Beyond these learning methods (adaptive critic) addressed above, we propose a new learning method, called creative learning. Creative learning is used to explore the unpredictable environment, permit the discovery of unknown problems, ones that are not yet recognized but may be critical to survival or success. It generalizes the highest level of human learning – imagination. The block diagram of the creative controller is shown in Figure 8. Experience with the guidance of a mobile robot have motivated this study to progress from simple line following to the more complex navigation and control in an unstructured environment.

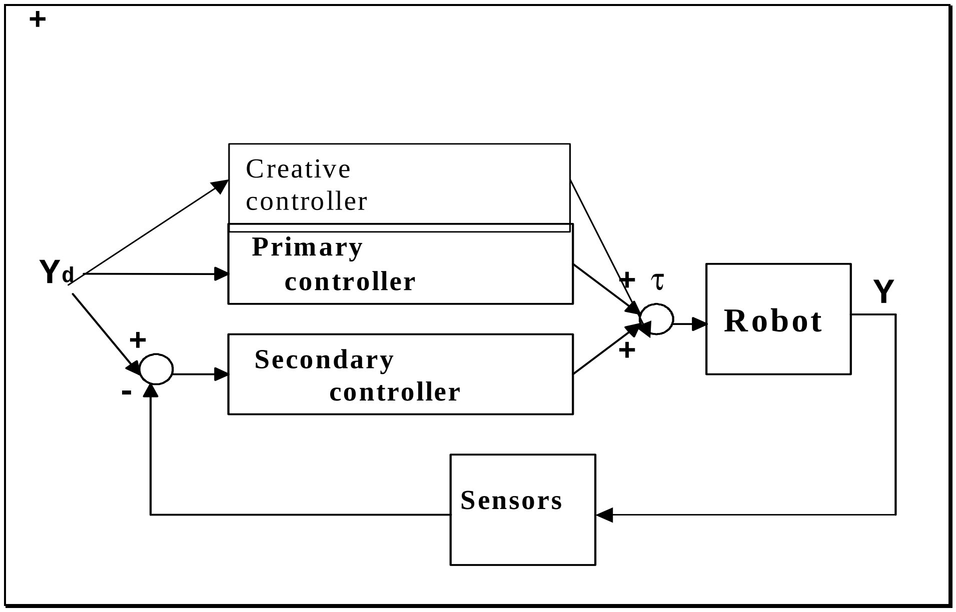

Figure 8 Block diagram of creative controller

5. Conclusions

In this paper, a review of learning machine (robot control strategies) is presented, from the original artificial neural network (ANN) to recurrent network as robot controllers. To design the intelligent robot controller, neural network approaches are addressed above, including supervised control, direct inverse control, neural adaptive control, backpropagation of utility, and adaptive critic method. As a most important optimal theory, we summarized the three advanced adaptive critic methods HDP, DHP, and GDHP according to its own "ladder". Beyond the adaptive critic approach, a creative learning theory is proposed in this paper. The significance of this approach is to generalize the highest level of human learning – imagination. We predict that the creative theory is going to be a real "emotional" or "expectations" component of a "brain-like" intelligent system.

References

1. E.L. Hall, and B. C. Hall, *Robotics: A User-Friendly Introduction*, pp. 1-8, Saunders College Publishing, Holt, Rienhart and Wilson, Orlando FL, 1985.
2. T. Kohonen, "Introduction to neural computing", *Neural Networks*, vol. **1**, pp. 3-16, 1988.
3. A. J. Koivo, *Fundamentals for Control of Robotic Manipulators*, pp. 306-338, John Wiley & Sons, Inc., New York NY, 1989.
4. D. Psaltis, A. Sideris, and A. A. Yamamura, "A multilayered neural network controller", *IEEE Control Systems Magazine*, vol. **8(2)**, pp. 17-21, 1988.
5. R. P. Lippman, "An introduction to computing with neural nets", *IEEE ASSP Magazine*, vol. **4(2)**, pp. 4-22, April 1987.
6. M. Chester, *Neural Networks: A Tutorial*, Prentice Hall, Englewood Cliffs New Jersey, 1993.
7. D.O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*, Wiley, New York NY, 1949.
8. V. Vemuri, "Artificial neural networks: an introduction", *Artificial Neural Networks: Theoretical Concepts*, V. Vemuri (ed.), pp. 1-12, IEEE Computer Society Press, 1988.
9. B. Widrow, and M.A. Lehr, "30 years of adaptive neural networks: Perceptron, madaline, and backpropagation", *Proceedings of IEEE*, vol. 78, pp. 1415-1442, 1990.
10. M.L. Minsky, and S.A. Papert, *Perceptrons*, Cambridge MA 1969.
11. J.J. Hopfield, and T.W. Tank, "'Neural' computation of decisions in optimization problems", *Biological Cybernetics*, vol. **52**, pp. 141-152, 1985.
12. S. Grossberg, (ed.), *Neural Networks and Natural Intelligence*, The MIT Press, Cambridge MA, 1988.
13. S. Grossberg, “Studies of Mind and Brain”, *Boston Studies in the Philosophy of Science*, D. Reidel Publishing Company, Boston MA, V.70, 1982.
14. T. Kohonen, "Self-organized formation of topologically correct feature maps", *Biological Cybernetics*, vol. **43**, pp. 59-69, 1982.
15. B. Kosko, "Bi-directional Associative Memories", *IEEE Transactions on System, Man, and Cybernetics*", vol. **18(1)**, pp. 49-60, Jan-Feb. 1988.
16. R. Hecht-Nielsen, Neurocomputing, pp. 182-190, Addison-Wesley, Reading MA, 1990.
17. D. E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representation by error propagation", *Parallel distributed processing: exploration in the microstructure of cognition*, D.E. Rumelhart and J.L. McClelland (eds.), vol. **1**, pp. 318-362, MIT Press, Cambridge MA, 1986.
18. C. Lau, (Ed.), *Neural Networks: Theoretical Foundations and Analysis*, IEEE Press, New York NY, 1992.
19. P. Chapnick, "Lots of neural nets books", *AI Expert*, pp. 21-23, June 1992.
20. G. Josin, D. Charney, and D. White, "Robot control using neural networks", *IEEE international conference on neural networks*, vol. 2, pp. 625-631, 1988.
21. M. Kuperstein, and J. Wang, "Neural controller for adaptive movements with unforeseen payload", *IEEE Transactions on Neural Networks*, vol. **1(1)**, pp. 137-142, 1990.
22. A. Guez, Z. Ahmad, and J. Selinsky, "The application of neural networks to robotics", *Neural Networks: Current Applications*, P.G.J. Lisboa (ed.), pp. 111-122, Chapman & Hall, London, 1992.
23. W. Golnazarian, E.L. Hall, and R.L. Shell, "Robot control using neural networks with adaptive learning steps", *SPIE Conference Proceedings, Intelligent Robots and Computer Vision XI: Biological, Neural Net, and 3-D Methods*, vol. 1826, pp. 122-129, 1992.
24. R.A. Jacobs, "Increased Rates of Convergence Through learning rate adaptation", *Neural Networks*, vol. **1**, pp. 295-307, 1988.
25. N. Baba, "A new approach for finding the global minimum of error function of neural networks", *Neural Networks*, vol. **2**, pp. 367-373, 1989.
26. K.Y. Goldberg, and B.A. Pearlmutter, "Using backpropagation with temporal windows to learn the dynamics of the CMU direct-drive arm II", *Advances in Neural Information Processing Systems I*", D.S. Touretzky (ed.), pp. 356-

363, Morgan Haufmann Publishers Inc., Palo Alto CA, 1989.
27. F.J. Pineda, "Recurrent backpropagation and the dynamical approach to adaptive neural computation", *Neural Computation*, vol. **1**, pp. 161-172, 1989.
28. M. Caudell, and C. Butler, *Understanding Neural Networks: Computer Exploration, Advanced Networks,* vol. 2, pp. 79-112, The MIT Press, Cambridge MA, 1992.
29. K.S. Narendra, and K. Parthasarathy, "Identification and control of dynamical systems using Neural networks", *IEEE Transactions on Neural Networks*, vol. **1(1),** pp. 4-27, 1990.
30. T. Yabuta, and T. Yamada, "Neural network controller characteristics with regard to adaptive control", *IEEE Transactions on System, Man, and Cybernetics*", vol. **22(1)**, pp. 170-176, January 1992.
31. A. Guez, and J. Selinsky, "A trainable Neuromorphic Controller", *Journal of Robotic Systems*, Vol. **5(4)**, pp. 363-388, 1988.
32. S. Kung and J. Hwang, "Neural network architectures for robotic applications", *IEEE Transactions on Robotics and Automation*, vol. **5(5)**, pp. 641-657, February 1989.
33. F. Chen, "Back-propagation neural networks for nonlinear self-tuning adaptive control", *IEEE Control Systems Magazine*, pp. 44-48, April 1990.
34. P. Werbos, "Backpropagation through time: what it does and how it does it", *Proceedings of the IEEE*, vol. 78, pp. 1550-1560, 1990.
35. P. Werbos, "An overview of neural networks for control", *IEEE Control Systems Magazine*, vol. **11(1)**, pp. 40-42, January 1991.
36. M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement", *Biological Cybernetics*, vol. **57**, pp. 169-185, 1987.
37. H. Miyamoto, M. Kawato, T. Setoyama, and R. Suzuki, "Feedback error learning neural network model for trajectory control of a robotic manipulator", *Neural Networks*, vol**. 1**, pp. 251-265, 1988.
38. R.T. Newton, and Y. Xu, "Real-time implementation of neural network learning control of a flexible space manipulator", *IEEE International Conference on Robotics and Automation*, pp. 135-141, Atlanta Georgia, May 2-6, 1993.
39. F.L. Lewis, K. Liu, and Yesildirek, "Neural net robot controller with guaranteed tracking performance", *IEEE Transactions on Neural Networks*, vol. **6(3)**, pp. 703-715, May 1995.
40. M. Kawato, "Feedback-error-learning neural network for supervised motor learning", *Advanced Neural Computers*, R. Eckmiller (ed.), pp. 365-372, Elsevier Science Publishers B.V. (North-Holland), 1990.
41. D. Patino, R. Carelli, and B. Kuchen, "Stability analysis of neural networks based adaptive controllers for robot manipulators", *Proceedings of the American Control Conference*, pp. 609-613, Baltimore MD, June 1994.
42. W. Golnazarian, *Time-varying Neural Networks for Robot Trajectory Control*, Ph.D. Dissertation, University of Cincinnati OH, 1995.
43. P. Werbos, “Optimal neurocontrol: practical benefits, new results and biological evidence”, *Wescon Conference Record* 1995 p.580-585
44. P. Werbos, “Backpropagation and neurocontrol: a review and prospectus”, *IJCNN Int Jt Conf Neural Network* 1989 p.209-216
45. P. Werbos, “New directions in ACDs: key to intelligent control an understanding the brain”, *Proceedings of the International Joint Conference on Neural Networks* v 3 2000 p.61-66
46. B. Widrow, N. Gupta & S. Maitra, “Punish/reward: learning with a critic in adaptive threshold systems”, *IEEE Trans. Systems, Man, Cybemetics*, 1973, Vol.**5** p.455-465.
47. D. White & D. Sofge, *Handbook of Intelligent Control*, Van Nostrand, 1992
48. D.Prokhorov & D.Wunsch, “Adaptive critic designs”, *IEEE Trans. Neural Networks_*, Vol. **8**, No.5, p.997-1007, 1997.