Transformation Invariant On-Line Target Recognition

Khan M. Iftekharuddin, *Senior Member, IEEE*

***Abstract*— Transformation invariant automatic target recognition (ATR) has been an active research area due to its widespread applications in defense, robotics, medical imaging and geographic scene analysis. The primary goal for this paper is to obtain an on-line ATR system for targets in presence of image transformations, such as rotation, translation, scale and occlusion as well as resolution changes. We investigate biologically inspired adaptive critic design (ACD) neural network (NN) models for on-line learning of such transformations. We further exploit reinforcement learning (RL) in ACD framework to obtain transformation invariant ATR. We exploit two ACD designs, such as heuristic dynamic programming (HDP) and dual heuristic dynamic programming (DHP) to obtain transformation invariant ATR. We obtain extensive statistical evaluations of proposed on-line ATR networks using both simulated image transformations and real benchmark facial image database, UMIST, with pose variations. Our simulations show promising results for learning transformations in simulated images and authenticating out-of plane rotated face images. Comparing the two on-line ATR designs, HDP outperforms DHP in learning capability and robustness and is more tolerant to noise. The computational time involved in HDP is also less than that of DHP. On the other hand, DHP achieves a 100% success rate more frequently than HDP for individual targets, and the residual critic error in DHP is generally smaller than that of HDP. Mathematical analyses of both our RL-based on-line ATR designs are also obtained to provide a sufficient condition for asymptotic convergence in a statistical average sense.**

***Index Terms*— Active on-line learning, automatic target recognition, dual heuristic dynamic programming, face authentication, heuristic dynamic programming, image transformation invariance, reinforcement learning.**

I. Introduction

MODERN automatic target recognition (ATR) systems have witnessed an unprecedented growth in complexity over the last few decades. The ongoing advancements in increased computer processing capabilities, new understanding of low-level sensor phenomenology, improved signal processing and information fusion algorithms, and an expanded communication infrastructure have contributed to complexity of current ATR systems. The robust operation of a complex ATR system containing numerous, disparate image sensing and operating conditions necessitates efficient processing, optimization, and on-line learning [1], [2].

On the other hand, biological vision routinely accomplishes difficult sensing, processing and decision-making tasks in a holistic manner, enabling them to scale up to larger problems and implying efficient underlying mechanisms to process information. Among many sensory recognition functionalities that are performed by the visual system, we are primarily interested in transformations, such as rotation, translation, scale and resolution-invariant ATR. One of the nontrivial challenges that continues to persist is the lack of efficient on-line learning algorithms and heuristics to support transformation invariant ATR [2]–[5]. The unsurpassed recognition ability of a biological vision system in transient and nonlinear environments may become an important inspiration for the design of on-line ATR systems.

The parallel computing capacity of neural networks (NNs) makes it an effective tool for the implementation of transformation invariant ATR [3]–[5]. Among many different NN-based ATR implementations, wavelet NNs (WNNs) have recently attracted great interest. The WNNs are universal approximators, achieve faster convergence and are capable of dealing with the so-called curse of dimensionality more effectively [6]. However, in a classical training-testing setup using most conventional NNs the ATR performance largely depends on the range of transformations involved in training data [7]. A serious bottleneck in the on-line ATR application is that there may not be enough data available for the training of NN or even no training data at all. To alleviate this rather nontrivial problem, a biologically inspired reinforcement learning (RL) approach exploiting minimal training data for on-line training may be preferable for transformation invariant ATR [8]–[11]. Furthermore, traditional ATR requires that the target of interest must be known *a priori*. The RL can be an alternate approach toward breaking that barrier such that the model of the environment may not be needed *a priori* [12]. In RL approach, the learning is performed on-line while interacting with the environment [13]. Compared to conventional NN-based ATR systems, the on-line ATR with the RL ability may be closer to working principles of human visual system. Such dynamic system can adapt quickly in a changing and unpredictable environment. The RL has made major advancement to implementing the temporal difference (TD) learning methods [14]–[18]. The RL has been investigated successfully in various applications, such as neurocomputing and control [19]–[21]. On the other hand, RL has recently found limited applications in distortion related ATR [10]–[12].

Manuscript received December 24, 2010; accepted March 6, 2011. Date of publication May 13, 2011; date of current version June 2, 2011. This work was supported in part by the Herff Endowment Funds, the Whitaker Foundation under Grant TF-04-0026, and the National Science Foundation under Grant ECCS-0715116.

The author is with the Intelligent Systems and Image Processing Laboratory, Department of Electrical and Computer Engineering, University of Memphis, Memphis, TN 38152 USA (e-mail: iftekhar@memphis.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2011.2132737

Consequently, in this paper, we investigate an RL-based on-line ATR system for transformation-invariant ATR. We exploit a biologically plausible adaptive critic design (ACD) framework for implementing our RL-based on-line ATR system. An ACD approximates Bellman's cost function effectively using an action-critic network framework [22]. We exploit two specific ACD algorithms such as a heuristic dynamic programming (HDP) and a dual heuristic dynamic programming (DHP) in our design of the on-line ATR model. In HDP, the critic network estimates the cost-to-go function using Bellman equation of dynamic programming, while in DHP, the critic network estimates the derivative of the cost function with respect to the states of the system.

This paper is organized as follows. After a brief review of the theoretical background, the two implementations of the RL model for the ATR system are discussed in Section II, followed by the simulation results and numerical analysis in Section III. Next, Section IV provides the statistical performance evaluation of HDP and DHP using a benchmark face database. Finally, the conclusion is presented in Section V. In addition, the appendix addresses the mathematical characterization of the asymptotic convergence of the two algorithms under appropriate technical conditions.

A. Background Review

Our proposed RL-based transformation-invariant on-line ATR is partly inspired by the invariance processing in primate vision. We briefly review the relevant background in vision and the root of RL in visual processing. We then present a short review of major RL-based implementations, primarily in ACD framework.

B. Biology of Invariance Learning

The brain senses through many different modalities by extracting relevant patterns, such as shapes, sounds, odors, and so on, from a noisy, nonstationary and often, unpredictable environment [4], [5], [8]. The primate visual organs make use of retina which transform visual stimuli into nerve impulses representing tonic (static) and phasic (changing) temporal imagery. The three primary visual pathways, such as P, M, and K-koniocellular that terminate at the striate visual cortex (V1), process information collected in retina in parallel [23]. Neuropsychological findings suggest that invariance emerges in the hierarchy of visual areas in the primate visual system in a series of successive stages under the influence of the attentional window. One important aspect of invariant processing is 'feature binding' such that the local spatial arrangements of features are preserved. Further, extensive anatomical, physiological and theoretical evidences show that the cerebellum is a specialized organism for supervised learning, the basal ganglia are for RL and the cerebral cortex is for unsupervised learning for a rapid parallel forward recognition [24]. Evidence also exists that the outcome of the learning process may also drive a feedback controlled refinement loop all the way back to the retina [25].

In classical neuroscience, predictive learning occurs whenever a stimulus is paired with a reward or punishment. However, more recent analyses of associative learning argue that a 'prediction error' between the prediction stimulus and the actual reinforcer is also required [12]. Thus, the reinforcer pairings play an important role in human conditioning, causal learning, animal conditioning, and artificial neural learning. Striking similarities in teaching signals and learning behavior between the computational and biological studies suggest that dopamine-like adaptive-critic reward responses may serve as effective learning and higher-level perception in primate cognitive behavior [25]. There are various nonmathematical and mathematical theories to explain the role of prediction in RL. A few of these mathematical theories attempt to explain attention a consequence of RL in an ACD formulation [23].

C. ACD for Invariance Processing

The RL in the ATR implementation can be viewed from a control perspective as an optimal control problem of dynamic stochastic systems. Dynamic programming is identified as an effective technique for optimization in a stochastic and noisy environment. Dynamic programming can be traced back to Bellman [26] and Bertsekas [17]. However, the computational complexity of the dynamic programming can be enormous for many problems resulting in the so-called "curse of dimensionality." In addition, the choice of performance measure in dynamic programming is critical for achieving optimization [18]. Further, another difficulty involving dynamic programming is that a stochastic model of the system is often required [20]. To address these rather nontrivial challenges, the ACD has been explored as an effective tool to approximate dynamic programming in a generic environment. ACD is categorized into three main families such as HDP, DHP, and Globalized DHP (GDHP) [22].

The HDP is the most basic form of ACD and uses the parametric structure called actor (or action network) to approximate the policy, and another parametric structure called critic (or critic network) to approximate the cost function. The role of the critic can be compared with that of "reinforcement" function. The critic provides the actor with a performance measure or cost for its present actions by anticipating and evaluating future events. On the other hand, the DHP uses the critic to approximate the derivatives of the cost function with respect to the state. The actor is used to approximate the policy similar to the HDP. Each family has its action dependent (AD) form if the action network is directly connected to the critic network. AD forms of the three original designs are denoted by ADHDP, ADDHP, and ADGDHP, respectively. A more elaborate description on different types of ACD models, including HDP and DHP, can be found in [22].

II. Proposed Models

In this paper, we propose an ACD-based on-line ATR model for the recognition of objects in the presence of different transformations. Specifically, we investigate two models, such as HDP- and DHP-based techniques. In the RL/ACD setting, a critic network "critiques" the action value in order to optimize a future cost-to-go function and the action value is

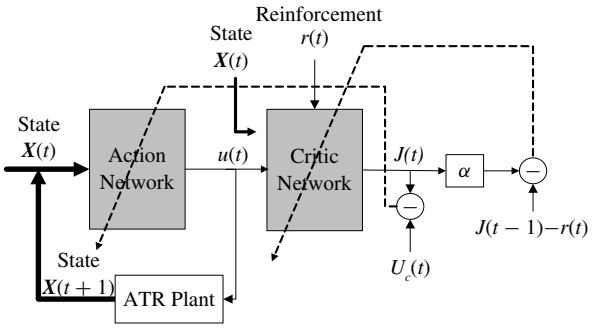

Fig. 1. Schematic diagram for the ATR system using HDP design.

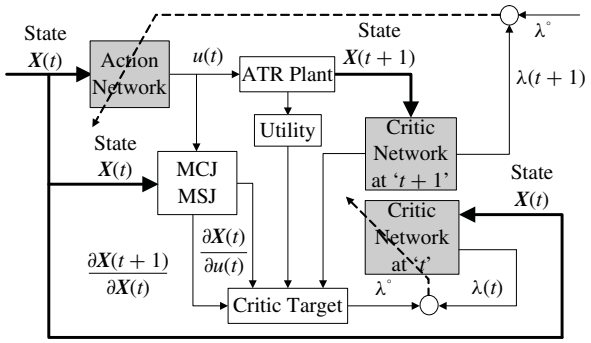

Fig. 2. Schematic diagram for the ATR system using DHP design.

determined on a step-by-step basis. Note step-by-step operation of the action network refers to iteration-by-iteration in our simulation. This critic and action network operation is consistent with on-line training strategy. In both ACD-based ATR implementations, our action output is directly connected to the critic network. Therefore, these implementations can be best categorized as ADHDP and ADDHP [22]. We discuss the major steps of the relevant network structures of HDP and DHP models for on-line ATR implementation below.

A. Network Structure for HDP-Based ATR

Fig. 1 provides a schematic diagram for the implementation of the HDP-based ATR system. The three major components in the HDP-based ATR are the action network, the critic network and the ATR plant. We discuss the ATR plant in a separate subsection below. The objective of the critic network is to approximate the total discounted cost-to-go function as given by

$$R(t) = r(t+1) + \alpha r(t+2) + \cdots \tag{1}$$

where $R(t)$ is the accumulative discounted future cost evaluated at time t, the function $r(t)$ is the reinforcement value at time t and α is a discount factor between 0 and 1. The weights of the critic network are adapted to obtain an approximate cost-to-go function $J^*(X(t))$ according to a modified Bellman equation [10], [20]

$$J^*(X(t)) = \min_{u(t)}\{J^*(X(t+1)) + g(X(t), X(t+1)) - U_0\} \tag{2}$$

where $g(X(t), X(t+1))$ is the immediate cost incurred by action $u(t)$ at time t, $X(t)$ is input state vector and U_0 is a heuristic constant term that is used only in infinite-time-horizon problems [27]. In this paper, we do not use U_0 since the on-line target recognition does not involve an infinite time-horizon. Note the output of the critic network, $J(t)$ in Fig. 1 [or, j^* in modified Bellman's (2)], approximates the total discounted cost-to-go function $R(t)$ in (1). For the critic network, its prediction error is defined as [10], [20]

$$e_c(t) = \alpha J(t) - [J(t-1) - r(t)]. \tag{3}$$

The objective function to be minimized in the critic network is [10], [20]

$$E_c(t) = \frac{1}{2}e_c^2(t) \tag{4}$$

while that in the action network is [10], [20]

$$E_a(t) = \frac{1}{2}e_\infty^2(t) = \frac{1}{2}\big[J(t) - U_c(t)\big]^2 \tag{5}$$

where $U_c(t)$ is defined as the desired ultimate objective.

Following the example in [20], we choose an artificial NN for the implementation of the action and critic networks. The specific structure of the NNs for both the action and the critic networks are implemented as a nonlinear multilayer feedforward NN. It consists of three layers of nodes, the input layer, the hidden layer and the output layer. The input and output nodes have linear activation, whereas the hidden layer has sigmoidal activation. For the critic network, the input into the network is the state vector $X(t)$ and the current action $u(t)$ at time t. The output $J(t)$ in Fig. 2 is given as [10], [20]

$$J(t) = \sum_{i=1}^{N_h} w_{c_i}^{(2)}(t)\, p_i(t) \tag{6}$$

where

$$p_i(t) = \frac{1 - e^{-q_i(t)}}{1 + e^{-q_i(t)}}, \qquad i = 1, 2, \ldots, N_h \tag{7}$$

and

$$q_i(t) = \sum_{j=1}^{n+1} w_{c_i}^{(1)}(t) x_j(t), \qquad i = 1, 2, \ldots, N_h \tag{8}$$

where q_i is the i-th hidden node input to the critic network, w_c is the critic weight, p_i is the i-th corresponding output of the hidden node, N_h is the number of hidden nodes in the critic network, and $n+1$ is the total number of inputs into the critic network and $x_{n+1}(t) = u(t)$.

The action network is implemented similarly by a feedforward artificial NN, the input to the NN is the n-dimensional state vector $X(t)$, and the output is the action value $u(t)$ at time t [10], [20]

$$u(t) = \frac{1 - \mathrm{e}^{-\nu(t)}}{1 + \mathrm{e}^{-\nu(t)}} \tag{9}$$

where

$$\nu(t) = \sum_{i=1}^{N_h} w_{a_i}^{(2)}(t)\, g_i(t) \tag{10}$$

$$g_i(t) = \frac{1 - e^{-h_i(t)}}{1 + e^{-h_i(t)}}, \qquad i = 1, 2, \ldots, N_h \tag{11}$$

and

$$h_i(t) = \sum_{i=1}^{n+1} w_{a_{ij}}^{(1)}(t) x_j(t), \qquad i = 1, 2, \ldots, N_h \tag{12}$$

where v, g, and h are input or output variables in the NN and w_a is the actor weight. Specifically, v is the input to the action node, g_i is the output to the i-th hidden node, and h_i is the input to the i-th hidden node. The detail weight update rules for both critic and action networks as well as training with back propagation are discussed in [20].

B. Network Structure for DHP-Based ATR

Compared to HDP, DHP is an incremental RL algorithm. Fig. 2 shows the schematic diagram for DHP-based ATR system. The state vector $\mathbf{X}(t)$ describes the state dynamics of the ATR plant, and it is an input to both the action and critic networks. The output of the critic is the derivative of the cost function. The 'utility' box generates the utility function and its derivatives and is used in the calculation of target value for the critic network. The critic network also involves computing of model critic Jacobian (MCJ) and model state Jacobian (MSJ) matrices, respectively. The role of the critic network is to supply a performance measure for action network's output, which in this case is the derivative of the cost function, denoted as λ. Comparison of various techniques to train the critic and action networks can be found in [28]–[31].

The ACD network of DHP is similar to that of the HDP. In implementation of DHP, the critic network approximates the derivative of cost function with respect to plant state. For DHP, the objective of the critic network is to approximate the discounted total cost-to-go function $R(t)$. We re-write $R(t)$ in (1) as follows:

$$R(t) = U(t+1) + \alpha U(t+2) + \cdots \tag{13}$$

where $U(t)$ is the utility function generated at time t and α is a discount factor between 0 and 1. The prediction error of the critic network is defined as [11], [22]

$$e_c(t) = \frac{\partial J(t)}{\partial X(t)} - \alpha \frac{\partial J(t+1)}{\partial X(t)} - \frac{\partial U(t)}{\partial X(t)} \tag{14}$$

where each term is defined as above and the critic error to be minimized is given as

$$E_c(t) = \sum e_c(t) e_c^T(t). \tag{15}$$

In training the network, a target value is generated for the output of the critic network and is given as [11]

$$\lambda^\circ(t) = \frac{\partial J(t)}{\partial X(t)} = \frac{\partial}{\partial X(t)}(U(t) + J(t+1)). \tag{16}$$

Application of chain rule yields [11]

$$\lambda^\circ(t) = \frac{\partial U(t)}{\partial X(t)} + \alpha \frac{\partial J(t+1)}{\partial X(t+1)} \times \left[\frac{\partial X_s(t+1)}{\partial X(t)} + \frac{\partial X(t+1)}{\partial u(t)} \frac{\partial u(t)}{\partial X(t)} \right]. \tag{17}$$

The first term is the derivative of the utility function $U(t)$ with respect to the current state vector $X(t)$. This term $\partial u(t)/\partial X(t)$ can be obtained through back propagation from the action network. The second term $\partial J(t+1)/\partial X(t+1)$ is calculated by passing the next state through the critic network. It can be represented as $\lambda(t+1)$ as shown in Fig. 2, i.e., the output of the critic network at $t+1$ is used to train the network at t. The weight update rules for critic and action networks for DHP as well as training with back propagation are discussed in [32].

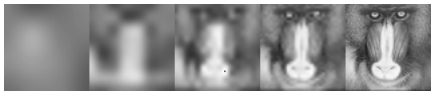

Fig. 3. Image bank for multiresolution images.

C. ATR Plant

In our proposed on-line HDP- and DHP-based systems, the ATR plant is the primary recognition element. Note unlike conventional NN, such as a WNN [6], our ATR performs on-line training alleviating the need for large training sample images. In general, an ATR consists of three major components, such as the sensory/data acquisition, the pre-processing/feature extraction and the identification/recognition. The selection of ACD structure as the identification component in the ATR system is in part a continuation of our long-term research goal to building a dynamic biologically inspired distortion invariant ATR system [8]–[11]. This RL/ACD type ATR implementation follows inspiration from earlier works reported by Werbos [27] and Barto [13]. In these works, the authors emphasize the prospect of ACD to duplicate the important aspect of human intelligence which is capable of coping with large number of variables simultaneously in real time within a nonlinear and noisy environment [9].

Our proposed on-line ATR plant simulates different types of transformations in sensor images, such as multiresolution, rotation, scaling, and translation. The plants in both Figs. 1 and 2 are simple ATR systems that either update the states of the system or maintain the same state according to the action decision made in the subsequent action network based on the RL signal. If the action value $u(t)$ is less than or equal to 0, the plant state $X(t+1)$ proceeds one step forward to the next state $i+1$, otherwise, it maintains its current state i. The reinforcement signal $r(t)$ takes a simple binary form of '0' and '-1' corresponding to 'reward' and 'punishment.' Note there is no need for extensive training of the network following the on-line learning paradigm. Furthermore, we do not pre-process the images, rather each raw image belonging to the corresponding state is used for recognition application as discussed below.

Furthermore, for DHP-based ATR implementation, as shown in Fig. 3, the computation of MCJ and MSJ terms may require a model of the plant [28]. The elements of the state Jacobian matrix, i.e., $\partial X(t+1)/\partial X(t)$, can be calculated from the plant model. In case the model of the plant is not present, they can be calculated by training a separate NN to simulate the plant and by back propagation [29]. In this paper, we approximate the derivatives of the state vector

as follows:

$$\frac{\partial X(t+1)}{\partial X(t)} \approx \frac{X(t+1)-X(t)}{X(t)-X(t-1)}. \quad (18)$$

In this paper, the ATR plant has been modeled to handle a single image distortion at a time. Therefore, the computation of Jacobian matrix using a difference operator as shown in (18) is justified in our application. Note such simplicity in the MCJ and MSJ computations also stems from the fact that our simple ATR model involves target selection from one distorted image to the next as discussed above. We now discuss one such image distortion for resolution change below.

For an example of state transition in the ATR plan, we discuss the multiresolution case. In an ATR system these multiresolution images can be considered as the incoming unknown test images with only one varying parameter, that is, the resolution. The multiresolution images are obtained using wavelet transform as follows [33]. 2-D transform requires a scaling function $\varphi(x, y)$ and its corresponding wavelet ψ. If the scaling function $\varphi(x, y)$ is assumed to be separable $\varphi(x, y) = \varphi(x)\varphi(y)$, the cross multiplication of scaling and the corresponding wavelet produce three other wavelet functions $\psi^H(x, y)$, $\psi^V(x, y)$, and $\psi^D(x, y)$. These functions describe variations along horizontal, vertical and diagonal directions of the transformed image and are given by

$$\psi^H(x, y) = \psi(x)\varphi(y)$$
$$\psi^V(x, y) = \varphi(x)\psi(y)$$

and

$$\psi^D(x, y) = \psi(x)\psi(x). \quad (19)$$

The corresponding dyadic scaling and wavelet functions are given as

$$\varphi_{d,m,n}(x, y) = 2^{\frac{d}{2}}\varphi(2^d x - m, 2^d y - n) \quad (20)$$
$$\psi^i_{d,m,n}(x, y) = 2^{\frac{d}{2}}\varphi^i(2^d x - m, 2^d y - n), i = \{H, V, D\} \quad (21)$$

where superscript i of H, V, and D stand for directional notation in horizontal, vertical and diagonal directions, d indicates the index of dilation scale, while m and n indicates the translation parameters in the 2-D space. Then the discrete 2-D wavelet decomposition of image $f(x, y)$ of size $M \times N$ is given by

$$W_\varphi(d_0, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)\varphi_{d_0,m,n}(x, y) \quad (22)$$

with d_0 is any arbitrary initial scale, and the scaling decomposition of the same image is given as

$$W^i_\psi(d, m, n) = \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)\psi^i_{d,m,n}(x, y),$$
$$i = \{H, V, D\}. \quad (23)$$

The transform product is quadrants starting from upper left to right and lower left to right as the approximation W_φ, horizontal W^H_ψ, vertical W^V_ψ, and diagonal W^D_ψ images. Using (21) and (22) the original image $f(x, y)$ can be obtained as

$$f(x, y) = \frac{1}{\sqrt{MN}} \left[\sum_{X=0}^{M-1} \sum_{y=0}^{N-1} W_\varphi(d_0, m, n)\varphi_{d_0,m,n}(x, y) + \sum_{i=H,V,D} \sum_{j=d_0}^{\infty} \sum_{m} \sum_{n} W^i_\psi(d, m, n)\psi^i_{d,m,n}(x, y) \right]. \quad (24)$$

In our multiresolution ATR implementation, 2-D discrete wavelet function Daubechies level 4 is used to decompose the image. Reconstruction of approximation image W_φ of increasingly higher resolution levels forms the multiresolution image set as a state of the ATR plant. For example multiresolution images are shown in Fig. 4 and the images are denoted as sub-image 1–5. At time t, the sub-image as current state $X(t)$, serves as input to action and critic network. The implementation of the remaining transformations, such as rotation, translation, scaling and occlusion, are straightforward and the methods are discussed in the following section.

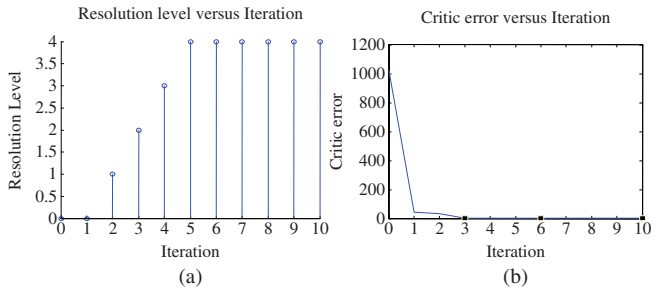

Fig. 4. HDP learning for resolution changes. (a) Resolution level versus iteration. (b) Critic error.

III. Results and Discussions

In this section, we discuss the performance of our transformation invariant HDP and DHP-based on-line ATR systems. For both the ATR systems, the following selections of parameters apply. The initial critic and action network learning rates $l_c(0)$ and $l_a(0)$ are both set to 0.25, and these parameters decrease as $l_c(t) = (1/t)l_c(0)$ and $l_a(t) = (1/t)l_a(0)$, respectively. The numbers of nodes in the hidden layer of both active and critic networks are set to 6. The discount factor α is chosen as 0.95. Some of the sample runs are demonstrated in this section. Note these parameters are selected on trial and error basis, however, we provide detail analysis on the choice of some of these parameters in Section 3.5.

A. Multiresolution

The images at different resolution levels are constructed from the approximation image at each level, and used in the ATR system as the state. The multiresolution image bank consists of five images of gradually increasing resolution level, as shown in Fig. 3.

In Fig. 3 the resolutions are numbered from 0, for the lowest resolution, to 4, for the highest one. The image at the highest

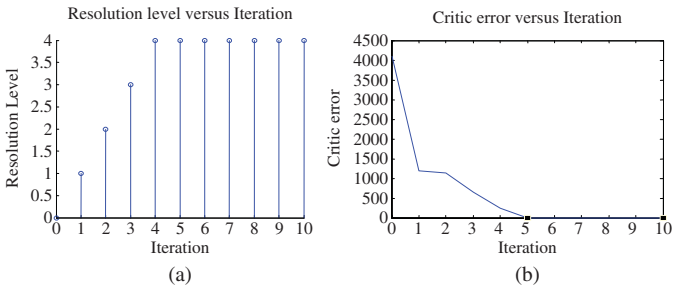

Fig. 5. DHP learning for resolution changes. (a) Resolution level versus iteration. (b) Critic error.

resolution serves as the target to the system. The initial state of the system is at resolution level 0, and the state dynamics of the system changes according to the action value from the action network $u(t)$ throughout the learning process. Figs. 4 and 5 show a typical run of HDP and DHP for the recognition of multiresolution images. The change of resolution level and the critic error are plotted as functions of iteration number. Note the error critics are obtained using (4) and (15) for HDP and DHP ATR networks.

In both learning algorithms, as the iteration progresses, the network increases resolution level and maintains the states after it reaches the target level. In both cases, the ACD networks have successfully recognized the image and locked the target once it accomplishes successful learning. Fig. 4(a) shows that in this run of HDP design, the system takes five steps to recognize the highest resolution level, while in the DHP run illustrated in Fig. 5(a), it takes four steps to reach the reference level. A summary of statistical comparison for the HDP and DHP-based learning algorithms is provided in Section IV. The optimization process of the system can be illustrated by the dynamics of the critic error as it changes with the learning steps. The critic error for HDP and DHP are shown in Figs. 4(b) and 5(b). In both plots, the initial error is very high, and the critic error decreases rapidly once the system starts to learn. In both designs, the critic errors decrease monotonously during the learning process, and approach minimum once the system recognizes the target. In the HDP case, the residual critic error is $E_c = 0.0075$ even after the system locks the target, while that in the DHP, the critic error goes to zero and remains zero thereafter.

B. Translation and Scale

In ATR, target image translation and scale are some of the known artifacts. In our simulation, we study image translation and scale separately as shown in Figs. 6 and 9. Fig. 6 shows the image bank composed of five images with translation of 40%, 30%, 20%, and 10% of the image width from left to right. The original image (with no translation) serves as the target image. The images are numbered 0–4 from left to right.

Figs. 7 and 8 show the corresponding HDP and DHP results for translation case. In HDP algorithms, the learning starts are iteration 2, and then it goes backwards to its initial state in the steps afterwards. The critic network continues an effort to minimize the critic error until it reaches a local

Fig. 6. Image bank for translation.

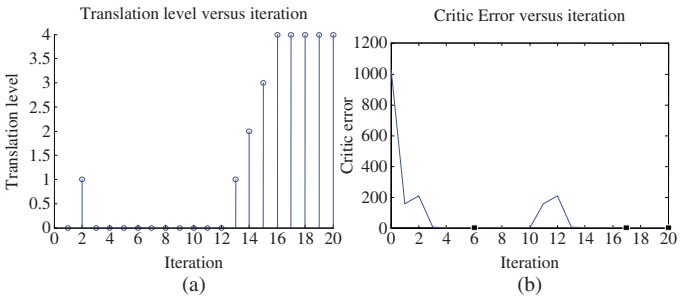

Fig. 7. HDP learning of translating object. (a) Translation level. (b) Critic error.

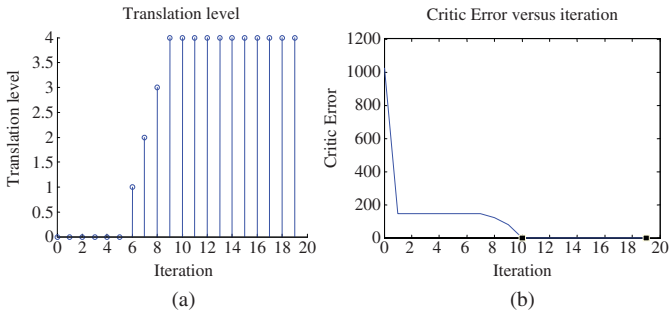

Fig. 8. DHP learning of translating object. (a) Translation level. (b) Critic error.

Fig. 9. Image bank for scaling image.

minimum at iteration 6. At this stage, the network is stuck at a local minimum. In iteration 10, the system gets out of the local minimum, restarts the learning process at iteration 12, performs a successful learning in the second trial, and reaches the global minimum when the system locks the target. This example in Fig. 7 shows that the system has a capability of getting out of a local minimum and proceeding to the global optimization in the learning process. Note we employ same RL mechanism as discussed above in this example. In Fig. 8, the system is stuck in a local minimum until iteration 6, and the learning process starts.

The image bank for the scaling case is shown in Fig. 9 the original image with five different scale levels, 2, 1.8, 1.6, 1.4, 1.2, and 1.0. The images are numbered 0–4 from left to right, with image of the original size (with a scaling factor 1.0) serves as the target. The image bank includes five images constructed by scaling. In both learning algorithms, the ACD networks are able to recognize the image with scale variations and lock the target once it accomplishes successful learning. In the HDP design, the critic error shows some fluctuation in

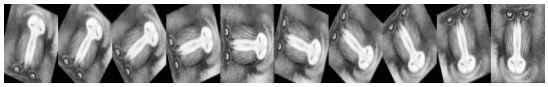

Fig. 10. Image bank for in plane rotation up to 180°.

Fig. 11. Image bank with occlusion.

the learning process and a small residual error exists even after the system locks the target, while in the DHP, the critic error decreases monotonously and goes to zero when the learning task is successfully accomplished.

C. Rotation and Occlusion

Recognition of target with pose variations remains a challenging problem in ATR. One category of the pose variations can be modeled by the in-plane rotation of a 2-D image. In our experiment, 180° in-plane rotation is considered as shown in Fig. 10.

In Fig. 10, the rotation levels are numbered from 0 for the leftmost image to 9 for the rightmost image, and the rightmost image is also the target for the system. The initial state of the system is at rotation level 0, and both the HDP- and DHP-based ATR networks are explored to recognize the target image at rotation level 9. As the iteration progresses, the state of the network changes closer to the target and locks the target once the target is found. In both designs, once the system recognizes the target, the critic error converges to the minimum. In the HDP case, there is some very small residual critic error even after the system locks the target, while in the DHP case, the critic error goes to zero and remains zero thereafter.

The image bank for the occlusion case is shown in Fig. 11. It is composed of five images with different levels of occlusion, such as 40%, 30%, 20%, 10%, and 0% of the whole image area. The original image with no occlusion serves as the target. The images are also numbered 0–4 from left to right. In the HDP design, a small residual critic error exists after the system locks the target, while in the DHP, the critic error decreases monotonously and goes to zero when the learning task is successfully accomplished.

IV. Statistical Evaluation of HDP- and DHP-Based ATR Designs

In this section, we provide an overall statistical evaluation of our ATR systems using the above synthetic transformations as well as real pose variations in face datasets.

A. Evaluation of HDP- and DHP-Based Systems Using Simulated Transformations

To compare the HDP- and DHP-based systems statistically, two batch learning experiments are performed. The experiment is set up as follows. For each individual transformation, each batch learning program consists of 100 runs. In each run, either

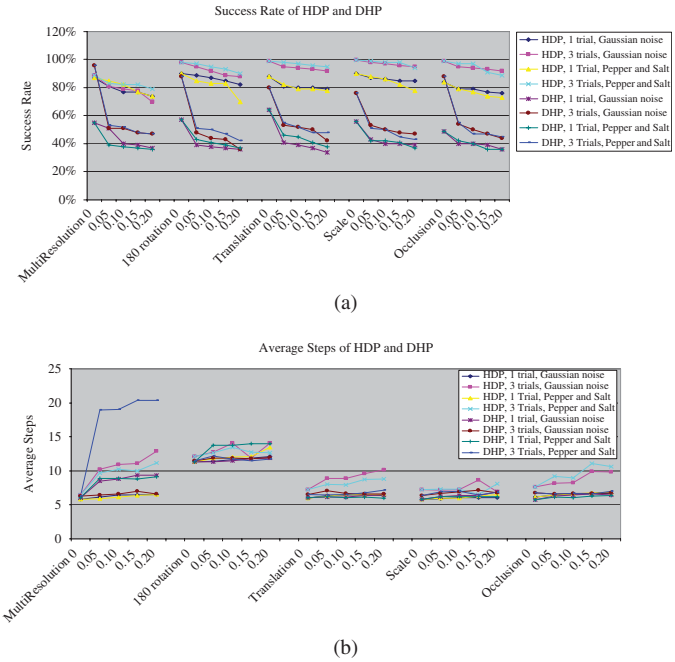

Fig. 12. Comparison of HDP- and DHP-based systems with noise. (a) Success rate versus noise variance for different distortions. (b) Average steps versus noise variance for different distortions.

one single trial, or a maximum of three consecutive trials are permitted. A run is considered successful if it can learn and lock the target within admissible trials. The ATR system is considered to lock the target if it can maintain the state in the target level for at least five steps after recognizing the target within the last admissible trial.

In 100 runs, the network weights are set to random initial conditions at the beginning of each individual run, learning task is recorded, and the average number of steps. If a run is successful, the number of steps it takes to perform corresponds to the one averaged over all the successful runs. We also define success rate as the percentage of successful runs out of 100. Note the success rate is influenced by the critic error among other factors. When three trials are attempted within one run, the number of trials it takes to perform such a successful run is also recorded. The average number of trials is then obtained by averaging over all the successful runs. The success rate can provide a meaningful insight into the precision of a learning system, and the average number of steps to successfully perform a task can indicate how fast a system can learn. We summarize the average iterations (steps) needed to obtain success rates for both HDP and DHP in Table I.

Table I presents that given three trials, the success rates across HDP and DHP are all improved compared to one trial cases. At the same time, the average steps taken in a successful run are also increased. This observation confirms that given more chances, the system can learn better, however, more computational resources are also needed in the process. Note for rotation transformation, the number of steps needed is always the highest for both networks suggesting that rotation is the most difficult among the distortions studied in this paper. The reason is that rotation introduces the most dissimilarity resulting in the hardest challenge for the ATR systems.

TABLE I
OVERALL SUCCESS RATES AND AVERAGE STEPS

Algorithm	Transformation	One trial		Three trials		
		Success Rate	Average Steps	Success Rate	Average Trials	Average Steps
HDP	Multiresolution	84%	6.14	89%	1.01	6.42
	180° rotation	84%	11.96	93%	1.29	13.78
	Translation	79%	6.41	97%	1.22	7.15
	Scale	87%	5.97	99%	1.16	6.75
	Occlusion	83%	6.20	97%	1.27	7.73
	Average	**83%**	**7.34**	**95%**	**1.19**	**8.37**
DHP	Multiresolution	55%	6.04	96%	1.80	6.28
	180° rotation	57%	11.30	88%	2.10	11.48
	Translation	64%	6.07	80%	2.06	6.55
	Scale	56%	5.82	76%	2.18	6.34
	Occlusion	48%	5.71	88%	2.15	6.74
	Average	**56%**	**6.99**	**86%**	**2.06**	**7.48**

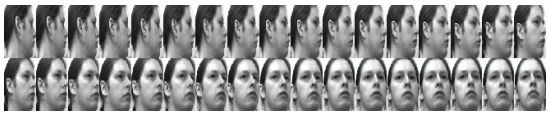

Fig. 13. Example image bank for subject 1a [25].

To demonstrate the robustness of the algorithms, two additive noises, e.g., the Gaussian noise and pepper-and-salt noise are considered. In the simulation, four different noise levels are considered for each transformation cases. For Gaussian noise, the zero-mean Gaussian noise with standard deviations $\sigma = 0.05, 0.10, 0.15$, and 0.20 are added to the images to be recognized. For pepper-and-salt noise, the different noise level is applied by adding pepper-and-salt noise with different densities $d = 0.05, 0.10, 0.15$, and 0.20. Note the corresponding normalized standard deviation of the image is 0.3021. The changes of the success rate versus the different noise level, and the average number of steps to perform a successful task, are plotted in Fig. 12.

In Fig. 12, note that when given three trials, the success rates of both HDP and DHP are improved compared to one-trial cases, at the same time, it also takes more steps on average to finish a successful run. It confirms that given more chances, the system can learn better, but will also consume more computational resources in the process. Comparing the success rate of HDP and DHP across different transformations, although the precision can be improved by applying more trials within one run, the success rate of HDP is higher than that of DHP in general. Therefore, HDP demonstrates overall better precision than DHP.

The learning capability of the algorithms is characterized by the number of steps to successfully perform a recognition task. Comparing the average number of steps to perform a successful run, in the cases of the multi-resolution, translation and occlusion, the average number of steps in DHP is less than that of HDP, for 180° in-plane rotation and scale, there is no noticeable difference. In Fig. 12, both HDP and DHP degrade in performance in presence of noise. However, DHP demonstrates more sensitivity to noise. The low accuracy manifested by the low success rate may make a DHP-based ATR system unattractive, even impractical in some scenarios. In summary, the robustness of the HDP algorithm is better than that of DHP in a noisy environment.

B. Evaluation of HDP and DHP-Based Systems Using UMIST Face Database

We further evaluate our HDP- and DHP-based ATR designs for face authentication using out-of-plane rotation of facial images in the UMIST face database [34]. Precise and fast face authentication is critical in applications involving personal identification and verification. A comprehensive discussion on face authentication using 2-D data can be found in [35]. The UMIST database consists of 2-D face data from 20 subjects, each covering a range of poses from profile to frontal views. The subjects are labeled as '1a', '1b' through '1t'. The original face images are approximately 220×220 pixels in 256 gray levels [34]. The face images are cropped into a uniform size of 112×92 pixels. In order to manage the face dataset better (i.e., to make the data arrangement experiment, we group the sequence in a pattern of 90° rotation as follows: the sequences with a rotation of more than 90° are truncated, while the same with a 180° rotation are split and rearranged into two subsequences of 90° rotation from profile view to frontal view. For example, the two subsequences of subject 'g', subject 'j' and subject 'n' are obtained in this fashion. The resulting image arrays are then sorted in a 90° rotation from profile view to frontal view before it is supplied to the ATR plant as system state. The image sequence used for subject '1a', for an example, is shown in Fig. 13.

Similar to synthetic transformation case above, for both algorithms, two batch simulations are performed, each containing 100 runs. The experiment results for a few example subjects in the UMIST database are shown in Figs. 14 and 15. Fig. 14 shows that given three trials, the success rate for both HDP and DHP algorithms improve.

In Fig. 14, the number of steps for both HDP and DHP to perform a successful learning is also increased over time. Comparing the success rates of HDP and DHP, taking both

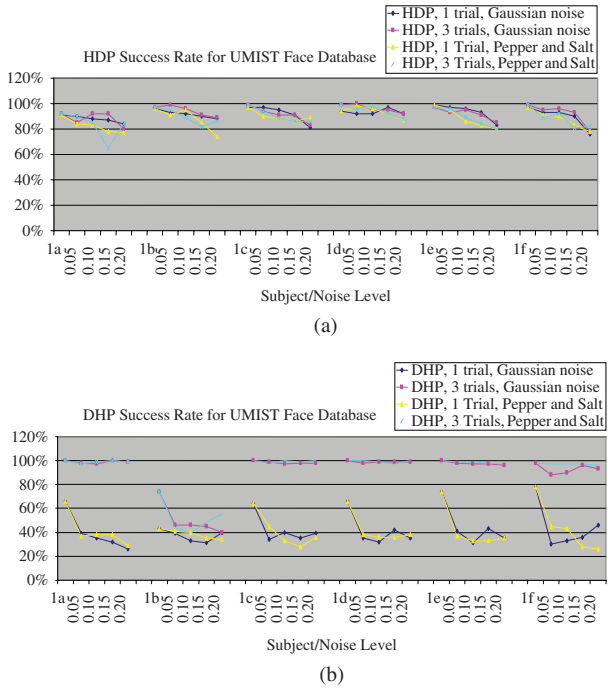

Fig. 14. Success rate versus noise variance for a few example subjects in UMIST face database, subject (a) 1a–1f using HDP and (b) 1a–1f using DHP.

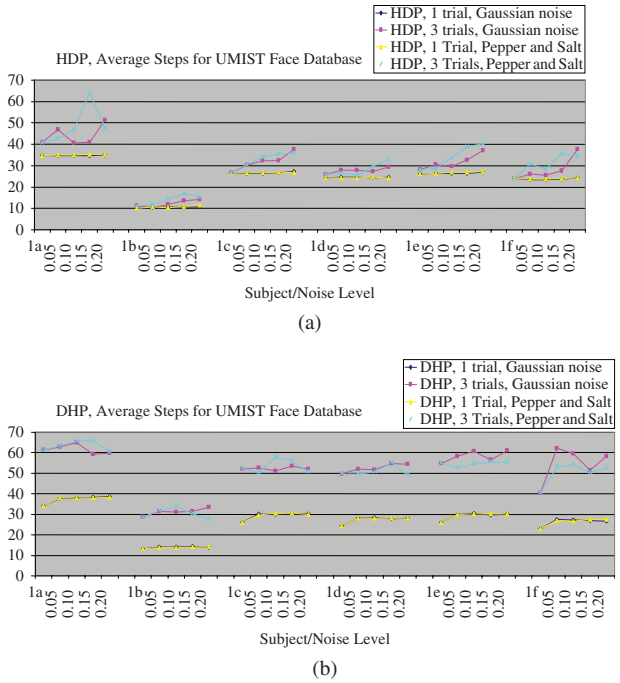

Fig. 15. Average steps versus noise variance for a few example subjects in UMIST face database, subject (a) 1a–1f using HDP and (b) 1a–1f using DHP.

one-trial and three-trial simulations into consideration, we observe that the success rate of DHP improves faster over that of HDP. However, DHP also demands more computational time for such an improvement. Comparing the HDP and HDP, the overall success rates of HDP across all different subjects in the UMIST face database are higher. On the other hand, DHP-based ATR achieves a 100% success rate more often (higher frequency) for individual subjects when there is no noise, as shown in Fig. 15. Therefore, overall HDP-based system learns faster than DHP throughout the variation of subject data. However, DHP-based system achieves more frequent 100% accurate face authentication for more of the subjects in the database. This demonstrates a trade-off in learning between precision in general and accuracy in particular for this example case.

Comparing the HDP and DHP number of steps needed for all the subjects in our face database to perform a successful face authentication task, performance varies among different subjects. For the overall simulations with the face database, the average number of steps needed for HDP to perform a successful learning is smaller than that of DHP. The derivative computation in DHP learning generally offers more precision when there is no noise or uncertainty, however, it also makes DHP more sensitive to noise and unpredictability in the environment. Consequently, DHP has more potential to improve the system performance, but the resources demanded for such an improvement is also increased. Furthermore, the stochastic nature of the real data may pose a challenge on the performance of DHP learning.

In summary, HDP outperforms DHP in learning capability, and also demonstrates better robustness when applied in a stochastic and uncertain environment. On the other hand, DHP provides more 100% precision across different subjects, therefore offers more precision and accuracy in recognition and authentication. In general, the computational time involved in DHP is more than HDP.

V. Conclusion

In this paper, we proposed transformation invariant on-line training and ATR using RL. Implementation of two ACD-based learning designs, such as HDP and DHP, demonstrate that RL is an effective approach for on-line transformation invariant ATR. Comparing the two designs, HDP outperforms DHP in precision, learning capability and robustness when applied to wide variations of transformations. Mathematical analysis of both the RL-based ATR algorithms provides a sufficient condition for asymptotic convergence in a statistical average sense, and an upper bound for the optimization error of the cost-to-go function is also obtained.

Our work using HDP- and DHP-based on-line ATR inherently offers advantage over conventional NN-based ATR [6], [36]–[38]. Conventional NN-based ATR systems have traditionally been computationally expensive since such networks need large training samples to learn and also do not implement RL. In addition, these ATR systems often times require cleverly crafted pre-processing and feature extraction on image data [39], [40]. In general, the on-line pose estimation and tracking techniques involve complex frontend feature extraction, multiple views of distortion related images and variants of Kalman filtering (KF) techniques [41], [42]. These algorithms are computationally involved and require estimation of the priors to compute the associated KF. Furthermore, from a regression perspective, it is impossible to predict individual ATR outcomes for all possible image transformations since

samples are only available for small subsets of the total space. Given this limitation, and absent a purely theoretical model meaningfully matched to the true complexity of this problem, the authors in [43] examine the empirical behavior of scores across various distortions, and identify trends and characteristics of the scores that are apparently predictable. Therefore, these and other similar works do not obtain specific on-line ATR for distorted objects.

In comparison, this paper attempted to implement a less computationally involved ATR algorithm such that we do not need expensive and large number of training samples to train our network. We also do not require any pre-processing, frontend feature extraction of input image data and generation of prior models. Furthermore, the supervised RL in our system is amenable to handling on-line affine related transformation invariant ATR. Directions for future work include ATR of different combinations of various transformations as well as feature-based ATR. Finally, a more robust and realistic formulation of the on-line ATR plant that simulates multiple distortions simultaneously and facilitates unsupervised RL may also be warranted. Note design and implementation of such complex plant systems will require appropriate formulation and training of the ATR models following examples in control and power systems [28], [44].

APPENDIX

A. Asymptotic Convergence of HDP and DHP for ATR

In this section, we discuss analytical characteristics of the action and critic networks used in our HDP- and DHP-based on-line ATR systems. Convergence analyses of similar dynamic programming algorithm examples have been obtained under certain conditions [13], [15], [16], [20]. In [20], the sufficient condition in the Robbins-Monroe algorithm is applied in the context of RL to obtain asymptotic convergence of the action and critic networks in a statistical average sense, assuming that the dynamics of the action and critic networks can be observed independently. On the other hand, the idea of value function approximation is applied to analyze gradient method in the action-critic network [13], [15], [16]. In value function approximation setting, the critic network can be seen as a parameterized representation of the cost function, the action network acts as a parameterized controller whose output is involved in changing the dynamics of the system. Both HDP and DHP designs can be seen as a special case of the TD learning TD(λ) when the parameter $\lambda = [13], [15], [16]$, only the objective functions to approximate are different. Therefore, the convergence results in [13], [15], and [16] can be used to the ACD structures in this paper.

Consider the ATR plant as a discrete time dynamic system as follows:

$$X(t+1) = f(X(t), u(t)) \tag{A1}$$

where $u(t)$ is a control decision generated by the action network. Denote the finite state and control spaces by $\mathbb{X}$ and $\mathbb{U}$, respectively, the reward function $r : \mathbb{X} \times \mathbb{U} \to \mathbb{R}$ associates a reward $r(X(t), u(t))$ with a control decision $u(t)$ taken at state $X(t)$. The action network provides a mapping $\phi : \mathbb{X} \to \mathbb{U}$ that generates state dependent decisions.

1) Convergence Assumption of TD Learning in Markov Reward Process: To characterize the asymptotic convergence of the critic network, the analysis for the convergence of TD learning for Markov decision process as proposed in [15] and [16] is followed. Consider a discrete time, finite-state Markov chain $\{x_n\}$, whose transition probability depends on a parameter vector $\theta \in \mathbb{R}^k$, and are denoted by

$$t_{ij}(\theta) = P\Big(x_n = j | x_{n-1} = i, \theta\Big). \tag{A2}$$

If a Markov chain satisfies the following properties [16].

1) The Markov chain corresponding to every $P \in \bar{\mathcal{P}}$ is aperiodic, and there exist a recurrent state i^* for every such Markov chain.
2) For every pair of states $i, j \in \mathbb{X}$, the functions $t_{ij}(\theta)$ and $r_i(\theta)$ are bounded, twice differentiable, and have bounded first and second derivatives.
3) For any pair of states $i, j \in \mathbb{X}$, there exists a bounded function $L_{ij}(\theta)$ such that $\nabla t_{ij}(\theta) = t_{ij}(\theta) L_{ij}(\theta)$ for all θ.
4) The step size is a sequence of positive numbers which satisfies the following:
   a) $\lim_{t\to\infty} l(t) = 0$;
   b) $\sum_{t=1}^{\infty} l(t) = \infty$;
   c) $\sum_{t=1}^{\infty} l^2(t) < \infty$.

Then $\lim_{t\to\infty}(\partial J/\partial\theta) = 0$, and the iteration, $\theta(t+1) = \theta(t) + l(t) e_c(t)(\partial J/\partial\theta)$ converges asymptotically with probability 1.

2) Convergence of HDP-Based ATR: Consider the ATR plant as a discrete time dynamic system as shown in (1). The state of the system $X(t)$ evolves as a Markov process with the following parameterized transition matrices:

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}_{m\times m} \quad \text{for} \quad u < 0,$$

and

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \ddots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}_{m\times m} \quad \text{for} \quad u > 0 \tag{A3}$$

where $m = \text{card}(X)$ is the number of states in the system. Note all the parameters in the Appendix are the same as defined in the main body unless otherwise noted. The objective of the critic network is to approximate the cost-to-go function $R(t)$ as given in (1) by

$$R(t) = r(t+1) + \alpha r(t+2) + \cdots \tag{A4}$$

which is a discounted total of the reward. The binary reward function in our simulation is defined as

$$r(t) = \begin{cases} 0, & : X = X_r \\ -1, & : \text{otherwise} \end{cases} \tag{A5}$$

where X_r is the reference state of the system. For each policy ϕ, the output of the critic network $J(t)$ provides an approximation to $R(t)$. Let us define

$$J(X,\phi,t) = \mathbb{E}\left[\sum_{i=0}^{\infty} \alpha^i r(X(t+1+i) \phi(t+1+i))\big|X(t)=X\right] = \mathbb{E}\Big[R(X,\phi,t)|X(t)=X\Big] \quad \text{(A6)}$$

where $\mathbb{E}$ is the mathematical expectation. The optimal cost function is defined by

$$J^*(X,t) = \max_{\phi} J(X,\phi,t). \quad \text{(A7)}$$

From the standard dynamic programming computation, the optimal control policy is given as

$$\phi^*(X,t) = \arg\max_{u\in\mathbb{U}} \left[r(X(t),u(t)) + \alpha J^*(X,\phi,t)\right] \quad \text{(A8)}$$

with

$$J^*(X,t) = J(X,\phi^*,t). \quad \text{(A9)}$$

In our proposed HDP design, the output of the critic network is a linear parameterization of the cost function and given as

$$J(t) = \sum_{i=1}^{N_h} w_{c_i}^{(2)}(t) p_i(t). \quad \text{(A10)}$$

The approximation $J(t)$ is represented by a set of basis functions $p_i(t)$, and the bases are also parameterizations as the following:

$$p_i(t) = \frac{1-\mathrm{e}^{-q_i}(t)}{1+\mathrm{e}^{-q_i}(t)} \quad \text{(A11)}$$

and

$$q_i(t) = \sum_{j=1}^{n+1} w_{c_i}^{(1)}(t) x_j(t). \quad \text{(A12)}$$

The critic error is defined as

$$e_c(t) = r(t) + \alpha J(t) - J(t-1). \quad \text{(A13)}$$

The updating rule

$$\mathbf{\Delta} W_c(t) = -l_c(t)\alpha e_c(t)\frac{\partial J(t)}{\partial W_c(t)} \quad \text{(A14)}$$

can be seen as a special case of the TD learning TD(λ) when the parameter $\lambda = 0$. The gradient $\partial J(t)/\partial W_c(t)$ can be viewed to provide a direction for the updating of $W_c(t)$ such that the previous approximation $J(t-1)$ moves toward a more desirable prediction $\alpha J(t) + r(t)$.

For the ATR setup in our simulation, the transition matrix P is aperiodic, and the transition probabilities $t_{ij}(t)$ and reinforcement $r_i(t)$ are functions of the state $X(t)$, which are not an explicit function of $W_c(t)$, therefore, Conditions 1–3 in Section A.1.1 are satisfied. Let us choose the learning rate sequence $l_c(t)$ that satisfies Condition 4 in Section A.1.1, we have $\lim_{t\to\infty}(\partial J(t)/\partial \mathbf{W}_c(t)) = 0$.

The critic network converges asymptotically with probability 1. The updating rule for the action network is given as

$$\mathbf{\Delta W}_a(t) = -l_a(t)\frac{\partial E_a(t)}{\partial \mathbf{W}_a(t)} = -l_a(t)J(t)\frac{\partial J(t)}{\partial \mathbf{W}_a(t)}. \quad \text{(A15)}$$

It is generally assumed in interpretation of this algorithm that the critic network evolves much faster than the action network. Therefore, from an action network's perspective, the critic network has already converged to an approximation of $R(t)$ during the action network update. If we choose the learning rate for the action network $l_a(t)$ that satisfies condition (4). Under (1)–(4), according to [15] and [16], similar results holds for the action network. The action network converges with probability 1 and that the following is satisfied, $\lim_{t\to\infty}(\partial J(t)/\partial \mathbf{W}_a(t)) = 0$.

3) Convergence of DHP-Based ATR: The ATR plant is the same as described in Section II. The primary difference between HDP and DHP is that the critic network in HDP is to approximate the cost function, while that of the DHP is to approximate the derivative as the cost function. For the critic network, the prediction error in DHP is defined, in terms of differentiations, as

$$e_c(t) = \frac{\partial J(t)}{\partial \mathbf{X}(t)} - \alpha\frac{\partial J(t+1)}{\partial \mathbf{X}(t)} - \frac{\partial U(t)}{\partial \mathbf{X}(t)}. \quad \text{(A16)}$$

In our implementation, the ATR plant is modeled as a discrete state system as shown in (18) and the differentiation in prediction error is computed as

$$e_c(t) = \frac{\Delta J(t)}{\Delta \mathbf{X}(t)} - \alpha\frac{\Delta J(t+1)}{\Delta \mathbf{X}(t)} - \frac{\Delta U(t)}{\Delta \mathbf{X}(t)}. \quad \text{(A17)}$$

The objective function to be minimized in the critic network is

$$E_c(t) = \sum e_c(t) e_c^T(t). \quad \text{(A18)}$$

The convergence result in [13], [15], and [16] can also be applied to the DHP algorithm in this paper. Conditions 1 and 3 in Section A.1.1 remain the same as in HDP. In the DHP setup, condition 2 in Section A.1.1 is changed into the following: for every pair of states $i, j \in \mathbb{X}$, the functions $t_{ij}(t)$ and $(\Delta r_i/\Delta \mathbf{X})\theta$ are bounded, twice differentiable, and have bounded first and second derivatives.

For every pair of states $i, j \in \mathbb{X}$, the functions $t_{ij}(t)$ and $(\Delta r_i/\Delta \mathbf{X})\theta$ are bounded. Furthermore, as the transition probabilities $t_{ij}(t)$ and reinforcement $r_i(t)$ are functions of the state $\mathbf{X}(t)$, which are not an explicit function of $\mathbf{W}_c(t)$, then the first three conditions are satisfied. If we choose the learning rate for critic network $l_c(t)$ that satisfies condition 4, we then have

$$\lim_{t\to\infty} \frac{\partial}{\partial \mathbf{W}_c(t)}\frac{\partial J(t)}{\partial \mathbf{X}(t)} = 0. \quad \text{(A19)}$$

The critic network converges asymptotically with probability 1. Assuming that the critic network dynamics evolves much faster than the action network, and choosing the learning rate $l_a(t)$ satisfying condition 4, the action network also converges asymptotically with probability 1.

4) Upper Bound of Estimation Error of the Cost-to-Go Function: Assume that action network has completed its learning with optimal action policy $u^*(t) = \phi^*(\mathbf{X})$ Also, assuming convergence of the critic network in this case, an upper bound for the critic error can be derived as [15]

$$\left\| J(\mathbf{w}_c^*) - J^* \right\|_\mu \leq \frac{1}{\sqrt{1-\alpha^2}} \left\| \Pi J^* - J^* \right\|_\mu \tag{A20}$$

where $\mathbf{W}_c^*(t)$ is the convergence limit of $\mathbf{W}_c(t)$, α is the constant discount factor, and $\|g\|_\mu$ is a weighted norm defined by

$$\left\| \nu \right\|_\mu = \left(\sum_{X \in \mathbb{X}} \pi(\mathbf{X}) \nu^2(\mathbf{X}) \right)^{\frac{1}{2}} \tag{A21}$$

with $\pi(\mathbf{X})$ being the steady state probability of system state $\mathbf{X}$. The norm $\|g\|_\mu$ induces a distance metric. The projection ΠJ^* with respect to the norm $\|g\|_\mu$ is

$$\Pi J^* = \arg\min_{\overline{J}} \left\| J^* - \overline{J} \right\|_\mu. \tag{A22}$$

As the error for the projection is always minimal, the upper bound in (A22) therefore states that the error associated with $\mathbf{W}_c^*(t)$ is within a constant factor of the best possible case.

In the ATR setup, the critic network provides an accurate approximation of $R(t)$ to improve the performance of the action network. The critic in combination with an action network helps to reduce variances in gradient descent algorithms [13]. At each cycle, the actor's parameters are updated according to a better prediction provided by the critic network. As the approximation of $J(t)$ is represented by a linear combination of basis functions $p_i(t)$, in the critic only structure with fixed basis functions, the selection of basis is critical to obtain a good approximation of the reward function $R(t)$. In the action-critic structure proposed in this paper, however, the basis function $p_i(t)$ is contingent upon the state of the system $\mathbf{X}(t)$ as well as the policy $u(t)$ generated by the action network. Therefore, the choice of basis functions is automatically refined at each iteration as the action network evolves. Note the updating rule for the action network is

$$\begin{aligned} \Delta \mathbf{W}_a(t) &= -l_a(t) \frac{\partial E_a(t)}{\partial \mathbf{W}_a(t)} = -l_a(t) e_a(t) \frac{\partial J(t)}{\partial \mathbf{W}_a(t)} \\ &= -l_a(t) J(t) \frac{\partial J(t)}{\partial \mathbf{W}_a(t)}. \end{aligned} \tag{A23}$$

The gradient is then given as

$$\frac{\partial E_a(t)}{\partial \mathbf{W}_a(t)} = J(t) \frac{\partial J(t)}{\partial \mathbf{W}_a(t)} \tag{A24}$$

and can be seen as a projection of $J(t)$ with respect to the inner product defined as

$$\left\langle J(t), \frac{\partial J(t)}{\partial \mathbf{W}_a(t)} \right\rangle = J(t) \frac{\partial J(t)}{\partial \mathbf{W}_a(t)}. \tag{A25}$$

Therefore, $\partial J(t)/\partial \mathbf{W}_a(t)$ can be represented as

$$\frac{\partial E_a(t)}{\partial \mathbf{W}_a(t)} = \left\langle J(t), \frac{\partial J(t)}{\partial \mathbf{W}_a(t)} \right\rangle. \tag{A26}$$

The gradient method implemented in the action network also helps to provide an approximate projection which in turn improves the evolution of basis function in the critic network.

ACKNOWLEDGMENT

The author would like to thank Y. Li for running part of the heuristic dynamic programming (HDP) simulation, noise related simulation and convergence analysis, and F. Siddiqui for running part of the dual heuristic dynamic programming (DHP) simulation in this paper. He also thanks Profs. Si and Lendaris for making HDP and dual heuristic DHP code publicly available.

REFERENCES

[1] D. Casasent and D. Psaltis, "Position, rotation and scale invariant optical correlator," *Appl. Opt.*, vol. 15, no. 7, pp. 1795–1799, Jul. 1976.

[2] K. M. Iftekharuddin, F. Ahmed, and M. A. Karim, "Amplitude-coupled mace for automatic target recognition applications," *Opt. Eng.*, vol. 35, no. 4, pp. 1009–1014, 1996.

[3] J. Shaik and K. M. Iftekharuddin, "Detection and tracking of targets in infrared images using Bayesian techniques," *Opt. Laser Technol.*, vol. 41, no. 6, pp. 832–842, Sep. 2009.

[4] S. L. Diab, M. A. Karim, and K. M. Iftekharuddin, "Multiobject detection of targets with fine details, scale and translation variations," *Opt. Eng.*, vol. 37, no. 3, pp. 876–883, 1998.

[5] K. M. Iftekharuddin, C. Rentala, and A. Dani, "Determination of exact rotation angle and discrimination for rotated images," *Opt. Laser Technol.*, vol. 34, no. 4, pp. 313–327, Jun. 2002.

[6] Q. Zhang, "Using wavelet network in nonparametric estimation," *IEEE Trans. Neural Netw.*, vol. 8, no. 2, pp. 227–236, Mar. 1997.

[7] M. Bryant, S. Worrell, and A. Dixon, "MSE template size analysis for MSTAR data," *Proc. SPIE*, vol. 3370, pp. 396–405, Apr. 1998.

[8] K. M. Iftekharuddin and G. Power, "A biological model for distortion-invariant target recognition," in *Proc. Int. Joint Conf. Neural Netw.*, vol. 1. Washington D.C., Jul. 2001, pp. 559–564.

[9] K. M. Iftekharuddin and R. P. Malhotra, "Role of multiresolution attention in automated object recognition," in *Proc. Int. Joint Conf. Neural Netw.*, Honolulu, HI, May 2002, pp. 2225–2230.

[10] K. M. Iftekharuddin and T. Widjanarko, "Reinforcement learning in multiresolution object recognition," in *Proc. Int. Joint Conf. Neural Netw.*, vol. 2. Jul. 2004, pp. 1085–1090.

[11] F. Siddiqui and K. M. Iftekharuddin, "Multiresolution object recognition using dual heuristic programming," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2006, pp. 4262–4268.

[12] Z. Ge, "Automated target recognition by reinforcement learning without a prior model," Ph.D. thesis, Texas Technol. Univ., Dept. Electr. Comput. Eng., Lubbock, 1997.

[13] A. G. Barto, "Reinforcement learning and adaptive critic method," in *Handbook of Intelligent Control*, D. A. Sofge and D. A. White, Eds. New York: Van Nostrand, 1992, pp. 469–491.

[14] J. N. Tsitsiklis and B. Van Roy, "Feature-based methods for large scale dynamic programming," *Mach. Learn.*, vol. 22, nos. 1–3, pp. 59–94, Jan.–Mar. 1996.

[15] B. Van Roy, "Neuro-dynamic programming: Overview and recent trends," in *Handbook of Markov Decision Processes: Methods and Applications*, E. Feinberg and A. Shwartz, Eds. Norwell, MA: Kluwer, 2001.

[16] P. Marbach and J. N. Tsitsiklis, "Simulation-based optimization of Markov reward processes," *IEEE Trans. Autom. Control*, vol. 46, no. 2, pp. 191–209, Feb. 2001.

[17] D. P. Bertsekas, *Dynamic Programming: Deterministic and Stochastic Models*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[18] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.

[19] R. Ens and J. Si, "Apache helicopter stabilization using neuro dynamic programming," Amer. Inst. Aeronaut. Astronaut., Reston, VA, Tech. Rep. G6316, 2000.

[20] J. Si and Y.-T. Wang, "Online learning control by association and reinforcement," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 264–276, Mar. 2001.

[21] G. K. Venayagamoorthy, R. G. Harley, and D. C. Wunsch, "Implementation of adaptive critic-based neurocontrollers for turbogenerators in a multimachine power system," *IEEE Trans. Neural Netw.*, vol. 14, no. 5, pp. 1047–1064, Sep. 2003.

[22] D. V. Prokhorov and D. C. Wunsch, "Adaptive critic designs," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 997–1007, Sep. 1997.

[23] E. Körner, M.-O. Gewaltig, U. Körner, A. Richter, and T. Rodemann, "A model of computation in neocortical architecture," *Neural Netw.*, vol. 12, nos. 7–8, pp. 989–1005, Oct. 1999.
[24] K. Doya, "What are the computations of the cerebellum, the basal ganglia and the cerebral cortex?" *Neural Netw.*, vol. 12, nos. 7–8, pp. 961–974, Oct. 1999.
[25] W. Schultz, "Predictive reward signal of dopamine neurons," *J. Neurophysiol.*, vol. 80, no. 1, pp. 1–27, Jul. 1998.
[26] R. E. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1957.
[27] P. J. Werbos, "Neurocontrol and supervised learning: An overview and evaluation," in *Handbook of Intelligent Control*, D. A. Sofge and D. A. White, Eds. New York: Van Nostrand, 1992, pp. 65–89.
[28] G. G. Lendaris and C. Paintz, "Training strategies for critic and action neural networks in dual heuristic programming method," in *Proc. Int. Conf. Neural Netw.*, vol. 2. Houston, TX, Jun. 1997, pp. 712–717.
[29] G. G. Lendaris, C. Paintz, and T. Shannon, "More on training strategies for critic and action neural networks in dual heuristic programming method," in *Proc. IEEE Conf. Syst. Man, Cybern., Comput. Cybern. Simul.*, vol. 4. Orlando, FL, Oct. 1997, pp. 3067–3072.
[30] G. C. Lendaris and T. Shannon, "Application considerations for the DHP methodology," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2. Anchorage, AK, May 1998, pp. 1013–1018.
[31] S. Shervais, T. T. Shannon, and G. G. Lendaris, "Adaptive critic based approximate dynamic programming: A new tool for smart manufacturing," in *Proc. Int. Joint Conf. Artif. Intell.*, Seattle, WA, Aug. 2001, pp. 1–7.
[32] W. Liu, G. K. Venayagamoorthy, and D. C. Wunsch, "A heuristic dynamic programming based power system stabilizer for a turbogenerator in a single machine power system," in *Proc. IEEE-IAS 38th Annu. Meet. Conf. Ind. Appl.*, vol. 1. Oct. 2003, pp. 270–276.
[33] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2002, ch. 7, pp. 386–394.
[34] D. B. Graham and N. M. Allinson, "Characterizing virtual eigensignatures for general purpose face recognition," in *Face Recognition: From Theory to Applications* (Computer and Systems Sciences), vol. 163, H. Wechsler, P. J. Phillips, V. Bruce, F. Fogelman-Soulie, and T. S. Huang, Eds. Berlin, Germany: Springer-Verlag, 1998, pp. 446–456.
[35] K. I. Chang, K. W. Bowyer, and P. J. Flynn, "Face recognition using 2-D and 3-D facial data," in *Proc. ACM Workshop Multimod. User Authen.*, Dec. 2003, pp. 25–32.
[36] D.-T. Lin, J. E. Dayhoff, and P. A. Ligomenides, "Trajectory production with the adaptive time-delay neural network," *Neural Netw.*, vol. 8, no. 3, pp. 447–461, 1995.
[37] E. Artyomov and O. Yadid-Pecht, "Modified high-order neural network for invariant pattern recognition," *Pattern Recognit. Lett.*, vol. 26, no. 6, pp. 843–851, May 2005.
[38] R. Foltyniewicz, "Efficient high order neural network for rotation, translation and distance invariant recognition of gray scale images," in *Computer Analysis of Images and Patterns* (Lecture Notes in Computer Science), vol. 970. New York: Springer-Verlag, 1995, pp. 424–431.
[39] D. Casasent and Y.-C. Wang, "A hierarchical classifier using new support vector machines for automatic target recognition," *Neural Netw.*, vol. 18, nos. 5–6, pp. 541–548, Jul.–Aug. 2005.
[40] L. S. Wu and L. Jain, "Size invariant shape recognition by modulated competitive neural circuit," *Int. J. Inf. Commun. Technol.*, vol. 1, no. 1, pp. 76–88, 2007.
[41] S. Augenstein and S. M. Rock, "Simultaneous estimation of target pose and 3-D shape using the fastSLAM algorithm," in *Proc. Amer. Inst. Aeronaut. Astronaut. Guid., Navigat., Control Conf.*, 2009, pp. 1–15.
[42] J. Lee, S.-M. Baek, C. Choi, and S. Lee, "Particle filter based robust recognition and pose estimation of 3-D objects in a sequence of images," in *Recent Progress in Robotics: Viable Robotic Service to Humans* (Lecture Notes in Control and Information Sciences), vol. 370. New York: Springer-Verlag, 2009, pp. 241–253.
[43] V. I. Kaufman, T. D. Ross, E. M. Lavely, and E. P. Blasch, "Score-based SAR ATR performance model with operating condition dependencies," *Proc. SPIE Algor. Synth. Apert. Radar Imagery*, vol. 6568, no. 14, pp. 65680Z-1–65680Z-12, 2007.
[44] J.-W. Park, R. G. Harley, and G. K. Venayagamoorthy, "A novel dual heuristic programming based optimal control of a series compensator in the electric power transmission system," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 4. Jul. 2003, pp. 2976–2981.

Khan M. Iftekharuddin (SM'02) received the B.Sc. degree from the Bangladesh Institute of Technology, Rajshahi, Bangladesh, in 1989, and the M.S. and Ph.D. degrees in electrical engineering from the University of Dayton, Dayton, OH, in 1991 and 1995, respectively.

He is an Associate Professor with the Department of Electrical and Computer Engineering, University of Memphis (UoM), Memphis, TN. He holds a joint appointment with the joint program in biomedical engineering at UoM and the University of Tennessee, Memphis. He also serves as a Faculty Member for the Bioinformatics Program and the Institute of Intelligent Systems, UoM. He is the principal author of more than 100 refereed journal papers, conference proceedings, and multiple book chapters. His research has been funded by different agencies, such as the National Science Foundation, the National Institutes of Health, Army Research Office, Air Force Research Laboratory, Wright-Patterson Air Force Base, OH, the Whitaker Foundation, Arlington, VA, and the FedEx Institute of Technology, Memphis. His current research interests include intelligent systems and reinforcement learning, medical image analysis, computational modeling, bioinformatics, sensor signal and image analysis, computing and interconnection, automatic target recognition (ATR), and biologically inspired ATR.

Prof. Iftekharuddin serves as an Associate Editor for *Optical Engineering*, the *International Journal of Image Processing*, and the *International Journal of Tomography and Statistics*. He is a fellow of the Society of Photo-Optical Instrumentation Engineers and a member of the Optical Society of America.