# REPORT DOCUMENTATION PAGE

1

Form Approved OMB NO. 0704-0188



| 1. REPORT DATE (DD-MM-YYYY) | 2. REPORT TYPE | 3. DATES COVERED (From - To) |
|---|---|---|
| | New Reprint | - |

4. TITLE AND SUBTITLE
Heuristic dynamic programming with internal goal representation

5a. CONTRACT NUMBER
W911NF-12-1-0378

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS
Zhen Ni, Haibo He

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
University of Rhode Island
Sponsored Projects
70 Lower College Road, Suite II
Kingston, RI 02881 -1967

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
61817-CS.26

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
In this paper, we analyze an internal goal structure based on heuristic dynamic programming, named GrHDP, to tackle the 2-D maze navigation problem. Classical reinforcement learning approaches have been introduced to solve this problem in literature, yet no intermediate reward has been assigned before reaching the final goal. In this paper,

15. SUBJECT TERMS
Goal representation heuristic dynamicprogramming (GrHDP), Maze navigation/path planning, Adaptive dynamic programming (ADP), Reinforcement learning (RL)

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 15. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | Haibo He |
| UU | UU | UU | UU | | 19b. TELEPHONE NUMBER 401-874-5844 |

**Report Title**

Heuristic dynamic programming with internal goal representation

**ABSTRACT**

In this paper, we analyze an internal goal structure based on heuristic dynamic programming, named GrHDP, to tackle the 2-D maze navigation problem. Classical reinforcement learning approaches have been introduced to solve this problem in literature, yet no intermediate reward has been assigned before reaching the final goal. In this paper, we integrated one additional network, namely goal network, into the traditional heuristic dynamic programming (HDP) design to provide the internal reward/goal representation. The architecture of our proposed approach is presented, followed by the simulation of 2-D maze navigation (10*10) problem. For fair comparison, we conduct the same simulation environment settings for the traditional HDP approach. Simulation results show that our proposed GrHDP can obtain faster convergent speed with respect to the sum of square error, and also achieve lower error eventually.

# REPORT DOCUMENTATION PAGE (SF298)
# (Continuation Sheet)

Continuation for Block 13

ARO Report Number 61817.26-CS
Heuristic dynamic programming with internal go...

Block 13: Supplementary Note


Approved for public release; distribution is unlimited.

Soft Comput (2013) 17:2101–2108
DOI 10.1007/s00500-013-1112-9

FOCUS

# Heuristic dynamic programming with internal goal representation

**Zhen Ni · Haibo He**

Published online: 3 September 2013


**Abstract** In this paper, we analyze an internal goal structure based on heuristic dynamic programming, named GrHDP, to tackle the 2-D maze navigation problem. Classical reinforcement learning approaches have been introduced to solve this problem in literature, yet no intermediate reward has been assigned before reaching the final goal. In this paper, we integrated one additional network, namely goal network, into the traditional heuristic dynamic programming (HDP) design to provide the internal reward/goal representation. The architecture of our proposed approach is presented, followed by the simulation of 2-D maze navigation (10*10) problem. For fair comparison, we conduct the same simulation environment settings for the traditional HDP approach. Simulation results show that our proposed GrHDP can obtain faster convergent speed with respect to the sum of square error, and also achieve lower error eventually.

**Keywords** Goal representation heuristic dynamic programming (GrHDP) · Maze navigation/path planning · Adaptive dynamic programming (ADP) · Reinforcement learning (RL)

## 1 Introduction

In the past decades, reinforcement learning (RL) and adaptive dynamic programming (ADP) techniques have been frequently employed for the prediction and optimization to find the optimal control policy over time. For instance, heuristic dynamic programming (HDP), dual heuristic dynamic programming (DHP), and globalized dual heuristic dynamic programming (GDHP), have been proposed in Werbos (1992, 1990) to seek the optimal control policy (solution for Bellman's equation). Various versions of ADP, such as the action-dependent (AD) designs and model-based designs, are developed and presented with the learning and control capabilities on various applications. More recently, high-level understanding of ADP also discussed the fundamental principles for ADP on the optimization and learning capabilities over time Werbos (2013, 2008, 2009). For instance, the online direct HDP was proposed and developed in Si and Wang (2001), Yang et al. (2009), Liu et al. (2012), where the authors took the advantages of the potential scalability of the adaptive critic designs and the intuitiveness of Q-learning. It was also an online learning scheme that simultaneously updated the value function and the control policy. For model-based DHP/GDHP design, the authors demonstrated the convergent analysis in terms of cost function and control law in Liu et al. (2012), Wang et al. (2012), Liu and Wei (2013). In addition, the performance comparison among HDP, DHP and GDHP are studied and presented with the the autolander helicopter problem in Prokhorov and Wunsch (1997), Prokhorov (1997), Prokhorov et al. (1995). Recent research books provided the deep overview of RL and ADP on both stability/convergent analysis and various of complex industrial applications Si et al. (2004), Lewis and Liu (2013).

Recent papers on the exploration of internal reward (goal) have demonstrated the significance in ADP/RL communities. It has been proposed and demonstrated in He et al. (2011, 2012a,b) that a three-network architecture can achieve better control performance comparing with the traditional ADP design on several balancing benchmarks. In addition, hierarchical HDP design is presented with significant improvement with respect to the average successful trial number, compar-

Communicated by C. Alippi, D. Zhao and D. Liu.

Z. Ni · H. He (✉)
Department of Electrical, Computer and Biomedical Engineering, University of Rhode Island, Kingston, RI 02881, USA
e-mail: ni@ele.uri.edu; he@ele.uri.edu

ing with both three-network design and the traditional ADP design in Ni et al. (2012a,b), He et al. (2012c). Furthermore, people also showed that the performance of ADP controller could be improved by second-order learning algorithm on complex industrial system in Fu et al. (2011a,b,c). In addition, stability analysis on dual-critic design with tracking controller has been provided and verified on numerical simulation benchmarks in Ni et al. (2013). Real-time tracking control with this dual-critic design on visual reality/simulink platform is also presented in Ni et al. (2013), Fang et al. (2012).

Maze navigation is the typical Markov decision process (MDP) benchmark and has been tested in the ADP/RL community. For instance, in Pang and Werbos (1996), Wunsch (2000), it has been proposed to learn the value table with adaptive-critic designs in a closed-loop form with simultaneous recurrent neural network (SRN). In Ilin et al. (2006, 2007, 2008), the authors proposed to improve the learning process with cellular SRN and Kalman filter integrating into ADP design on the same problem. Furthermore, in Wiering and Van Hasselt (2007), the authors compared classical Q-learning, Sarsa($\lambda$), conventional actor-critic design and the proposed QV-learning on the maze navigation benchmark, and showed the improved learning process with the proposed approach. Although recent advancements of ADP research have demonstrated many critical applications across different domains, it has been recognized that the 2-D maze navigation problem is a significant challenge for the society Werbos and Pang (1996).

In this paper, we extend our previous work on goal representation design for MDP benchmarks Ni et al. (2013), and focus on the comparison between the proposed approach and the traditional HDP approach. From the viewpoint of proof-of-concept, we conduct the same simulation environment setting for both approaches and adopt the gradient descent method as the learning algorithm. In specific, there are three neural networks in our proposed goal representation HDP (GrHDP) approach: an action network, a critic network, and a goal network. The motivation is to represent the detailed goal signal, which can be able to be tuned adaptively and efficiently, according to the system state. In this way, rather than a discrete reinforcement signal in the traditional ADP and RL approach, our goal network can automatically provide an internal goal signal based on the (external) reinforcement signal, to achieve optimal action selection. For fair comparison, we evaluate our proposed GrHDP approach and regular HDP approach with the same parameter settings. The learning curves show that GrHDP and HDP can both learn the value table online. However, our proposed GrHDP approach can not only show faster convergent speed, but also achieve lower sum of squared error in the end.

The rest of the paper is organized as follows: Sect. 2 shows the architecture design of our proposed GrHDP framework on the maze navigation problem, and also provides the learning algorithm for the goal network. Simulation results on GrHDP and HDP under the same environment settings are presented and compared in Sect. 3. Finally, the conclusion is provided in Sect. 4.

## 2 GrHDP structure for maze navigation

We provide the interaction diagram between the proposed GrHDP design and the maze/environment in Fig. 1. From this figure, we can see that the action network observes the system state from the maze/environment and provides the action based on the current state. The (external) reward will be provided by the environment based on the performance of the corresponding action. As for the HDP controller (i.e. the middle part in Fig. 1), we keep the similar design with traditional the HDP in Si and Wang (2001). That is to say, we adopt model-free action dependent (AD) design for our GrHDP and also use the gradient descent algorithm for the learning of all the neural networks. Instead of the traditional (external) discrete reward assignment in maze navigation, our proposed GrHDP design integrates a goal network to learn from (external) reward $r$, and provide the critic network with a detailed internal reward $s$. In this paper, we defined the (external) reward as

$$r = \begin{cases} 1, & reach\ the\ goal \\ -0.2, & out\ of\ bound \\ 0, & regular\ move \end{cases} \tag{1}$$

The explicit explanation for this reward on the maze is also presented in the lower part of Fig. 1.

We employ typical multi-layer perceptron (MLP) structure for the all the neural networks here. In order to closely connect the goal network with the critic network, we set the internal reward $s$ to be one of the inputs for the critic network. Therefore, the input of the goal network and critic network can be denoted as $x_g = [X,\ u]$ and $x_c = [X,\ u,\ s]$, respectively. The inputs for the action network is the current system state vector, and the outputs of the action network refer to the four directions (i.e. the outputs of the action network is a $4 * 1$ vector). In order to show our original contribution, we will only discuss the learning algorithm of goal network and briefly provide the error (objective) function for both critic network and action network.

### 2.1 Learning in goal network

In literature, people generally assign the instant reward to be 0 unless the agent reaches the goal in the maze navigation problem Mitchell (1997), Sutton and Barto (1998). In recent years, there seems to be growing attention to see if there is any improvement if a non-zero instant reward is assigned

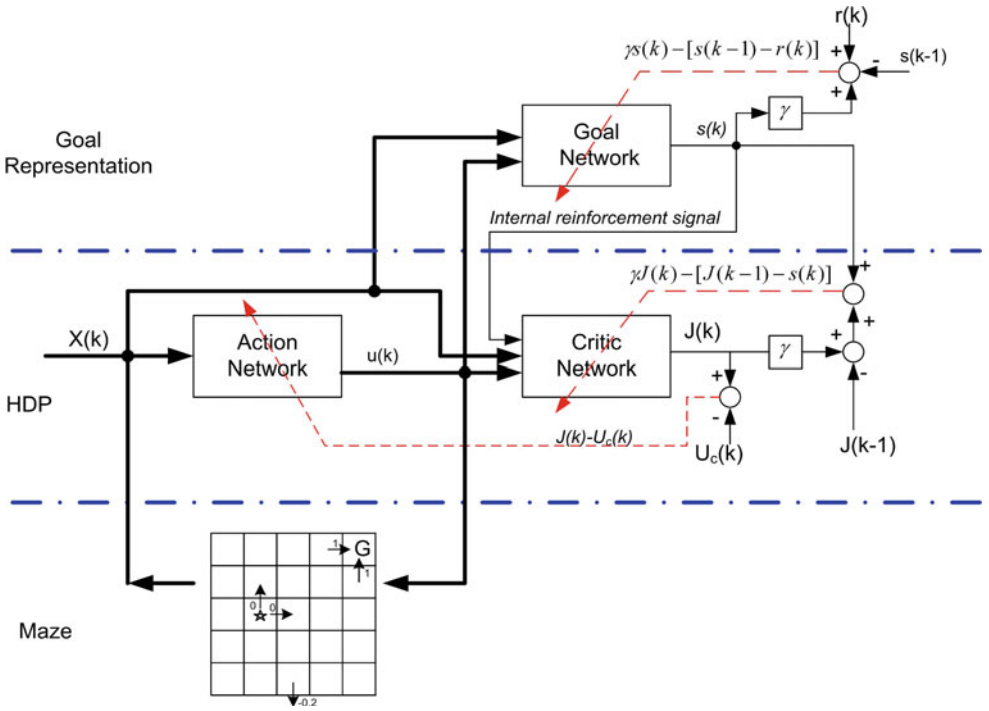


**Fig. 1** The proposed GrHDP framework on maze navigation. Two *dash lines* separate the diagram into three parts: goal representation, traditional HDP design, maze navigation benchmark

for the agent during the learning process He (2011), He et al. (2012c). Various reward/cost functions are defined according to different applications, however, such reward/cost functions are strongly domain-oriented and it is difficult to define such a proper function in general. Therefore, it is desirable to find a general reward/cost function that can be able to learn and self-adjust in various environment. In this paper, we propose to build a general mapping between the system state (including the control action) and the internal goal signal by using a neural network. In addition, we integrate such a network into the HDP framework. The internal goal signal can then be represented as

$$s = f(X, u). \tag{2}$$

The motivation of this design is to introduce the goal network to represent the internal reward/goal, and approximate the discounted total future reward. Thus we define the internal reward/goal as

$$s(k) = r(k+1) + \gamma r(k+2) + \gamma^2 r(k+3) + \cdots \tag{3}$$

where $\gamma$ is the discounted factor, and $r$ is the (external) reward signal defined in (1). Here the sequence of $r(k+1)$, $r(k+2)$, $r(k+3)$... are the future reward signals.

Therefore, the error function for goal network is defined as

$$e_g(k) = \gamma s(k) - [s(k-1) - r(k)]. \tag{4}$$

and

$$E_g(k) = \frac{1}{2} e_g^2(k). \tag{5}$$

The state vector is $X = [x_1, x_2, \ldots x_n]$, where $n$ is number of element in state vector, and the control action is $u = [u_1, u_2, \ldots u_m]$, where $m$ is the number of element in control vector. The input vector for the goal network is defined as $x_g = [X, u]$ and the output is the internal goal $s$, which is a scalar here. Sigmoid function is defined as

$$\phi(x) = \frac{1 - e^{-x}}{1 + e^{-x}}. \tag{6}$$

to constrain the output into $[-1, 1]$. Here sigmoid function is applied on all hidden nodes and the output node as presented in Fig. 2. The forward paths of goal network is provided as follows.

$$\begin{aligned} s(k) &= \phi(l(k)) \\ l(k) &= \sum_{i=1}^{N_{gh}} \omega_{g_i}^{(2)}(k) y_i(k) \\ y_i(k) &= \phi(z_i(k)), \qquad i = 1, \ldots, N_{gh} \\ z_i(k) &= \sum_{j=1}^{n} \omega_{g_{i,j}}^{(1)}(k) x_j(k) + \sum_{j=1+n}^{m+n} \omega_{g_{i,j}}^{(1)}(k) u_{j-n}(k) \end{aligned} \tag{7}$$

where $z_i$ and $y_i$ refer to the input and the output of the $i$-th hidden node. $l$ is the input for the output node. $\omega_g^{(1)}$ and $\omega_g^{(2)}$ denote the weights of the input to hidden layer and the hidden to output layer in the goal network, respectively. $N_{gh}$ is the number of hidden node in goal network. We have denoted

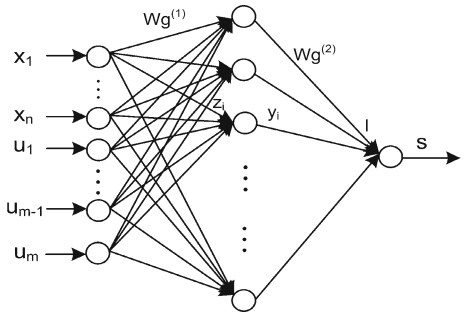


**Fig. 2** MLP structure of goal network. Sigmoid function is applied for both hidden nodes and output nodes

these parameters in Fig. 2, where readers can easily follow the forward learning paths in (7) for the goal network.

We adopt gradient descent method to minimize the approximation error in (5). The weights from hidden to output layer are tuned as

$$\frac{\partial E_g(k)}{\partial w_{g_i}^{(2)}(k)} = \frac{\partial E_g(k)}{\partial s(k)} \frac{\partial s(k)}{\partial l(k)} \frac{\partial l(k)}{\partial w_{g_i}^{(2)}(k)} \tag{8}$$

The weights from input to hidden layer are tuned as

$$\frac{\partial E_g(k)}{\partial w_{g_i}^{(1)}(k)} = \frac{\partial E_g(k)}{\partial s(k)} \frac{\partial s(k)}{\partial l(k)} \frac{\partial l(k)}{\partial y_i(k)} \frac{\partial y_i(k)}{\partial z_i(k)} \frac{\partial z_i(k)}{\partial w_{g_{i,j}}^{(1)}(k)} \tag{9}$$

The weights are tuned in the order of goal network, critic network and action network. After the weights in goal network are tuned, we fix these weights thereafter and start to tune the weights in the critic network and action network.

### 2.2 Learning in critic network and action network

The critic network and the action network in our design are similar with the existing designs in Si and Wang (2001), Yang et al. (2009), He and Jagannathan (2007). We introduce one more input, namely internal reward signal $s$, for the critic network and aim to help the value function approximation. In addition, the error (objective) function for critic network is different with those in existing designs. As the critic network is set to approximate the discounted total future internal reward/goal $s$ with value function $J$, we can write the value function as

$$J(k) = s(k+1) + \gamma s(k+2) + \gamma^2 s(k+3) + \cdots \tag{10}$$

Then the error function of critic network can be defined as

$$e_c(k) = \gamma J(k) - [J(k-1) - s(k)]. \tag{11}$$

and

$$E_c(k) = \frac{1}{2} e_c^2(k). \tag{12}$$

We apply the same gradient algorithm to minimize $E_c$ as above. Once we finish the weights-tuning in the critic network, we will start the online learning of action network.

As the objective of the action network is to maximize the total reward, we define the error function of action network as

$$e_a(k) = J(k) - U_c. \tag{13}$$

and

$$E_a(k) = \frac{1}{2} e_a^2(k). \tag{14}$$

where $U_c$ is the ultimate utility function. The same as the critic network, we adopt the gradient descent algorithm here to minimize (14).

### 2.3 Learning on maze navigation

In maze navigation benchmark, we assume that the agent starts with the initial state from the updating sequence (each updating sequence is assumed to visit all the state enough times). Our proposed GrHDP controller learn to provide the action based on the position of the agent. We apply winner-take-all (WTA) method to determine the direction for the agent to go. The goal network will provide the internal goal based on the direction and the updated state of the agent. This internal goal signal is set as one of the input for the critic network, which will then evaluate the performance of the agent for the corresponding action.

In our learning process, we keep checking if the agent is out of bound or reaching the goal after the action. If the agent is out of bound, we will set the punishment and start another trial. If the agent reaches the goal, we assign the reward and regard this as the end of the trial. Otherwise, the agent is kept moving forward (i.e. we adopt the infinite step looking-ahead). We update the $J(x, u)$ value table after each trial, and compare the learned value table with the reference value table to show the learning process. We will terminate the learning process when the trial number satisfies the maximum number we assign. In this simulation, we set independent runs to show that the learning process could be duplicated. We show the learning curves and value tables by taking the average of the results in different runs.

## 3 Simulation results and analysis

### 3.1 Algorithm implementation

The environment of the 2-D maze navigation is presented in the lower part of Fig. 1. In this simulation, we denote that the instant reward between any two state $x$ and $x'$ by taking the action $u$ as $r(x, x')$ or $r(x, u)$. We assume that there are $N$ possible states in the maze. The transition probabilities

between the two state $x$ and $x'$ can only take the value of 0 or 1 (i.e. the maze navigation problem here is a deterministic and finite MDP). Thus, the Bellman's equation can be rewritten as

$$J^*(x,u) = \arg\max_u \left( r(x,u) + \gamma \sum_{j=1}^{N} J^*(x',u') \right) \tag{15}$$

where $J^*(x,u)$ is the maximum total reward at state $x$ by taking the action $u$.

In this maze navigation benchmark, our objective is to employ learning algorithms to learn the value table online so that the agent can move according to the direction that maximizes the total reward (towards the goal location). For fair comparison, we conduct our proposed GrHDP and traditional HDP algorithm with the same environment settings and initial parameters. The algorithms and parameter settings for both approaches are summarized as:

1. HDP: Online model-free HDP proposed in Si and Wang (2001) is used here. The initial learning parameters are set as: $l_c = 0.005$ and $l_a = 0.01$, where $l_c$ and $l_a$ refer to the learning rate of critic network and action network, respectively. The stopping criteria are: $N_c = 20$, $N_a = 30$, $T_c = 1e-4$ and $T_a = 1e-4$. That is to say that the learning process of critic/action network will be terminated either if the error drops under the threshold $T_c/T_a$ or the iteration number meets the threshold $N_c/N_a$.
2. GrHDP: The same parameters are applied here if our proposed GrHDP approach has the same architecture as those of the HDP approach above. In addition, the initial parameters for the goal network are: $l_g = 0.012$, $T_g = 1e-4$ and $N_g = 25$. Furthermore, for fair comparison, we also set that the GrHDP and HDP start with the same initial weights between $[-0.3,\ 0.3]$ and the same updating sequence.

### 3.2 Simulation setup and study

In the simulation, we assume that (1) every state in the maze has been visited enough times; (2) every action (up, down, left, right) has been taken enough times for each state; (3) for every initial state, the agent can go infinite steps forward unless it reaches the goal or it hits the bound. The input for the action network is the current state vector that

$$x_a = [x_1,\ x_2] \tag{16}$$

The input for the goal network and the critic network are that

$$x_g = [x_1,\ x_2,\ u_1,\ u_2,\ u_3,\ u_4] \tag{17}$$

and

$$x_c = [x_1,\ x_2,\ u_1,\ u_2,\ u_3,\ u_4,\ s] \tag{18}$$

respectively. In this benchmark study, we define the the system state and control action as follow:

$x_1$: the coordinate of horizontal ($x$) axis;
$x_2$: the coordinate of horizontal ($y$) axis;
$u_1$: the direction—up;
$u_2$: the direction—down;
$u_3$: the direction—left;
$u_4$: the direction—right.

We assign that $U_c = 1$ and normalize the inputs for the action network to be in $[0,\ 2]$. We set 10 independent updating sequences for 10 runs (i.e. the updating sequence in each runs is independent). Each run includes 500 trials and each trial starts with the initial state loaded from the updating sequence. We set infinite step looking-ahead in our simulation studies. Each trial can only be terminated when the agent reaches the goal or hits the bound. Therefore, the steps that the agent move in each trial are not necessary the same. The $J(x,u)$ table is initialized as all zero at the very beginning and is only updated after the agent finish each trial. We then normalized the $J(x,u)$ values to be in $[0,\ 1]$ to show the difference with the reference value table.

We assume that both our proposed GrHDP approach and the traditional HDP approach start with the same initial weights (uniformly initialized in $[-0.3,\ 0.3]$) and the same updating sequence. The learning rates and internal stopping criteria for both approaches are also set to be the same. Adaptive learning rate (ALR) is used in our simulation. The initial learning rates for the action network, critic network and the goal network are set to be $5e-3$, $1e-2$ and $1.2e-2$, respectively, and they will be decreased by dividing 2 every 10 trials. We keep the learning rates to be $1e-10$ thereafter if they are under $1e-10$ after dividing. In addition, we also set a counter for the four actions/directions taken for all the states. For a specific state, for instance, if any action (i.e. up, down, left, right) is taken over a preset number (like 30 in this case study), we will randomly pick up another direction from the remaining choices as the final decision. We hope that all the directions could be tried enough times to guarantee that the agent can learn from both failure and success.

In this simulation study, we introduce Q reference value table according to the distance between the current location and the goal as that in Ilin et al. (2008), Ni et al. (2013). The values for the states that are one step from the goal are assigned to be 1 and the values for the other states will drop $\frac{1}{L+W}$ for each step, where $L$ and $W$ refer to the length and width of the maze, respectively. For maze size of $10*10$, the difference between each step is set as 0.05. Therefore, we define the Q reference table as

$$Q_{ref}(x_1,x_2) = 1 - \frac{1}{L+W} \cdot (L - x_1 + W - x_2 - 1) \tag{19}$$

Furthermore, we define the sum of squared error as

$$E_{sum} = \frac{1}{2} \sum_{x_1=1,x_2=1}^{x_1=L,x_2=W} \Big( J(x_1, x_2) - Q_{ref}(x_1, x_2) \Big)^2 \qquad (20)$$

In this case study, we set the maze size as $10 * 10$ and the goal locates at [10, 10]. The learning curves are presented in Fig. 3, where the $x$ axis refers to the number of the trial and the $y$ axis refers the sum of squared error. Note that all the curves presented here are the average values of 10 independent runs.

From Fig. 3, we can see that both approaches start with the same sum of squared errors as we initialize $J(x, u)$ to be all zero at the very beginning. When the agent starts to move, one can see that our proposed GrHDP approach shows faster convergent speed with regard to the sum of squared errors. Moreover, our proposed GrHDP approach can also achieve lower steady error than that with HDP approach. In addition, we also provide the histogram of internal goal $s_r$ in 10 independent runs in Fig. 4. The statistical results show that the internal goals are within the range that is almost symmetric with zero point. Furthermore, we provide the value table learned with our proposed GrHDP approach in Fig. 5, the values in which are the average of 10 runs. We can obtain the tendency that the values become larger if the the agent approaches the goal in the upper-right corner. The surface plot of this value table (i.e. the same value table in Fig. 5)

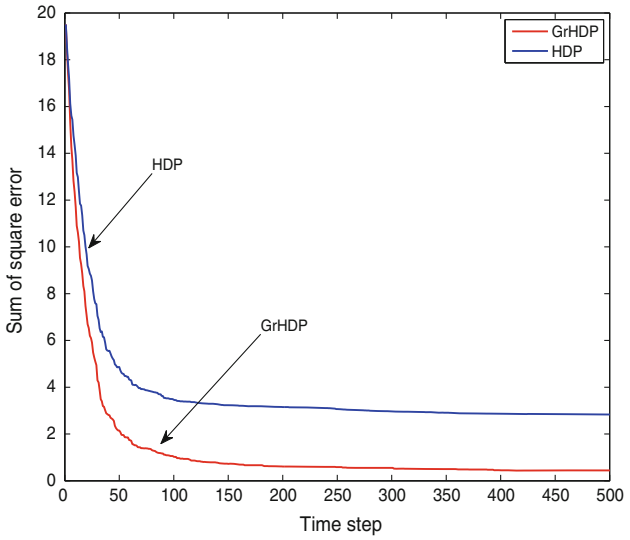


**Fig. 3** The learning *curves* for the sum of squared errors with GrHDP and HDP approaches, respectively

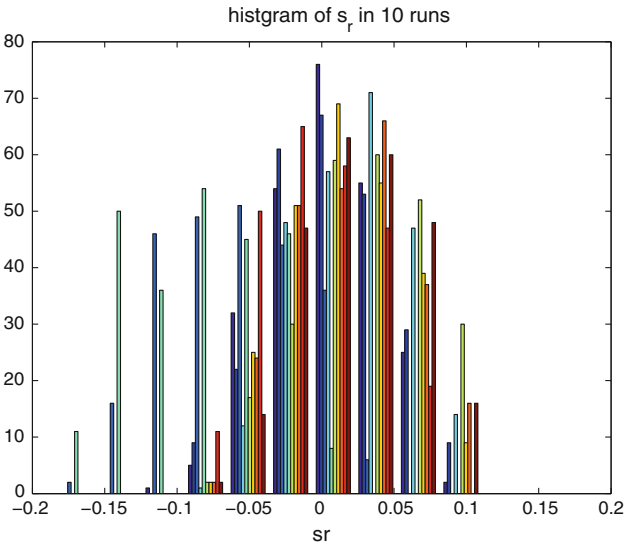


**Fig. 4** The histogram of internal goal in ten independent runs. Ten different colors are adopted to represent ten independent runs

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0.78 | 0.81 | 0.85 | 0.88 | 0.87 | 0.90 | 0.92 | 0.95 | 0.96 | 0 |
| 0.72 | 0.76 | 0.80 | 0.82 | 0.82 | 0.84 | 0.88 | 0.90 | 0.91 | 0.92 |
| 0.67 | 0.68 | 0.70 | 0.74 | 0.76 | 0.78 | 0.81 | 0.84 | 0.86 | 0.88 |
| 0.60 | 0.62 | 0.64 | 0.67 | 0.71 | 0.73 | 0.76 | 0.79 | 0.81 | 0.83 |
| 0.53 | 0.55 | 0.58 | 0.61 | 0.65 | 0.67 | 0.71 | 0.74 | 0.76 | 0.79 |
| 0.45 | 0.48 | 0.52 | 0.55 | 0.59 | 0.62 | 0.65 | 0.69 | 0.72 | 0.74 |
| 0.38 | 0.41 | 0.45 | 0.49 | 0.53 | 0.57 | 0.60 | 0.64 | 0.67 | 0.70 |
| 0.31 | 0.35 | 0.39 | 0.43 | 0.47 | 0.51 | 0.55 | 0.58 | 0.62 | 0.65 |
| 0.24 | 0.29 | 0.33 | 0.37 | 0.41 | 0.45 | 0.49 | 0.53 | 0.56 | 0.60 |
| 0.18 | 0.22 | 0.27 | 0.31 | 0.36 | 0.40 | 0.44 | 0.48 | 0.51 | 0.55 |

**Fig. 5** The value table for the $10 * 10$ maze. The value in each blank refers to the max value that the agent can obtain among the four possible directions

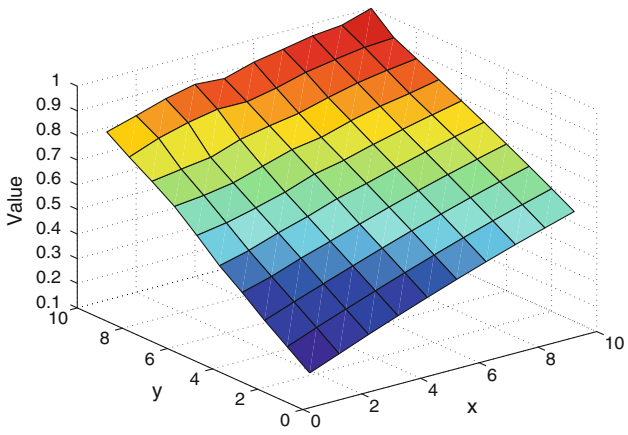


**Fig. 6** The surf plot of the value table for the $10 * 10$ maze. The value of the goal location has been set to 1 in this plot

is presented in Fig. 6, where $x$-axis and $y$-axis refer to the coordinates of the agent and $z$-axis refers to the corresponding J value. It is clear to see the smooth surface and the value goes bigger from the origin (i.e. [0, 0]) to the goal location (i.e. [10, 10]).

In addition, we have also compared the computational cost for both approaches. Here we only count the computation time for the part of learning algorithms (i.e. only the weights tuning are counted in both approaches). As the learning procedure is focus in the first 200 trials, we would like to compare the time-cost in this region for both approaches. Simulation results show that our proposed GrHDP approach requires $0.023s$ per trial, comparing with $0.016s$ per trial with HDP approach (simulations are conducted based on Sun server with 16 GB memory, Intel Xeon CPU 3.60 GHz and Matlab R2013a). Certainly, our proposed GrHDP approach include one more neural network and thus could take additional memory space for this goal network. Our key interests from this perspective in this paper are the convergent speed and the optimal policy, in which our proposed approach achieves much better performance compared to the regular HDP approach.

## 4 Conclusions

In this paper, a goal representation heuristic dynamic programming is introduced and analyzed for a classical maze navigation benchmark. We studied the GrHDP architecture design and its learning algorithms. In order to demonstrate the improved performance, we compare the learning results of our proposed GrHDP approach with that of traditional HDP approach based on a $10 * 10$ maze navigation benchmark under the same environment settings. The learning curves and the value table justify the improved performance comparing with traditional HDP approach.

**Acknowledgments** This work was supported by the National Science Foundation (NSF) under grant CAREER ECCS 1053717, Army Research Office (ARO) under grant W911NF-12-1-0378, and NSF-DFG Collaborative Research on "Autonomous Learning" (a supplement grant to CNS 1117314).

## References

Fang X, He H, Ni Z, Tang Y (2012) Learning and control in virtual reality for machine intelligence. In: International conference intelligent control and information processing (ICICIP'12), IEEE, Dalian, China, pp 63–67

Fu J, He H, Zhou X (2011) Adaptive learning and control for mimo system based on adaptive dynamic programming. IEEE Trans Neural Netw 22(7):1133–1148

Fu J, He H, Liu Q, Ni Z (2011) An adaptive dynamic programming approach for closely-coupled mimo system control. In: Int symp neural networks (ISNN'11), pp 1–10

Fu J, He H, Ni Z (2011) Adaptive dynamic programming with balanced weights seeking strategy. In: IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL), IEEE symposium series on computational intelligence (SSCI), France

He P, Jagannathan S (2007) Reinforcement learning neural-network-based controller for nonlinear discrete-time systems with input constraints. IEEE Trans Syst Man Cybern Part B-Cybern 37(2):425–436

He H (2011) Self-adaptive systems for machine intelligence. Wiley, New York

He H, Ni Z, Fu J (2012) A three-network architecture for on-line learning and optimization based on adaptive dynamic programming. Neurocomputing 78(1):3–13

He H, Ni Z, Zhao D (2012) Reinforcement learning and approximate dynamic programming for feedback control, ch. learning and optimization in hierarchical adaptive critic design. Wiley-IEEE Press, Hoboken

He H, Ni Z, Prokhorov DV (2011) Actor-critic design for on-line learning and optimization for machine intelligence. In: International conference on cognitive and neural systems (ICCNS'11), Boston

He H, Ni Z, Zhao D (2012) Data-driven learning and control with multiple critic networks. In: The 10th world congress on, intelligent control and automation (WCICA'12), pp 523–527

Ilin R, Kozma R, Werbos P (2008) Beyond feedforward models trained by backpropagation: a practical training tool for a more efficient universal approximator. Neural Netw IEEE Trans 19(6):929–937

Ilin R, Kozma R, Werbos (2006) Cellular SRN trained by extended Kalman filter shows promise for ADP. In: Proceedings of the IEEE international joint conference on neural networks (IJCNN), IEEE, pp 506–510

Ilin R, Kozma R, Werbos P (2007) Efficient learning in cellular simultaneous recurrent neural networks-the case of maze navigation problem. In: IEEE international symposium on approximate dynamic programming and reinforcement learning (ADPRL), IEEE, pp 324–329

Lewis F, Liu D (eds) (2013) Reinforcement learning and approximate dynamic programming for feedback control. Wiley-IEEE Press, Hoboken

Liu F, Sun J, Si J, Guo W, Mei S (2012) A boundedness result for the direct heuristic dynamic programming. Neural Netw 32:229–235

Liu D, Wang D, Zhao D, Wei Q, Jin N (2012) Neural-network-based optimal control for a class of unknown discrete-time nonlinear systems using globalized dual heuristic programming. IEEE Trans Autom Sci Eng 9(3):628–634

Liu D, Wei Q (2013) Finite-approximation-error-based optimal control approach for discrete-time nonlinear systems. IEEE Trans Cybern 43(2):779–789

Mitchell TM (1997) Machine learning. McGraw-Hill, Inc, New York

Ni Z, He H, Wen J (2013) Adaptive learning in tracking control based on the dual critic network design. IEEE Trans Neural Netw Learn Syst 6(24):913–928

Ni Z, Fang X, He H, Zhao D, Xu X (2013) Real-time tracking control on adaptive critic design with uniformly ultimately bounded condition. In: IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL'13). IEEE symposium series on computational intelligence (SSCI), USA

Ni Z, He H, Prokhorov DV, Fu J (2011) An online actor-critic learning approach with Levenberg-Marquardt algorithm. In: The 2011 international joint conference on neural networks (IJCNN), IEEE, pp 2333–2340

Ni Z, He H, Prokhorov DV (2012) Adaptive learning with goal generator network based on heuristic dynamic programming. In: Internatinal conference on cognitive and neural systems (ICCNS'12), Boston

Ni Z, He H, Wen J, Xu X (2013) Goal representation heuristic dynamic programming on maze navigation. IEEE Trans Neural Netw Learn Syst (to be published)

Ni Z, He H, Zhao D, Prokhorov D (2012) Reinforcement learning control based on multi-goal representation using hierarchical heuristic dynamic programming. In: The 2012 international joint conference on neural networks (IJCNN), IEEE, pp 1–8

Pang X, Werbos PJ (1996) Neural network design for j function approximation in dynamic programming. In: Mathematical modelling and scientific computing. http://arxiv.org/pdf/adap-org/9806001.pdf

Prokhorov DV (1997) Adaptive critic designs and their applications, PhD. Dissertation. PhD thesis

Prokhorov DV, Santiago RA, Wunsch DC II (1995) Adaptive critic designs: a case study for neurocontrol. Neural Netw 8(9):1367–1372

Prokhorov D, Wunsch D (1997) Adaptive critic designs. IEEE Trans Neural Netw 8(5):997–1007

Si J, Wang Y-T (2001) Online learning control by association and reinforcement. IEEE Trans Neural Netw 12(2):264–276

Si J, Barto AG, Powell WB, Wunsch DC (eds) (2004) Handbook of learning and approximate dynamic programming. Wiley, New York

Sutton R, Barto A (1998) Reinforcement learning: an introduction. MIT Press, Cambridge

Wang D, Liu D, Wei Q, Zhao D, Jin N (2012) Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming. Automatica 48:1825–1832

Werbos PJ (1990) Consistency of HDP applied to a simple reinforcement learning problem. Neural Netw 3(2):179–189

Werbos PJ (1992) Handbook of itelligent control, ch. Approximate dynamic programming for real-teim control and nerual modeling. Van Nostrand Reinhold, New York

Werbos PJ (2008) Adp: the key direction for future research in intelligent control and understanding brain intelligence. IEEE Trans Syst Man Cybern Part B-Cybern 38(4):898–900

Werbos PJ (2009) Intelligence in the brain: a theory of how it works and how to build it. Neural Netw 22(3):200–212

Werbos P (2013) Reinforcement learning and approximate dynamic programming for feedback control, ch. reinforcement learning and approximate dynamic programming (RLADP)-foundations, common misconceptions and challenges ahead. Wiley-IEEE Press, Hoboken

Werbos P, Pang X (1996) Generalized maze navigation: SRN critics solve what feedforward or Hebbian nets cannot. In: Systems, man, and cybernetics, 1996, IEEE international conference on, vol 3, pp 1764–1769

Wiering M, Van Hasselt H (2007) Two novel on-policy reinforcement learning algorithms based on td ($\lambda$)-methods. In: IEEE international symposium on approximate dynamic programming and reinforcement learning (ADPRL), IEEE, pp 280–287

Wunsch D (2000) The cellular simultaneous recurrent network adaptive critic design for the generalized maze problem has a simple closed-form solution. In: Proceedings of the IEEE international joint conference on neural networks (IJCNN), IEEE, vol 3, pp 79–82

Yang L, Si J, Tsakalis KS, Rodriguez AA (2009) Direct heuristic dynamic programming for nonlinear tracking conrol with filtered tracking error. IEEE Trans Syst Man Cybern Part B-Cybern 39(6):1617–1622