Incremental Model-Based Global Dual Heuristic Programming for Flight Control

Sun, Bo; Van Kampen, Erik Jan

DOI
10.1016/j.ifacol.2019.12.613

Publication date
2019

Document Version
Final published version

Published in
IFAC-PapersOnline

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 52-29 (2019) 7–12

Incremental Model-Based Global Dual Heuristic Programming for Flight Control ⋆

Bo Sun * **Erik-Jan van Kampen** **

* *Department of Control and Operations, Delft University of Technology, Delft, 2629HS, The Netherlands (e-mail: b.sun-1@tudelft.nl).*

** *Department of Control and Operations, Delft University of Technology, Delft, 2629HS, The Netherlands (e-mail: E.vanKampen@tudelft.nl).*

Abstract: This paper proposes a novel adaptive dynamic programming method, called Incremental model-based Global Dual Heuristic Programming, to generate a self-learning adaptive controller, in the absence of sufficient prior knowledge of system dynamics. An incremental technique is employed for online model identification, instead of the artificial neural networks commonly used in conventional Global Dual Heuristic Programming. The incremental model has the capability of tackling nonlinearity and uncertainty of the plant, but can also guarantee high precision of online identification without the requirement of offline training. On the basis of the identified model, two neural networks are adopted to facilitate the implementation of the self-learning controller, by approximating the cost-to-go and its derivatives and the control policy, respectively. Both methods are applied to a tracking control problem of a nonlinear aerospace system and the results show that the proposed method outperforms conventional Global Dual Heuristic Programming in online learning speed, tracking precision and robustness to variation of initial system states and network weights.

Keywords: Adaptive dynamic programming, adaptive control, incremental technique, global dual heuristic programming, artificial neural network.

1. INTRODUCTION

Recently, a number of Reinforcement Learning (RL) methods have been developed which solve nonlinear, optimal control problems and achieve certain levels of robustness and fault-tolerance (Wang (2019); Valadbeigi et al. (2019)). RL methods link bio-inspired artificial intelligence techniques to control problem so as to overcome some of the limitations and challenges of control methods that require accurate models. One branch of them is Adaptive/Approximate Dynamic Programming (ADP), which, based on Reinforcement Learning (RL) theory but different from traditional discrete RL approaches, aims to solve adaptive optimal control problems with large or continuous state spaces (Sutton and Barto (2018)). ADP methods approximate the value of states and/or control policy and obtain nearly optimal solutions of the Hamilton-Jacobi-Bellman (HJB) equations. In this way, ADP methods can deal with the so-called "curse of dimensionality".

As a class of ADP methods, Adaptive Critic Designs (ACDs), which separate policy evaluation (critic) and policy improvement (actor), have shown success in optimal adaptive control of aerospace systems (Ferrari and Stengel (2004); Van Kampen et al. (2006); Zhou et al. (2016); Zhou et al. (2018)). ACDs can be categorized into three groups (Prokhorov and Wunsch (1997)): Heuristic Dynamic Programming (HDP), Dual Heuristic Programming (DHP) and Global Dual Heuristic Programming (GDHP). HDP is the most basic form and most used structure, which uses the critic to approximate the cost-to-go. The critic in DHP approximates the derivatives of the cost-to-go with respect to the critic inputs, and in many practical applications it outperforms HDP in success rate and precision (Venayagamoorthy et al. (2002)), but increases computational burden of backpropagation through the actor. GDHP approximates both the cost-to-go and its derivatives, and has several forms (Prokhorov and Wunsch (1997)). In this paper the most straightforward form, in which the critic approximates the cost-to-go and its derivatives simultaneously (Yi et al. (2019)), is applied to illustrate the proposed method. It not only allows the derivatives of the cost-to-go to impact the critic weights update, but also avoids complexity in the actor weights update process.

The three groups of ACDs usually build a third module to identify the global model dynamics, often with Artificial Neural Networks (ANNs) (Liu et al. (2012)). Although ANNs can approximate nonlinear functions with arbitrary precision, many samples are required before the weights converge for online identification, which is problematic at the start of training because the other function approximators are then trained based on the incorrect model. For more complex systems, offline training is often involved to

⋆ The first author is financially supported for this Ph.D. research by China Scholarship Council with the project reference number of 201806290007.

10.1016/j.ifacol.2019.12.613

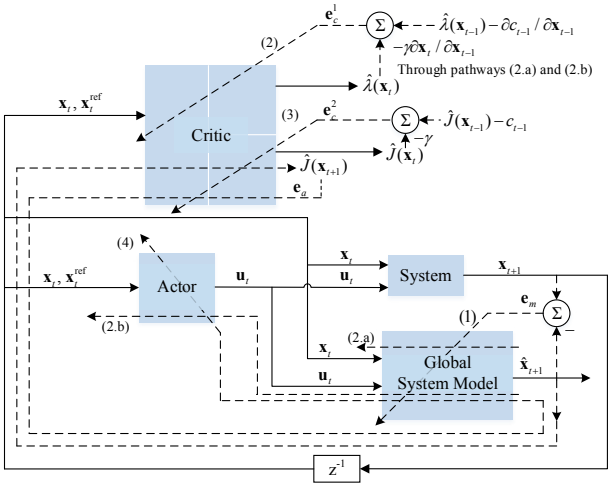

Fig. 1. The architecture of GDHP

obtain primary model, which, however, often cannot be applied to realistic system directly due to "reality gap".

The contribution of this paper is an Incremental model based Global Dual Heuristic Programming method (IGDHP) based on (Zhou et al. (2017); Zhou et al. (2018)). Different from conventional GDHP, an incremental model is involved for adaptive control to deal with the absence of full system information. Incremental techniques are able to accurately identify nonlinear system dynamics online, preventing the controllers from initial failure, and have been successfully applied to design adaptive flight controllers, such as Incremental Nonlinear Dynamic Inversion (INDI)(Smeur et al. (2015)), Incremental Back-Stepping (IBS) (Sonneveldt (2010)) and Incremental Sliding Mode Control (Wang et al. (2018)). Assuming sufficiently high sampling rate for discretization, IGDHP is able to achieve adaptive flight control, with this linear time-varying approximation.

The remainder of this paper is structured as follows. Section 2 presents the basic formulation of GDHP with three ANNs. Section 3 introduces the incremental method for online identification and uses it to simplify the weight update process of the actor and critic networks. Section 4 provides the experiment setup, while section 5 validates the approaches by applying both GDHP and IGDHP on a tracking control problem and comparing their results. Finally section 6 summarizes the paper and puts up possibilities for future research.

2. GLOBAL DUAL HEURISTIC PROGRAMMING IMPLEMENTATION

GDHP, which combines HDP and DHP, is also a kind of model free technique with three ANNs, namely model, critic and actor. One architecture of GDHP is illustrated in Fig. 1. The actor network outputs a control action, the model network approximates the states at the next time step, and then the outputs of the critic network at the next time step are computed. All weights of the ANNs are updated in a back-propagation way according to gradient-descent algorithm (Liu et al. (2012)).

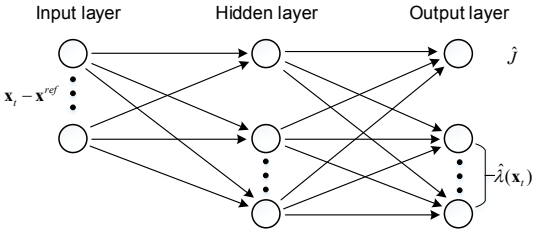

Fig. 2. Structure of critic neural network

2.1 Global Model

For a full-state feedback system, the inputs of the system model can be the current state vector $\mathbf{x}_t \in \mathbb{R}^n$ and current control vector $\mathbf{u}_t \in \mathbb{R}^m$, while the output is the estimated next state vector $\hat{\mathbf{x}}_{t+1} \in \mathbb{R}^n$. The network weights are updated by minimizing the difference between the measured state vector $\mathbf{x}_t$ and the estimated state vector $\hat{\mathbf{x}}_t$:

$$E_m(t) = \frac{1}{2}\boldsymbol{e}_m(t)^{\mathrm{T}} Q_m \boldsymbol{e}_m(t) \tag{1}$$

where

$$\boldsymbol{e}_m(t) = \mathbf{x}_t - \hat{\mathbf{x}}_t \tag{2}$$

and $Q_m \in \mathbb{R}^{n\times n}$ is a positive definite matrix. For simplicity, Q_m is usually defined as a diagonal matrix, i.e. $Q_m = diag\{\zeta_1, \zeta_2, \cdots, \zeta_n\}$, where the elements respectively select and weigh the approximating errors. The weights are updated as follows:

$$\mathbf{w}_m(t+1) = \mathbf{w}_m(t) - \eta_m \cdot \frac{\partial E_m(t)}{\partial \mathbf{w}_m(t)} \tag{3}$$

where η_m is the learning rate, and

$$\frac{\partial E_m(t)}{\partial \mathbf{w}_m(t)} = \frac{\partial E_m(t)}{\partial \hat{\mathbf{x}}_t} \cdot \frac{\partial \hat{\mathbf{x}}_t}{\partial \mathbf{w}_m(t)} = \boldsymbol{e}_m^{\mathrm{T}}(t) \cdot \frac{\partial \hat{\mathbf{x}}_t}{\partial \mathbf{w}_m(t)} \tag{4}$$

2.2 The Critic

GDHP combines HDP and DHP and requires information of both the cost-to-go $J(\tilde{\mathbf{x}}_t)$ and its derivatives with respect to the network inputs $\tilde{\mathbf{x}}_t$, where $\tilde{\mathbf{x}}_t = \mathbf{x}_t - \mathbf{x}_t^{\mathrm{ref}}$ stands for the tracking error vector. There are several forms to present it (Prokhorov and Wunsch (1997)), and in this paper, a straightforward critic structure is introduced to show the effects of the incremental method, i.e. the critic network outputs the approximation of $J(\tilde{\mathbf{x}}_t)$ and $\lambda(\tilde{\mathbf{x}}_t) = \frac{\partial J(\tilde{\mathbf{x}}_t)}{\partial \tilde{\mathbf{x}}_t}$ simultaneously, as shown in Fig. 2.

The first neuron of the output layer approximates the true cost-to-go $J(\tilde{\mathbf{x}}_t)$, which is the cumulative sum of future rewards r_t from any initial state $\tilde{\mathbf{x}}_t$:

$$J(\tilde{\mathbf{x}}_t) = \sum_{l=t}^{\infty} \gamma^{l-t} r_l \tag{5}$$

where $\gamma \in (0,1)$ is discount factor, used to control the extent to which the short-term cost or long-term cost is concerned. The other neurons of the output layer approximate the derivatives of $J(\tilde{\mathbf{x}}_t)$ with respect to the input vector $\tilde{\mathbf{x}}_t$ and the number of these output neurons equals the dimension of $\tilde{\mathbf{x}}_t$.

The goal of the experimental setup is to track the reference states contained in $\mathbf{x}_t^{\mathrm{ref}}$, so a one-step cost function with a quadratic form is designed:

$$r_l = r(\mathbf{x}_t, \mathbf{x}_t^{\text{ref}}) = (\mathbf{x}_t - \mathbf{x}_t^{\text{ref}})^{\mathrm{T}} Q_c (\mathbf{x}_t - \mathbf{x}_t^{\text{ref}}) = \tilde{\mathbf{x}}_t^{\mathrm{T}} Q_c \tilde{\mathbf{x}}_t \quad (6)$$

where $Q_c \in \mathbb{R}^{n\times n}$ is a non-negative definite matrix.

Because future rewards are required, Temporal Difference (TD) method is introduced to iteratively update the critic network (Sutton and Barto (2018)). The principle is to minimize the temporal difference error. The critic errors are shown as follows:

$$e_{c1}(t) = \hat{J}(\tilde{\mathbf{x}}_{t-1}) - r_{t-1} - \gamma \hat{J}(\tilde{\mathbf{x}}_t) \quad (7)$$

and

$$\begin{aligned} \boldsymbol{e}_{c2}(t) &= \frac{\partial[\hat{J}(\tilde{\mathbf{x}}_{t-1}) - r_{t-1} - \gamma \hat{J}(\tilde{\mathbf{x}}_t)]}{\partial \tilde{\mathbf{x}}_{t-1}} \\ &= \hat{\lambda}(\tilde{\mathbf{x}}_{t-1}) - \frac{\partial r_{t-1}}{\partial \tilde{\mathbf{x}}_{t-1}} - \gamma \hat{\lambda}(\tilde{\mathbf{x}}_t) \frac{\partial \tilde{\mathbf{x}}_t}{\partial \tilde{\mathbf{x}}_{t-1}} \end{aligned} \quad (8)$$

where $e_{c1}(t)$ is the current TD error of first neuron of the output layer while $\boldsymbol{e}_{c2}(t)$ is the current TD error vector of other neurons of the output layer. GDHP combines both of them by a overall error function $E_c(t)$:

$$E_c(t) = \beta \frac{1}{2} e_{c1}^2(t) + (1-\beta)\frac{1}{2} \boldsymbol{e}_{c2}^{\mathrm{T}}(t) \boldsymbol{e}_{c2}(t) \quad (9)$$

where β is a scalar indicating the importance. If $\beta = 1$, then it becomes pure HDP, if $\beta = 0$, then the back-propagation channel of the actor is cut, so normally β is chosen from $(0, 1)$ for GDHP and in this paper it is set to be 0.5.

The critic weights are updated using a gradient-descent algorithm with a learning rate η_c to minimize the overall error $E_c(t)$:

$$\mathbf{w}_c(t+1) = \mathbf{w}_c(t) - \eta_c \cdot \frac{\partial E_c(t)}{\partial \mathbf{w}_c(t)} \quad (10)$$

where

$$\begin{aligned} \frac{\partial E_c(t)}{\partial \mathbf{w}_c(t)} &= \frac{\partial E_c(t)}{\partial \hat{J}(\tilde{\mathbf{x}}_{t-1})} \cdot \frac{\partial \hat{J}(\tilde{\mathbf{x}}_{t-1})}{\partial \mathbf{w}_c(t)} + \frac{\partial E_c(t)}{\partial \hat{\lambda}(\tilde{\mathbf{x}}_{t-1})} \cdot \frac{\partial \hat{\lambda}(\tilde{\mathbf{x}}_{t-1})}{\partial \mathbf{w}_c(t)} \\ &= \beta e_{c1}(t) \cdot \frac{\partial \hat{J}(\tilde{\mathbf{x}}_{t-1})}{\partial \mathbf{w}_c(t)} + (1-\beta) \boldsymbol{e}_{c2}^{\mathrm{T}}(t) \cdot \frac{\partial \hat{\lambda}(\tilde{\mathbf{x}}_{t-1})}{\partial \mathbf{w}_c(t)} \end{aligned} \quad (11)$$

Consequently, both kinds of critic outputs will have an influence on weights tuning.

2.3 The Actor

The goal of the actor network is to produce an optimal control policy by minimizing the error between the current approximated cost-to-go $\hat{J}(\tilde{\mathbf{x}}_t)$ and the ideal one $J^*(t)$, which depends on the given reward function and is set to be zero in this paper:

$$\mathbf{u}_t^* = \arg\min_{\mathbf{u}_t} E_a(t) \quad (12)$$

where $E_a(t)$ is the overall actor error function and defined as follows:

$$E_a(t) = \frac{1}{2} \boldsymbol{e}_a^2(t), \qquad e_a(t) = \hat{J}(\tilde{\mathbf{x}}_t) - J^*(t) \quad (13)$$

There are two optional back-propagation ways to update the actor weights in the GDHP design of this paper. One is to use the first neuron $\hat{J}(\tilde{\mathbf{x}}_t)$ of the critic outputs directly, and another one, similar to DHP, involves $\hat{\lambda}(\tilde{\mathbf{x}}_t)$ and its process is more complicated. Although the latter has a higher accuracy in theory (Prokhorov and Wunsch (1997)), in this paper, the simpler structure is utilized.

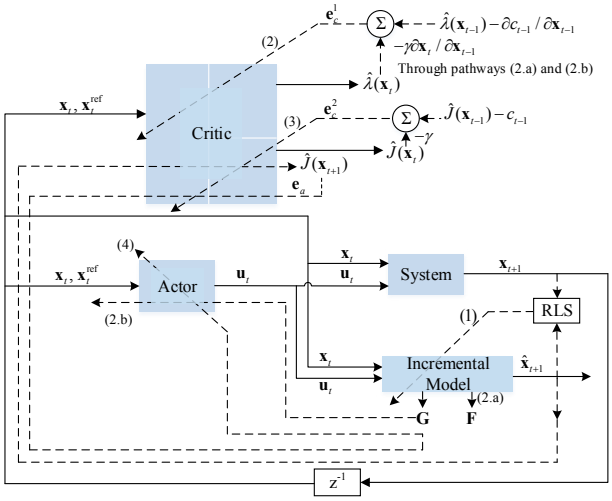

Fig. 3. The architecture of IGDHP

The actor network outputs control action $\mathbf{u}_t$, which is an input of the model network, and thus it will affect the critic outputs at the next time-step. As illustrated in Fig. 1, the actor weights are updated along the 4th back-propagation direction with a learning rate η_a:

$$\mathbf{w}_a(t+1) = \mathbf{w}_a(t) - \eta_c \cdot \frac{\partial E_a(t+1)}{\partial \mathbf{w}_a(t)} \quad (14)$$

where

$$\begin{aligned} \frac{\partial E_a(t+1)}{\partial \mathbf{w}_a(t)} &= \frac{\partial E_a(t+1)}{\partial \hat{J}(\tilde{\mathbf{x}}_{t+1})} \cdot \frac{\partial \hat{J}(\tilde{\mathbf{x}}_{t+1})}{\partial \hat{\mathbf{x}}_{t+1}} \cdot \frac{\partial \hat{\mathbf{x}}_{t+1}}{\partial \mathbf{u}_t} \cdot \frac{\partial \mathbf{u}_t}{\partial \mathbf{w}_a(t)} \\ &= \hat{J}(\tilde{\mathbf{x}}_{t+1}) \cdot \frac{\partial \hat{J}(\tilde{\mathbf{x}}_{t+1})}{\partial \hat{\mathbf{x}}_{t+1}} \cdot \frac{\partial \hat{\mathbf{x}}_{t+1}}{\partial \mathbf{u}_t} \cdot \frac{\partial \mathbf{u}_t}{\partial \mathbf{w}_a(t)} \end{aligned} \quad (15)$$

3. INCREMENTAL GLOBAL DUAL HEURISTIC PROGRAMMING IMPLEMENTATION

Nonlinear function approximators, such as ANNs, have the capability of generalization and approximation. However, considering complexity and nonlinearity of the system, online identification may fail to obtain accurate enough results, especially at the start of training, when the system has not been fully excited. An offline identified model, on the other hand, is less robust when applied to a realistic system with uncertainties. In this section, an incremental technique is introduced to ensure a quick approximation of a locally linearized model (Fig. 3), which also reduces computational burden of the network weight update processes.

3.1 Incremental Model

With the assumption of sufficiently high sampling frequency and slow time-varying dynamics, one can represent a continuous nonlinear plant with a discrete incremental model and retain high enough precision.

Consider a nonlinear continuous system described by:

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t)] \quad (16)$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the current state vector and $\mathbf{u}(t) \in \mathbb{R}^m$ is the current control vector. $f[\mathbf{x}(t), \mathbf{u}(t)] \in \mathbb{R}^n$ provides the physical dynamics of the state vector over time.

The general form can be used to describe dynamic and kinematic equations of complicated aerospace systems.

By taking the first order Taylor series expansion of (16) around time t_0 and omitting higher-order terms, the system is linearized approximately as follows:

$$\begin{aligned}\dot{\mathbf{x}}(t) \approx \dot{\mathbf{x}}(t_0) + \mathbf{F}[\mathbf{x}(t_0), \mathbf{u}(t_0)][\mathbf{x}(t) - \mathbf{x}(t_0)] \\ +\mathbf{G}[\mathbf{x}(t_0), \mathbf{u}(t_0)][\mathbf{u}(t) - \mathbf{u}(t_0)]\end{aligned} \tag{17}$$

where $\mathbf{F}[\mathbf{x}(t_0), \mathbf{u}(t_0)] = \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{x}(t)}|_{\mathbf{x}(t_0), \mathbf{u}(t_0)} \in \mathbb{R}^{n \times n}$ is the system transition matrix of the linearized model and $\mathbf{G}[\mathbf{x}(t_0), \mathbf{u}(t_0)] = \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{u}(t)}|_{\mathbf{x}(t_0), \mathbf{u}(t_0)} \in \mathbb{R}^{n \times m}$ is the control effectiveness matrix. Assuming the states and state derivatives of the system are measurable, i.e. $\Delta\dot{\mathbf{x}}(t)$,$\Delta\mathbf{x}(t)$,$\Delta\mathbf{u}(t)$ are measurable, an incremental model can be used to describe the above system:

$$\Delta\dot{\mathbf{x}}(t) \simeq \mathbf{F}[\mathbf{x}(t_0), \mathbf{u}(t_0)]\Delta\mathbf{x}(t) + \mathbf{G}[\mathbf{x}(t_0), \mathbf{u}(t_0)]\Delta\mathbf{u}(t) \tag{18}$$

With a constant, high data sampling frequency, i.e. the sampling time Δt is sufficiently small, the plant model can be written approximately in a discrete form(Zhou et al. (2018)):

$$\frac{\mathbf{x}_{t+1} - \mathbf{x}_t}{\Delta t} \approx \mathbf{F}_{t-1} \cdot (\mathbf{x}_t - \mathbf{x}_{t-1}) + \mathbf{G}_{t-1} \cdot (\mathbf{u}_t - \mathbf{u}_{t-1}) \tag{19}$$

where $\mathbf{F}_{t-1} = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}|_{\mathbf{x}_{t-1}, \mathbf{u}_{t-1}} \in \mathbb{R}^{n \times n}$ is the system transition matrix and $\mathbf{G}_{t-1} = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}|_{\mathbf{x}_{t-1}, \mathbf{u}_{t-1}} \in \mathbb{R}^{n \times m}$ is the input distribution matrix at time step $t-1$ for the discretized systems. From (19), following incremental form of the new discrete nonlinear system can be obtained:

$$\Delta\mathbf{x}_{t+1} \approx \mathbf{F}_{t-1}\Delta t \cdot \Delta\mathbf{x}_t + \mathbf{G}_{t-1} \cdot \Delta t \cdot \Delta\mathbf{u}_t \tag{20}$$

The continuous nonlinear plant is simplified into a linear incremental dynamic equation and the plant model can be identified online with Recursive Least Squares (RLS) technique. Although some information is omitted, such as state variation related nonlinear terms and higher-order terms in their Taylor series expansion, with the identified $\hat{\mathbf{F}}_{t-1}$ and $\hat{\mathbf{G}}_{t-1}$ matrix, one can predict the next system state with relatively high precision:

$$\hat{\mathbf{x}}_{t+1} = \mathbf{x}_t + \hat{\mathbf{F}}_{t-1} \cdot \Delta t \cdot \Delta\mathbf{x}_t + \hat{\mathbf{G}}_{t-1} \cdot \Delta t \cdot \Delta\mathbf{u}_t \tag{21}$$

3.2 Online Identification using Recursive Least Square

RLS is applied to identify the system transition matrix $\mathbf{F}_{t-1}$ and the input distribution matrix $\mathbf{G}_{t-1}$ online with the assumption of full-state feedback. The incremental form of the states in (20) is:

$$\Delta\mathbf{x}_{t+1} \approx \begin{bmatrix}\Delta\mathbf{x}_t^{\mathrm{T}} & \Delta\mathbf{u}_t^{\mathrm{T}}\end{bmatrix} \cdot \begin{bmatrix}\mathbf{F}_{t-1}^{\mathrm{T}} \\ \mathbf{G}_{t-1}^{\mathrm{T}}\end{bmatrix} \cdot \Delta t \tag{22}$$

Since all increments of the states share the same covariance matrix, the parameters can be identified together as $\Theta_{t-1} = \begin{bmatrix}\mathbf{F}_{t-1}^{\mathrm{T}} \\ \mathbf{G}_{t-1}^{\mathrm{T}}\end{bmatrix} \in \mathbb{R}^{(n+m)\times n}$. Therefore, the state prediction equation (21) can be rewritten as follows:

$$\Delta\hat{\mathbf{x}}_{t+1} = \mathbf{X}_t^{\mathrm{T}} \cdot \hat{\Theta}_{t-1} \cdot \Delta t \tag{23}$$

where $\mathbf{X}_t = \begin{bmatrix}\Delta\mathbf{x}_t \\ \Delta\mathbf{u}_t\end{bmatrix} \in \mathbb{R}^{(n+m)\times 1}$ is the input information of the incremental model, and it is assumed to be measured directly.

The main procedure of the RLS approach is presented as follows:

$$\epsilon_t = \Delta\mathbf{x}_{t+1}^{\mathrm{T}} - \Delta\hat{\mathbf{x}}_{t+1}^{\mathrm{T}} \tag{24}$$

$$\hat{\Theta}_t = \hat{\Theta}_{t-1} + \frac{Cov_{t-1}\mathbf{X}_t}{\gamma_{RLS} + \mathbf{X}_t^{\mathrm{T}} Cov_{t-1}\mathbf{X}_t} \frac{\epsilon_t}{\Delta t} \tag{25}$$

$$Cov_t = \frac{1}{\gamma_{RLS}}\left(Cov_{t-1} - \frac{Cov_{t-1}\mathbf{X}_t\mathbf{X}_t^{\mathrm{T}}Cov_{t-1}}{\gamma_{RLS} + \mathbf{X}_t^{\mathrm{T}}Cov_{t-1}\mathbf{X}_t}\right) \tag{26}$$

where $\epsilon_t \in \mathbb{R}^{1\times n}$ stands for the prediction error, also called *innovation*, $Cov_t \in \mathbb{R}^{(n+m)\times(n+m)}$ is the estimation covariance matrix and it is symmetric and semi-positive definite, and γ_{RLS} is the forgetting factor for this RLS approach.

For most ACD designs, sufficient exploration of the state space guarantees good performance. Although RLS depends less on the global exploration, it is better to satisfy the Persistent Excitation (PE) condition (Zhou et al. (2018)) for identifying the incremental model. A 3211 disturbance signal is introduced to excite the system modes at the start of training.

3.3 Network Update Simplification

Considering (8), the last term $-\gamma\hat{\lambda}(\tilde{\mathbf{x}}_t)\frac{\partial\tilde{\mathbf{x}}_t}{\partial\tilde{\mathbf{x}}_{t-1}}$ needs to be dealt with carefully, because there are two pathways for $\tilde{\mathbf{x}}_{t-1}$ to affect $\tilde{\mathbf{x}}_t$. One is through the model network directly (pathway 2.a), and another one firstly goes through the actor network and then through the model network (pathway 2.b), as shown in both Figs. 1 and 3:

$$\frac{\partial\tilde{\mathbf{x}}_t}{\partial\tilde{\mathbf{x}}_{t-1}} = \frac{\partial\mathbf{x}_t}{\partial\mathbf{x}_{t-1}} = \underbrace{\frac{\partial\mathbf{x}_t}{\partial\mathbf{x}_{t-1}}|_m}_{pathway\ (2.a)} + \underbrace{\frac{\partial\mathbf{x}_t}{\partial\mathbf{u}_{t-1}}|_m \cdot \frac{\partial\mathbf{u}_{t-1}}{\partial\mathbf{x}_{t-1}}|_a}_{pathway\ (2.b)} \tag{27}$$

In conventional GDHP, the two system model derivative terms in (27) are calculated back through the global system model, while IGDHP introduces the identified incremental model information directly to approximate them, partly avoiding complicated computation:

$$\frac{\partial\tilde{\mathbf{x}}_t}{\partial\tilde{\mathbf{x}}_{t-1}} \approx \hat{\mathbf{F}}_{t-1} \cdot \Delta t + \hat{\mathbf{G}}_{t-1} \cdot \Delta t \cdot \frac{\partial\mathbf{u}_{t-1}}{\partial\mathbf{x}_{t-1}}|_a \tag{28}$$

Similarly, the actor weight update process can also be simplified by the incremental information. Specifically, the term $\frac{\partial\hat{\mathbf{x}}_{t+1}}{\partial\mathbf{u}_t}$ in (15) can be approximated by the identified input distribution matrix $\hat{G}_{t-1}$ directly:

$$\frac{\partial\hat{\mathbf{x}}_{t+1}}{\partial\mathbf{u}_t} = \hat{G}_{t-1} \cdot \Delta t \tag{29}$$

Therefore, with the identified system transition matrix $\hat{\mathbf{F}}_{t-1}$ and input distribution matrix $\hat{\mathbf{G}}_{t-1}$, one can simplify the update processes of the critic network and actor network and thus accelerates the learning.

4. NUMERICAL EXPERIMENTS SETUP

4.1 Aerospace System Model

IGDHP can be applied to highly nonlinear aerospace systems of the form:

$$\dot{\mathbf{x}}(t) = f\left[\mathbf{x}(t), \mathbf{u}(t) + \mathbf{d}(t)\right] \tag{30}$$

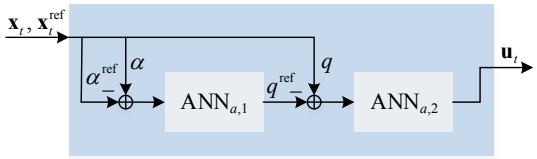

Fig. 4. The architecture of the cascaded actor network (adapted from Zhou et al. (2018))

where $\mathbf{d}(t)$ is the external disturbance and is set to be the excitation noise in this paper.

To verify the proposed method, a second order nonlinear model of a generic surface-to-air missile (Zhou et al. (2017); Sonneveldt (2010)) is introduced, which is a specific example of (30). It consists of the longitudinal force and moment equations, with angle of attack α and pitch rate q as states, and one input: elevator deflection δe. The nonlinear model is simulated around a steady wings-level flight condition in a valid flight envelope of $\alpha \in (-10°, 10°)$ and $M_a \in (1.8, 2.6)$, at an altitude of approximately 6000 meters:

$$\dot{\alpha} = q + \frac{\bar{q}S}{mV_T} C_z(\alpha, q, M_a, \delta_e) \tag{31}$$

$$\dot{q} = \frac{\bar{q}Sd_l}{I_{yy}} C_m(\alpha, q, M_a, \delta_e) \tag{32}$$

where $\bar{q}$, S, m, V_T, d_l, I_{yy} are dynamic pressure, reference area, mass, speed, reference length and pitching moment of inertia respectively and M_a is Mach number, which is set to be 2.0 thereafter. C_z and C_m are the aerodynamic force and moment coefficients. The control surface actuator is modeled as a first order system.

4.2 Network Structure

The ANNs of the actor, critic and global model are fully connected and consist of only one hidden layer. The activation function σ in the nodes of the hidden layer is a sigmoid function:

$$\sigma(o) = \frac{1 - e^{-o}}{1 + e^{-o}} \tag{33}$$

The actor is implemented as a hierarchical structure, or specifically a cascaded actor network (Van Kampen et al. (2006), Zhou et al. (2018)), as shown in Fig. 4. This hierarchical structure takes advantage of the physical properties of the system, by putting some prior knowledge into the design of the controller, which in theory will reduce the complexity of the problem. To improve stability, the output layers of the actor also adopt a sigmoid function as activation function, which is different from the global model and the critic where linear functions are employed, to add restrictions to the pitch rate reference and the control action. The pith rate and the elevator deflection commands are bounded in the range of $[-20°/s, 20°/s]$ and $[-15°, 15°]$ respectively.

The critic and actor networks in both GDHP and IGDHP have the same settings. More neurons will improve approximation precision, but can also increase computational burden or even lead to overfitting. As a trade off, the number of hidden layer neurons in the actor is 6, while

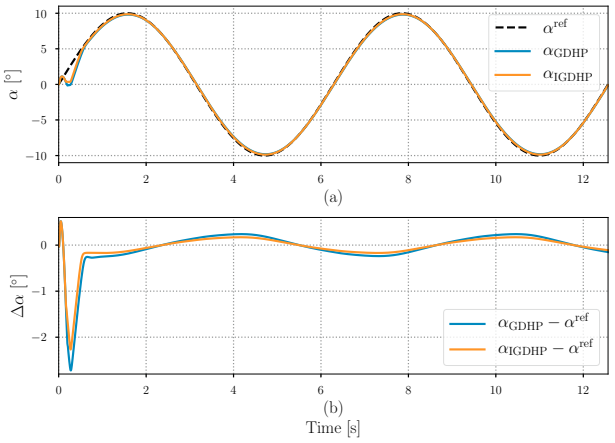

Fig. 5. Online tracking control with zero initial states

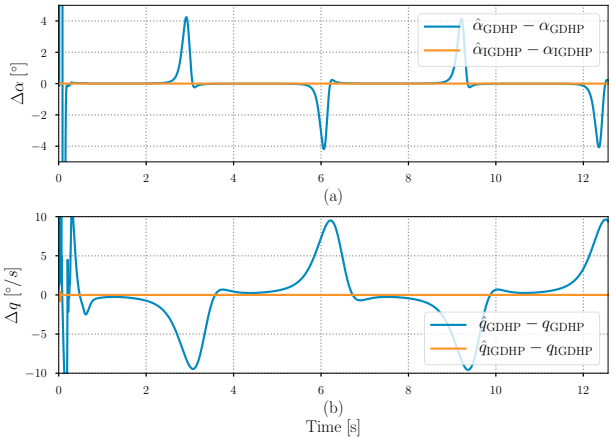

Fig. 6. Prediction errors of the states α and q

in both the critic and the global system it is 12. To guarantee effective learning, learning rates have to be chosen carefully. A descending method is applied, which means that the initial learning rates are set to be large numbers which gradually decrease as the weights are updated.

5. RESULTS AND DISCUSSION

To compare the performance of conventional GDHP and the proposed IGDHP approach, an online tracking control problem is used. The controller has to learn an optimal policy to minimize the cost-to-go J online, so as to control the angle of attack α to track a reference signal α^{ref}, which is a sine function with amplitude of 10 degrees and the period of 2π seconds.

The results of online tracking using GDHP and IGDHP are illustrated in Fig. 5. The global model and incremental techniques are implemented for online dynamic identification and Fig. 6 presents the prediction errors of the states α and q in subplots (a) and (b). IGDHP allows for a more precise identification of the local model, leading to faster learning and smaller tracking error, while the global model converges slower and has less precise results. When the sign of the tracking error changes, variations exist in the outputs of the global model network, leading to larger prediction errors and increased tracking error.

Initial weights of the neural networks have an influence on the learning. In this paper, all weights are randomly

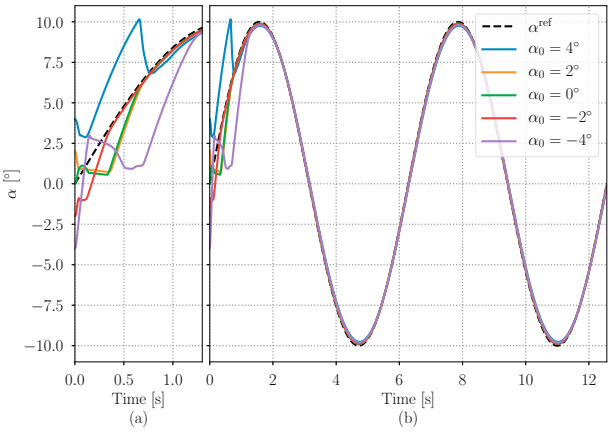

Fig. 7. Online tracking control with different initial states

initialized within $[-0.002, 0.002]$ to reduce the impact of initialization, and bounded within $[-15, 15]$ to prevent sudden failure in the learning process. Nevertheless, bad initialization can still lead to failure. As illustrated in table 1, success ratio (Van Kampen et al. (2006)) is introduced to indicate the performance of the two methods. Keeping all parameters intact ($\alpha_0 = 0°$), the success ratio is 99.4% for IGDHP and merely 37.2% for GDHP, which means that the global model is not robust enough for online application.

Table 1. Effect of initial state on success ratio

$\alpha_0/[°]$	4	2	0	2	4
GDHP	9.4%	11.6%	37.2%	12.3%	8.2%
IGDHP	50.6%	99.1%	99.2%	99.5%	26.7%

The IGDHP approach can deal with different initial α within the range of $[-2°, 2°]$ without apparent loss of accuracy and success rate, as shown in table 1 and Fig. 7. On the other hand, GDHP cannot guarantee the completion of the online tracking task even at the zero initial state.

6. CONCLUSION

This paper develops a novel approach, called Incremental model based Global Dual Heuristic Programming (IGDHP), to generate an adaptive model free flight controller. Different from traditional Global Dual Heuristic Programming (GDHP) designs, which often use an artificial neural network to approximate the system dynamics, IGDHP adopts incremental approaches instead to identify the plant model online. For illustrative validation, both methods are applied to an online tracking problem of a nonlinear second-order aerospace system, whose full dynamics are unknown. The simulation results show that compared to conventional GDHP, IGDHP accelerates the online learning process, improves tracking precision and has apparent advantage in success ratio for a wider range of initial states.

This study generalizes the basic form of the IGDHP but still has limitations for realistic applications. Further research should, therefore, concentrate on the investigation of various critic structures, the improvement of stability and expansion to other application scenarios.

REFERENCES

Ferrari, S. and Stengel, R.F. (2004). Online adaptive critic flight control. *Journal of Guidance, Control, and Dynamics*, 27(5), 777–786.

Liu, D., Wang, D., Zhao, D., Wei, Q., and Jin, N. (2012). Neural-network-based optimal control for a class of unknown discrete-time nonlinear systems using globalized dual heuristic programming. *IEEE Transactions on Automation Science and Engineering*, 9(3), 628–634.

Prokhorov, D.V. and Wunsch, D.C. (1997). Adaptive critic designs. *IEEE transactions on Neural Networks*, 8(5), 997–1007.

Smeur, E.J., Chu, Q., and de Croon, G.C. (2015). Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles. *Journal of Guidance, Control, and Dynamics*, 38(12), 450–461.

Sonneveldt, L. (2010). *Adaptive Backstepping Flight Control for Modern Fighter Aircraft*. Ph.D. thesis, Delft University of Technology.

Sutton, R.S. and Barto, A.G. (2018). *Reinforcement learning: An introduction*. MIT press.

Valadbeigi, A.P., Sedigh, A.K., and Lewis, F.L. (2019). H static output-feedback control design for discrete-time systems using reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*. Early access.

Van Kampen, E., Chu, Q., and Mulder, J. (2006). Online adaptive critic flight control using approximated plant dynamics. In *2006 International Conference on Machine Learning and Cybernetics*, 256–261. IEEE.

Venayagamoorthy, G.K., Harley, R.G., and Wunsch, D.C. (2002). Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator. *IEEE Transactions on Neural Networks*, 13(3), 764–773.

Wang, D. (2019). Intelligent critic control with robustness guarantee of disturbed nonlinear plants. *IEEE transactions on cybernetics*. Early access.

Wang, X., van Kampen, E., Chu, Q., and Lu, P. (2018). Incremental sliding-mode fault-tolerant flight control. *Journal of Guidance, Control, and Dynamics*, 42(2), 244–259.

Yi, J., Chen, S., Zhong, X., Zhou, W., and He, H. (2019). Event-triggered globalized dual heuristic programming and its application to networked control systems. *IEEE Transactions on Industrial Informatics*, 15(3), 1383–1392.

Zhou, Y., van Kampen, E., and Chu, Q. (2016). Incremental model based heuristic dynamic programming for nonlinear adaptive flight control. In *Proceedings of the International Micro Air Vehicles Conference and Competition 2016, Beijing, China*.

Zhou, Y., van Kampen, E., and Chu, Q. (2017). Launch vehicle adaptive flight control with incremental model based heuristic dynamic programming. In *68th International Astronautical Congress (IAC), Adelaide, Australia*.

Zhou, Y., van Kampen, E., and Chu, Q. (2018). Incremental model based online dual heuristic programming for nonlinear adaptive control. *Control Engineering Practice*, 73, 13–25.