Adaptive Dynamic Programming for Flight Control

van Kampen, Erik Jan; Sun, Bo

DOI
10.1007/978-3-031-39767-7_10

Publication date
2023

Document Version
Final published version

Published in
Advances in Industrial Control

Citation (APA)
van Kampen, E. J., & Sun, B. (2023). Adaptive Dynamic Programming for Flight Control. In A. L'Afflitto, G. Inalhan, & H.-S. Shin (Eds.), *Advances in Industrial Control* (pp. 269-292). (Advances in Industrial Control; Vol. Part F1768). Springer. https://doi.org/10.1007/978-3-031-39767-7_10

Chapter 10
Adaptive Dynamic Programming for Flight Control

Erik-Jan van Kampen and Bo Sun

10.1 Introduction to Adaptive Dynamic Programming

Optimality is a property that a control system should always pursue. A generic way to solve optimal control problems is involving the Hamilton–Jacobi–Bellman (HJB) equation. However, for nonlinear systems, it is, in general, impossible to find an analytical solution to the HJB equation [1, 2]. Nevertheless, adaptive dynamic programming (ADP) offers a promising tool to attain satisfying numerical solutions by incorporating artificial neural networks (ANNs). As its name suggests, ADP not only inherits the capability of the classic dynamic programming for pursuing optimality but also empowers the controller to be adaptive. Through iterations between policy improvement and policy evaluation, ADP can address optimal control problems in a numerical way. Wang et al. [3] provided a comprehensive survey on ADP and also discusses the many names under which these methods are known, such as Approximate Dynamic Programming, Neuro-Dynamic Programming, Adaptive Critic Designs, and RL. Although there are differences in the meanings of these terms, they have all at some point been used to describe the same class of methods. In this chapter, the term ADP will be used, but we consider it a form of reinforcement learning (RL) because ADP essentially maximizes returns [4, 5].

With the development of technology, the complexity of aircraft has largely increased. For example, making aviation sustainable has become a popular topic in recent days. To achieve this goal, wings with morphing functions [6] or large aspect ratios [7] are usually designed, which, however, involves more states that introduce nonlinearities, uncertainties, and constraints. An equilibrium for different

E.-J. van Kampen (✉) · B. Sun
Delft University of Technology, Delft, Netherlands
e-mail: E.vanKampen@tudelft.nl

B. Sun
e-mail: B.Sun-1@tudelft.nl

A. L'Afflitto et al. (eds.), *Control of Autonomous Aerial Vehicles*, Advances in Industrial Control, https://doi.org/10.1007/978-3-031-39767-7_10

demands should be attained, which becomes an optimal control problem. With its power of approximation and interaction, ADP is promising to overcome these new challenges.

This section presents some of the basic derivations of ADP, which will be used as a starting point for derivations of more advanced ADP methods in Sect. 10.3. Consider a generic continuous, nonlinear, affine in the control system

$$\dot{x}(t) = f(x(t)) + g(x(t))\, u(t), \tag{10.1}$$

where $x(t) \in \mathbb{R}^n$ denotes the *state vector*, $u(t) \in \Omega_u \subset \mathbb{R}^m$ the *control vector*, and f and g are differentiable functions describing the effect of the state and input and the state derivatives.

Based on optimal control theory, the *utility function* U is introduced to account for the cost of states and inputs

$$U(x(t), u(t)) \triangleq Q(x(t)) + u^{\mathrm{T}}(t)\, Ru(t). \tag{10.2}$$

The infinite-horizon cost function related to this utility function should be minimized, leading to the *optimal cost function*

$$J^*(x) \triangleq \min_{u \in A(\Omega)} \int_t^\infty U(x(\tau), u(\tau))\, d\tau. \tag{10.3}$$

Bellman's optimality principle states that the admissible control that achieves this optimal value is the one that sets the Hamiltonian of system (10.1)

$$H(x, u(x), \nabla J(x)) \triangleq U(x, u(x)) + (\nabla J(x))^{\mathrm{T}} [f(x) + g(x)\, u(x)] \tag{10.4}$$

to zero, that is,

$$u^*(x) = \arg\min_{u \in A(\Omega)} H\left(x, u(x), \nabla J^*(x)\right) = -\frac{1}{2} R^{-1} g^{\mathrm{T}}(x)\, \nabla J^*(x). \tag{10.5}$$

The cost or value function J is approximated by the critic, often in the form of an artificial neural network (ANN). Instead of the analytical expression for the control action, as given in (10.5), an actor ANN is often used to approximate the control law and to reduce model dependency. Together with a model of the plant that is to be controlled, both the actor and the critic form the basic framework for ADP, as shown in Fig. 10.1.

The structure of the remainder of this chapter is as follows. First, some early applications of ADP to flight control will be presented in Sect. 10.2. The main contributions of this chapter are presented in Sect. 10.3, which will describe four recent developments in ADP applied to flight control, namely, the use of incremental models (Sect. 10.3.1), the analytical approach to ADP (XGDHP) (Sect. 10.3.2), how

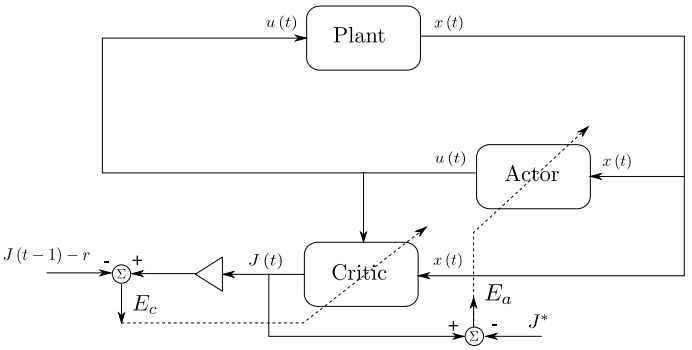


Fig. 10.1 Basic actor–critic ADP framework: action-dependent heuristic dynamic programming (AD-HDP)

to deal with input constraints (Sect. 10.3.3), and an event-triggered ADP control (Sect. 10.3.4). The chapter is concluded by Sect. 10.4.

10.2 Early Applications of ADP to Flight Control

This section will show some of the early applications of ADP methods to flight control tasks. These applications form the foundation of the recent developments that will be shown in Sect. 10.3. One of the early works on applications of ADP to flight control is a paper by Enns and Si on Neuro-Dynamic Programming applied to helicopter flight control [8]. Although they do not use the name ADP themselves, and opt for *Neuro-Dynamic* to stress the use of artificial neural networks as function approximators, it is a classic AD-HDP approach, in which the actor output is fed into the critic for easier back-propagation through the critic to update the actor weights. This is a model-free approach, which does not require a model of the plant in the control loop. A block diagram of their approach is shown in Fig. 10.2. The controller

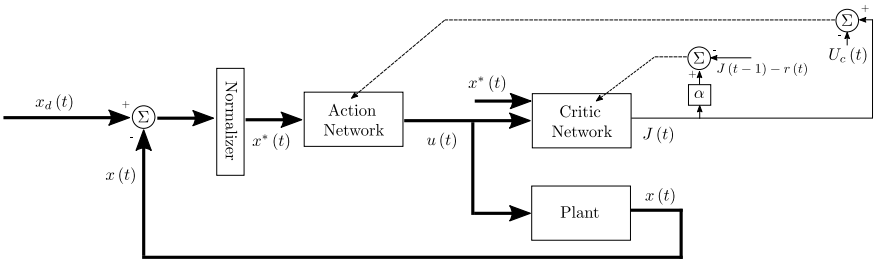


Fig. 10.2 Action-dependent heuristic dynamic programming architecture for helicopter control. For additional details on this architecture, see [8]

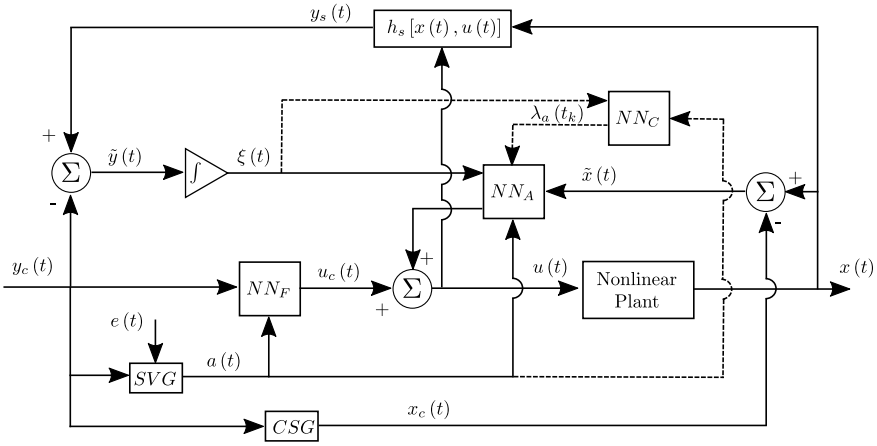


Fig. 10.3 Dual heuristic dynamic programming architecture for fixed-wing aircraft control. The scheduling variable generator (SVG) produces the significance scheduling vector and the command state generator (CSG) provides secondary state elements that are compatible with the reference. For additional details on this architecture, see [9]

is applied to stabilizing tasks, i.e., tasks where the aim is to drive a set of states to zero. The controller is trained offline, with success rates varying between 78 and 96%, depending on the task. After training, the weights of a successful run are frozen and an online evaluation takes place.

Another key publication in the history of ADP for flight control is the work by Ferrari and Stengel on the application of ADP for control of a business jet-type aircraft model [9]. The work from Ferrari and Stengel is different from the work from Enns and Si in several important aspects. First of all, they use a more advanced form of ADP, Dual Heuristic Dynamic Programming, in which the critic network approximates the derivative of the value function with respect to the state λ, instead of the value function itself, see Fig. 10.3. Secondly, Ferrari and Stengel do not have the action dependency in the critic, and, hence, model information, in the form of transition matrices, is needed to update the actor network. Another difference with previous work is that the controller is not just used for stabilization, but also for tracking reference signals.

A final example of some of the early work on ADP for flight control can be found in [10]. In this paper, a direct comparison is made between action-dependent and action-independent heuristic dynamic programming applied to a simulation model of the F-16 fighter aircraft. By removing the action dependency, i.e., by cutting the back-propagation path through the critic to the actor, an approximated plant model is required, as in the work from Ferrari and Stengel. The results in [10] show that the success rate for convergence was nearly doubled by removing the action dependency. It also shows an improvement in tracking performance for the action-independent HDP approach. The three examples of early ADP applied to flight control, which

were shown in this section, have shaped a lot of the later developments, some of which are discussed in Sect. 10.3.

10.3 Recent Developments in ADP

This section describes four recent developments in ADP applied to flight control. The use of incremental models and the analytical approach to ADP will be introduced in Sects. 10.3.1 and 10.3.2, respectively. Then, the improvement of the recent ADP in dealing with input constraints and incorporating event-triggered control (ETC) will be presented in Sects. 10.3.3 and 10.3.4, respectively.

10.3.1 Incremental Model

ADP can be model free if it is action dependent, which means the control signals are also introduced as the inputs of the critic network [11]. Nevertheless, to achieve a model-free application, an alternative is building a third module to approximate the plant dynamics, and ANNs are often regarded as the first choice [10]. The authors in [10] show that this three-network structure outperforms the action-dependent approach if rewards only depend on system states. Although ANNs can approximate the nonlinear function with arbitrary precision, many samples are required before the weights converge for online identification of complex plant dynamics, such as aerospace systems, which can be dangerous especially at the start of training because the critic and actor networks are then trained based on the incorrect model. For these complex systems, offline training is normally involved to obtain a primary model and it often remains constant in applications [10], which, however, cannot achieve adaptive control when facing unforeseen uncertainties and sudden disturbances in realistic application [12]. Modern processors work in a discrete way, leading to discrete measurements and computations. With the assumption of sufficiently high sampling frequency and relatively slow time-varying dynamics, one can represent a continuous nonlinear plant with a discrete incremental model and retain high enough precision [13].

By taking the first-order Taylor series expansion of (10.1) around time t_0 and omitting higher order terms, the system is linearized approximately as follows:

$$\dot{x}(t) \approx \dot{x}(t_0) + F(x(t_0), u(t_0))(x(t) - x(t_0)) + G(x(t_0), u(t_0))(u(t) - u(t_0)), \tag{10.6}$$

where

$$F(x(t_0), u(t_0)) = \left.\frac{\partial f(x(t), u(t))}{\partial x(t)}\right|_{x(t_0),u(t_0)}, \tag{10.7a}$$

$$G(x(t_0), u(t_0)) = \left.\frac{\partial f(x(t), u(t))}{\partial u(t)}\right|_{x(t_0),u(t_0)}, \tag{10.7b}$$

$F(x(t_0), u(t_0)) \in \mathbb{R}^{n\times n}$ denotes the *system matrix* and $G(x(t_0), u(t_0)) \in \mathbb{R}^{n\times m}$ denotes the *control effectiveness matrix*. Assuming the states and state derivatives of the system are measurable, i.e., $\Delta\dot{x}(t)$, $\Delta x(t)$ and $\Delta u(t)$ are measurable, the following incremental model can be used to describe the above system:

$$\Delta\dot{x}(t) \simeq F(x(t_0), u(t_0))\Delta x(t) + G(x(t_0), u(t_0))\Delta u(t). \tag{10.8}$$

With a constant, high sampling frequency, i.e., if the sampling time Δt is sufficiently small, then the plant model can be written approximately in the discrete form

$$\frac{x_{t+1} - x_t}{\Delta t} \approx F_{t-1}(x_t - x_{t-1}) + G_{t-1}(u_t - u_{t-1}), \tag{10.9}$$

where $F_{t-1} \triangleq \left.\frac{\partial f(x,u)}{\partial x}\right|_{x_{t-1},u_{t-1}} \in \mathbb{R}^{n\times n}$ denotes the *system transition matrix* and $G_{t-1} \triangleq \left.\frac{\partial f(x,u)}{\partial u}\right|_{x_{t-1},u_{t-1}} \in \mathbb{R}^{n\times m}$ denotes the *input distribution matrix* at the time step $t-1$ for the discretized system. From (10.9), the following incremental form of the new discrete nonlinear system can be obtained as follows:

$$\Delta x_{t+1} \approx F_{t-1}\Delta t\,\Delta x_t + G_{t-1}\Delta t\,\Delta u_t. \tag{10.10}$$

This way, the continuous nonlinear global plant is simplified into a linear incremental dynamic equation. The resulting local plant model can be identified online by the recursive least squares (RLS) technique, to take advantage of its adaptability to cope with time variations in the regression parameters and fast convergence speed, so as to avoid training a complex ANN [13]. Although some information is omitted, such as state variation-related nonlinear terms and higher order terms in their Taylor series expansion, with the identified $\hat{F}_{t-1}$ and $\hat{G}_{t-1}$ matrix, the next system state can be predicted by

$$\hat{x}_{t+1} = x_t + \hat{F}_{t-1}\Delta t\,\Delta x_t + \hat{G}_{t-1}\Delta t\,\Delta u_t. \tag{10.11}$$

The combination of the incremental model with HDP and DHP leads to IM-HDP [15] and IM-DHP [13, 14, 16–18], respectively. The structural diagram of IM-DHP is illustrated in Fig. 10.4 [14], where the discrete-time (DT) state x_t is represented by s_t. Both IM-HDP and IM-DHP have successfully been applied to a variety of aerospace systems, including launch vehicles, satellites, airplanes, etc. In particular,

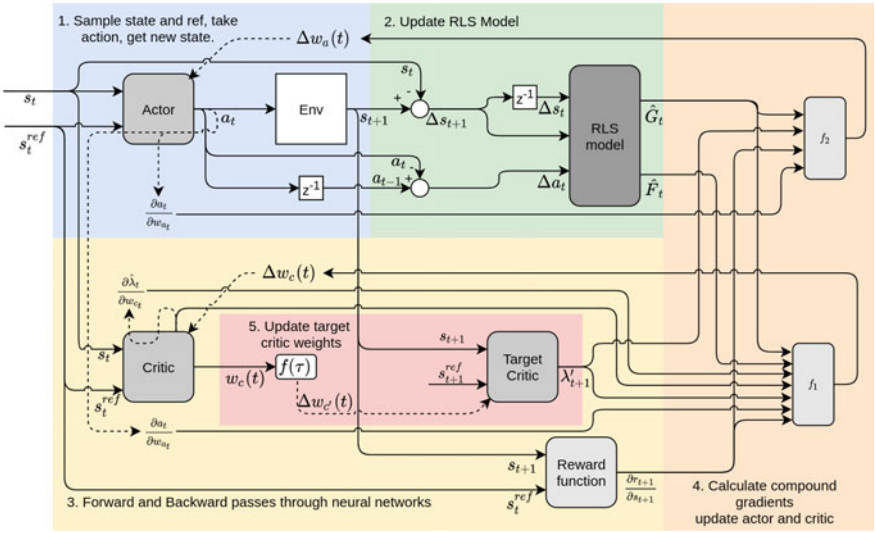


Fig. 10.4 Flowchart of the information during a single complete time step of the IM-DHP learning framework. Solid lines capture feedforward information, while dashed lines indicate feedback update paths For additional details on this architecture, see [14]

Fig. 10.5 The PH-LAB research aircraft operated by Delft University of Technology. The Cessna Citation 550 is a CS-25 certified aircraft

the application of IM-DHP in Cessna Citation 550 (shown in Fig. 10.5) has been gaining attention [16, 18], and, therefore, is selected to demonstrate the applicability of the incremental model-based ADP.

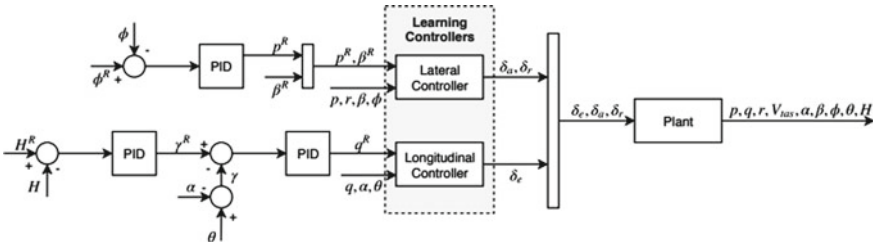


Fig. 10.6 General layout of the flight controller. The inner control loop consists of the reinforcement learning agent and provides body rate control. The outer loop consists of PID controllers converting the desired altitude and bank angle to body rate references for the inner control loop. For additional details on this architecture, see [18]

The adaptive learning framework is applied to the pitch and roll rate, augmented with an outer control loop, as illustrated in Fig. 10.6. Rate control exhibits the lowest learning complexity due to the direct dynamic relation between the angular rates and control surfaces. The outer control loop consists of conventional proportional–integral–derivative (PID) controllers and provides a higher level control interface, which enables reference tracking of an altitude and roll angle profile. Under the assumption of a symmetric aircraft, a decoupled design is employed for longitudinal and lateral learning controllers. Furthermore, the aircraft's yaw damper is disabled to provide the agent with full control authority over the control surfaces. The engine's thrust setting is controlled by an internal airspeed controller. The simulation model is run with a sampling frequency of 50 Hz.

The online learning simulation is conducted at the trimmed condition where the true airspeed is 90 m/s and the altitude is 2 km. More detailed settings can be found in [18]. It is noteworthy that persistent excitation (PE) is essential to both state-space exploration in the learning process and dynamic excitation in the system identification process of the incremental model [12]. Exponentially decaying, sinusoidal excitation is applied to the elevator and ailerons to excite the system during the initial training phase. As the agent learns to track the dynamic reference signals, the excitation on the elevator and ailerons is reduced. As illustrated in Fig. 10.7, both the longitudinal and lateral controllers are able to follow the reference signals after less than 30 s of training.

10.3.2 XGDHP

The main character of the (X)GDHP technique is that it makes use of both the cost function and its derivative information to update the critic network. The structural diagram of the present XGDHP implementation that is combined with IM is depicted in Fig. 10.8.

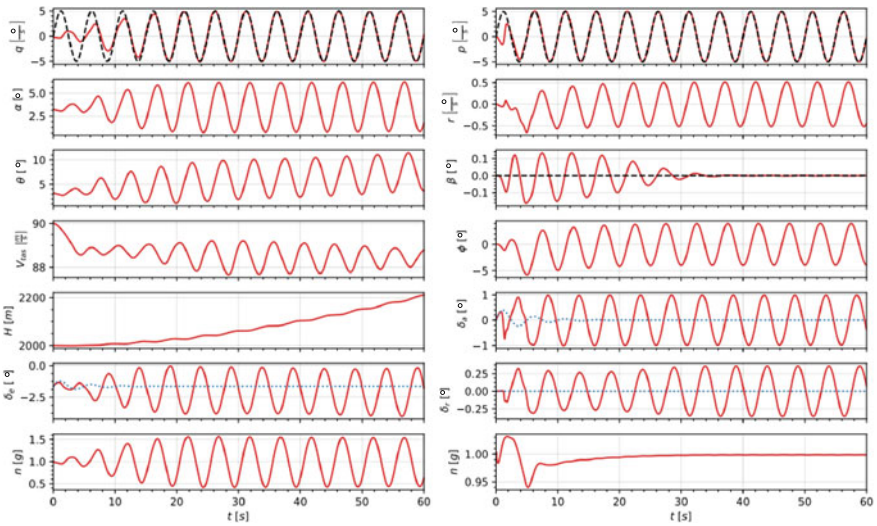


Fig. 10.7 Online training procedures of longitudinal (left) and lateral (right) controller starting at the trimmed operation condition. Reference and excitation signals are illustrated by black dashed and blue dotted lines, respectively. For additional details, see [18]

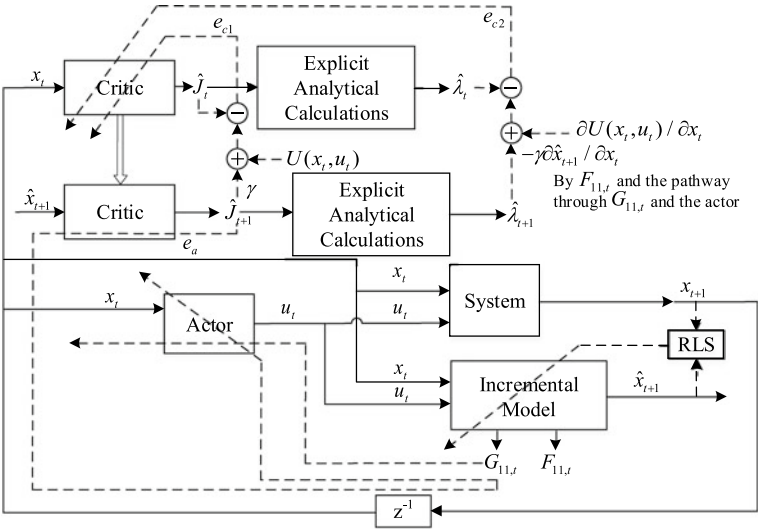


Fig. 10.8 The architecture of the IM-XGDHP algorithm, where solid lines represent the feedforward flow of signals, dashed lines are back-propagation pathways, and the thick arrow represents the weight transmission. For additional details on this architecture, see [19]

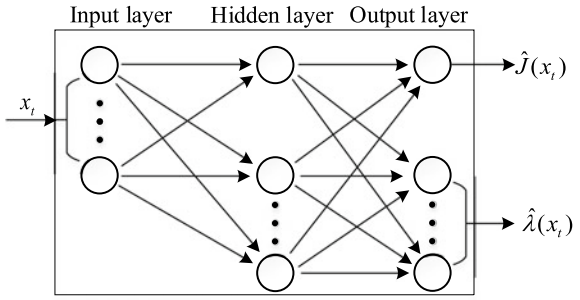


Fig. 10.9 Traditional straightforward critic network

As depicted in Fig. 10.9, for the conventional straightforward GDHP technique, the critic network outputs the approximation of cost function and its derivatives simultaneously [20], whose description is given by

$$\begin{bmatrix} \hat{J}(x_t) \\ \hat{\lambda}(x_t) \end{bmatrix} = \begin{bmatrix} \hat{w}_{c2,J} \\ \hat{w}_{c2,\lambda} \end{bmatrix}^{\mathrm{T}} \sigma\left(\hat{w}_{c1}^{\mathrm{T}} x_t\right), \tag{10.12}$$

where $\hat{w}_{c1} \in \mathbb{R}^{n \times l_c}$, $\hat{w}_{c2,J} \in \mathbb{R}^{l_c}$, and $\hat{w}_{c2,\lambda} \in \mathbb{R}^{l_c \times n}$, respectively, denote the estimation of the ideal weights $w_{c1} \in \mathbb{R}^{n \times l_c}$ $w_{c2,J} \in \mathbb{R}^{l_c}$, and $w_{c2,\lambda} \in \mathbb{R}^{l_c \times n}$, and $\sigma(\cdot)$ is the *activation function* in the hidden layer. In this structure, two kinds of outputs share the same inputs and hidden layer but have different pathways between the hidden layer and the output layer. However, this sharing does not make any physical sense but only makes them strongly coupled, which is undesirable. Indeed, in general, the weights update processes for two kinds of outputs should be relatively independent. Besides, due to the inevitable approximation error, $\hat{J}(x_t)$ and $\hat{\lambda}(x_t)$ approximated in this way cannot exactly provide the derivative relationship, which is called suffering from the *inconsistency error* [12].

Therefore, inspired by [12, 19], a novel XGDHP technique that takes the advantage of explicit analytical calculations is developed. The architecture of the critic network of the XGDHP technique is illustrated in Fig. 10.10, where the critic network only approximates the cost function as

$$\hat{J}(x_t) = \hat{w}_{c2}^{\mathrm{T}} \sigma(\hat{w}_{c1}^{\mathrm{T}} x_t), \tag{10.13}$$

where $\hat{w}_{c2} \in \mathbb{R}^{l_c}$ denotes the estimation of the ideal weight matrix $w_{c2} \in \mathbb{R}^{l_c}$. By taking the explicit analytical calculations, we obtain that

$$\hat{\lambda}(x_t) = \frac{\partial \hat{J}(x_t)}{\partial x_t} = w_{c1}\left(\hat{w}_{c2} \odot \sigma'(w_{c1}^{\mathrm{T}} x_t)\right), \tag{10.14}$$

where $\odot$ denotes the *Hadamard product*.

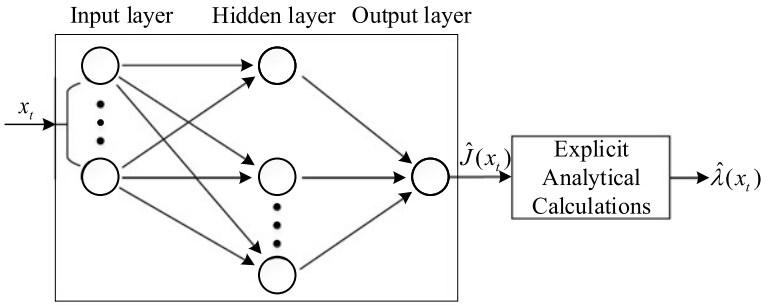


Fig. 10.10 Critic network of the XGDHP technique. For additional details, see [12]

XGDHP makes use of the cost function and its derivative information, and, therefore, the critic network is expected to minimize the performance measure

$$E_{c,t} = \beta \frac{1}{2} e_{c1,t}^2 + (1-\beta)\frac{1}{2} e_{c2,t}^{\mathrm{T}} e_{c2,t}, \tag{10.15}$$

where

$$e_{c1,t} = \hat{J}(x_t) - U(x_t, u(x_t)) - \gamma \hat{J}(\hat{x}_{t+1}), \tag{10.16a}$$

$$e_{c2,t} = \frac{\partial[\hat{J}(x_t) - U(x_t, u(x_t)) - \gamma \hat{J}(\hat{x}_{t+1})]}{\partial x_t}, \tag{10.16b}$$

and $\beta \in [0, 1]$. If $\beta = 1$, then XGDHP becomes pure HDP, and if $\beta = 0$, then the weight matrix is tuned merely based on the computed derivatives $\hat{\lambda}(x_t)$ and, consequently, XGDHP is equivalent to DHP [19].

For precise calculations, the partial derivative of $u(x_t)$ with respect to x_t should also be into consideration in the critic network updating procedure. Thus, define $U(x_t, u(x_t))$ as in (10.2), in which $Q(x_t) = x^{\mathrm{T}} Q x$. By applying the chain rule, it follows from (10.16b) that

$$\begin{aligned} e_{c2,t} = {} & \hat{\lambda}(x_t) - 2Qx_t - \frac{\partial u(x_t)}{\partial x_t} \cdot 2Ru(x_t) \\ & - \gamma \left(\frac{\partial \hat{x}_{s_k+1}}{\partial x_t} + \frac{\partial u(x_t)}{\partial x_t} \frac{\partial \hat{x}_{s_k+1}}{\partial u(x_t)} \right) \hat{\lambda}(x_{s_k+1}), \end{aligned} \tag{10.17}$$

where $\partial u(x_t)/\partial x_t$ is computed with the facilitation of the actor network, and $\partial \hat{x}_{s_k+1}/\partial x_t$ and $\partial \hat{x}_{s_k+1}/\partial u(x_t)$ are computed through the model network.

Given a learning rate $\eta_c > 0$, the weight updating algorithm is given by

$$\Delta \hat{w}_{c2} = -\eta_c \frac{\partial E_{c,t}}{\partial \hat{w}_{c2}}, \ \Delta \hat{w}_{c1} = -\eta_c \frac{\partial E_{c,t}}{\partial \hat{w}_{c1}}, \tag{10.18}$$

and

$$\frac{\partial E_{c,t}}{\partial \hat{w}_{c2}} = \beta \frac{\partial \hat{J}(x_t)}{\partial \hat{w}_{c2}} e_{c1,t} + (1-\beta) \frac{\partial \hat{\lambda}(x_t)}{\partial \hat{w}_{c2}} \mathrm{e}_{c2,t}, \tag{10.19a}$$

$$\frac{\partial E_{c,t}}{\partial \hat{w}_{c1}} = \beta \frac{\partial \hat{J}(x_t)}{\partial \hat{w}_{c1}} e_{c1,t} + (1-\beta) \frac{\partial \hat{\lambda}(x_t)}{\partial \hat{w}_{c1}} \mathrm{e}_{c2,t}, \tag{10.19b}$$

where $\partial \hat{\lambda}(x_t)/\partial \hat{w}_{c2}$ and $\partial \hat{\lambda}(x_t)/\partial \hat{w}_{c1}$ are the second-order mixed gradients of the cost function $\hat{J}(x_t)$, and are computed as

$$\frac{\partial \hat{\lambda}(x_t)}{\partial \hat{w}_{c2}} e_{c2,t} = (\hat{w}_{c1}^{\mathrm{T}} \mathrm{e}_{c2,t}) \odot \sigma'(\hat{w}_{c1}^{\mathrm{T}} x_t), \tag{10.20a}$$

$$\frac{\partial \hat{\lambda}(x_t)}{\partial \hat{w}_{c1}} e_{c2,t} = e_{c2,t}(\hat{w}_{c2} \odot \sigma'(\hat{w}_{c1}^{\mathrm{T}} x_t))^{\mathrm{T}} - x_t \left(\hat{w}_{c1}^{\mathrm{T}} x_t \odot \hat{w}_{c2} \odot \sigma(\hat{w}_{c1}^{\mathrm{T}} x_t) \odot \sigma'(\hat{w}_{c1}^{\mathrm{T}} x_t) \right)^{\mathrm{T}}, \tag{10.20b}$$

respectively [21]. The computation method in (10.20) is simpler than the approach presented in [12, 19, 22], where the Kronecker product and tensor operations are involved, and matrix dimensionality transformation is required. Through mathematical derivation, it can be found that (10.20) is equivalent to the method developed in [12, 19, 22].

The above introduction is presented based on the stabilization control problem. The IM-XGDHP approach can also be applied to tracking control problems such as the angle of attack (AOA) tracking problem of the F-16 Fighting Falcon aircraft [12]. In this case, the input to the networks and the identifier is the tracking error instead of the system state x_t. The AOA reference signal varies around the trimmed condition, namely, 2.6638°. The partial observability (PO) condition is taken into consideration, i.e., the pitch rate q is not directly measurable. To handle this PO problem, an augmented IM is adopted by extending IM with previous controls and observations. All simulations are performed with a sampling frequency of 1 kHz and using Euler's method. In order to achieve the PE condition, a 3211 disturbance signal [12, 13] is introduced as the exploration signal at the initial exploration stage. Furthermore, it is noted that the learning process is performed totally online. More detailed settings can be found in [19]. Although the initial condition can have an impact on the controller, the IM-XGDHP-PO approach can deal with a wide range of initial states within $[-10, 15]^\circ$ without loss of precision. As presented in Fig. 10.11, the AOA can track the given reference signal α^{ref} in less than 2 s for all initial conditions using the IM-XGDHP-PO approach, which is indicative of its competent adaptability and robustness.

Nevertheless, only when the task is successfully performed, the results presented above make sense. Random factors, such as initial weights of ANNs and measurement noises, can have impacts on performance and, occasionally, even trigger divergence and failure. A concept of success ratio is therefore introduced as a performance index [19] to show the robustness of the IM-XGDHP-PO by comparing it with XGDHP-

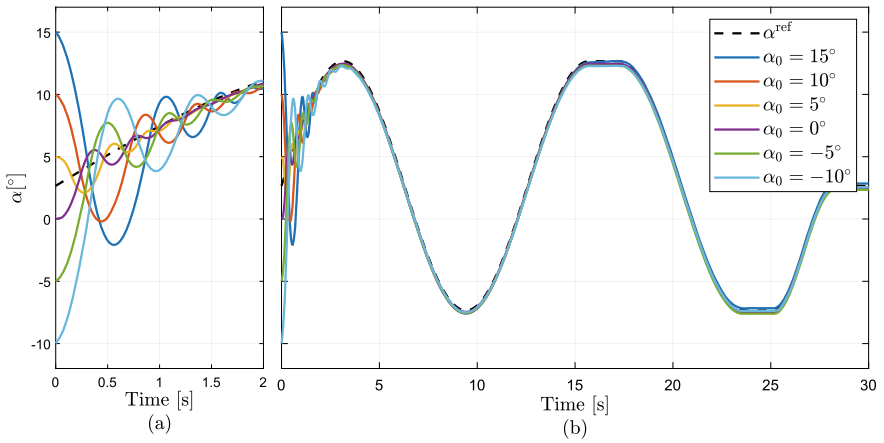


Fig. 10.11 Online AOA tracking control with different initial states using the IM-XGDHP-PO approach

Table 10.1 Success ratio comparison for different initial states with 1000 times of Monte Carlo simulation

α_0 [°]	−10	−5	0	2.6638[a]	5	10	15
IM-XGDHP-FSF	100%	64.5%	78.5%	100%	100%	99.1%	99.4%
IM-XGDHP-PO	92.7%	46.3%	73.5%	100%	99.3%	99.2%	99.6%
XGDHP-PO	25.7%	38.2%	45.3%	51.2%	44.3%	41.4%	50.5%

[a] AOA value starting at the trimmed condition

PO using a model network and IM-XGDHP with full state feedback (FSF). One thousand Monte Carlo simulations are executed with seven different initial AOAs and equal reference to evaluate the robustness of these approaches toward initial tracking errors. The results regarding success ratio are illustrated in Table 10.1.

Both IM-XGDHP-FSF and IM-XGDHP-PO have a success ratio of 100% starting at the trimmed condition, which implies these approaches are stable for this tracking control problem. Thanks to the fast convergence and accurate approximation of IM, both IM approaches outperform the XGDHP-PO which uses a model network to approximate system dynamics. However, it should also be noted that in multiple cases the success ratio is not 100%, which is mainly owing to the fact that it is arduous to accomplish optimal PE condition due to the circular argument between PE condition, accurate system information, and stable control policy [12]. Nevertheless, there is still a prospect of full success and the results presented in Table 10.1 are obtained based on current settings. The development of various aspects can benefit the stability and improve the success ratio, such as sensor precision, exploration signal, parameter initialization, and learning rates.

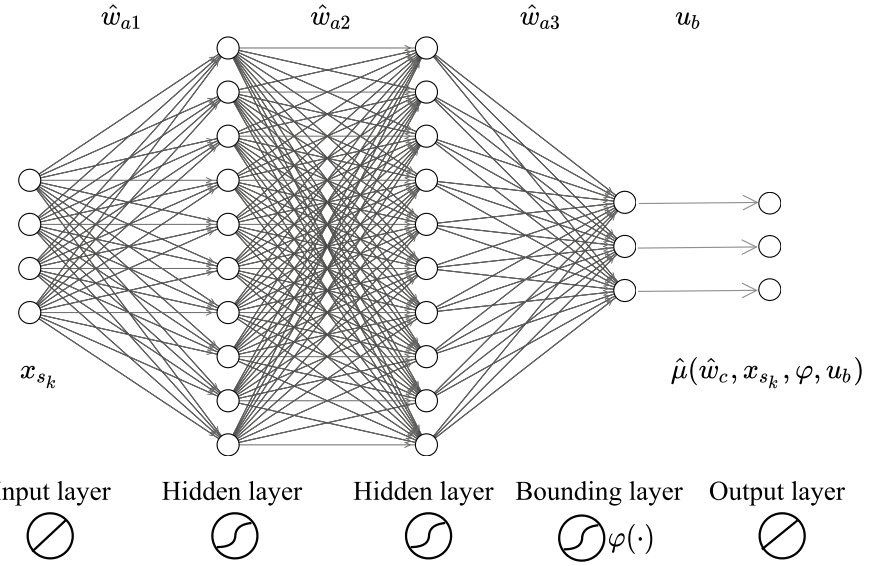


Fig. 10.12 The structure of the actor network, where $\hat{w}_{a1}$, $\hat{w}_{a2}$, and $\hat{w}_{a3}$ are the weight matrices to be updated. It is assumed that there are 4 system states and 3 control inputs, and the number of neurons in both hidden layers is 10

10.3.3 Input Constraints

Saturation constraints commonly exist in aerospace systems [22], but are rarely tackled in the traditional optimal control because of their nonlinear nature. By incorporating ANNs, ADP acquires a more powerful generalization capability and can deal with input constraints as well. For DT systems, the actor–critic scheme is adopted, and an actor network is required. A bounding layer can be added to the actor output to improve the actor network such that the outputted control command can be bounded by the activation function of the bounding layer [12, 19] as shown in Fig. 10.12.

However, for continuous-time (CT) systems, by solving the HJB equation, the single critic network architecture is able to perform ADP with lower computational cost and eliminate the approximation error introduced by the actor network [23]. Different from the actor–critic architecture, where the input saturation constraints are addressed by the bounded actor neurons, the single critic network structure ordinarily utilizes a non-quadratic cost function, such that the control inputs derived from the solution of the HJB equation can be bounded by a hyperbolic tangent function [23].

For the CT system (10.1), assume that $u(x(t)) \in \Omega_u$ and $\Omega_u = \{u | u \in \mathbb{R}^m, |u_i| < u_b, i = 1, \ldots, m\}$, where u_b denotes the *control saturating bound*. For simplicity, the input constraints are assumed identical and symmetric for each input element in this chapter. Different and asymmetric input constraints are introduced in [22] and [21], respectively. To deal with input constraints, a new infinite-horizon cost function

is defined for system (10.1) as

$$J(x) = \int_t^{\infty} x^{\mathrm{T}} Q x + Y(u) \mathrm{d}\tau, \tag{10.21}$$

where $Q \in \mathbb{R}^{n \times n}$ is positive semi-definite and is set to be a diagonal matrix in this chapter, and $Y(u)$ is a positive semi-definite integrand function utilized to handle control input constraints. We choose $U(x, u(x)) = x^{\mathrm{T}} Q x + Y(u)$ as the utility function, and $U(x, u(x))$ satisfies $U(x, u) \geq 0$ and $U(0, 0) = 0$. Therefore, we define

$$Y(u) \triangleq 2u_b \int_0^u \tanh^{-\mathrm{T}}(\upsilon/u_b) R \mathrm{d}\upsilon = 2u_b \sum_{i=1}^{m} \int_0^{u_i} \tanh^{-\mathrm{T}}(\upsilon_i/u_b) r_i \mathrm{d}\upsilon_i, \tag{10.22}$$

where $\tanh^{-\mathrm{T}}(\cdot)$ stands for $(\tanh^{-1}(\cdot))^{\mathrm{T}}$, and $\tanh^{-1}(\cdot)$ denotes the *inverse hyperbolic tangent function*, which is a monotonic odd function; $R = \mathrm{diag}([r_1, \cdots, r_m]) \in \mathbb{R}^{m \times m}$ is a positive-definite weight matrix, where $\mathrm{diag}(\cdot)$ reshapes the vector to a diagonal matrix; remarkably $Y(u) \geq 0$, and $Y(u) = 0$ only if $u = 0$.

The Hamiltonian and the optimal cost function $J^*(x_t)$ still take forms as (10.4) and (10.3), respectively. Applying the first-order optimality condition

$$\partial H\left(x, u(x), \nabla J(x)\right) / \partial u(x) = 0,$$

we deduce that the corresponding optimal feedback control solution is given by

$$\begin{aligned} u^*(x) &= \arg\min_{u(x) \in \Omega_u} H(x, u(x), \nabla J^*(x)) \\ &= -u_b \tanh(D^*), \end{aligned} \tag{10.23}$$

where $\tanh(\cdot)$ denotes the *hyperbolic tangent function*, and D^* is given by

$$D^* = \frac{1}{2u_b} R^{-1} g^{\mathrm{T}}(x) \nabla J^*(x). \tag{10.24}$$

The control input u^* is bounded by u_b, and the non-quadratic cost (10.22) evaluated at u^* is given by

$$Y(u^*(x)) = u_b \nabla J^{*\mathrm{T}}(x) g(x) \tanh(D^*) + u_b^2 \underline{R} \ln\left(\underline{\mathbf{1}} - \tanh^2(D^*)\right), \tag{10.25}$$

where $\nabla J^{*\mathrm{T}}(x)$ denotes $(\nabla J^*(x))^{\mathrm{T}}$ and $\underline{R} = [r_1, \cdots, r_m]^{\mathrm{T}}$. Substituting (10.23) and (10.24) into the HJB equation produces

$$0 = x^{\mathrm{T}} Q x + u_b^2 \underline{R} \ln(\underline{\mathbf{1}} - \tanh^2(D^*)) + \nabla J^{*\mathrm{T}}(x) f(x) \tag{10.26}$$

with $J^*(0) = 0$ that leads to $H(x, u^*(x), \nabla J^*(x)) = 0$. Since the system is CT, only a single critic network is required to approximate the cost function $J(x)$, and the

bounded approximate optimal control can directly be obtained through (10.23) and (10.24) by substituting the approximated cost function $\hat{J}(x)$.

10.3.4 Event-Triggered Control

Although time-based ADP approaches provide a mature and normative solution to nonlinear optimal control problems, to enhance the resource utilization and reduce the computational burden, ETC has been evolved as an alternate control paradigm and acquired more attentions in recent days [21]. Originating from networked control systems [25], ETC originally aims to deal with the limitation of communication bandwidth [21, 26]. A cross fertilization of ETC and ADP produces event-triggered ADP, which has successfully been implemented for optimal stabilization of both discrete-time aerospace systems [21] and CT aerospace systems [27]. The key attribute of the event-triggered mechanism lies in that the control signals are updated only when a certain condition is triggered [27]. Therefore, designing a sound triggering condition is the principal task of ETC.

Considering the event-triggered scheme, we define a sequence of triggering instants $\{s_k\}_{k=0}^{\infty}$, where s_k satisfies $s_k < s_{k+1}$ with $k \in \mathbb{N}$. The output of the sampled-data module is $x(s_k) \triangleq x_k$ for all $t \in [s_k, s_{k+1})$. Subsequently, we define the gap function using the event error:

$$e_k(t) = x_k - x, \forall t \in [s_k, s_{k+1}). \tag{10.27}$$

We denote $e_k(t)$ briefly by e_k hereafter. Every time when a certain triggering condition is satisfied, the event-triggered state vector is updated and the event error e_k is reset to zero. At every triggering instant (instead of time instant), the state feedback control law $u(x(s_k)) = u(x_k)$ is accordingly updated. By introducing a zero-order holder (ZOH), the control sequence $\{u(x_k)\}_{k=0}^{\infty}$ actually turns to be a piece-wise signal that remains constant during the time interval $[s_k, s_{k+1})$, $\forall k \in \mathbb{N}$. Based on the control signal $u(x_k)$, system (10.1) takes the form:

$$\dot{x} = f(x) + g(x)u(x + e_k), \forall t \in [s_k, s_{k+1}). \tag{10.28}$$

For system (10.1), with the infinite-horizon cost function represented by (10.21), we define a triggering condition as

$$\|e_k\|^2 > \|e_{\text{Thr}}\|^2, \tag{10.29}$$

where e_{Thr} denotes the threshold to be determined. The event is said to be triggered if (10.29) is satisfied.

The methods to design a threshold for CT systems and DT systems are different. For CT systems, the triggering threshold is designed to incorporate the ADP

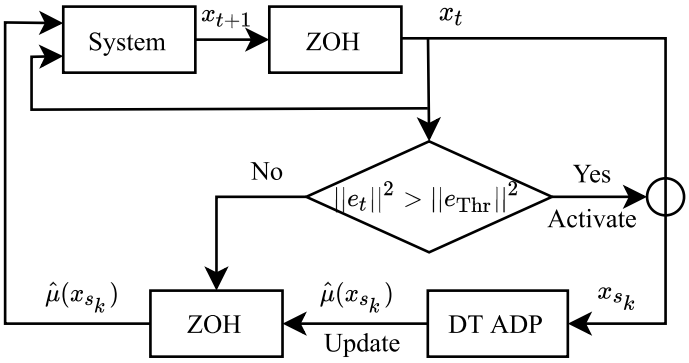


Fig. 10.13 Simple diagram of the ETC scheme incorporating the DT ADP algorithm

controller, which means the control policy is always computed although the control input is updated based on the triggering condition. For CT systems adopting ETC methods, the inter-execution time can be zero, resulting in the accumulation of event times. This is the infamous Zeno phenomenon that must be avoided in the controller design [27]. The main benefit to involve ETC is to decrease the communication cost between the controller and the controlled system.

On the contrary, for DT systems the design of the triggering threshold can be separated from the common control law. As presented in Fig. 10.13, only when an event is triggered will the ADP algorithm be activated and the control policy is computed. Therefore, not only the communication cost is decreased, but also the computational load is saved for DT systems.

Since the triggering condition is determined independently for DT ADP, it can have a common form. By assuming that there exists a positive constant $C \in (0, 0.5)$ such that

$$\|f(x_t, \mu(e_t + x_t))\| \leq C\|x_t\| + C\|e_t\|, \tag{10.30}$$

where $\|e_t\| \leq \|x_t\|$, the triggering condition for DT systems can be defined as [21]

$$\|e_t\| > e_{\text{Thr}} = C\frac{1-(2C)^{t-s_k}}{1-2C}\|x_{s_k}\|. \tag{10.31}$$

The effectiveness of the event-triggered ADP has been verified in an aeroelastic system as an example for CT systems [27]. With the wide usage of composite materials, high aspect-ratio aircraft wing can suffer from aeroelastic instability phenomena, including the LCOs [28, 29]. If not suppressed by active control, LCOs can lead to structural failure and even flight accidents [30]. The schematic of an aeroelastic wing section controlled by a single trailing-edge flap is illustrated in Fig. 10.14 [24], where c.m. is the abbreviation of center of mass. It has two degrees of freedom: the plunge displacement h and the pitch angle θ. In this problem, it is assumed in the undisturbed case that the freestream is along the airfoil chord, and thus pitch angle θ is equal to

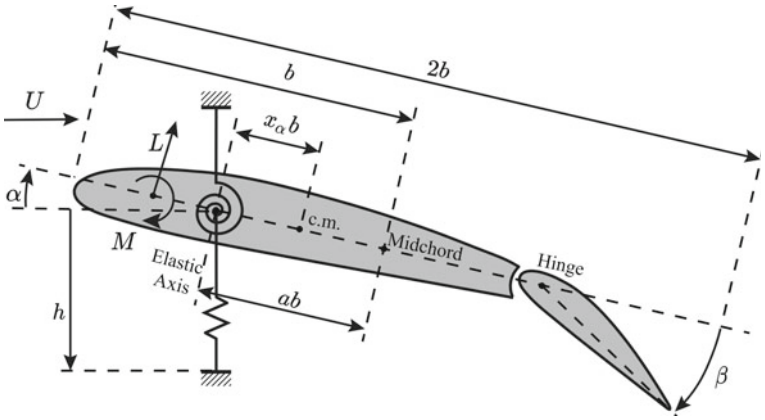


Fig. 10.14 A two-degree-of-freedom aeroelastic system with one control surface. For additional details on this mechanical analogy between fluttering of a wing and the connection of a wing with a translational and torsional spring, see [24]

AOA α. The task of the controller is to stabilize the system with input constraints in the ETC scheme and the detailed settings can be found in [27].

For presenting the advantage of the ETC scheme, a time-based approach is adopted for comparison, whose settings are exactly same as the proposed intelligent critic control approach except for the event-triggered scheme, i.e., the time-based control approach updates the control input at each time instant. The simulation is conducted in an online manner, which means that the control policy improves in a closed-loop way. It can be observed from Fig. 10.15 that the convergence of the weight vector occurs around 1s, which illustrates the feasibility of online learning. The inter-execution time is depicted in Fig. 10.16. It is worth mentioning that 800 samples are utilized by the time-based controller, whereas the proposed event-triggered approach only requires 366 samples. Therefore, the event-triggered method reduces the control updates in the learning process up to 54.25%, and thus improves the resource utilization.

Figures 10.17 and 10.18 present the aeroelastic system states trajectory divided into plunge and pitch motion, respectively. It can be observed that, although the event-triggered controller utilizes fewer data samples, the state variables eventually converge to a small vicinity of zero without deteriorating the converge rate.

Figure 10.19 presents the control command directly generated by the controller, u (β_c). The developed event-triggered approach has an overall comparable curve to the time-based approach. Due to the event-triggered mechanism, the control command signal is step-wise. Nevertheless, the control command signal has to go through an actuator and the real deflection is adequately smooth for the wing surface control. Furthermore, the control command (incorporating exploration noise) is bounded by the pre-designed saturation constraints, i.e., $|u| < u_b$. Therefore, it can be concluded that the control input constraints problem has been overcome.

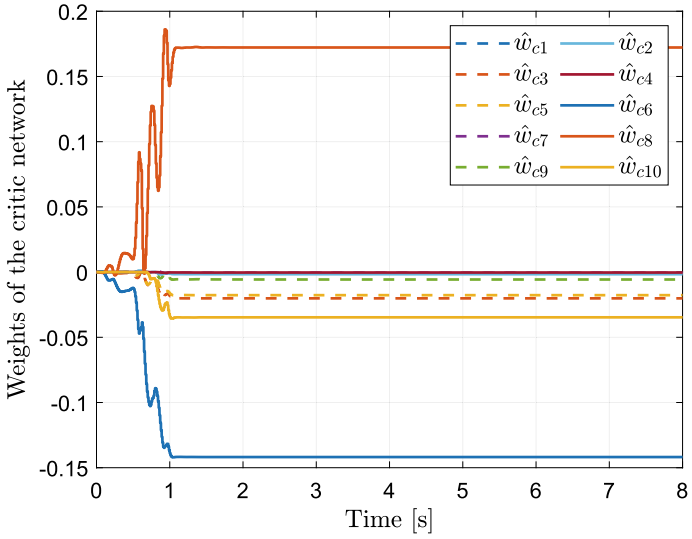


Fig. 10.15 Convergence process of the critic weights

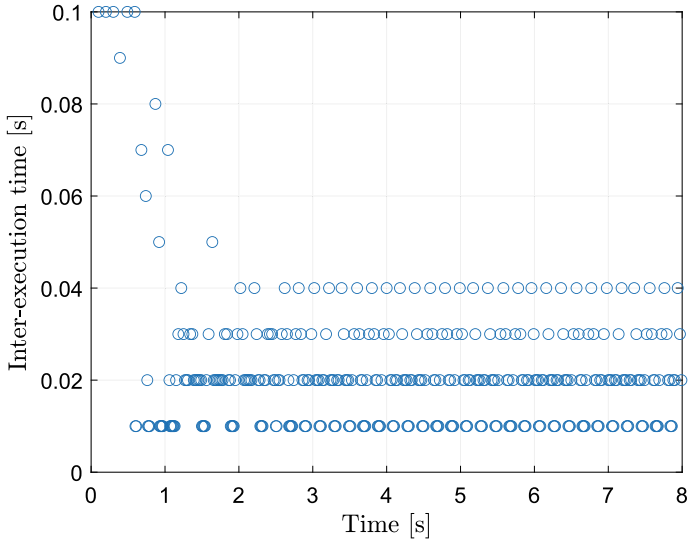
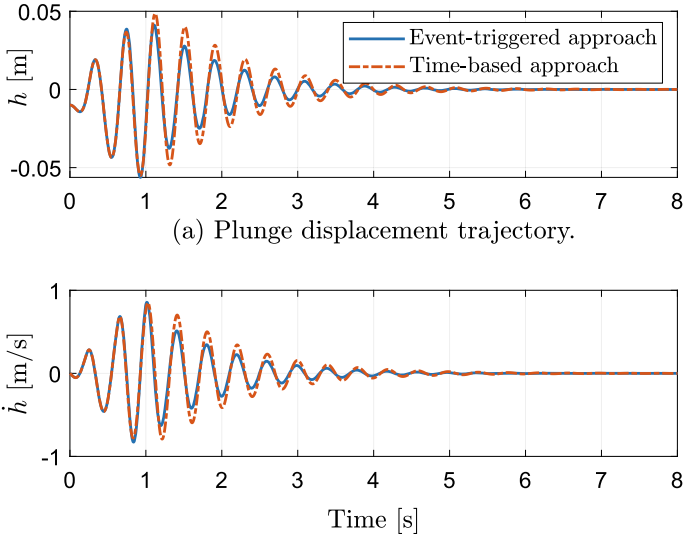


Fig. 10.16 Evolution of the inter-execution time

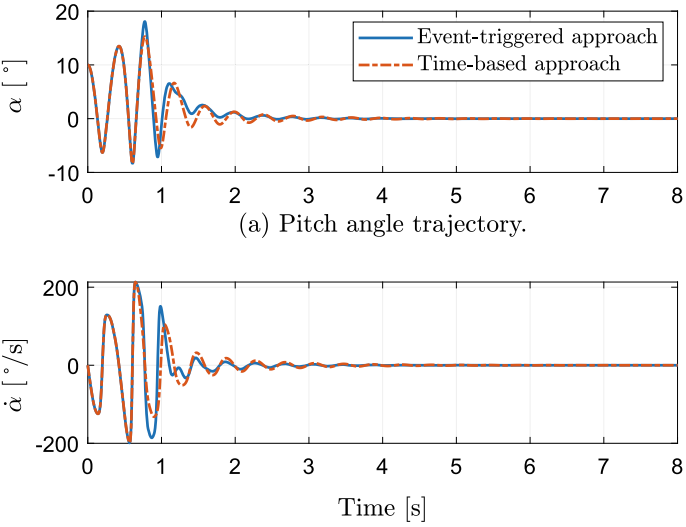
The phase portraits of plunge motions are illustrated in Fig. 10.20. It appears that the trajectories of the proposed method and the open-loop simulation almost coincide at the beginning. This phenomenon is due to the collective effect caused by LCOs and the initial unlearned policy and disappears quickly as the weight vector updates. Then, all states are stabilized to a small vicinity of the equilibrium point.



(a) Plunge displacement trajectory.

(b) Plunge velocity trajectory.

Fig. 10.17 Evolution of the plunge motion states



(a) Pitch angle trajectory.

(b) Pitch rate trajectory.

Fig. 10.18 Evolution of the pitch motion states

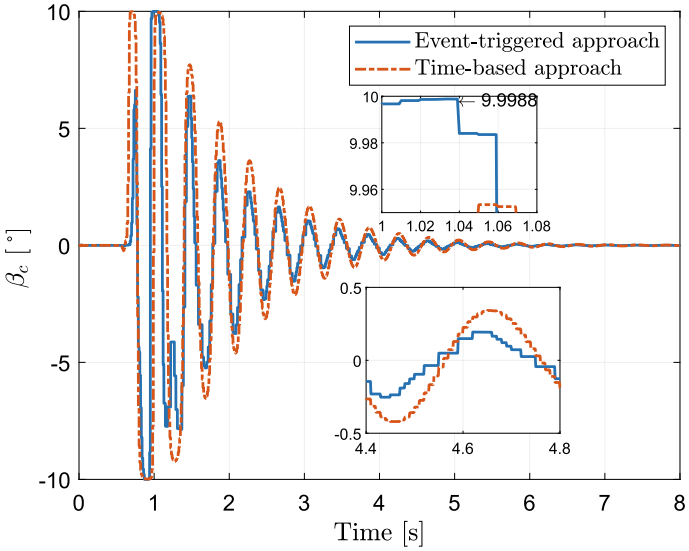


Fig. 10.19 Control command generated by the controller

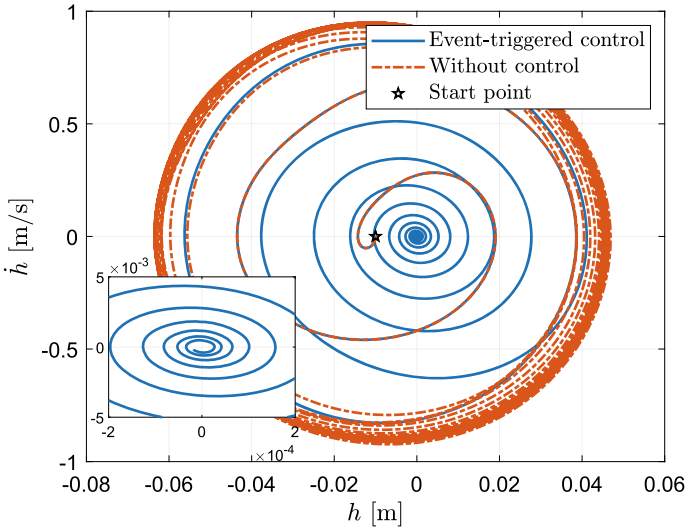


Fig. 10.20 Phase portrait of the plunge motion states

The simulation results collectively verify the feasibility and the effectiveness of the event-triggered constrained-input ADP control approach.

10.4 Conclusions and Future Work

This chapter provided an overview of how adaptive dynamic programming has been applied to flight control in the past 20 years. The introduction of incremental plant models in the ADP loop has allowed fast online learning and enabled the design of providing fault-tolerant flight control systems. Many developments have been made to increase the convergence success rate and performance of ADP controllers, for example, by using more advanced structures such as XGDHP (explainable global dual heuristic dynamic programming).

There are still several challenges that have to be addressed before ADP can find more widespread applications in flight control. For this reason, considerable effort is continuing in developing ADP controllers with guaranteed converge properties. These guarantees will improve safety at the cost of some flexibility in the learning strategy. An additional aspect that needs improvement concerns the learning efficiency, i.e., how many samples of data are required to learn a good policy. For low-level tasks, such as pitch rate or pitch angle control, the efficiency is high enough to allow for fast online learning. However, increasing the complexity of the control task, by adding more states and actions, can cause the learning to be too slow for pure online learning, meaning the aircraft can lose control before a recovering policy is learned. In these situations, a combination of offline and online learning can be used, in order to provide some baseline control performance while the online agent is trained.

Having addressed a control system for a single autonomous aircraft, such as ADP, which is at the intersection between optimal control, adaptive control, and RL, the next chapter will discuss how adaptive control can be successfully employed to control heterogeneous systems of autonomous aircraft.

References

1. Zhao J, Na J, Gao G (2020) Adaptive dynamic programming based robust control of nonlinear systems with unmatched uncertainties. Neurocomputing 395:56–65
2. Wang D, Ha M, Qiao J (2020) Self-learning optimal regulation for discrete-time nonlinear systems under event-driven formulation. IEEE Trans Autom Control 65(3):1272–1279
3. Wang D, He H, Liu D (2017) Adaptive critic nonlinear robust control: a survey. IEEE Trans Cybern 47(10):3429–3451
4. Kiumarsi B, Vamvoudakis KG, Modares H, Lewis FL (2018) Optimal and autonomous control using reinforcement learning: a survey. IEEE Trans Neural Netw Learn Syst 29(6):2042–2062
5. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction, 2nd edn. MIT Press, Cambridge, MA
6. Sun B, Mkhoyan T, van Kampen E-J, De Breuker R, Wang X (2022) Vision-based nonlinear incremental control for a morphing wing with mechanical imperfections. Trans Aerosp Electron Syst 58(6):5506–5518
7. Ruiz Garcia A, Vos R, de Visser C (2020) Aerodynamic model identification of the flying V from wind tunnel data. In: Aviation forum, Orlando, FL, pp 1–18
8. Enns R, Si J (2000) Neuro-dynamic programming applied to helicopter flight control. In: Guidance, navigation, and control conference and exhibit. AIAA, Orlando, FL, pp 1–11

9. Ferrari S, Stengel RF (2004) Online adaptive critic flight control. J Guidance Control Dyn 27(5):777–786
10. van Kampen E-J, Chu Q, Mulder J (2006) Online adaptive critic flight control using approximated plant dynamics. In: International conference on machine learning and cybernetics. IEEE, pp 256–261
11. Abouheaf M, Gueaieb W, Lewis F (2018) Model-free gradient-based adaptive learning controller for an unmanned flexible wing aircraft. Robotics 7(4):1–22
12. Sun B, van Kampen E-J (2020) Incremental model-based global dual heuristic programming with explicit analytical calculations applied to flight control. Eng Appl Artif Intell 89:103425
13. Zhou Y, van Kampen E-J, Chu Q (2018) Incremental model based online dual heuristic programming for nonlinear adaptive control. Control Eng Pract 73:13–25
14. Helder B, van Kampen E-J, Pavel M (2021) Online adaptive helicopter control using incremental dual heuristic programming. In: Scitech Forum. AIAA, Orlando, FL, pp 1–18
15. Zhou Y, van Kampen E-J, Chu Q (2020) Incremental model based online heuristic dynamic programming for nonlinear adaptive tracking control with partial observability. Aerosp Sci Technol 105:1–14
16. Lee JH, van Kampen E-J (2021) Online reinforcement learning for fixed-wing aircraft longitudinal control. In: Scitech Forum. AIAA, Orlando, FL, pp 1–20
17. Li H, Sun L, Tan W, Jia B, Liu X (2020) Switching flight control for incremental model-based dual heuristic dynamic programming. J Guidance Control Dyn 43(7):1352–1358
18. Heyer S, Kroezen D, van Kampen E-J (2020) Online adaptive incremental reinforcement learning flight control for a CS-25 class aircraft. In: Scitech Forum. AIAA, pp 1–20
19. Sun B, van Kampen E-J (2021) Intelligent adaptive optimal control using incremental model-based global dual heuristic programming subject to partial observability. Appl Soft Comput 103:107153
20. 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(8):1825–1832
21. Sun B, van Kampen E-J (2022) Event-triggered constrained control using explainable global dual heuristic programming for nonlinear discrete-time systems. Neurocomputing 468:452–463
22. Sun B, van Kampen E-J (2021) Reinforcement-learning-based adaptive optimal flight control with output feedback and input constraints. J Guidance Control Dyn 44(9):1685–1691
23. Heydari A, Balakrishnan SN (2013) Finite-horizon control-constrained nonlinear optimal control using single network adaptive critics. IEEE Trans Neural Netw Learn Syst 24(1):145–157
24. Lhachemi H, Chu Y, Saussié D, Zhu G (2017) Flutter suppression for underactuated aeroelastic wing section: nonlinear gain-scheduling approach. J Guidance Control Dyn 40(8):2102–2109
25. Li L, Song L, Li T, Fu J (2021) Event-triggered output regulation for networked flight control system based on an asynchronous switched system approach. IEEE Trans Syst Man Cybern Syst 51(12):7675–7684
26. Wang J, Wang P, Ma X (2020) Adaptive event-triggered control for quadrotor aircraft with output constraints. Aerosp Sci Technol 105:1–13
27. Sun B, Wang X, van Kampen E-J (2022) Event-triggered intelligent critic control with input constraints applied to a nonlinear aeroelastic system. Aerosp Sci Technol 120:1–11
28. Li D, Xiang J, Guo S (2011) Adaptive control of a nonlinear aeroelastic system. Aerosp Sci Technol 15(5):343–352
29. Sanches L, Guimarães TA, Marques FD (2019) Aeroelastic tailoring of nonlinear typical section using the method of multiple scales to predict post-flutter stable LCOs. Aerosp Sci Technol 90:157–168
30. Lee KW, Singh SN (2018) Robust finite-time continuous control of an unsteady aeroelastic system. J Guidance Control Dyn 41(4):978–986

Erik-Jan van Kampen received his BSc, MSc, and PhD degrees from Delft University of Technology in 2004, 2006, and 2010, respectively. He is currently an Assistant Professor in the Department of Control and Operations, Delft University of Technology, Delft, Netherlands. His research interests are intelligent flight control, reinforcement learning and incremental nonlinear control.

Bo Sun received a B.S. degree and the M.S. degree in aerospace engineering from Northwestern Polytechnical University, Xi'an, China, 2016 and 2019, respectively. He is currently pursuing a Ph.D. degree with the Department of Control and Operations, Delft University of Technology, Delft, Netherlands. He has a broad interest in adaptive dynamic programming, reinforcement learning, and aerospace engineering.