TU Delft

Delft University of Technology

Online intelligent flight control using reinforcement learning

Sun, B.

DOI
10.4233/uuid:1a6101dd-d5d2-4182-94ef-761329d4bdc0

Publication date
2024

Document Version
Final published version

Citation (APA)
Sun, B. (2024). *Online intelligent flight control using reinforcement learning*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:1a6101dd-d5d2-4182-94ef-761329d4bdc0

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Online Intelligent Flight Control Using Reinforcement Learning

Online Intelligent Flight Control Using Reinforcement Learning

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus, prof. dr. ir. T. H. J. J. van der Hagen,
chair of the Board for Doctorates
to be defended publicly on

Monday 3 June 2024 at 17:30 o'clock

by

Bo SUN

Master of Science in Navigation, Guidance and Control
Northwestern Polytechnical University, China,
born in Chifeng, Nei Mongol, China.

This dissertation has been approved by the promotors

promotor: Prof. dr. G.C.H.E. de Croon
copromotor: Dr. ir. E. van Kampen

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. G.C.H.E. de Croon,	Delft University of Technology, promotor
Dr. ir. E. van Kampen,	Delft University of Technology, copromotor
Independent members:	
Prof. dr. B. Clement,	ENSTA Bretagne, France
Prof. dr. H.S. Shin,	Cranfield University, the United Kingdom
Prof. dr. M.T.J. Spaan,	Delft University of Technology
Dr. G. Spigler,	Tilburg University
Dr. ing. J. Kober,	Delft University of Technology
Prof. dr. M. Kotsonis,	Delft University of Technology, reserve member

Keywords: Reinforcement learning; Aerial vehicles; Adaptive dynamic programming; Adaptive optimal control; Global dual heuristic programming; Event-triggered control; Incremental model; Nonlinear control.

Printed by: Ipskamp Printing

Cover: Our Friday Goldfish Studio

Style: TU Delft House Style, with modifications by Moritz Beller.

The author set this dissertation in LaTeX using the Libertinus and Inconsolata fonts. The cover is designed with the Hylia Serif font.

Copyright © 2024 by Bo Sun. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior permission in writing from the proprietor.

ISBN 978-94-6384-588-5

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

To my beloved parents and the alluring world

Contents

Summary

Advancements in aerial vehicles have presented new challenges for flight control system design. The disturbed airflow caused by rotors and flapping wings and the nonlinearity and uncertainty increased by morphing components impede the identification of a globally accurate model of these vehicles. Additionally, complex components can increase the risk of structural damage and faults and pursuing large wingspans and a high aspect ratio can result in unstable flight dynamics because of structural flexibility. Moreover, current control systems are limited in their ability to adapt to unanticipated or changing circumstances.

Given these challenges, the aerospace community has turned to intelligent methods to enhance the autonomy of flight controllers. Reinforcement learning (RL) has been popular in recent years because it offers self-learning approaches that improve agent policies through interaction with the environment. Fruitful cross-fertilization of RL and control theory produced adaptive dynamic programming (ADP), which has been successfully applied to a wide range of nonlinear control systems, including aerial vehicles. ADP inherits dynamic programming's optimality while enabling adaptability. Furthermore, with the facilitation of artificial neural networks (ANNs), ADP can handle complex control demands and demonstrate the capability of online learning. Consequently, the main research goal of this dissertation is:

To improve the adaptability and online learning capability of flight control systems by designing nonlinear ADP approaches.

RL and ADP are relatively new in the field of flight control. Despite the promising benefits, applying ADP to aerial vehicles in flight control presents challenges. Online learning efficiency is a concern due to uncertainties, disturbances, and sudden faults during flight. Imperfect measurements and partial observability further complicate the application of ADP. The lack of a unified paradigm for different categories of ADP also hinders learning efficiency. Therefore, these challenges lead to the first research sub-question:

1. How to synthesize and improve current online ACDs to cope with dynamic and measurement uncertainties, partial observability, external disturbances and sudden faults?

This research question is answered through the development of global dual heuristic programming (GDHP) with explicit analytical calculations and an incremental model, which synthesizes current online ACDs without requiring prior knowledge of system dynamics. It uses a critic network to approximate the cost function and analytically computes derivatives based on the approximation, while incorporating an incremental model to estimate state transitions using instant and historical data sets. Through applications to attitude control problems in a nonlinear aircraft, the proposed method demonstrates the ability to handle imperfect measurements, unknown and time-varying dynamics, diverse initial conditions, and sudden faults while maintaining efficient control.

The second challenge in Adaptive Dynamic Programming (ADP) is ensuring online learning stability for aerial vehicles. As system complexity increases, accurate modeling becomes difficult, and control input constraints must be considered. However, theoretical analysis of control stability is limited due to the complexity of Artificial Neural Networks (ANNs). RL flight controllers also have a low success ratio in online learning control due to stochastic processes. These challenges lead to the second research sub-question:

2. How to achieve optimality and stability of ADP for discrete-time (DT) and continuous-time (CT) systems, while dealing with input constraints of aerial vehicles?

This research question is answered through the construction of an enhanced non-quadratic integral utility function and a bounding layer in the actor network. The new utility function allows for analytical calculations of constrained control inputs. A bounding layer with activation functions is designed for DT systems, addressing asymmetric control input constraints. Three solutions are proposed to enhance stability. Firstly, the incremental model and improved GDHP technique enable quick online learning with a success ratio exceeding 99%. Secondly, the iterative adaptive critic algorithm improves performance through multiple iterations. Besides, simplified analytical calculations and the concept of explainable GDHP (XGDHP) are introduced. Finally, for continuous-time nonlinear systems, an improved update criterion eliminates the need for initial admissible control while ensuring Lyapunov closed-loop stability considering network approximation errors.

The third challenge in online ADP flight control is the high computational load. This dissertation introduces the event-triggered control (ETC) scheme to optimize computational utilization, and applies ETC to aerial vehicles with intelligent optimal control. ETC reduces update frequency and affects policy exploration, ultimately impacting control performance. Therefore, the third research sub-question is formulated:

3. How to save computational and communication load by event-triggered ADP without significantly affecting overall performance?

This research question is answered through the respective investigation of DT and CT systems. This research combines ETC with XGDHP for DT systems, with ETC acting as the outer loop of XGDHP. The triggering condition for stability is improved and a more specific proof is provided. For CT systems, ETC is incorporated in the ADP algorithm with a novel triggering condition considering input constraints. The Zeno phenomenon is analyzed and avoided based on the derived triggering condition. The closed-loop stability of the ETC scheme is theoretically proven and verified through simulation. Results show that ADP benefits from ETC in terms of computational and communication load while achieving comparable control performance.

In conclusion, this dissertation develops ADP flight control methods to enhance the autonomy of aerial vehicles despite uncertainties and constraints, and successfully answers the following main research question:

How can aerial vehicles benefit from ADP to improve autonomy and online learning in an uncertain environment, while satisfying requirements on input constraints, adaptability, stability and computational efficiency?

1

Introduction

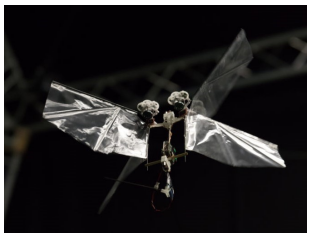
1.1 Autonomous Control in Aerial Vehicles

The last hundred years have witnessed an unprecedented surge of automated control in various domains, ranging from household appliances to industrial manufacturing and aerospace systems. Since the great industrial revolution, automation has now been widely accepted and has freed people from tedious and monotonous physical activities and thus decreased costs. In the information age, great importance has been attached to higher automation for better economic benefits and a more comfortable life. These incentives encourage increased automation in all social sectors.

The development of aerial vehicles is no exception to this trend. Since their first prototypical versions, which consisted only of mechanical components and depended heavily on manual operations, a variety of aerial vehicles has been created for a multitude of usages, as shown in Fig. 1.1. Taking Delft University of Technology (TU Delft) as an example, there are various unmanned aerial vehicles (UAVs) and manned aircraft developed or under development in recent years. In terms of UAVs, the Delfy Nimble bio-inspired ornithopter (Fig. 1.1a) is one of the most famous, which imitates the tailless structure of the fruit fly and can achieve agile flying [1]. As to manned aircraft, sustainable aviation is becoming a concept recognized by more people, and therefore energy-efficient aircraft are receiving more attention, such as the Flying-V prototype aircraft (Fig. 1.1b). In addition to these complete aircraft, research on aircraft components is also underway. For instance, the SmartX project develops a biomimetic seamless morphing wing (Fig. 1.1d) that can actively optimize its shape, minimize the air resistance and thus save energy [2, 3]. Researchers in other parts of the world also investigate novel aerial vehicles that can work in various environments. For instance, for urban air mobility, Airbus develops the CityAirbus demonstrator (Fig. 1.1c), which is a multi-passenger, autonomous piloted electric vertical takeoff and landing vehicle [4]. For multimodal environments, scientists at Harvard University developed an insect-scale robot with a pair of flapping wings, which is capable of hybrid aerial and aquatic locomotion [5] (Fig. 1.1e).

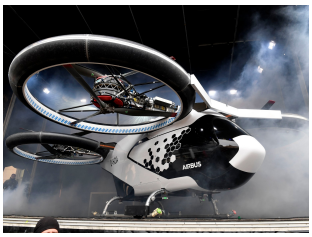
Nonetheless, with the rise and success of these projects, they also become more complex in terms of their structure and working environment and bring about new challenges for the control system. For example, conventional flight control methods require piecewise

1



(a) DelFly Nimble ornithopter [1]

(b) Flying-V aircraft ©TU Delft



(c) CityAirbus demonstrator [4]

(d) SmartX-Alpha seamless morphing wing ©T. Mkhoyan

(e) Hybrid RoboBee robot [6]

Figure 1.1: Examples of recently developed aerial vehicles and components.

mathematical models of the physical system and generate control laws around each operating point by appropriate methods such as stability analysis and manual tuning [7]. However, rotors and flapping wings disturb the airflow that interacts with the UAV body, which impedes the analysis of the force and moment mechanisms. Simultaneously, morphing components in the morphing wing also increase the nonlinearity and uncertainty, which complicates the system identification. Therefore, constructing a globally accurate model is intractable and often not practical for these structurally complex systems. Furthermore, these complex components may also increase the risk of structural damage and sudden faults [8], as presented in Fig. 1.2a. Moreover, although the high-precision aerodynamics and kinematics of fixed-wing aircraft can be identified offline, the pursuit of a large wingspan and a high aspect ratio for aerodynamic efficiency often leads to structural flexibility. When encountered with moderate air turbulence, the interaction of the structural and rigid-body modes can result in unstable flight dynamic modes [9] like the phugoid mode, which in extreme situations can cause oscillations and even lead to a crash, such as the loss of NASA's Helios prototype aircraft [10] shown in Fig. 1.2b. Under these unanticipated or changing circumstances, current control systems are seriously challenged because they often lack the capability to adapt to situations for which they have not been designed. Consequently, with the trend of designing novel and advanced aerial vehicles, more stringent, improved and intelligent approaches to autonomous control are required to deliver guaranteed performance and the satisfaction of prescribed targets, instead of merely automated control.

The difference between automated control and autonomous control is not always clear

(a) A quadcopter with a single rotor detached, ©S. Sun

(b) Helios at high wing dihedral prior to structural failure and in-flight breakup, ©NASA

Figure 1.2: Examples of structural failure of aerial vehicles during flight.

due to the nearness of their definitions. They are overlapping but different concepts and have already been distinguished by lots of researchers [11–14]. Specifically, automated control can utilize limited logic or feedback to react accordingly to several predetermined circumstances with limited-to-no human interaction. Autonomous control mentioned in this dissertation encompasses forms of automated control but additionally allows for decision making in circumstances that are not explicitly accounted for by designers or can find solutions that are not predetermined or preprogrammed by designers [14]. In other words, autonomous control extends the concept of automated control with aspects of *adaptive* or *online learning* capability, which is also named as *intelligence*. Therefore, autonomous control has more potential to overcome the challenges brought about by the increase in the complexity of aerial vehicles.

Recent years have witnessed significant developments and innovations in artificial intelligence (AI) and machine learning (ML), which provide promising solutions to the increase of autonomy, and therefore the involvement of AI/ML in aerial vehicles has become an active research field. For instance, the European Union Aviation Safety Agency (EASA) has published a human-centric AI technical roadmap, intending to depict the development prospect of AI in aviation [15]. As shown in Fig. 1.3, AI and ML play a vital role in achieving more autonomous flight in this vision. Nonetheless, many ML approaches rely on offline training with prior-collected data, which is limited when facing complex unforeseen circumstances. Therefore, what if an intelligent approach exists to enable the controller to adapt and learn online by interacting with the environment?

1.2 Reinforcement Learning

One solution to address the aforementioned issues is reinforcement learning (RL), a class of ML algorithms that draws its inspiration from the way humans or animals learn [16]. In the context of this dissertation, the capability of adapting or online learning is defined as that an agent can improve its policy and thus performance via experience gained from interaction with its environment. RL by nature learns from the environment and therefore this intelligent approach can empower the controller to be autonomous [17].

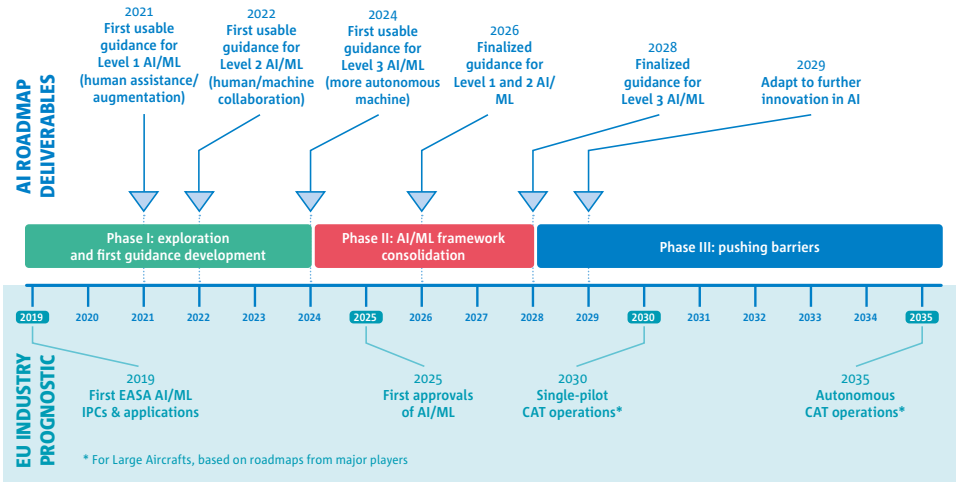


Figure 1.3: Human-centric AI roadmap published by EASA in 2020 [15].

1.2.1 Working principle of RL

As the complexity of aerial vehicles increases, the need for the individual components to work with each other is inevitable. Flight control systems thus tend to seek optimality in terms of a specific performance indicator. Dynamic programming (DP) lays out the fundamentals of optimal control and is an important theoretical support for early RL theory [16, 18, 19]. However, DP is not natural for implementation in real time because it is fundamentally an offline approach that requires an accurate model. Moreover, DP calculates the policy in a time-backwards way and does not offer methods for solving optimal decision problems online in a forward manner. Furthermore, traditional DP approaches are devised for discrete state and action spaces by using a lookup table [16]. With real-world applications, in particular aerial vehicle control problems, traditional DP approaches have been confronted with high-dimensional or continuous spaces, which can result in the infamous "curse of dimensionality", a phenomenon where the number of states and actions grows exponentially [7, 16, 19–22].

Fortunately, this issue can be handled by some RL methods. RL is closely linked with optimal control, in that learning behaviour that gets maximum reward is equal to pursuing a control policy that optimizes certain performance-based cost functions. The working principle of RL is visualized in Fig. 1.4, where the agent, represented by a quadcopter, takes an action that interacts with the environment, and then accordingly perceives the new state and collects the feedback (reward or penalty) corresponding to the state transition [16]. This process can be iterative and therefore provides the agent with a guideline to learn rewarding behaviours online.

One family of online RL methods is built with the actor–critic structure, in which the actor applies an action or a control policy to the environment, whereas the critic assesses the value of that action [23]. With these two components, the temporal-difference (TD) approach can be employed to perform time-forward calculations. Therefore, the actor–critic structure is particularly well adapted for solving optimal decision problems

in real time through RL techniques [21, 23]. Adaptive dynamic programming (ADP) is a common alias of practical actor–critic methods for solving optimal control problems in real time in the control engineering [22, 24]. For handling the curse of dimensionality, ADP often utilizes function approximations to develop practical algorithms for complex systems with disturbances and uncertain dynamics [23, 24]. As its name suggests, ADP not only inherits the capability of DP for pursuing optimality, but also empowers the controller to be adaptive. Developed by hybridization between RL and control theory, ADP has become a key direction for future research in building autonomous systems.

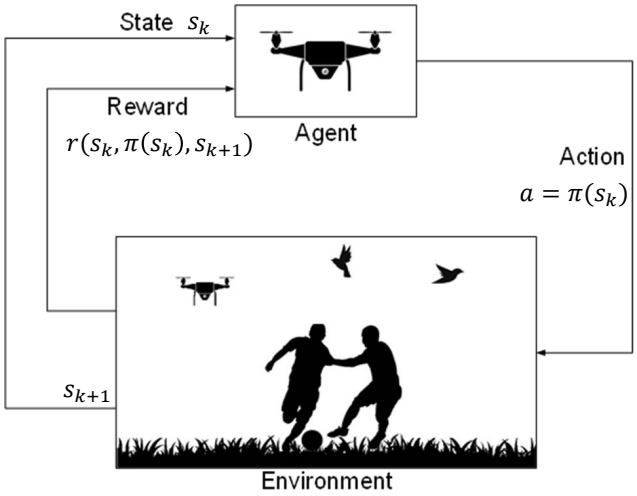


Figure 1.4: Working principle of RL visualized by an example. In this representation, the quadcopter agent aims to keep tracking shots of athletes steadily in a changing environment defined by everything included in the graph, such as the ground, athletes, the football, birds and even the status of the quadcopter itself. The agent observes its state in this environment and may receive rewards for successfully recording video meanwhile keeping a safe flight, and subsequently decides what actions to take for transitioning its state in the environment.

1.2.2 Adaptive dynamic programming

When reviewing the literature, there is often confusion about ADP-related terminologies due to its several synonyms [25], such as adaptive critic design [26], approximate dynamic programming [27], neuro-dynamic programming [28, 29], relaxed dynamic programming [30], actor-critic control [31] and also reinforcement learning [21, 32, 33]. Despite slight differences in detail and emphasis, these terminologies are essentially the same type of approach in the context of optimal control. This dissertation focuses on adaptability and online learning of control systems. Therefore, to highlight adaptability and to distinguish it from the larger concept of RL, this dissertation refers to such approaches uniformly as adaptive dynamic programming (ADP) and gives its definition as follows:

Definition 1.1. Adaptive dynamic programming (ADP) is a class of adaptive optimal control methods which use function approximations to approximate values or policies and solve the Bellman equation in a time-forward way via a TD approach.

1

It is noteworthy that the implementation of ADP shows differences when applied to discrete-time (DT) and continuous-time (CT) in terms of dealing with the time-bootstrapping property, in that for CT systems the impact of bootstrapping is often neglected. To distinguish these situations, this dissertation exploits the adaptive critic design (ACD) to describe the situation when the system or controller is DT, and this terminology is widely used in previous work [7, 26, 34]. The taxonomy of these terminologies is visualized in Fig. 1.5.

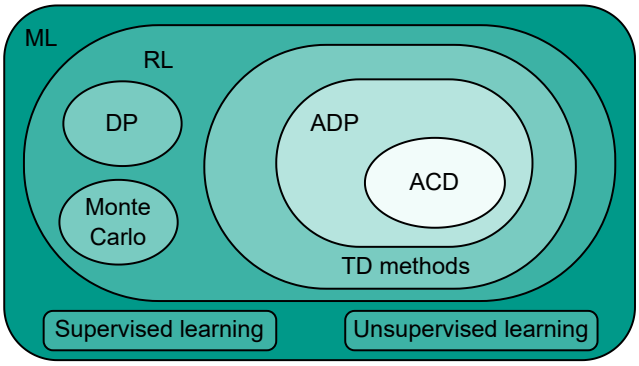


Figure 1.5: Taxonomy of the different methodologies of ML, RL, ADP and ACD.

ADP falls within the scope of TD methods, as illustrated in Fig. 1.5, owing to their shared foundational principles and iterative approaches in updating the value function. Both ADP and TD methods employ bootstrapping techniques, enabling online learning from incomplete experiences. Throughout this dissertation, all developed ADP methodologies adhere rigorously to Definition 1.1, which encapsulates the essential characteristics of ADP as outlined in [21]: function approximation and TD-based approaches. Consequently, ADP specifically encompasses a subset of TD methods that utilises approximators, such as artificial neural networks (ANNs), to approximate solutions of the Bellman equation, marking a key distinction within the broader TD framework.

All RL approaches, including ADP, are developed based on the Markov assumption. Therefore, the mechanism depicted in Fig. 1.4 can accordingly be modeled as a Markov decision process (MDP). Choose the DT system as an example. The Markov property guarantees that the state transition towards a new state s_{k+1} and the acquired reward r depend uniquely on the current state s_k and the action a_k generated by current control policy $\pi(s_k)$ [35]. Hence, the target of the controller is to maximize the total discounted sum of future rewards (or to minimize the cost) and therefore a value function can be implicitly expressed as follows:

$$V^{\pi}(s_k) = \lim_{N\to\infty} E_{\pi}\left[\sum_{l=k}^{N-1} \gamma^{l-k} r(s_l, \pi(s_l), s_{l+1})\right], \; k \geq 0 \tag{1.1}$$

where $E_{\pi}[\cdot]$ denotes the expected value of a random variable given that the agent follows policy π, and $\gamma \in (0, 1]$ is a discount factor determining to what extent future rewards can affect current behaviour [16]. According to Bellman's principle of optimality [16, 18, 19],

the optimal value $V^*(s_k)$ is compliant with the Bellman equation:

$$V^*(s_k) = \max_{a_k} (r(s_k, a_k, s_{k+1}) + \gamma V^*(s_{k+1})), \ \forall k. \tag{1.2}$$

Solving Eq. (1.2) is the key procedure when designing an optimal controller, and it is impossible for generic nonlinear systems to obtain the analytical solution. ADP solves it iteratively based on the TD error by subdividing the Bellman equivalence into multiple updates as:

$$V(s_k) \longleftarrow V(s_k) + \eta(r(s_k, a_k, s_{k+1}) + \gamma V(s_{k+1}) - V(s_k)), \tag{1.3}$$

where η denotes a learning rate whose role is to facilitate convergence [35]. At each update step, the control policy $\pi(s_k)$ is also accordingly updated. These iterative procedures, updating value function and control policy, are called policy evaluation and policy improvement, respectively [16], which rely on each other to gradually converge to the optimal policy.

From Eq. (1.3), it can be observed that the update process relies on the result from the future, i.e., it is bootstrapping regarding time. To cope with this issue, the actor-critic scheme is commonly employed with the facilitation of ANNs [36]. For DT systems, two ANNs, namely actor network and critic network, are constructed to respectively generate and evaluate the control policy. Policy improvement and policy evaluation procedures are conducted when updating these ANNs. This kind of control approach is classified as ACDs in this dissertation, which is a subclass of ADP.

For CT systems, when the time interval tends to 0, the update of states will also tend to 0, which approximately eliminates the impact of bootstrap and the control architecture can be simplified. Specifically, only a single critic network is required for CT control systems to evaluate the control policy, and building an actor network is often not necessary, because the control policy can accordingly be computed through the critic network by solving the Bellman equation [37].

1.3 Challenges of ADP for Flight Control

Unlike Monte Carlo methods that rely on sampling the whole episode and DP methods that require a time-backwards calculation, ADP makes use of TD error and function approximation to enable online learning. That is to say that ADP can accomplish the objective of online intelligent optimal control. Nevertheless, ADP for flight control is often confronted with three main challenges to be dealt with, as will be illustrated in this section.

1.3.1 The challenge of online learning efficiency

When arriving at a new situation or meeting with emergencies, humans can often adapt quickly with very few iterations. ML approaches, however, usually require many samples. In emergencies, it is vital to control the aircraft before it crashes, so there are not many samples to learn from, which is the main reason why we need to improve the sample efficiency.

When people talk about RL, it is often accompanied by the model-free concept because some RL methods can directly learn from the environment without building a system model to achieve direct adaptive optimal control [38], which can also be accomplished by ADP. Nevertheless, being model-free does not mean that the system dynamics are totally

uninvolved. Actually, the system and environment dynamics are implicitly incorporated in the learned control policy in these approaches [39]. A typical example of model-free RL is Q-learning [40, 41], which employs a $Q(s_k, a_k)$ function to calculate the quality of a state–action combination instead of the value function $V(s_k)$ formulated in Eq. (1.1). Since the policy evaluation in these approaches is directly based on not only states but also actions, in the ADP field, the class of methods is also called action-dependent (AD) [26, 39, 42].

However, such direct learning approaches have drawbacks that decrease online learning efficiency from both theoretical and practical perspectives. Since the control policy in AD-ADP methods implicitly incorporates the system dynamics, the complexity of the critic network [7] and also the sampling complexity of the algorithm are increased, particularly when using high-dimensional function approximators, which tends to limit their applicability to physical systems [40]. Furthermore, previous studies on ACD and AD-ACD have reported that ACD outperforms its AD form in success ratio and adaptability [26, 39]. Consequently, another module is often constructed in the ADP scheme to approximate the unknown system dynamics and provide state transition information, normally based on ANNs [43–45]. Due to the demand for efficient online adaptability, the approximated dynamics should be as accurate as possible. However, uncertainties, disturbances and even sudden faults can happen during flight [8, 46], which requires fast adaptation of the ADP controller, especially the identifier. Furthermore, conventional ADP assumes the availability of full state feedback [39, 43–45]. However, either output feedback (OPFB) where there exist unmeasurable inner states in deterministic systems [47, 48] or partial observability (PO) that involves stochastic or time-varying dynamics [49, 50] can be encountered by aerial vehicles. All of these factors bring about challenges in the system identification for efficient online intelligent flight control.

Despite the importance of the learned network parameters, the network structure plays a fundamental role. Compared to the actor network that maps states to actions, the critic network is often more complex in the low-level control task because it bears the responsibility for policy evaluation which instructs policy improvement. Based on different targets to be approximated by the critic network, current ADP (especially ACD) methods can be categorized into different types, such as heuristic dynamic programming (HDP) [42], dual heuristic programming (DHP) [34, 36] and global DHP (GDHP) [43, 45]. Each method has benefits and drawbacks, but all these methods do not share the same network structure so they cannot directly be transferred to each other without being redesigned. Therefore, a unified structure is needed to cope with complex situations, which will be covered in this dissertation.

1.3.2 The challenge of online learning stability

The stability of the flight is crucial. For the flight controller, stability can be evaluated separately from both theoretical and experimental perspectives. In this dissertation, a controller is said to be stable if a control method is able to achieve a 100% success ratio through simulation verification or if the closed-loop stability can pass a rigorous theoretical analysis. It is noted that the success ratio depends on how realistic the simulation environment is, and this dissertation will take the random and unanticipated factors that affect the stability of online learning into consideration.

When interacting with the real world, the automatic flight control system is significant for preventing aerial vehicles from getting into dangerous situations. A traditional and common solution is to build a simulation environment with accurate system dynamics such that a primitive control policy can be trained offline to guarantee safety, after which, online fine-tuning is implemented for better performance [51–53]. However, the reality gap between simulation and the real world broadens as the system becomes complex. As analyzed in Section 1.1, it is difficult to construct a globally accurate model for aerial vehicles, let alone the unforeseen disturbances and faults. Consequently, enhancing online learning stability is crucial in these situations.

In many cases, enhancing flight stability is consistent with learning efficiency in that aerial vehicles can swiftly adjust to new conditions. Nevertheless, the prerequisite is that the controller knows what the proper behaviour is. Therefore, designing a sound reward is important, which however is still an open research topic in the RL field. Furthermore, aerial vehicles always suffer from control input constraints [2], which should be considered in the controller design. Because of the complexity of ANNs, literature in ADP-based flight control rarely theoretically analyzes the control stability, which mostly is demonstrated through tests [32, 34, 39, 42]. However, it is difficult to make comparisons among these research works because of different systems and different ways to define success. Therefore, this dissertation intends to enhance flight control stability by increasing the success ratio compared to existing methods, or preferably by providing theoretical proof.

1.3.3 The challenge of computational load

One of the widely known shackles of AI is its high computational load. For instance, it took the famous AlphaGo Zero 40 days of computations on multiple high-performance processors to achieve its impressive performance [41]. The computational restrictions are harsher for aerial vehicles because they are systems with limited resources and continuous state and action spaces. For a quadcopter, months of training can be spent before a stable flight RL-based controller is obtained [52]. This has seriously hampered the application and dissemination of intelligent flight control methods.

Computational load is also closely related to learning efficiency. In addition to the factors mentioned in Subsection 1.3.1, another cause of high computational load is the low sampling efficiency that requires a huge amount of interactions [54]. Similar to other RL approaches, ADP can suffer from low sampling efficiency, which may result in unbearable computational costs or even infeasibility in online flight control applications. However, different from those cases where the reward is sparse, such as the Go game in which, only after the game ends, a reward can be collected [41], flight control, particularly low-level control, often rich feedback is used to speed up learning. Rich feedback requires high bandwidth sensors. Conventionally, the calculation regarding ANNs parameters update is conducted once the feedback is obtained so as to evaluate and update the control policy. However, if the performance is sufficiently good at present, continued update of the ANNs is a waste of computational resources. So why not only calculate and update when necessary?

There exists such a technique called event-triggered control (ETC). Arising from networked control systems, ETC was originally designed to enhance communication resource utilization [55, 56]. Recently, it has been introduced to spacecraft control problems [57, 58], but they are not solved with optimal control and the spacecraft is different from aerial ve-

hicles in aerodynamics and disturbances. The application of event-triggered ADP methods to aircraft has not yet been explored. Furthermore, ETC decreases the update frequency and at the same time has an impact on policy exploration, which will ultimately affect the control performance. Therefore, this dissertation will investigate the application of event-triggered ADP to flight control.

1.4 Research Questions, Methodology and Scope

This section illustrates the objectives, the contributions and the limitations of the dissertation. Sub-section 1.4.1 proposes the research goal and derives the research questions. Then the research approaches and contributions in this dissertation are enumerated in Sub-section 1.4.2. Finally, Sub-section 1.4.3 defines the research scope of the dissertation and clarifies the limitations.

1.4.1 Research goal and questions

This dissertation focuses on the control approach development for aerial vehicles with a big-picture aim of an autonomous system. The challenges stated in Section 1.1 motivate the main research goal of this dissertation:

Research goal

To improve the adaptability and online learning capability of flight control systems by designing nonlinear ADP approaches.

While ADP offers promising benefits, effectively leveraging its potential in the context of aerial vehicles requires addressing the challenges outlined in Section 1.3. Hence, the main research question of this dissertation focuses on dealing with the challenges of ADP and is formulated as follows:

Main research question

How can aerial vehicles benefit from ADP to improve autonomy and online learning in an uncertain environment, while satisfying requirements on input constraints, adaptability, stability and computational efficiency?

Specifying further, the main research question can be decomposed into 3 sub-questions concerning the previously discussed challenges of ADP for flight control from different aspects:

- **learning efficiency**,
- **control stability,**
- **computational load.**

Accordingly, the research sub-questions can be formulated as:

Research sub-questions

RQ1: How to synthesize and improve current online ACDs to cope with dynamic and measurement uncertainties, partial observability, external disturbances and sudden faults?

RQ2: How to achieve optimality and stability of ADP for DT and CT systems, while dealing with input constraints of aerial vehicles?

RQ3: How to save computational and communication load by event-triggered ADP without significantly affecting overall performance?

Each of the sub-questions is answered in several chapters with a considerable overlap in this dissertation, since by design each method or task usually corresponds to more than one challenge.

1.4.2 Research approaches and contributions

This dissertation presents an online intelligent optimal controller design using the ADP approach for DT and CT aerial systems respectively. Improvements are made in system identification, algorithm training, computational reduction, constraint satisfaction and stability enhancement. The contributions can be elaborated from two aspects. On the one hand, it contributes to adaptive optimal control approaches:

- Proposal of a novel explainable global dual heuristic programming (XGDHP) approach for discrete-time (DT) aerospace systems. This method synthesizes and improves current ACD techniques. (RQ 1)
- Development of the incremental model and theoretical proof regarding its convergence. The incremental model is further combined with XGDHP to improve the adaptability and online learning of the controller, even under some unanticipated or changing circumstances. (RQ 1)
- Design of an incremental model-based XGHDP method to deal with tracking control problems of an aircraft with respect to full state measurements and partial observability. (RQ 1)
- Improvement of ADP with facilitation of ANNs to deal with nonlinear optimal control problems and actuator saturation constraints for both DT and CT systems. (RQ 2)
- Introduction of the event-triggered control scheme to both DT and CT systems, which can effectively decrease the computational burden and simultaneously save communication load between the host computer and the actuator. (RQ 3)

The developed approaches can be directly applied to other nonlinear uncertain systems complying with similar formulations and assumptions, i.e., the preceding contributions are applicable to generic systems, not necessarily aerial vehicles. On the other hand, the dissertation also makes contributions to flight control applications, which are listed as follows:

- Verification by simulations that the RL-based intelligent control framework can help aircraft adapt to a wide range of dynamic and measurement uncertainties, external disturbances, sudden faults and structural damages. (RQs 1-3)

- Design of an event-triggered constrained-input ADP controller for an aeroelastic system to stabilize limit cycle oscillations. Via theoretical proof, the Zeno phenomenon is avoided and the Lyapunov stability of the proposed method is guaranteed. (RQ 3)

1.4.3 Research scope and limitations

In order to achieve the research goal stated in Sub-section 1.4.1, the scope of this dissertation is restricted as follows:

Validation methodology: In control, RL and aerospace engineering literature, it is common to employ simulations to prove the validity of an algorithm. This is because, for theoretical development, simulations allow engineers to better understand the agent's behaviour under different circumstances. For physical systems, safety and cost also need to be handled carefully. Consequently, as well as to reduce the complexity of research, the approaches developed in this dissertation are investigated, tested and validated exclusively via simulations.

Targeted applications: This dissertation focuses on low-level control. The proposed approaches in this dissertation are originally designed for aerial vehicles, but the applications are not limited to this area. All aerial vehicle models utilized in this dissertation are publicly accessible and do not involve any confidentiality.

Online learning control: RL is by nature learning online from interactions with the environment. It can learn from rewards, penalties or even failures. In some literature, the learning is episode-based. However, this dissertation places more stringent demands on online learning, that at a single flight, the controller needs to update the control policy online recursively and learn a feasible policy before failure. This is because, in this way, the controller is able to adapt to unanticipated or changing circumstances. Nevertheless, this does not mean that pre-training cannot be applied, but this is out of the scope of this dissertation.

ANN: Every algorithm developed in this dissertation is facilitated by ANNs, which are also important cornerstones of modern AI technology development. Different types of ANNs have their specializations. For the low-level control problems in this dissertation, no great importance needs to be attached to feature extraction and abstraction. Therefore, simple feedforward neural networks with shallow layers are adopted, which are fast, easy to analyse, and sufficient to achieve satisfying performance.

1.5 Outline of Dissertation

The outline of this dissertation is visualized in Fig. 1.6. The research is divided into terms of DT and CT systems. Chapter 2 proposes a novel ACD algorithm for synthesizing and improving current ACDs. Then, Chapters 3 and 4 enhance the algorithm with improved incremental models to deal with imperfect measurements. Chapter 5 introduces the ETC

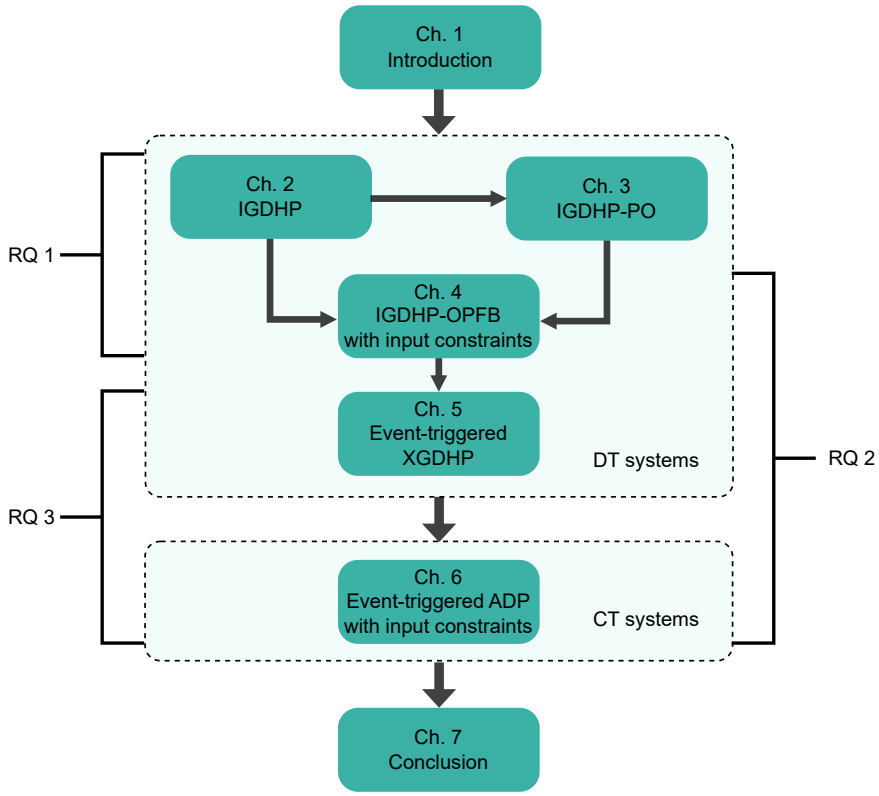


Figure 1.6: Architecture of the chapters in the dissertation.

scheme into XGDHP based on the work in previous chapters. Chapter 6 expands on the ETC scheme and the approach to dealing with input constraints in CT systems.

Chapter 2 proposes incremental model-based GDHP (IGDHP) to generate a self-learning adaptive flight controller, in the absence of sufficient prior knowledge of system dynamics. An incremental technique is employed for online local dynamics identification, instead of the ANNs commonly used in GDHP, to enable fast and precise learning. This chapter for the first time computes the derivatives of the performance function required by the GDHP technique with explicit analytical calculations based on differential operations. This chapter is based on full-state feedback (FSF) functions as a starting point for the rest of the DT system part.

In Chapters 3 and 4, the IGDHP is further enhanced by incorporating the previous data sets into the incremental model to deal with unmeasurable inner states. Chapter 3 expands the idea of IGDHP to the partially observable control system, whose tracking reference is assumed unknown, and the augmented incremental model is accordingly developed, whose feasibility and convergence are theoretically analyzed. The performance of IGDHP-PO is evaluated in multidimensional aspects such as different initial conditions, the presence of

noises as well as sudden disturbances and faults. Chapter 4 investigates the OPFB situation and incorporates the control input constraints into performance design. The technique to cope with input constraints is inherited in the following chapters.

Chapter 5 combines the ETC scheme and ADP to deal with the nonlinear control problems. ETC is able to decrease the amount of computation, and the stability analysis is provided in this chapter, with fewer assumptions compared to most existing ACD studies that investigate ETC for DT systems. Furthermore, this chapter simplifies the method to calculate the required derivatives of performances which is proposed in Chapter 2 and employed in Chapters 3 and 4, and originally proposes the concept of XGDHP. Besides, the performance function is also improved to handle asymmetric input constraints.

Chapter 6 enhances the application section of ADP in that the approach is originally designed particularly for an aeroelastic system with input constraints. This chapter designs a CT ADP approach that uses a single critic network, with a novel triggering condition that considers input constraints, avoiding the Lipschitz assumption on the inverse hyperbolic tangent function. This chapter also employs an improved weight updating criterion, which can eliminate the requirement of initial admissible control. Furthermore, the closed-loop Lyapunov stability is guaranteed and the Zeno phenomenon is avoided through theoretical proof.

Finally, Chapter 7 summarizes the results and findings of the previous chapters and demonstrates how this dissertation addresses the research goal of improving the adaptability and online learning of the flight control system through proposed ADP approaches. In addition, this chapter proposes several points of further development for the autonomy of aerial vehicles.

All of the chapters are created from published journal papers, which are listed as follows:

- Chapter 2 is based on the following article:

  B. Sun and E. van Kampen, "Incremental Model-Based Global Dual Heuristic Programming with Explicit Analytical Calculations Applied to Flight Control", *Engineering Applications of Artificial Intelligence*, vol. 89, pp. 103425, 2020. Doi: 10.1016/j.engappai.2019.103425.

- Chapter 3 is based on the following article:

  B. Sun and E. van Kampen, "Intelligent Adaptive Optimal Control Using Incremental Model-Based Global Dual Heuristic Programming Subject to Partial Observability", *Applied Soft Computing*, vol. 103, pp. 107153, 2021. Doi: 10.1016/j.asoc.2021.107153.

- Chapter 4 is based on the following article:

  B. Sun and E. van Kampen, "Reinforcement-Learning-Based Adaptive Optimal Flight Control with Output Feedback and Input Constraints", *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 9, pp. 1685-1691, 2021. Doi: 10.2514/1.G005715.

- Chapter 5 is based on the following article:

  B. Sun and E. van Kampen, "Event-Triggered Constrained Control Using Explainable Global Dual Heuristic Programming for Nonlinear Discrete-Time Systems", *Neurocomputing*, vol. 468, pp. 452-463, 2022. Doi: 10.1016/j.neucom.2021.10.046.

- Chapter 6 is based on the following article:

  B. Sun, X. Wang and E. van Kampen, "Event-Triggered Intelligent Critic Control with Input Constraints Applied to A Nonlinear Aeroelastic System", *Aerospace Science and Technology*, vol. 120, pp. 107279, 2022. Doi: 10.1016/j.ast.2021.107279.

2

IGDHP with Explicit Analytical Calculations Applied to Flight Control

Incremental Model-Based Global Dual Heuristic Programming with Explicit Analytical Calculations Applied to Flight Control

The first research question of how to synthesize and improve current online adaptive critic designs (ACDs) will be addressed in this chapter. As discussed in Chapter 1, there is a demand for efficient online dynamics identification for flight control systems, and there is a lack of a method of unifying existing ACDs that allows easy conversion between different methods. Therefore, this chapter will first introduce an incremental model as the online local dynamics identifier, instead of the common artificial neural networks (ANNs) to enable a fast and precise identification. Then, a novel structure will be proposed in this chapter to synthesize ACDs, called global dual heuristic programming (GDHP) with explicit analytical calculations. Finally, this chapter will verify the method with numerical simulations on a tracking control task of a nonlinear aerial vehicle, by taking measurement noises and sudden faults into consideration.

This chapter is based on the following article:
B. Sun and E. van Kampen, "Incremental Model-Based Global Dual Heuristic Programming with Explicit Analytical Calculations Applied to Flight Control", *Engineering Applications of Artificial Intelligence*, vol. 89, pp. 103425, 2020. Doi: 10.1016/j.engappai.2019.103425 [59].

Abstract

A novel adaptive dynamic programming method, called incremental model-based global dual heuristic programming, is proposed to generate a self-learning adaptive flight controller, in the absence of sufficient prior knowledge of system dynamics. An incremental technique is employed for online local dynamics identification, instead of the artificial neural networks commonly used in global dual heuristic programming, to enable a fast and precise learning. 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 the control policy, respectively. The required derivatives of cost-to-go are computed by explicit analytical calculations based on differential operations. Both methods are applied to an online attitude tracking control problem of a nonlinear aerospace system and the results show that the proposed method outperforms conventional global dual heuristic programming in tracking precision, online learning speed, robustness to different initial states and adaptability for fault-tolerant control problems.

2.1 Introduction

Controller design for aerospace systems, especially airplanes, is challenging for many reasons. One of the most challenging parts is the difficulty of modeling the dynamics of the system. Especially for complex, nonlinear vehicles, global plant information may be impossible to obtain. The aircraft can be susceptible to uncertainties, sudden faults and structural damages, which changes the real plant compared to the previously obtained model [60]. To deal with these problems, one promising solution is to create learning or adaptive controllers.

Reinforcement learning (RL), for instance, which links several bio-inspired artificial intelligence techniques, can make a system learn desired policies without accurate models of its dynamics or environment and can adapt to changing situations [16]. For these reasons, there have been a number of RL methods developed to enable model-free flight control in various types of aerospace systems [34, 52, 61–63]. However, different from ground robots, it is difficult to employ end-to-end training approaches on aerospace systems because of their special working environments. Consequently, a combination of RL and traditional control theory is a promising strategy to improve adaptive flight control. The combination of an actor-critic structure, dynamic programming, and neural networks, results in the adaptive/ approximate dynamic programming (ADP) algorithm, which is regarded as an effective technique to design adaptive optimal controllers and can achieve certain levels of adaptiveness and fault-tolerance [25, 33, 64, 65]. According to optimal control theory, for a nominal system, given a cost function, the analytical optimal solution can be obtained by solving the Hamilton-Jacobi-Bellman (HJB) equation. However, for complex or nonlinear systems, analytical solutions to the HJB equation can be difficult or even impossible to get, let alone in real time. To conquer the difficulty of directly solving the HJB equation online for general nonlinear systems, the adaptive critic framework and artificial neural networks (ANNs) are often involved in order to approximate the HJB solution [25, 66, 67].

As a class of ADP methods, adaptive critic designs (ACDs), which separate policy evaluation (critic) and policy improvement (actor), have shown great success in optimal adaptive control of nonlinear aerospace systems [34, 36, 39, 62, 63]. ACDs can generally

be categorized into several groups [26]: heuristic dynamic programming (HDP), dual heuristic programming (DHP) and global dual heuristic programming (GDHP) and their action-dependent (AD) versions. HDP is the most basic form and most often used structure, which employs 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 [68]. GDHP, which approximates both the cost-to-go and its derivatives so as to take advantage of two kinds of information, has several different forms [26]. Among them, the straightforward form, where the critic approximates the cost-to-go and its derivatives simultaneously [43, 63, 69], is most commonly used because of its simple structure. In this architecture, two kinds of outputs share the same inputs and hidden layers, making them strongly coupled. Although these outputs have explicit mathematical relationships, without analytical calculations, weight update processes can suffer from this coupling due to inconsistent errors, and sometimes it even leads to instability. In this chapter, explicit analytical calculations of the mixed second-order derivatives of the critic network outputs with respect to its input vector and weight matrices are introduced for adaptive flight control. [26] and [70] illustrate how to calculate these derivatives in an element-wise way. However, [71] clarifies that vectors and matrices more often appear as a whole in practical applications rather than multivariable functions, and this holistic view is easier to obtain the chain rule. Therefore, one contribution of this chapter is deriving a direct method based on differential operation [71] to compute these second-order derivatives in a holistic way so as to tackle the inconsistent errors between the approximated cost-to-go function and its derivatives.

ACDs can be model-free if the critic is an AD network, which means the control signals are also introduced as network inputs [72, 73]. Nevertheless, to achieve model-free application, an alternative is building a third module to approximate the plant dynamics and ANNs are often regarded as the first choice [39, 43–45, 74]. [39] illustrates how this three-network structure outperforms AD ones 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 like 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 [39, 43, 44], which, however, cannot achieve adaptive control when facing unforeseen uncertainties and sudden disturbances in realistic application.

The main contribution of this chapter is an incremental model-based GDHP (IGDHP) method that enables online model-free flight control based on our latest work [63]. Different from conventional GDHP, an incremental model is involved for adaptive control to deal with the absence of full system information. Assuming sufficiently high sampling rate for discretization, incremental techniques are able to accurately identify 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) [75], incremental back-stepping (IBS) [76], incremental sliding mode control (ISMC) [46, 76] and IADP [47, 50], IACDs [34, 42, 63]. However, these existing incremental methods have some limitations. For instance, INDI, IBS and ISMC cannot deal with optimal control

problems, and IADP requires linear quadratic reward and offline training process. Compared to existed methods, IGDHP develops current IACDs to achieve online adaptive optimal control. In summary, the primary advantages lie in that the novel algorithm speeds up online policy learning without knowing system dynamics or offline training a model network, and the analytical calculations make use of the information of cost-to-go function and its derivatives without introducing inconsistent errors.

The remainder of this chapter is structured as follows. Section 2.2 presents the basic formulation of three-network GDHP with explicit analytical calculations. Section 2.3 introduces the incremental method for online identification and uses it to simplify the weight update process of the actor and critic networks. Then Section 2.4 provides the necessary information for verification, where an F-16 Fighting Falcon model is built and possible noises, faults and damages during flight are explained. Section 2.5 verifies the approaches by applying both GDHP and IGDHP on a longitudinal attitude tracking task in various conditions and analyzing their results. Finally Section 2.6 summarizes the chapter and puts up possibilities for future research.

2.2 GDHP Implementation

GDHP, which combines the advantages of HDP and DHP, can be implemented as a model-free technique with three ANNs, namely model, critic and actor. The variables or pathways corresponding to these ANNs are denoted by the subscripts m, c and a, respectively. The architecture of GDHP with explicit analytical calculations is illustrated in Fig. 2.1. Based on current states, the actor network generates an action to control both real system and plant model. The model network estimates the states at the next time step, which are connected to the critic network to approximate cost-to-go, whose derivatives are computed analytically. All weights of the ANNs are updated in a back-propagation way according to the gradient-descent algorithm [26].

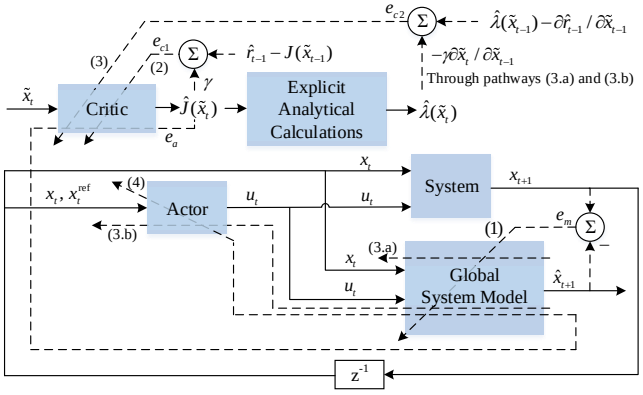


Figure 2.1: The architecture of GDHP with explicit analytical calculations, where solid lines represent the feedforward flow of signals, and dashed lines represent the adaptation pathways.

2.2.1 Global model

For a full-state feedback system, the inputs of the system model are the current state vector $x_t \in \mathbb{R}^n$ and current control vector $u_t \in \mathbb{R}^m$, while the output is the estimated next state vector $\hat{x}_{t+1} \in \mathbb{R}^n$. However, because the future true value of state vector x_{t+1} is unavailable at time step t, the update is implemented with current and previous data, i.e. the network weights are updated by minimizing the difference between the current measured state vector x_t and the estimated state vector $\hat{x}_t$:

$$E_m(t) = \frac{1}{2} e_m^{\mathsf{T}}(t) Q_m e_m(t), \tag{2.1}$$

where $e_m(t) = \hat{x}_t - x_t$, 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. $Q_m = \mathrm{diag}(\zeta_1, \zeta_2, \cdots, \zeta_n)$, where the elements respectively select and weigh the approximating errors.

The weights are updated by a gradient-descent algorithm:

$$w_m(t+1) = w_m(t) - \eta_m \cdot \frac{\partial E_m(t)}{\partial w_m(t)} \tag{2.2}$$

where η_m is the learning rate, and

$$\frac{\partial E_m(t)}{\partial w_m(t)} = \frac{\partial E_m(t)}{\partial \hat{x}_t} \cdot \frac{\partial \hat{x}_t}{\partial w_m(t)} = e_m^{\mathsf{T}}(t) \cdot \frac{\partial \hat{x}_t}{\partial w_m(t)}. \tag{2.3}$$

2.2.2 The critic network

GDHP combines HDP and DHP and requires the information of both the cost-to-go $J(\tilde{x}_t)$ and its derivatives with respect to the network inputs $\tilde{x}_t$, where $\tilde{x}_t = \hat{x}_t - x_t^{\mathrm{ref}}$ stands for tracking error vector. The critic network only employs $\hat{J}(\tilde{x}_t)$ to approximate the true cost-to-go $J(x_t, x_t^{\mathrm{ref}})$, which is the cumulative sum of future rewards r_t from any initial state $\tilde{x}_t$:

$$J(x_t, x_t^{\mathrm{ref}}) = \sum_{l=t}^{\infty} \gamma^{l-t} r_l, \tag{2.4}$$

where $\gamma \in (0, 1)$ is the discount factor, used to control the extent to which the short-term cost or long-term cost is concerned. The derivative of the cost-to-go with respect to the input vector $\hat{\lambda}(\tilde{x}_t)$ is shown in Appendix 2.A .

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

$$r_t = r(x_t, x_t^{\mathrm{ref}}) = (x_t - x_t^{\mathrm{ref}})^{\mathsf{T}} Q_c (x_t - x_t^{\mathrm{ref}}) = \tilde{x}_t^{\mathsf{T}} Q_c \tilde{x}_t, \tag{2.5}$$

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

Because future rewards are required, a temporal difference (TD) method is introduced to iteratively update the critic network [16]. The principle is to minimize the temporal difference error, the error between the current and successive estimates of the state value. Similar to the model network, the weights of the critic network are updated with current and previous data. The critic errors are as follows:

$$e_{c1}(t) = \hat{J}(\tilde{x}_{t-1}) - \hat{r}_{t-1} - \gamma \hat{J}(\tilde{x}_t), \tag{2.6}$$

and

$$e_{c2}(t) = \frac{\partial[\hat{J}(\tilde{x}_{t-1}) - \hat{r}_{t-1} - \gamma\hat{J}(\tilde{x}_t)]}{\partial\tilde{x}_{t-1}} = \hat{\lambda}(\tilde{x}_{t-1}) - \frac{\partial\hat{r}_{t-1}}{\partial\tilde{x}_{t-1}} - \gamma\hat{\lambda}(\tilde{x}_t)\frac{\partial\tilde{x}_t}{\partial\tilde{x}_{t-1}}, \tag{2.7}$$

where $e_{c1}(t)$ is the TD error of the estimated cost-to-go $\hat{J}(\tilde{x}_{t-1})$ with current network weights, while $e_{c2}(t)$ is the TD error of the computed derivatives $\hat{\lambda}(\tilde{x}_{t-1})$ with current network weights. $\hat{r}$ denotes the estimated reward, because true states are unavailable to evaluate updated control policy. GDHP combines both of them in an overall error function $E_c(t)$:

$$E_c(t) = \beta\frac{1}{2}e_{c1}^2(t) + (1-\beta)\frac{1}{2}e_{c2}^{\mathsf{T}}(t)e_{c2}(t), \tag{2.8}$$

where β is a scalar indicating the importance within a range of $[0,1]$. If $\beta = 1$, then it becomes pure HDP. If $\beta = 0$, then the tuning of weights merely depends on the TD error of computed derivatives $\hat{\lambda}(\tilde{x}_t)$, and consequently it is equivalent to DHP, which is different from the straight form in [43, 63, 69], where if $\beta = 0$, the back-propagation channel of the actor is cut.

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

$$w_c(t+1) = w_c(t) - \eta_c \cdot \frac{\partial E_c(t)}{\partial w_c(t)}, \tag{2.9}$$

where

$$\frac{\partial E_c(t)}{\partial w_c(t)} = \frac{\partial E_c(t)}{\partial\hat{J}(\tilde{x}_{t-1})}\cdot\frac{\partial\hat{J}(\tilde{x}_{t-1})}{\partial w_c(t)} + \frac{\partial E_c(t)}{\partial\hat{\lambda}(\tilde{x}_{t-1})}\cdot\frac{\partial\hat{\lambda}(\tilde{x}_{t-1})}{\partial w_c(t)} = \beta e_{c1}(t)\frac{\partial\hat{J}(\tilde{x}_{t-1})}{\partial w_c(t)} + (1-\beta)e_{c2}^{\mathsf{T}}(t)\frac{\partial\hat{\lambda}(\tilde{x}_{t-1})}{\partial w_c(t)}, \tag{2.10}$$

where $\partial\hat{\lambda}(\tilde{x}_{t-1})/\partial w_c(t)$ represents the second-order mixed gradient of the estimated cost-to-go $\hat{J}(\tilde{x}_{t-1})$. An example of how to obtain it is illustrated in Appendix 2.A.

2.2.3 The actor network

The actor network outputs control action u_t, which is an input of the model network, and thus it will affect the critic outputs at the next time-step. The goal of the actor network is to produce an optimal control policy by minimizing the error between the current approximated cost-to-go $\hat{J}(\tilde{x}_t)$ and the ideal one $J^*(t)$, which depends on the given reward function and is set to be zero in this chapter:

$$u_t^* = \arg\min_{u_t} E_a(t), \tag{2.11}$$

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

$$E_a(t) = \frac{1}{2}e_a^2(t), \tag{2.12}$$

where

$$e_a(t) = \hat{J}(\tilde{x}_t) - J^*(t). \tag{2.13}$$

Different from the straight form in [43, 63, 69] where the actor network can be trained through the pathways either leading from $\hat{J}(\tilde{x}_t)$ or carried out by $\hat{\lambda}(\tilde{x}_t)$, there is only one

back-propagation way for GDHP with explicit analytical calculations to update the actor weights and the information from both $\hat{J}(\tilde{x}_t)$ and $\hat{\lambda}(\tilde{x}_t)$ can be utilized. As illustrated in Fig. 2.1, the actor weights are updated along the 4th back-propagation direction with a learning rate η_a:

$$w_a(t+1) = w_a(t) - \eta_a \cdot \frac{\partial E_a(t)}{\partial w_a(t)}, \tag{2.14}$$

where

$$\frac{\partial E_a(t)}{\partial w_a(t)} = \frac{\partial E_a(t)}{\partial \hat{J}(\tilde{x}_t)} \cdot \frac{\partial \hat{J}(\tilde{x}_t)}{\partial \hat{x}_t} \cdot \frac{\partial \hat{x}_t}{\partial u_{t-1}} \cdot \frac{\partial u_{t-1}}{\partial w_a(t)} = \hat{J}(\tilde{x}_t) \cdot \frac{\partial \hat{J}(\tilde{x}_t)}{\partial \hat{x}_t} \cdot \frac{\partial \hat{x}_t}{\partial u_{t-1}} \cdot \frac{\partial u_{t-1}}{\partial w_a(t)}. \tag{2.15}$$

2.3 IGDHP Implementation

Aerospace systems have complex nonlinear dynamics for which ANNs can fail to achieve accurate online identification fast enough. In this section, an incremental technique is introduced to ensure a quick and accurate approximation using locally linearized models (Fig. 2.2). In addition to online learning and quick adaptation, it also reduces the computational burden of the network weight update processes.

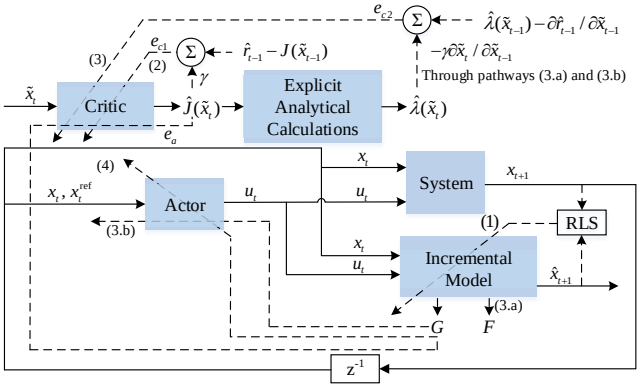


Figure 2.2: The architecture of IGDHP with explicit analytical calculations, where solid lines represent the feedforward flow of signals, and dashed lines represent the adaptation pathways.

2.3.1 Incremental model

Although most physical systems are continuous, 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. The derivation [34] can be generally given as follows:

Consider a nonlinear continuous system described by:

$$\dot{x}(t) = f(x(t), u(t)), \tag{2.16}$$

where $f[x(t), u(t)] \in \mathbb{R}^n$ provides the 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 Eq. (2.16) around time t_0 and omitting higher-order terms, the system is linearized approximately as follows:

$$\dot{x}(t) \approx \dot{x}(t_0) + F(x(t_0), u(t_0))(x(t) - x(t_0)) + G(x(t_0), u(t_0))(u(t) - u(t_0)), \tag{2.17}$$

where

$$F(x(t_0), u(t_0)) = \frac{\partial f(x(t), u(t))}{\partial x(t)}|_{x(t_0), u(t_0)}, \tag{2.18}$$

$$G(x(t_0), u(t_0)) = \frac{\partial f(x(t), u(t))}{\partial u(t)}|_{x(t_0), u(t_0)}, \tag{2.19}$$

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

$$\Delta\dot{x}(t) \simeq F(x(t_0), u(t_0))\Delta x(t) + G(x(t_0), u(t_0))\Delta u(t). \tag{2.20}$$

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

$$\frac{x_{t+1} - x_t}{\Delta t} \approx F_{t-1} \cdot (x_t - x_{t-1}) + G_{t-1} \cdot (u_t - u_{t-1}), \tag{2.21}$$

where $F_{t-1} = \frac{\partial f(x,u)}{\partial x}|_{x_{t-1}, u_{t-1}} \in \mathbb{R}^{n\times n}$ is the system transition matrix and $G_{t-1} = \frac{\partial f(x,u)}{\partial u}|_{x_{t-1}, u_{t-1}} \in \mathbb{R}^{n\times m}$ is the input distribution matrix at time step $t-1$ for the discretized systems. From Eq. (2.21), the following incremental form of the new discrete nonlinear system can be obtained:

$$\Delta x_{t+1} \approx F_{t-1}\Delta t\Delta x_t + G_{t-1}\Delta t\Delta u_t. \tag{2.22}$$

In this way, the continuous nonlinear global plant is simplified into a linear incremental dynamic equation. The obtained local plant model can be identified online with 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 [66], so as to avoid training a complex ANN. 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:

$$\hat{x}_{t+1} = x_t + \hat{F}_{t-1}\Delta t\Delta x_t + \hat{G}_{t-1}\Delta t\Delta u_t. \tag{2.23}$$

2.3.2 Online identification using RLS

A RLS approach is applied to identify the system transition matrix F_{t-1} and the input distribution matrix G_{t-1} online with the assumption of full-state feedback. The incremental form of the states in Eq. (2.22) can be rewritten in a row-by-row form as follows:

$$\Delta x_{t+1} \approx \begin{bmatrix}\Delta x_t^\top & \Delta u_t^\top\end{bmatrix} \cdot \begin{bmatrix}F_{t-1}^\top \\ G_{t-1}^\top\end{bmatrix} \cdot \Delta t. \tag{2.24}$$

Since all increments of the states share the same covariance matrix, the parameters can be identified together as $\Theta_{t-1} = \begin{bmatrix} F_{t-1}^{\mathsf{T}} \\ G_{t-1}^{\mathsf{T}} \end{bmatrix} \in \mathbb{R}^{(n+m)\times n}$ [34]. Therefore, the state prediction equation Eq. (2.23) can be rewritten as follows:

$$\Delta\hat{x}_{t+1} = X_t^{\mathsf{T}}\hat{\Theta}_{t-1}\Delta t, \tag{2.25}$$

where $X_t = \begin{bmatrix} \Delta x_t \\ \Delta u_t \end{bmatrix} \in \mathbb{R}^{(n+m)\times 1}$ 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:

$$\epsilon_t = \Delta x_{t+1}^{\mathsf{T}} - \Delta\hat{x}_{t+1}^{\mathsf{T}}, \tag{2.26}$$

$$\hat{\Theta}_t = \hat{\Theta}_{t-1} + \frac{Cov_{t-1}X_t}{\gamma_{RLS} + X_t^{\mathsf{T}}Cov_{t-1}X_t} \cdot \frac{\epsilon_t}{\Delta t}, \tag{2.27}$$

$$Cov_t = \frac{1}{\gamma_{RLS}}\left(Cov_{t-1} - \frac{Cov_{t-1}X_tX_t^{\mathsf{T}}Cov_{t-1}}{\gamma_{RLS} + X_t^{\mathsf{T}}Cov_{t-1}X_t}\right), \tag{2.28}$$

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 global exploration, it is better to satisfy the persistent excitation (PE) condition [34] for identifying the incremental model. A 3211 disturbance signal is introduced to excite the system modes at the start of training.

2.3.3 Network update simplification

Considering Eq. (2.7), the last term $-\gamma\hat{\lambda}(\tilde{x}_t)\frac{\partial\tilde{x}_t}{\partial\tilde{x}_{t-1}}$ needs to be dealt with carefully, because there are two pathways for $\tilde{x}_{t-1}$ to affect $\tilde{x}_t$. One is through the model network directly (pathway 3.a), and another one firstly goes through the actor network and then through the model network (pathway 3.b), as shown in both Figs. 2.1 and 2.2:

$$\frac{\partial\tilde{x}_t}{\partial\tilde{x}_{t-1}} = \frac{\partial x_t}{\partial x_{t-1}} = \underbrace{\frac{\partial x_t}{\partial x_{t-1}}|_m}_{pathway\ (3.a)} + \underbrace{\frac{\partial x_t}{\partial u_{t-1}}|_m \cdot \frac{\partial u_{t-1}}{\partial x_{t-1}}|_a}_{pathway\ (3.b)}. \tag{2.29}$$

In conventional GDHP, the two system model derivative terms in Eq. (2.29) are calculated back through the global system model, while IGDHP introduces the identified incremental model information directly to approximate them, whose computation burden is decreased compared to GDHP:

$$\frac{\partial\tilde{x}_t}{\partial\tilde{x}_{t-1}} \approx \hat{F}_{t-1}\cdot\Delta t + \hat{G}_{t-1}\cdot\Delta t\cdot\frac{\partial u_{t-1}}{\partial x_{t-1}}|_a. \tag{2.30}$$

Similarly, the actor weight update process can also be simplified by the incremental information. Specifically, the term $\frac{\partial\hat{x}_{t+1}}{\partial u_t}$ in Eq. (2.15) can be approximated by the identified

input distribution matrix $\hat{G}_{t-1}$ directly:

$$\frac{\partial \hat{x}_{t+1}}{\partial u_t} = \hat{G}_{t-1} \cdot \Delta t. \tag{2.31}$$

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

References [46, 75, 76] demonstrate that under the assumption that the sampling rate is sufficiently high, in other words, Δt is small enough, the errors due to linearization and discretization can be ignored. In this chapter, Δt is set to be 1 ms, which is realistic and accurate enough. The stability analysis of the GDHP method is investigated in [43, 69]. However, to the best of our knowledge, the theoretical assurance for the closed-loop convergence of online model-free control algorithms is still an open problem. The parameter convergence of control policy requires accurate and stable model information, which in turn depends on the parameter convergence of the control policy, making a circular argument.

2.4 Numerical Experiments Setup

The first part in this section introduces a nonlinear longitudinal model of F-16 Fighting Falcon for evaluation of the proposed algorithms. The second part briefly introduces some potential uncertainties in practical flight. The third part discusses some related issues of the network structures for implementation of the aforementioned algorithms, including the activation function, the hierarchical actor network, etc.

2.4.1 Aerospace system model

IGDHP can be applied to nonlinear aerospace systems, whose dynamic and kinematic state equations can be generally represented as:

$$\dot{x}(t) = f(x(t), u(t), d(t)), \tag{2.32}$$

where $d(t)$ represents the disturbances and noises.

To verify the proposed method and compare the effect of these differences for a practical case, a nonlinear longitudinal model of F-16 Fighting Falcon [77, 78] is introduced. The model consists of the longitudinal force and moment equations, and it is a specific example of Eq. (2.32):

$$\begin{cases} \dot{V} = g\sin(\alpha-\theta) + \dfrac{\cos\alpha}{m}T + \dfrac{\bar{q}S\cos\alpha}{m}C_{x,v} + \dfrac{\bar{q}S\sin\alpha}{m}C_{x,v}, \\ \dot{\alpha} = \dfrac{g}{V}\cos(\alpha-\theta) - \dfrac{\sin\alpha}{mV}T + (1 + \dfrac{\bar{q}S\bar{c}}{2mV^2}C_{x,\alpha})q + \dfrac{\bar{q}S}{mV}C_{z,\alpha}, \\ \dot{q} = \dfrac{\bar{q}S\bar{c}}{2I_{yy}V}C_{x,q}q + \dfrac{\bar{q}S\bar{c}}{I_{yy}}C_{z,q}, \\ \dot{\theta} = q, \\ \dot{h} = V\sin(\alpha-\theta), \end{cases} \tag{2.33}$$

in which

$$\begin{cases} C_{x,v} = C_x(\alpha, \delta_e) + \dfrac{\bar{c}}{2V} C_{xq}(\alpha) q, \\ C_{z,v} = C_z(\alpha, \delta_e) + \dfrac{\bar{c}}{2V} C_{zq}(\alpha) q, \\ C_{x,\alpha} = C_{zq}(\alpha)\cos\alpha - C_{xq}(\alpha)\sin\alpha, \\ C_{z,\alpha} = C_z(\alpha, \delta_e)\cos\alpha - C_x(\alpha, \delta_e)\sin\alpha, \\ C_{x,q} = \bar{c} C_{mq}(\alpha) + \bar{c}(X_{cgr} - X_{cg}) C_{zq}(\alpha), \\ C_{z,q} = C_m(\alpha, \delta_e) + (X_{cgr} - X_{cg}) C_x(\alpha), \\ \bar{q} = \dfrac{1}{2}\rho V^2, \end{cases} \tag{2.34}$$

where V denotes the velocity, α and θ denote angle of attack and pitch angle, q denotes pitch rate, T denotes engine thrust, δ_e denotes elevator deflection, m denotes aircraft mass, $\bar{q}$ denotes dynamic pressure, ρ denotes air density, $\bar{c}$ denotes mean aerodynamic chord, S denotes wing planform area, Iyy denotes pitch moment of inertia, X_{cg} and X_{cgr} denote centre of gravity (CG) location and reference CG location, g denotes a gravitational constant, C_x, C_z, C_{xq}, C_{zq} are aerodynamic force coefficients and C_m, C_{mq} are aerodynamic moment coefficients. All parameters are determined by simulating this model around a steady wings-level flight condition at an altitude of approximately $15000ft$ with the speed of $600ft/s$ based on [78].

Although the model has multiple states, two main states are selected as identified system states, which are the angle of attack, which is to be controlled and the pitch rate q, the basic inner state. In this chapter, only one control input, elevator deflection δe, is considered, and engine thrust T is set to be constant. Before elevator deflection is practically adjusted, the control signal that is generated by the actor u, or δ_e^c in this attitude tracking problem, has to go through the actuator, which consists of a command saturation and a first-order filter with rate saturation, as shown in Fig. 2.3. δ_e^c is bounded in the range of $[-25^\circ, 25^\circ]$ and changing rate of elevator deflection is limited in the range of $[-60^\circ/s, 60^\circ/s]$ [78].

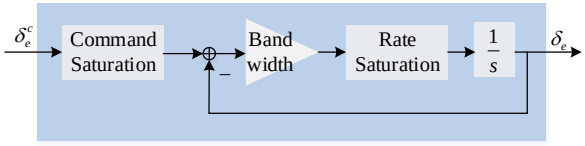


Figure 2.3: The dynamics of the actuator, which is a first-order filter with command and rate saturation.

2.4.2 Uncertainties

In flight control system design, system uncertainties, such as measurement uncertainties, unexpected changes in system dynamics or even sudden failures, need to be taken into account [34, 46]. This section will introduce the uncertainties the system has to deal with.

In practice, sensors have measurement uncertainties, and the magnitude of real-world phenomena used in this chapter is illustrated in Table 2.1 [79]. The bias acting on the feedback signals is based on the mean of the disturbances, while the noise acting on the signals is based on the standard deviation of the disturbances.

Table 2.1: Sensor uncertainties of F16 aircraft (adapted from [79]).

	Bias	Noise
q [°/s]	$1.7 \cdot 10^{-3}$	$3 \cdot 10^{-2}$
α [°]	$2.2 \cdot 10^{-1}$	$1.8 \cdot 10^{-3}$
δ_e [°]	$2.6 \cdot 10^{-1}$	$4 \cdot 10^{-2}$

Sudden partial damages might be encountered during flight. The effects of the aircraft structural damages have been investigated in [46]. It has been found that actuators, as moving components, can be affected by unforeseen faults. The first actuator fault considered in this chapter is the sudden decrease of elevator bandwidth, which is initially set to be 20.2 rad/s [76, 78]. The bandwidth, which numerically equals the reciprocal of the time constant of the first-order filter, denotes the maximum frequency of sinusoidal command that the elevator can follow. A smaller bandwidth may lead to a high gain control policy, which can result in oscillation or divergence. Another actuator fault scenario considered is the reduction of control effectiveness, which can be caused directly by actuator damages or indirectly by structural damages that change aerodynamics.

For longitudinal dynamics, damage of the horizontal stabilizer needs to be taken into consideration, which leads to significant loss in both static and dynamic stability on the directional axis with an approximately linear relationship with the scale of tip loss, whose effectiveness is reflected by the changes of the aerodynamic moment coefficients C_m and C_{mq}. Besides, these structural damages are usually accompanied by mass loss, instantaneously shifting the CG to a new location.

2.4.3 Network structure

ANNs, or more specifically multilayer perceptions (MLPs), are utilized to approximate the actor, critic and global model. For simplicity, the introduced ANNs are fully connected and consist of only three layers of nodes: an input layer, a hidden layer and an output layer. The activation function σ in the nodes of the hidden layer is a sigmoid function:

$$\varsigma(o) = \frac{1 - e^{-o}}{1 + e^{-o}}. \tag{2.35}$$

It is anti-symmetric, zero-center and differentiable, with output bounded between −1 and 1. Its derivative is continuous, differentiable and positive at every point:

$$\frac{\partial \varsigma(o)}{\partial o} = \frac{1}{2}(1 - \varsigma(o))^2. \tag{2.36}$$

The actor is implemented as a hierarchical structure, or specifically a cascaded actor network [34, 39, 63], as shown in Fig. 2.4. The first sub-network outputs a virtual reference signal of the pitch rate q, which is one input of the second sub-network, and the second

sub-network produces the control command. Compared to the "flat" actor with only one end-to-end network, the virtual reference signal q^{ref} provides a more direct instruction. 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 sub-networks adopt a sigmoid function as the activation function, to add restrictions to the pitch rate reference and the control action, which is different from the global model and the critic, where linear functions are employed. The pitch rate and the elevator deflection commands are bounded in the range of $[-20^\circ/s, 20^\circ/s]$ and $[-25^\circ, 25^\circ]$, respectively.

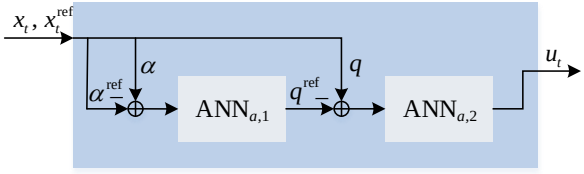


Figure 2.4: The architecture of the cascaded actor network [63], where the physical properties of aircraft dynamics are utilized.

The critic and actor networks in both GDHP and IGDHP have the same settings. The DHP technique generally surpasses HDP in tracking precision, convergence speed and success rate because the costate function is employed [34, 67]. Therefore, to take advantage of the information of derivatives, β is set to be 0.01. More neurons will improve approximation precision, but can also increase computational burden or even lead to overfitting, which will decrease the robustness of the controller. As a trade-off, the number of hidden layer neurons in the actor is 15, while in both the critic and the global system it is 25. Initial weights of the neural networks can have a great influence on the learning. In this chapter, all weights are randomly initialized within a small range of $[-0.01, 0.01]$ to reduce the impact of initialization, and bounded within the range of $[-20, 20]$ to prevent sudden failure in the learning process. 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.

2.5 Results and Discussion

Both the GDHP and the IGDHP algorithms are applied to a simulation of controlling the F16 aircraft longitudinal model. First, the flight controller learns to track a changing reference at different initial conditions. Then, these two methods are compared in the failure cases where actuator faults and structural damages take place. All numerical experiments are implemented in the presence of sensor uncertainties.

2.5.1 Different initial conditions

Figures 2.5-2.7 compare the performance of the IGDHP and GDHP approaches, when applied to control the F16 aircraft to track a given reference signal online at different initial states. To be more specific, the controllers are required to control the angle of attack α

to track the reference signal α^{ref}, which is a sinusoidal wave with the amplitude of 10 degrees and the period of 4π seconds. The sub-figures on the top present how the angle of attack α tracks the reference signal α^{ref} using these two approaches respectively, while the sub-figures on the bottom provide the tracking errors during these tasks.

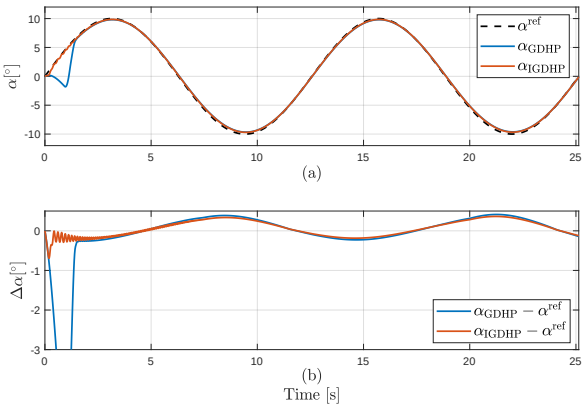


Figure 2.5: Online attitude tracking control with the zero initial state, $\alpha_0 = 0^\circ$ using GDHP and IGDHP approaches.

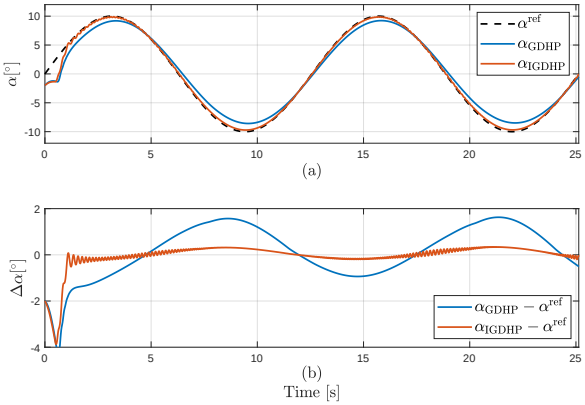


Figure 2.6: Online attitude tracking control with a negative initial state, $\alpha_0 = -2^\circ$ using GDHP and IGDHP approaches.

Take every two degrees as a scale to examine the influence of the initial states. When the initial state α_0 is $\pm 2^\circ$, both methods perform similarly to the simulation results with zero initial states, as shown in Figs. 2.5 and 2.6. When the initial state is 4°, although tracking precision decreases for the GDHP method, both methods can successfully complete the tracking task, as shown in Fig. 2.7. Nevertheless, compared to GDHP, IGDHP spends less time finding a feasible actor, leading to a smaller settling time. These results imply that incremental techniques can accelerate the learning process of the actor and critic networks.

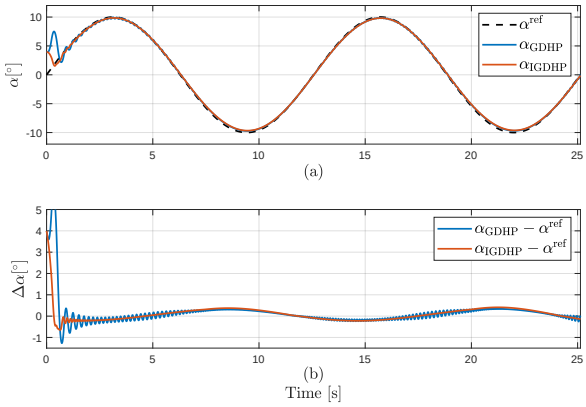


Figure 2.7: Online attitude tracking control with positive initial state, $\alpha_0 = 4^\circ$ using GDHP and IGDHP approaches.

Furthermore, when the initial state α_0 is beyond the range of $[-2^\circ, 4^\circ]$, the GDHP method cannot track the reference signal without oscillation. On the other side, the IGDHP method can deal with a wider range of initial states within $[-10^\circ, 10^\circ]$ without the loss of precision. As presented in Fig. 2.8, the angle of attack α can follow the given reference signal α^{ref} in less than 1 second in all initial conditions using the IGDHP approach, which shows that IGDHP is more robust than GDHP to various initial states.

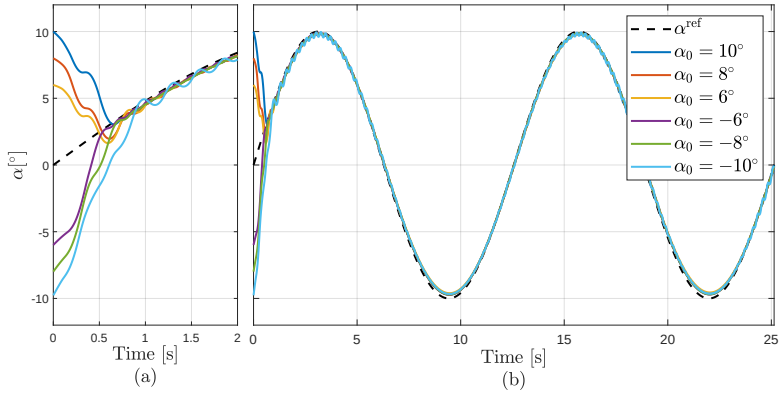


Figure 2.8: Online attitude tracking control with different initial states using the IGDHP approach.

However, Figs. 2.5-2.8 only present the nominal results when the task is successfully performed. Random factors, including initial weights of the neural networks and sensor noises, can affect the performance and sometimes can even lead to failure. To compare how robust the proposed algorithms are to these random factors, a concept of success ratio is introduced to indicate their performance, which has been widely used in [34, 39, 63]. The success ratio used in this chapter is defined as that the angle of attack α can track the

given reference signal α^{ref} after one period of α^{ref}, 4π seconds, and the tracking errors will not exceed $\pm 2^{\circ}$ hereafter. Without adjustment of parameters, 1000 times of Monte Carlo simulation are implemented to evaluate the performance of algorithms.

The results are illustrated in Table 2.2. The reason why the highest success ratio is not 100% lies in the difficulty of achieving optimal PE condition due to the circular argument between PE condition, accurate system information and stable control policy. Improving several factors can increase the success ratio and improve robustness, such as the performance of sensors, exploration noise, parameter initialization and learning rates. However, the improvement of these factors still remains an open problem and therefore this chapter only concentrates on the comparison of robustness between different methods. As presented in Table 2.2, the success ratios of IGDHP are higher than those of GDHP at any initial state. When applied to non-zero initial states, success ratios decrease dramatically for GDHP, which means that the global model is not robust enough for various initial conditions. However, the success ratios of both IGDHP and GDHP degrade heavily due to measurement uncertainties, and the impacts on IGDHP are even more severe. That is because, to achieve the quick and high-precision identified model, the incremental method adapts quickly to locally acquired data. Nevertheless, IGDHP still shows better performance.

Table 2.2: Success ratio comparison for different initial states and measurement uncertainties situations with 1000 times of Monte Carlo simulation.

	$\alpha_0/[^{\circ}]$	-2	0	2	4
Without uncertainties	GDHP	1.1%	50.3%	1.9%	1.4%
	IGDHP	32.4%	91.6%	41.3%	36.6%
With uncertainties	GDHP	0.7%	43.9%	1.6%	1.0%
	IGDHP	19.5%	54.3%	25.1%	19.8%

2.5.2 Fault-tolerant examination

The capability to adapt is one of the most important advantages of ACDs compared to other traditional control techniques. It allows the controller to learn sound policies automatically by tuning the weights of the networks and this merit makes ACDs suitable for fault-tolerant control (FTC). Fault diagnosis is a significant part of the FTC area, but will not be discussed in this chapter. It is assumed that when a sudden fault occurs, the controller can recognize it immediately. One challenge of FTC is that the controller is unable to change its policy quickly enough when faced with sudden changes in plant dynamics, while in this situation the originally learned control policy may even increase the instability of the closed loop plant. Therefore, in this chapter, the way the controller adapts to new situations is to reset the actor weights to small random numbers within the range of $[-0.01, 0.01]$ and to increase the corresponding learning rate as long as the fault is detected.

Figure 2.9 compares the online adaptability of the GDHP method and the IGDHP method in the presence of the sudden decrease of elevator bandwidth to 18 rad/s. This change is introduced after the convergence of the policy for the original system at 4π, 4.5π and 5π seconds, respectively, which corresponds to different values of the reference

signal. As shown in Fig. 2.9, GDHP shows poor adaptation performance and it results in divergence in sub-figure (c). Although the GDHP method can adapt in sub-figures (a) and (b), its tracking performance degrades and the adapted controller leads to unexpected oscillations due to a higher gain control policy. On the contrary, IGDHP is able to adapt to elevator faults, and continues to track the commands precisely. The adaptation of the actor weights during this online FTC task is presented in Fig. 2.10. There are in total 60 weights belonging to two cascaded networks. Figure 2.10 demonstrates how the controller achieves a convergent policy by adapting actor weights and sub-figures (b) and (c) take a closer look at the main learning processes at the beginning and after the elevator fault happens, respectively.

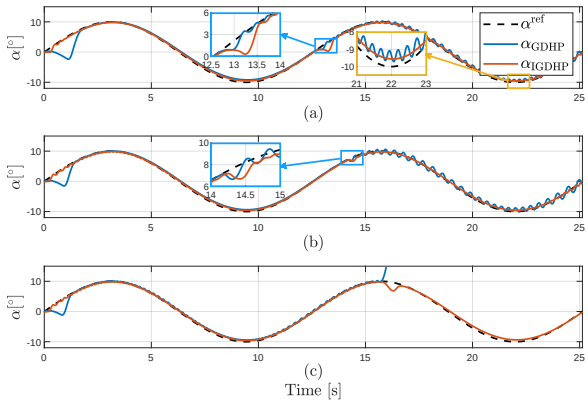


Figure 2.9: Online fault-tolerant tracking control using GDHP and IGDHP approaches in the presence of a sudden decrease of elevator bandwidth at 3 different times.

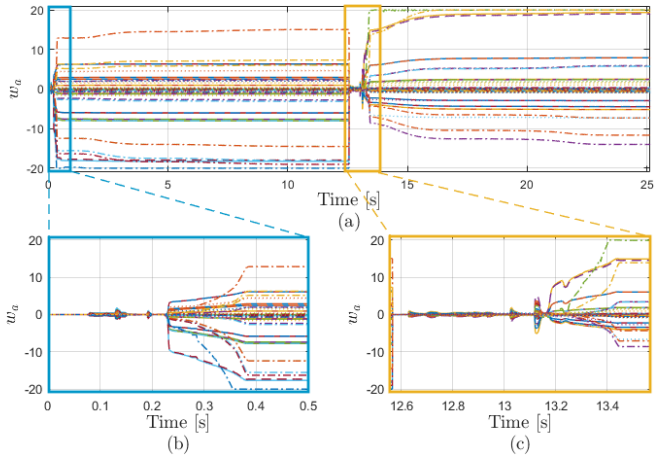


Figure 2.10: Convergence of the actor weights using the IGDHP approach when faced with a sudden decrease of elevator bandwidth at 4π seconds.

The second fault scenario considered is that the elevator suddenly loses 30% of its effectiveness during flight at 4π, 4.5π and 5π seconds, respectively. As can be seen from Fig. 2.11, only when this sudden fault happens at 4π seconds, GDHP can recover from this situation, but it encounters slightly growing oscillations thereafter, making the tracking errors have larger root mean square value. If the sudden damage occurs at the points where α^{ref} has a non-zero value, GDHP will suffer from divergence. On the other hand, IGDHP is able to rapidly adapt to the elevator fault with smaller tracking errors.

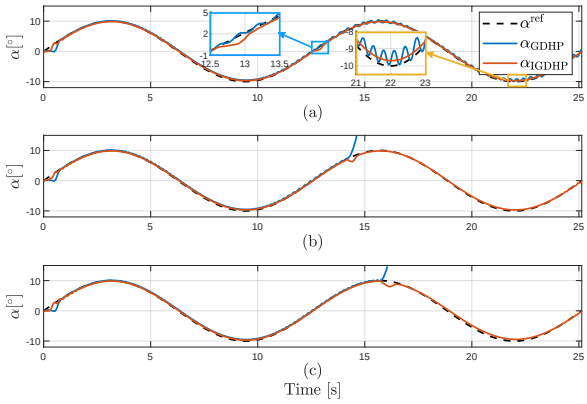


Figure 2.11: Online fault-tolerant tracking control using GDHP and IGDHP approaches in the presence of a sudden reduction of control effectiveness at 3 different times.

The last fault scenario considered is that at the three different times mentioned above, the left stabilator is damaged, while the right stabilator is still working normally. Accompanying the left stabilator damage, the CG shifts forward and to the right, producing both rolling and pitching moment increments. However, only the effects of pitching moment increments are considered for this longitudinal model and rolling effects are omitted. The reduced longitudinal damping and stability margin are also influencing the closed-loop system responses. The results are illustrated in Fig. 2.12, where at all three times, GDHP fails to adapt to the new dynamics while IGDHP shows satisfying performance.

2.6 Conclusion

This chapter 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), which often employs an artificial neural network to approximate the global system dynamics, IGDHP adopts incremental approaches instead to identify the local plant model online and to speed up policy convergence. Besides, this chapter derives a direct method from a holistic viewpoint based on differential operation, to explicitly analytically compute derivatives of cost-to-go function with respect to the critic inputs, rather than utilize conventional neural network approximation, so as to eliminate the inconsistent errors due to coupling.

Both methods are applied to an online longitudinal attitude tracking task of a nonlinear

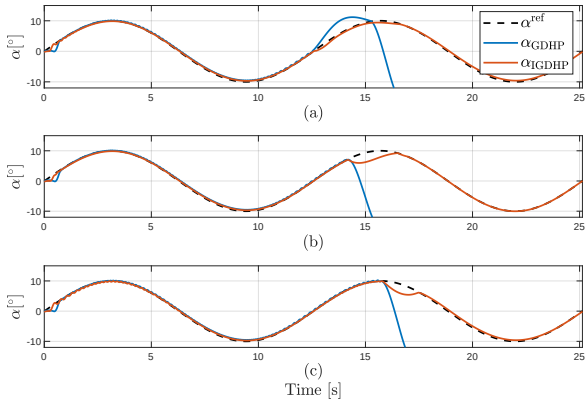


Figure 2.12: Online fault-tolerant tracking control using GDHP and IGDHP approaches when horizontal stabilizers are partially damaged at 3 different times.

F-16 Fighting Falcon system, whose dynamics are unknown to the controller. The numerical experiment results uniformly illustrate that, in comparison to conventional GDHP, IGDHP improves tracking precision, accelerates the online learning process, has advantages in robustness to different initial states and measurement uncertainties, and has increased capability to adapt when faced with unforeseen sudden faults.

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 types of function approximators, the improvement of stability and success ratio, and expansion to other application scenarios.

2.A Vector and Matrix Derivation

The following derivation process to calculate derivatives is based on differential operation [71]. Consider a critic network with a single hidden layer architecture the numbers of network inputs, neurons and outputs are n, p and 1, respectively. For convenience, define I_r as an identity matrix with a dimension of r, where r can be n, p and 1, respectively.

2.A.1 Computation of $\hat{\lambda}(\tilde{x}_t)$

The feed-forward process of the three-layer critic network is given as:

$$\hat{J} = w_{c2}^{\mathsf{T}}\sigma(w_{c1}^{\mathsf{T}}\tilde{x}), \tag{2.37}$$

where w_{c1} denotes the weight matrix connecting the input layer and the hidden layer, w_{c2} denotes the weight matrix connecting the hidden layer and the output layer, and $\sigma(\cdot)$ is an element-wise operation applied to the transfer function in the hidden layer. Compute the differentials of both sides:

$$\mathrm{d}\hat{J} = \mathrm{d}w_{c2}^{\mathsf{T}}\sigma(w_{c1}^{\mathsf{T}}\tilde{x}) + w_{c2}^{\mathsf{T}}\mathrm{d}\sigma(w_{c1}^{\mathsf{T}}\tilde{x}) = w_{c2}^{\mathsf{T}}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}) \odot (w_{c1}^{\mathsf{T}}\cdot \mathrm{d}\tilde{x})), \tag{2.38}$$

where $\odot$ denotes Hadamard product, and $\sigma'(\cdot)$ denotes the first order derivative of function σ. Perform trace operation on both sides:

$$\begin{aligned}\mathrm{tr}(\mathrm{d}\hat{J}) =&\mathrm{d}\hat{J} = \mathrm{tr}(w_{c2}^{\mathsf{T}}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})\odot(w_{c1}^{\mathsf{T}}\cdot\mathrm{d}\tilde{x})))\\ =&\mathrm{tr}((w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))^{\mathsf{T}}(w_{c1}^{\mathsf{T}}\mathrm{d}\tilde{x})) = \mathrm{tr}((w_{c1}(w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})))^{\mathsf{T}}\mathrm{d}\tilde{x}),\end{aligned} \tag{2.39}$$

The relationship between the derivative of scalar with respect to vector and their differentials, the first order derivative of critic network output with respect to its inputs, is:

$$\hat{\lambda}(\tilde{x}_t) = \frac{\partial\hat{J}}{\partial\tilde{x}} = w_{c1}(w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})). \tag{2.40}$$

2.A.2 Computation of $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_c$

Since the critic network has two weight matrices, the second-order derivatives have to be calculated separately. Perform the differential operation on both sides of Eq. (2.40):

$$\mathrm{d}\hat{\lambda}(\tilde{x}_t) = \mathrm{d}w_{c1}(w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})) + w_{c1}\cdot\mathrm{d}(w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})). \tag{2.41}$$

Firstly consider $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_{c2}$ and regard w_{c1} as a constant matrix, then Eq. (2.41) can be modified as:

$$\mathrm{d}\hat{\lambda}(\tilde{x}_t) = w_{c1}\cdot\mathrm{d}(w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})) = w_{c1}(\mathrm{d}w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))I_1. \tag{2.42}$$

Reshape both sides to column vectors:

$$\begin{aligned}\mathrm{vec}(\mathrm{d}\hat{\lambda}(\tilde{x}_t)) =&\mathrm{d}\hat{\lambda}(\tilde{x}_t) = \mathrm{vec}(w_{c1}\cdot\mathrm{d}(w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))I_1)\\ =&(I_1\otimes w_{c1})\cdot\mathrm{vec}(\mathrm{d}w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})) = (I_1\otimes w_{c1})\cdot\mathrm{diag}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))\cdot\mathrm{vec}(\mathrm{d}w_{c2})\end{aligned} \tag{2.43}$$

where $\mathrm{vec}(\cdot)$ is vector reshaping function, $\mathrm{diag}(\cdot)$ reshapes the vector to a diagonal matrix, and $\otimes$ is Kronecker product. $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_{c2}$ is the derivative of the vector $\hat{\lambda}(\tilde{x}_t)$ with respect to the matrix w_{c2}. Based on the relationship between $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_{c2}$ and the differentials of $\hat{\lambda}(\tilde{x}_t)$ and w_{c2} [71], $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_{c2}$ can be obtained:

$$\frac{\partial\hat{\lambda}(\tilde{x}_t)}{\partial w_{c2}} = ((I_1\otimes w_{c1})\cdot\mathrm{diag}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})))^{\mathsf{T}} = \mathrm{diag}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))(I_1\otimes w_{c1}^{\mathsf{T}}). \tag{2.44}$$

Then consider $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_{c1}$ and regard w_{c2} as a constant matrix. Perform the differential operation on both sides of Eq. (2.40):

$$\begin{aligned}\mathrm{d}\hat{\lambda}(\tilde{x}_t) =& \mathrm{d}w_{c1}(w_2\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})) + w_{c1}(\mathrm{d}w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}) + w_{c2}\odot\mathrm{d}\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))\\ =& I_n\cdot\mathrm{d}w_{c1}\cdot(w_{c2}\odot\sigma'(w_{c1}^{\mathsf{T}}\tilde{x})) + w_{c1}(w_{c2}\odot\mathrm{d}\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))I_1.\end{aligned} \tag{2.45}$$

Recall Eq. (2.36) and reshape both sides to column vectors:

$$\begin{aligned}
\mathrm{vec}(\mathrm{d}\hat{\lambda}(\tilde{x}_t)) =& \mathrm{d}\hat{\lambda}(\tilde{x}_t) \\
=& ((w_{c2} \odot \sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))^{\mathsf{T}} \otimes I_n)\mathrm{vec}(\mathrm{d}w_{c1}) + (I_1 \otimes w_{c1})\mathrm{vec}(w_{c2} \odot \mathrm{d}(\frac{1}{2}(1 - \sigma^{\circ 2}(w_{c1}^{\mathsf{T}}\tilde{x})))) \\
=& ((w_{c2} \odot \sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))^{\mathsf{T}} \otimes I_n) \cdot \mathrm{vec}(\mathrm{d}w_{c1}) + \\
& (I_1 \otimes w_{c1}) \cdot \mathrm{vec}(w_{c2} \odot (-\sigma(w_{c1}^{\mathsf{T}}\tilde{x}) \odot (\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}) \odot ((\mathrm{d}w_{c1})^{\mathsf{T}}\tilde{x})))) \\
=& ((w_{c2} \odot \sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))^{\mathsf{T}} \otimes I_n) \cdot \mathrm{vec}(\mathrm{d}w_{c1}) - \\
& (I_1 \otimes w_{c1})\mathrm{diag}(\mathrm{vec}(w_{c2}))\cdot \mathrm{diag}(\sigma(w_{c1}^{\mathsf{T}}\tilde{x}))\mathrm{diag}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))\mathrm{vec}(I_p(\mathrm{d}w_{c1})^{\mathsf{T}}\tilde{x}) \\
=& ((w_{c2} \odot \sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))^{\mathsf{T}} \otimes I_n - \\
& (I_1 \otimes w_{c1})\mathrm{diag}(\mathrm{vec}(w_{c2}))\mathrm{diag}(\sigma(w_{c1}^{\mathsf{T}}\tilde{x}))\mathrm{diag}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))(\tilde{x}^{\mathsf{T}} \otimes I_p)\mathrm{K})\mathrm{vec}(\mathrm{d}w_{c1}),
\end{aligned} \tag{2.46}$$

where $\cdot^{\circ 2}$ is an element-wise square operation, and K is a commutation matrix of w_{c1}. Similar to $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_{c2}$, based on the relationship between the derivative of the vector $\hat{\lambda}(\tilde{x}_t)$ with respect to the matrix w_{c1}, and the differentials of $\hat{\lambda}(\tilde{x}_t)$ and w_{c1}, $\partial\hat{\lambda}(\tilde{x}_t)/\partial w_{c1}$ is obtained as follows:

$$\begin{aligned}
\frac{\partial\hat{\lambda}(\tilde{x}_t)}{\partial w_{c1}} =& ((w_{c2} \odot \sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))^{\mathsf{T}} \otimes I_n - (I_1 \otimes w_{c1}) \cdot \\
& \mathrm{diag}(\mathrm{vec}(w_{c2}))\mathrm{diag}(\sigma(w_{c1}^{\mathsf{T}}\tilde{x})) \cdot \mathrm{diag}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))(\tilde{x}^{\mathsf{T}} \otimes I_p)\mathrm{K})^{\mathsf{T}} \\
=& (w_{c2} \odot \sigma'(w_{c1}^{\mathsf{T}}\tilde{x})) \otimes I_n - \mathrm{K}^{\mathsf{T}}(\tilde{x} \otimes I_p) \cdot \\
& \mathrm{diag}(\sigma'(w_{c1}^{\mathsf{T}}\tilde{x}))\mathrm{diag}(\sigma(w_{c1}^{\mathsf{T}}\tilde{x})) \cdot \mathrm{diag}(\mathrm{vec}(w_{c2}))(I_1 \otimes w_{c1}^{\mathsf{T}}).
\end{aligned} \tag{2.47}$$

Note that in order to obtain the derivative of a vector with respect to a matrix, tensor operation is involved, which, however, reduces the dimensionality of the original matrix. Therefore, after calculating the gradients of error with respect to weights (referring to Eq. (2.15) by chain rule), dimensionality analysis is required to reshape the results.

3

Intelligent Adaptive Optimal Control Using IGDHP with Partial Observability

Intelligent Adaptive Optimal Control Using Incremental Model-Based Global Dual Heuristic Programming Subject to Partial Observability

Chapter 2 developed an incremental model-based global dual heuristic programming (IGDHP) in the full-state feedback (FSF) situation. The situation of partial observability (PO) that involves immeasurable stochastic or time-varying dynamics will be further explored in this chapter. The incremental model will be augmented using historical observations to approximate the unknown dynamics. Then, the approximation error of local linearization and the convergence of the recursive least square algorithm that is used for updating the incremental model will be analysed. Finally, this chapter will make numerical comparisons between FSF and PO scenarios, and will verify the proposed algorithm under the circumstances of external disturbances and sudden faults.

This chapter is based on the following article:
B. Sun and E. van Kampen, "Intelligent Adaptive Optimal Control Using Incremental Model-Based Global Dual Heuristic Programming Subject to Partial Observability", *Applied Soft Computing*, vol. 103, pp. 107153, 2021. Doi: 10.1016/j.asoc.2021.107153 [80].

The identification method based on the incremental model developed in this chapter has been validated via an experiment in the following article:
B. Sun, T. Mkhoyan, E. van Kampen, R. De Breuker and X. Wang, "Vision-Based Nonlinear Incremental Control for A Morphing Wing with Mechanical Imperfections", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 6, pp. 5506–5518, 2022. Doi: 10.1109/TAES.2022.3175679 [3].

Abstract

The scarcity of information regarding dynamics and full-state feedback increases the demand for a model-free control technique that can cope with partial observability. To deal with the absence of prior knowledge of system dynamics and perfect measurements, this chapter develops a novel intelligent control scheme by combining global dual heuristic programming with an incremental model-based identifier. An augmented system consisting of the unknown nonlinear plant and unknown varying references is identified online using a locally linear regression technique. The actor-critic is implemented using artificial neural networks, and the actuator saturation constraint is addressed by exploiting a symmetrical sigmoid activation function in the output layer of the actor network. Numerical experiments are conducted by applying the proposed method to online adaptive optimal control tasks of an aerospace system. The results reveal that the developed method can deal with partial observability with performance comparable to the full-state feedback control, while outperforming the global model-based method in stability and adaptability.

3.1 Introduction

Conventional controller design for aerospace systems is commonly based on the known linearize models at different equilibrium or trimmed conditions and on PID controllers with human-scheduled gains to cover the complete operating envelope [81]. However, more complex demands such as optimization and adaptation have emerged recently, which cannot be tackled within this traditional scheme. The demand for optimization involves optimal control, in which the dynamic programming (DP) principle plays a fundamental role [82], and the Hamilton-Jacobi-Bellman (HJB) equation is often involved [83]. However, as a partial differential equation, the HJB equation is arduous to be solved analytically due to its nonlinear nature. Besides, DP-based approaches are by nature offline planning approaches in a backwards-in-time way and generally require the full knowledge of the system dynamics [24]. However, for complex systems, sometimes not only the internal dynamics, but also the information to infer its internal states can be unaccessible, i.e., full-state feedback (FSF) is no longer available[47, 50, 84]. These factors prevent traditional optimal control methods from further applications. On the other hand, adaptive control is another focal point of aerospace systems control [81, 85], which is generally considered a separate paradigm from optimal control [86]. Adaptive control concentrates on how the controller can adapt to uncertain system dynamics, and changing environments and tasks, and does not feature optimality as its paramount target. Both optimal control and adaptive control can be significant for aerospace systems. Therefore, the purpose of this chapter is to develop an adaptive optimal control approach so as to improve the optimal tracking performance without known system dynamics and perfect measurements.

Reinforcement learning (RL) is a class of bio-inspired artificial intelligence techniques, by which the agent improves its policy to maximize the received reward (or minimize the penalty) during interaction with the environment [16]. From a theoretical point of view, RL is closely linked with adaptive optimal control methods [21, 38]. A fruitful cross-fertilization of RL and control theory produces adaptive/approximate dynamic programming (ADP), whose essential goal lies in approximating the solutions of DP [25, 87]. With two essential ingredients of temporal difference (TD) error and value function approximation (VFA)

[21], ADP is a class of effective approaches to deal with adaptive optimal control problems. ADP divides the learning process into two parts, namely policy evaluation and policy improvement, which, in comparison to conventional DP, enables the controller to be computed forward in time and makes online computation feasible. Linear ADP (LADP) is a widely used technique to deal with linear optimal control problems with a quadratic performance index function [47, 50, 88, 89], and an explicit solution can be constructed [90]. Nevertheless, relying on the assumption that the dynamic system is linear time-invariant (LTI), LADP is not suitable for dealing with nonlinear or time-varying systems [50]. What is more, LADP is constricted to only employ a linear quadratic form cost, i.e., $x^\mathsf{T} Qx + u^\mathsf{T} Ru$, where Q is a positive semi-definite weight matrix and R is a positive definite weight matrix. This prevents LADP from further applications with different demands, such as addressing input saturation constraints [91–95], or releasing the input constraints [34, 59, 63, 84], i.e., R is positive semi-definite.

As an expansion of LADP, adaptive critic designs (ACDs) break the linear quadratic constraints that exist in LADP, and have demonstrated impressive success in adaptive optimal control problems [96, 97]. ACDs normally exploit artificial neural networks (ANNs) to approximate evaluation (critic) and improvement (actor) of the control policy, and consequently they can be applied to nonlinear system control problems with complicated rewards. Based on the information utilized by the critic network, ACDs are generally categorized as heuristic dynamic programming (HDP), dual heuristic programming (DHP), and global dual heuristic programming (GDHP) [26]. Among them, GDHP combines the information used by HDP and DHP and thus takes advantage of the two methods [59, 63, 98–100]. There are several structures of the critic network for GDHP [26] and the most widely used structure is the straightforward form that approximates the performance index function and its derivatives simultaneously [63, 99, 100]. However, this structure can introduce undesired inconsistent errors, so this chapter employs explicit analytical calculations derived in [59] to eliminate these inconsistent errors.

Directly learning from unknown real systems usually requires a lot of trials or episodes [97] and may even cause disasters such as misconvergence or even divergence in some extreme cases [39]. Therefore, for complex and delicate systems, such as aerospace systems, information about state transition is required. For instance, for a time-invariant affine system, given the explicit information of control effectiveness, a convergent control policy can be generated based on some assumptions [94, 101]. However, sometimes system dynamics are completely unknown. Consequently, an extra structure, such as an ANN, is introduced to approximate the system model in some literature [44, 45, 98, 100, 102]. Because training ANNs require some efforts before the parameters converge, the model network is trained offline and kept unchanged for online application in [98, 100, 102], which lacks the capability to adapt if the system is changed, whereas in [44, 45], the information of partial system dynamics is still required for online identification.

To tackle the limits of learning global system models and to achieve online fast adaptation, an incremental model is introduced in this chapter. According to a local linear regression (LLR) technique [103], the incremental model only approximates the local dynamics of the original nonlinear system instead of the global model, on the assumption of sufficiently high sampling frequency [59]. The incremental technique has been successfully combined with various classic control methods to obtain adaptive nonlinear control ap-

proaches, such as incremental nonlinear dynamic inversion (INDI) ([104]) and incremental sliding mode control (ISMC) [8, 46]. These approaches have shown success in the reduction of the model dependency and fault tolerance, but have still not addressed the optimality. On the other hand, the synthesis of incremental techniques and ACDs leads to the incremental model-based adaptive critic designs (IACDs) ([34, 59, 63]. These approaches have been applied to several flight control problems and performed well to generate adaptive optimal controllers with FSF. Nevertheless, real applications are often more complex and FSF can be unrealistic, which results in a partial observability (PO) problem. According to [50, 105], the methods coping with deterministic systems and measurements are often regarded as output feedback methods [45, 47, 48, 84, 86, 89, 106], whereas if stochastic time-varying dynamics are involved, the control problems are linked with partially observable Markov decision processes (POMDPs) [49, 50, 105, 107]. PO often occurs in aerospace systems, whose internal dynamics can be difficult to obtain and may be time-varying or stochastic, such as liquid sloshing in spacecraft with fuel tanks, infrared camera tracking with unpredictable target maneuvering, and unforeseen damages to aircraft structures changing system dynamics suddenly [50, 105]. In [50], the incremental model is for the first time applied to a flight control problem with only tracking errors directly measurable by improving LADP, and extends the approach by combining the HDP approach in [105]. However, HDP has shown inferiority in convergence speed and control precision compared to DHP and GDHP[34, 59]. In addition, the convergence of the identification technique is not analyzed in all existing literature adopting the incremental model.

The main contribution of this chapter is dealing with the PO condition in adaptive optimal tracking control of unknown nonlinear systems by introducing an augmented incremental model into the GDHP algorithm, such that an incremental model-based GDHP (IGDHP) approach is developed. The principal advantage of the IGDHP approach lies in that the incremental model accelerates online policy learning without knowing global system dynamics or offline training a model network, which allows for quick adaptation to system changes. Although some previous works are based on the incremental model [34, 47, 50, 59, 63, 84, 98, 105], this chapter discusses the convergence of the identification technique and achieves the highest 100% success ratio for the first time. The output layer of the actor network exploits a symmetrical sigmoid activation function, to satisfy the demands for tackling input saturation constraints, by multiplying an additive-determined weight vector. Different from [59], this chapter focuses on the PO situation, and improves the previous IGDHP approach by an augmented incremental model so as to deal with the unavailability of the information referring to inner system states and the unknown time-varying reference. The present research aims at bridging the gap between the discussed algorithms and real-world systems, by taking more realistic application scenarios into consideration for verification, including sensor noises, fault-tolerant tasks, parameter variations, load disturbances, and combination with other controllers in higher-level control.

The remainder of this chapter is structured as follows. Section 3.1 presents the basic formulation of the continuous optimal tracking control problem subject to input constraints. In Section 3.3, the incremental technique is introduced for online identification in both FSF and PO conditions. Section 3.4 presents the IGDHP algorithm with explicit analytical calculations and addresses the input constraints via the actor network. Then Section 3.5

verifies the developed IGDHP method by applying it to various control problems of an aerospace system. Finally, the conclusion and future research are presented in Section 3.6.

3.2 Problem Statement

Consider a nonlinear continuous system described by:

$$\dot{x} = f(x(t), u(t)), \tag{3.1}$$

where $x(t) \in \mathbb{R}^n$, and $u(t) \in \mathbb{R}^m$ are the state vector and control vector, respectively, and $f(x(t), u(t)) \in \mathbb{R}^n$ provides the physical evaluation of the state vector over time. Assume that f is Lipschitz continuous on a set $\Omega_s \subset \mathbb{R}^n$ and that the system (3.1) is controllable on Ω_s.

The output of the nonlinear system is represented as:

$$y(t) = h(x(t)), \tag{3.2}$$

where $y(t) \in \mathbb{R}^p$, and $h[x(t)] \in \mathbb{R}^p$ is the Lipschitz continuous output function. The system is also assumed to be observable.

The problem investigated in this study is in the framework of optimal tracking control problem, so the objective of the controller is to minimize the tracking error between system output $y(t)$ and reference trajectory $y^{\text{ref}}(t)$, which is defined as:

$$e(t) = y(t) - y^{\text{ref}}(t), \tag{3.3}$$

where $e(t) \in \mathbb{R}^p$ and $y^{\text{ref}}(t) \in \mathbb{R}^p$.

In the ADP scheme, the performance index function, also called cost-to-go, of optimal tracking control problem is usually presented as the cumulative sum of future costs from any initial time t:

$$J(e(t), \overline{u}(t : \infty)) = \int_t^{\infty} \gamma^{\tau - t} r(e(\tau), u(\tau)) \mathrm{d}\tau, \tag{3.4}$$

where $\overline{u}(t : \infty) = \{u(\tau) : t \le \tau < \infty\}$ denotes the system control produced by control law $\mu(e(\tau)) \in \mathbb{R}^m$ from time instant t to ∞, $r(e(t), u(t)) \in \mathbb{R}$ denotes the cost at the time instant t, and $\gamma \in (0, 1]$ is the discount factor that indicates the extent to which the short-term cost or long-term cost is concerned. For simplicity, $J(e(t), \overline{u}(t : \infty))$ is denoted by $J(t)$ and $r(e(t), u(t))$ is denoted by $r(t)$ in the following part.

Input constraints are taken into account in this chapter, which cannot be tackled merely by the linear quadratic cost. A non-quadratic functional is employed in [91, 93, 95] for regulation optimal control problems with input constraints. Nevertheless, this non-quadratic functional can relatively improve the complexity of the GDHP technique, in that the backpropagation processes need to compute partial derivatives. Moreover, in the existing standard solution to the optimal tracking control problems, a transformation is conducted with the aid of a desired control input $u_d(t)$ to build a regulation optimal control formation concerning the tracking error $e(t)$ and the feedback input $u_e(t) = u(t) - u_d(t)$. However, as claimed in [92], it is impossible to encode the input constraints into this new control problem simply by a non-quadratic functional, since only the feedback part of the control input $u_e(t)$ can be directly obtained by minimizing the performance function.

Therefore, a saturation function is directly imposed upon the control commands to satisfy the input constraints, which will be addressed by modifying the structure of the actor network in Section 3.4. In this way, the tracking problem is transformed into a regular optimal control problem subject to input constraints.

Based on TD technique [21], the cost-to-go can also be represented as:

$$J(t) = \int_t^{t+T} r(\tau)\mathrm{d}\tau + \gamma J(t+T), \tag{3.5}$$

where $T > 0$ is a time horizon. According to Bellman's optimality principle, the optimal cost-to-go is given as:

$$J^*(t) = \min_{\overline{u}(t:t+T)} \left(\int_t^{t+T} r(\tau)\mathrm{d}\tau + \gamma J^*(t+T) \right), \tag{3.6}$$

where J^* stands for the optimal value of J. Therefore, the optimal control law can be expressed as:

$$\mu^*(e(t)) = \arg\min_{\overline{u}(t:t+T)} J^*(t) = \arg\min_{\overline{u}(t:t+T)} \left(\int_t^{t+T} r(\tau)\mathrm{d}\tau + \gamma J^*(t+T) \right). \tag{3.7}$$

For nonlinear systems, the solution of Eq. (3.6) is usually intractable to be obtained analytically. Therefore, an IGDHP algorithm is introduced to iteratively solve this optimal control problem.

3.3 Incremental Model Implementation

The IGDHP algorithm requires the information of the cost at the next time instant, so the predictability of the system states is significant. This chapter considers a PO situation, where although the system is observable, the only measurement is the tracking error and even the reference can be unknown and changing. This scenario can happen in real applications in aerospace systems control problems. For instance, the docking sensors for automated transfer vehicles and the International Space Station are infrared cameras, which only measure the relative distance and angles between them as the navigation information [50]. Therefore, it is desired to build a new module to provide a mapping from the system input to the observation, which will be dealt with using the incremental model in this section.

3.3.1 Incremental model with FSF

The derivation of the incremental model starts from the FSF condition while for all incremental techniques, the following assumption is a prerequisite:

Assumption 3.1. *The sampling frequency is sufficiently high, i.e., the sampling time Δt is sufficiently small, and the system dynamics are relatively slowly time-varying.*

Remark 3.1. There are two important parts referring to Assumption 3.1. Firstly, a discrete incremental model can be introduced to represent a continuous nonlinear plant and retain high enough precision. Secondly, the discrete model does not change the properties of the original system, including controllability and observability.

It is assumed that the system is first-order continuous with respect to time at around time instant $t - \Delta t$ (denoted by t_0). Then, taking the first-order Taylor series expansion and omitting higher-order terms, the system dynamics of Eq. (3.1) at around time instant t_0 can approximately be linearized as follows :

$$\dot{x}(t) = \dot{x}(t_0) + F(x(t_0), u(t_0))(x(t) - x(t_0)) + G(x(t_0), u(t_0))(u(t) - u(t_0)) + O((\Delta x(t))^2, (\Delta u(t))^2), \tag{3.8}$$

where $F(x(t_0), u(t_0)) = \frac{\partial f^{\mathsf{T}}(x(t),u(t))}{\partial x(t)}|_{x(t_0),u(t_0)} \in \mathbb{R}^{n\times n}$ denotes the system transition matrix and $G(x(t_0), u(t_0)) = \frac{\partial f^{\mathsf{T}}(x(t),u(t))}{\partial u(t)}|_{x(t_0),u(t_0)} \in \mathbb{R}^{n\times m}$ denotes the control effectiveness matrix. $F(x(t_0), u(t_0))$ and $G(x(t_0), u(t_0))$ are bounded due to the Lipschitz continuity of f in Eq. (3.1).

For ADP-based methods, the control policy cannot be determined in advance, and therefore the higher-order term can behave as a perturbation term to affect the closed-loop performance. Nevertheless, as claimed in [46, 75], the higher-order term, $O((\Delta x(t))^2, (\Delta u(t))^2)$ satisfies:

$$\lim_{\Delta t \to 0} ||O((\Delta x(t))^2, (\Delta u(t))^2)||_2 = 0, \ \forall x \in \mathbb{R}^n, \forall u \in \mathbb{R}^m, \tag{3.9}$$

which means the norm value of the higher-order term is negligible given a sufficiently high sampling frequency. Equation (3.9) also indicates that $\forall \bar{O} > 0$, $\exists \overline{\Delta t} > 0$, satisfies that for all $0 < \Delta t \leq \overline{\Delta t}$, $\forall x \in \mathbb{R}^n$, $\forall u \in \mathbb{R}^m$, $\forall t \geq t_0$, $||O((\Delta x(t))^2, (\Delta u(t))^2)||_2 \leq \bar{O}$, i.e., there exists a Δt that guarantees the boundedness of the higher-order term and the bound can be further diminished with the increase of the sampling frequency. Besides, the LLR technique is adopted and the linearization errors will not accumulate but only affect the local system identification. Furthermore, the real-world experiments, including the ground robot [104] and aerospace systems [8, 108], have been successfully carried out based on this linearization process. Therefore, in the following part, the higher-order term is omitted for the convenience of controller design.

Assuming the states and state derivatives of the system are measurable, i.e., $\Delta\dot{x}(t)$, $\Delta x(t)$, $\Delta u(t)$ are measurable, an incremental model can be utilized to describe the system (3.8):

$$\Delta\dot{x}(t) \approx F(x(t_0), u(t_0))\Delta x(t) + G(x(t_0), u(t_0))\Delta u(t). \tag{3.10}$$

Despite the fact that physical systems are usually continuous, modern processors work in a discrete way, leading to discrete measurements and computations [59, 63]. Consequently, given a sufficiently small sampling time Δt, based on Assumption 3.1, the plant model (3.10) can be represented approximately in a discrete form:

$$\frac{x_{t+1} - x_t}{\Delta t} - \frac{x_t - x_{t-1}}{\Delta t} \approx F_{t-1}(x_t - x_{t-1}) + G_{t-1}(u_t - u_{t-1}), \tag{3.11}$$

in which the subscript t stands for the current sampling time instant, $F_{t-1} = \frac{\partial f^{\mathsf{T}}(x,u)}{\partial x}|_{x_{t-1},u_{t-1}} \in \mathbb{R}^{n\times n}$ and $G_{t-1} = \frac{\partial f^{\mathsf{T}}(x,u)}{\partial u}|_{x_{t-1},u_{t-1}} \in \mathbb{R}^{n\times m}$ denote the system transition matrix and the input distribution matrix at time instant $t-1$ for the discretized systems, respectively. From Eq. (3.11), the following incremental form of the new discrete system can be obtained:

$$\Delta x_{t+1} \approx (I_n + F_{t-1}\Delta t)\Delta x_t + G_{t-1}\Delta t \Delta u_t, \tag{3.12}$$

where I_n denotes an identity matrix and subscript n shows its dimension.

In the FSF situation, matrices F_{t-1} and G_{t-1} can be identified online with a recursive least square (RLS) algorithm [59] and each update only requires the latest data.

3.3.2 Augmented incremental model

This subsection will focus on the construction of the locally incremental model using tracking error and input measurements based on the augmented state.

Considering Eq. (3.2), the output of the system around time instant t_0 can be linearized with Taylor expansion:

$$y(t) \approx y(t_0) + H(x(t_0))(x(t) - x(t_0)), \tag{3.13}$$

where $H(x(t_0)) = \frac{\partial h^\top(x(t))}{\partial x(t)}|_{x(t_0)} \in \mathbb{R}^{p\times n}$ denotes the observation matrix. Given a sampling time Δt, the incremental dynamics of the system output can be written as:

$$\Delta y_{t+1} \approx H_t \Delta x_{t+1}, \tag{3.14}$$

in which $H_t = \frac{\partial h^\top(x)}{\partial x}|_{x_t} \in \mathbb{R}^{p\times n}$ denotes the discrete observation matrix. It has been examined that, if a nonlinear system is completely observable with its output, then the system can still be regarded as deterministic [47, 50], suggesting that the unmeasurable internal states can be reconstructed with adequate observations to provide transition information [84].

Lemma 3.1. *Given the measured input/output data over a long-enough time horizon,* $[t - N + 1, t]$, $N \geq n/p$, *the output increment* Δy_{t+1} *can uniquely be determined as follows:*

$$\Delta y_{t+1} \approx \underline{F}_t \overline{\Delta y}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N}, \tag{3.15}$$

where $\underline{F}_t \in \mathbb{R}^{p\times Np}$ *denotes the extended discrete system transition matrix,* $\underline{G}_t \in \mathbb{R}^{p\times Nm}$ *denotes the extended discrete input distribution matrix, and* $\overline{\Delta u}_{t,N} = [\Delta u_t^\top, \Delta u_{t-1}^\top, \cdots, \Delta u_{t-N+1}^\top]^\top \in \mathbb{R}^{Nm}$ *and* $\overline{\Delta y}_{t,N} = [\Delta y_t^\top, \Delta y_{t-1}^\top, \cdots, \Delta y_{t-N+1}^\top]^\top \in \mathbb{R}^{Np}$ *are the measured input/output data of* N *previous steps, respectively.*

Proof. Based on Assumption 3.1, the new discrete system described by Eqs. (3.12) and (3.14) is observable. Then the detailed proof can be found in [47, 88] and is omitted here. □

If the reference signal is slow-varying in comparison to the system dynamics, then in the time horizon $[t - N + 1, t]$, the increment of the reference signal can be ignored. Accordingly, considering Eqs. (3.3) and (3.15), the output tracking error at the next time instant can be written as:

$$\begin{aligned} e_{t+1} &= y_{t+1} - y_{t+1}^{\text{ref}} \\ &\approx y_t + \underline{F}_t \overline{\Delta y}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N} - (y_t^{\text{ref}} + \Delta y_{t+1}^{\text{ref}}) \\ &\approx e_t + \underline{F}_t \overline{\Delta y}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N} \\ &\approx e_t + \underline{F}_t \overline{\Delta e}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N}, \end{aligned} \tag{3.16}$$

where $e_{t+1} \in \mathbb{R}^p$ and $y_{t+1}^{\text{ref}} \in \mathbb{R}^p$. However, it is impossible to directly identify Matrices $\underline{F}_t$ and $\underline{G}_t$ since the reference is unknown, and put another way, the system output cannot

be measured separately. Furthermore, the last approximation in Eq. (3.16) relies on the assumption that the reference remains constant within the time horizon $[t-N+1, t]$, which can be invalid in numerous scenarios.

Consequently, a more general situation corresponding to POMDP is taken into account, and the following assumption is given:

Assumption 3.2. *The bandwidth of the reference signal is comparable with that of the system dynamics, and the dynamics of the reference signal can be represented as:*

$$\dot{y}^{\text{ref}} = f^{\text{ref}}(y^{\text{ref}}(t), y(t)), \tag{3.17}$$

where f^{ref} is Lipschitz continuous on a set $\Omega_r \subset R^p$, and differentiable almost everywhere except for finite isolated points.

The reference signal is often independent of the system output, while in some other cases the reference can partially be determined by the system output, such as moving targets equipped with anti-tracking systems [50]. Equation (3.17) provides a general reference description that can also be expressed by the time-based function, as long as the reference signal is continuous and piecewise differentiable. Similar to Eq. (3.12), the reference signal can be represented as a discrete incremental form by Taylor expansion and discretization:

$$\Delta y_{t+1}^{\text{ref}} \approx (I_p + F_{t-1}^{\text{ref}} \Delta t)\Delta y_t^{\text{ref}} + G_{t-1}^{\text{ref}} \Delta t \Delta y_t, \tag{3.18}$$

where $F_{t-1}^{\text{ref}} = \frac{\partial f^{\text{ref}\,\top}(y^{\text{ref}}, y)}{y^{\text{ref}}}|_{y_t^{\text{ref}}} \in \mathbb{R}^{p\times p}$, and $G_{t-1}^{\text{ref}} = \frac{\partial f^{\text{ref}\,\top}(y^{\text{ref}}, y)}{y}|_{y_t} \in \mathbb{R}^{p\times p}$. F_{t-1}^{ref} and G_{t-1}^{ref} can be time-varying and since f^{ref} is Lipschitz continuous, F_{t-1}^{ref} and G_{t-1}^{ref} are bounded. If the reference is independent of the controlled system, the matrix G_{t-1}^{ref} is a zero matrix.

Accordingly, an augmented system that consists of the system state and reference dynamics can be constructed by combining system representation Eqs. (3.12) and (3.14) and reference representation Eq. (3.18) [89]. Define $z_t = [x_t^\top, y_t^{\text{ref}\,\top}]^\top$ and $\Delta z_t = [\Delta x_t^\top, \Delta y_t^{\text{ref}\,\top}]^\top$, and then the following augmented system can be obtained:

$$\Delta z_{t+1} \approx \mathcal{F}_{t-1}\Delta z_t + \mathcal{G}_{t-1}\Delta u_t, \tag{3.19}$$

and

$$\Delta e_{t+1} \approx \mathcal{H}_t \Delta z_{t+1}, \tag{3.20}$$

where $\mathcal{F}_{t-1} = \begin{bmatrix} I_n + F_{t-1}\Delta t & 0 \\ G_{t-1}^{\text{ref}} H_{t-1}\Delta t & I_p + F_{t-1}^{\text{ref}}\Delta t \end{bmatrix} \in \mathbb{R}^{(n+p)\times(n+p)}$, $\mathcal{G}_{t-1} = [G_{t-1}^\top \Delta t, 0]^\top \in \mathbb{R}^{(n+p)\times m}$, and $\mathcal{H}_t = [H_t, -I_p] \in \mathbb{R}^{p\times(n+p)}$.

Hence, given the current time instant t, the increment of system state and output reference can uniquely be represented by the historical data as an augmented state equation:

$$\Delta z_{t+1} \approx \tilde{\mathcal{F}}_{t-1,t-M}\Delta z_{t-M+1} + \mathcal{U}_M \overline{\Delta u}_{t,M}, \tag{3.21}$$

where $\tilde{\mathcal{F}}_{t-a,t-b} = \prod_{i=t-a}^{t-b} \mathcal{F}_i$, and $\mathcal{U}_M = [\mathcal{G}_{t-1}, \mathcal{F}_{t-1}\mathcal{G}_{t-2}, \cdots, \tilde{\mathcal{F}}_{t-1,t-M+1}\mathcal{G}_{t-M}] \in \mathbb{R}^{(n+p)\times mM}$. Similarly, the tracking error can be represented by previous data:

$$\overline{\Delta e}_{t,M} \approx \tilde{\mathcal{V}}_M \Delta z_{t-M+1} + \tilde{\mathcal{T}}_M \overline{\Delta u}_{t,M}, \tag{3.22}$$

in which $\overline{\Delta e}_{t,M} = [\Delta e_t^\mathsf{T}, \Delta e_{t-1}^\mathsf{T}, \cdots, \Delta e_{t-M+1}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{pM}$, $\bar{\mathcal{V}}_M = [(\mathcal{H}_{t-1}\tilde{\mathcal{F}}_{t-2,t-M})^\mathsf{T}, (\mathcal{H}_{t-2}\tilde{\mathcal{F}}_{t-3,t-M})^\mathsf{T}, \cdots, \mathcal{H}_{t-M}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{pM\times(n+p)}$ and

$$\bar{\mathcal{T}}_M = \begin{bmatrix} 0 & \mathcal{H}_{t-1}\mathcal{G}_{t-2} & \mathcal{H}_{t-1}\mathcal{F}_{t-2}\mathcal{G}_{t-3} & \cdots & \mathcal{H}_{t-1}\tilde{\mathcal{F}}_{t-2,t-M+1}\cdot\mathcal{G}_{t-M} \\ 0 & 0 & \mathcal{H}_{t-2}\mathcal{G}_{t-3} & \cdots & \mathcal{H}_{t-2}\tilde{\mathcal{F}}_{t-3,t-M+1}\cdot\mathcal{G}_{t-M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \mathcal{H}_{t-M+1}\cdot\mathcal{G}_{t-M} \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{pM\times mM}.$$

3

Vectors $\overline{\Delta e}_{t,M}$ and $\overline{\Delta u}_{t,M}$ are the increments of observation and input sequences over the time interval $[t-M+1, t]$, which represent the available measured data. Then the following lemma is given:

Lemma 3.2. *Let the augmented system described by Eqs.* (3.19) *and* (3.20) *be observable. Then the increment of the system state and output reference is determined uniquely in terms of the previous data sequences over a sufficiently long time horizon* $[t-M+1, t]$, $M \geq (n+p)/p$.

Proof. Since the augmented system is observable, there exists a K, the observability index, such that $rank(\bar{\mathcal{V}}_M) < n+p$ for $M < K$, and that $rank(\bar{\mathcal{V}}_M) = n+p$ for $M \geq K$. Note that $K \geq (n+p)/p$. Therefore, let $M \geq K$, and there exists a matrix $\bar{M} \in \mathbb{R}^{(n+p)\times pM}$ such that:

$$\tilde{\mathcal{F}}_{t-1,t-M} = \bar{M}\bar{\mathcal{V}}_M. \tag{3.23}$$

Since $\bar{\mathcal{V}}_M$ has a full column rank, and its left inverse $\bar{\mathcal{V}}_M^{\text{left}}$ is given by:

$$\bar{\mathcal{V}}_M^{\text{left}} = \left(\bar{\mathcal{V}}_M^\mathsf{T}\bar{\mathcal{V}}_M\right)^{-1}\bar{\mathcal{V}}_M^\mathsf{T}, \tag{3.24}$$

then

$$\bar{M} = \tilde{\mathcal{F}}_{t-1,t-M}\bar{\mathcal{V}}_M^{\text{left}} + Z(I_{n+p} - \bar{\mathcal{V}}_M\bar{\mathcal{V}}_M^{\text{left}}) \equiv \bar{M}_0 + \bar{M}_1 \tag{3.25}$$

holds for any matrix Z, with $\bar{M}_0$ denoting the minimum norm operator and $P(\mathbb{R}^\perp(\bar{\mathcal{V}}_M)) = I_{n+p} - \bar{\mathcal{V}}_M\bar{\mathcal{V}}_M^{\text{left}}$ being the projection onto a range perpendicular to $\bar{\mathcal{V}}_M$ [88].

Note that $\tilde{\mathcal{F}}_{t-1,t-M}\Delta z_{t-M+1} = \bar{M}\bar{\mathcal{V}}_M\Delta z_{t-M+1}$ so that, according to Eq. (3.22),

$$\tilde{\mathcal{F}}_{t-1,t-M}\Delta z_{t-M+1} = \bar{M}\bar{\mathcal{V}}_M\Delta z_{t-M+1} \approx \bar{M}\overline{\Delta e}_{t,M} - \bar{M}\bar{\mathcal{T}}_M\overline{\Delta u}_{t,M}, \tag{3.26}$$

$$(\bar{M}_0 + \bar{M}_1)\bar{\mathcal{V}}_M\Delta z_{t-M+1} \approx (\bar{M}_0 + \bar{M}_1)\overline{\Delta e}_{t,M} - (\bar{M}_0 + \bar{M}_1)\bar{\mathcal{T}}_M\overline{\Delta u}_{t,M}. \tag{3.27}$$

Note, however, that $\bar{M}_1\bar{\mathcal{V}}_M = 0$ so that $\bar{M}\bar{\mathcal{V}}_M\Delta z_{t-M+1} = \bar{M}_0\bar{\mathcal{V}}_M\Delta z_{t-M+1}$, and applying $\bar{M}_1$ to Eq. (3.22) yields:

$$\bar{M}_1\overline{\Delta e}_{t,M} - \bar{M}_1\bar{\mathcal{T}}_M\overline{\Delta u}_{t,M} \approx 0. \tag{3.28}$$

Therefore,

$$\tilde{\mathcal{F}}_{t-1,t-M}\Delta z_{t-M+1} = \bar{M}_0\bar{\mathcal{V}}_M\Delta z_{t-M+1} \approx \bar{M}_0\overline{\Delta e}_{t,M} - \bar{M}_0\bar{\mathcal{T}}_M\overline{\Delta u}_{t,M} \tag{3.29}$$

independently of $\bar{M}_1$. Then from Eq. (3.21), it can be obtained that:

$$\Delta z_{t+1} \approx \bar{M}_0\overline{\Delta e}_{t,M} + (\mathcal{U}_M - \bar{M}_0\bar{\mathcal{T}}_M)\overline{\Delta u}_{t,M}. \tag{3.30}$$

This result expresses the increment of the system state and reference Δz_{t+1} in terms of the inputs and observations from time instant $t-N+1$ to time instant t, which ends the proof. □

Lemma 3.2 provides a deterministic relationship between the historical data and future states. To build a direct mapping from the historical observations and inputs to the future observations regardless of the inner states, the following theorem is presented based on Lemma 3.2:

Theorem 3.1. *Let the augmented system described by Eqs.* (3.19) *and* (3.20) *be observable. The tracking error increment* Δe_{t+1} *can be determined uniquely from the observations and control inputs over a sufficiently long time horizon,* $[t-M+1, t]$, $M \geq (n+p)/p$*:*

$$\Delta e_{t+1} \approx \underline{\mathcal{F}}_t \overline{\Delta e}_{t,M} + \underline{\mathcal{G}}_t \overline{\Delta u}_{t,M}, \tag{3.31}$$

where $\underline{\mathcal{F}}_t = \mathcal{H}_t \tilde{\mathcal{F}}_{t-1,t-M} \bar{\mathcal{V}}_M^{left} \in \mathbb{R}^{p \times Mp}$ *is the augmented transition matrix, and* $\underline{\mathcal{G}}_t = (\mathcal{H}_t \mathcal{U}_M - \mathcal{H}_t \tilde{\mathcal{F}}_{t-1,t-M} \bar{\mathcal{V}}_M^{left} \bar{\mathcal{T}}_M) \in \mathbb{R}^{p \times Mm}$ *is the augmented input distribution matrix.*

Proof. Substitute Eq. (3.30) into Eq. (3.20) and the dynamics of the measurement can directly be obtained:

$$\Delta e_{t+1} \approx \mathcal{H}_t \tilde{\mathcal{F}}_{t-1,t-M} \bar{\mathcal{V}}_M^{\text{left}} \overline{\Delta e}_{t,M} + (\mathcal{H}_t \mathcal{U}_M - \mathcal{H}_t \tilde{\mathcal{F}}_{t-1,t-M} \bar{\mathcal{V}}_M^{\text{left}} \bar{\mathcal{T}}_M) \overline{\Delta u}_{t,M}. \tag{3.32}$$

This completes the proof. □

Theorem 3.1 has a similar form to Lemma 3.1 but includes the reference signal in its representation, which enables the incremental model to predict tracking error without knowing the reference function. Matrices $\underline{\mathcal{F}}_t$ and $\underline{\mathcal{G}}_t$ in Eq. (3.31) can be identified using the RLS algorithm and then the one-step prediction of the tracking error can be made as:

$$\hat{e}_{t+1} = e_t + \hat{\underline{\mathcal{F}}}_{11,t} \Delta e_t + \hat{\underline{\mathcal{F}}}_{12,t} \overline{\Delta e}_{t-1,M-1} + \hat{\underline{\mathcal{G}}}_{11,t} \Delta u_t + \hat{\underline{\mathcal{G}}}_{12,t} \overline{\Delta u}_{t-1,M-1}, \tag{3.33}$$

where $\hat{\cdot}$ stands for the estimated or approximated value, $\hat{\underline{\mathcal{F}}}_{11,t} \in \mathbb{R}^{p \times p}$ and $\hat{\underline{\mathcal{F}}}_{12,t} \in \mathbb{R}^{p \times (M-1)p}$ are partitioned matrices from $\hat{\underline{\mathcal{F}}}_t$, and $\hat{\underline{\mathcal{G}}}_{11,t} \in \mathbb{R}^{p \times m}$ and $\hat{\underline{\mathcal{G}}}_{12,t} \in \mathbb{R}^{p \times (M-1)m}$ are partitioned matrices from $\hat{\underline{\mathcal{G}}}_t$.

In this way, the original continuous non-affine system is transformed approximately into a new discrete affine system, based on which, the IGHDP algorithm can design the control increment Δu_t.

3.3.3 Online identification with RLS algorithm

A RLS algorithm is applied to the pending matrices $\underline{\mathcal{F}}_t$ and $\underline{\mathcal{G}}_t$ online [50, 59]. For convenience, Eq. (3.31) is represented a row-by-row form as follows:

$$\Delta e_{t+1}^{\mathsf{T}} \approx \begin{bmatrix} \overline{\Delta e}_{t,M}^{\mathsf{T}} & \overline{\Delta u}_{t,M}^{\mathsf{T}} \end{bmatrix} \cdot \begin{bmatrix} \underline{\mathcal{F}}_t^{\mathsf{T}} \\ \underline{\mathcal{G}}_t^{\mathsf{T}} \end{bmatrix}. \tag{3.34}$$

Define $\overline{x}_t = [\overline{\Delta e}_{t,M}^{\mathsf{T}}, \overline{\Delta u}_{t,M}^{\mathsf{T}}]^{\mathsf{T}} \in \mathbb{R}^{M(p+m) \times 1}$ as the input information of the augmented incremental model identification, and $\underline{\Theta}_t = [\underline{\mathcal{F}}_t, \underline{\mathcal{G}}_t]^{\mathsf{T}} \in \mathbb{R}^{M(p+m) \times p}$ as the pending augmented matrix to be determined using the RLS algorithm.

A sliding window technique is employed to store sufficient historical data for online identification [84, 103]. Considering the demands for fast computation, identification and

adaptation, the width of the data window should be as small as possible with guaranteed accuracy. Consequently, according to Lemma 3.2 and Theorem 3.1, let $M = (n+p)/p$.

The main procedure of the RLS algorithm is presented as follows [63, 109]:

$$\Delta\hat{e}_{t+1}^{\mathsf{T}} = \overline{x}_t^{\mathsf{T}}\hat{\underline{\Theta}}_{t-1}, \tag{3.35}$$

$$\epsilon_t = \Delta e_{t+1}^{\mathsf{T}} - \Delta\hat{e}_{t+1}^{\mathsf{T}}, \tag{3.36}$$

$$\hat{\underline{\Theta}}_t = \hat{\underline{\Theta}}_{t-1} + \frac{\underline{\mathrm{Cov}}_{t-1}\overline{x}_t}{\gamma_{\mathrm{RLS}} + \overline{x}_t^{\mathsf{T}}\underline{\mathrm{Cov}}_{t-1}\overline{x}_t}\epsilon_t, \tag{3.37}$$

$$\underline{\mathrm{Cov}}_t = \frac{1}{\gamma_{\mathrm{RLS}}}\left(\underline{\mathrm{Cov}}_{t-1} - \frac{\underline{\mathrm{Cov}}_{t-1}\overline{x}_t\overline{x}_t^{\mathsf{T}}\underline{\mathrm{Cov}}_{t-1}}{\gamma_{\mathrm{RLS}} + \overline{x}_t^{\mathsf{T}}\underline{\mathrm{Cov}}_{t-1}\overline{x}_t}\right), \tag{3.38}$$

where $\epsilon_t \in \mathbb{R}^p$ denotes the prediction error, $\underline{\mathrm{Cov}}_t \in \mathbb{R}^{(p+m)M\times(p+m)M}$ stands for the estimation covariance matrix, which is symmetric and positive definite, and $\gamma_{\mathrm{RLS}} \in (0,1]$ is the forgetting factor in the RLS algorithm. It is noted that $\Delta\hat{e}_{t+1}$ is approximated by $\hat{\underline{\Theta}}_{t-1}$ during the implementation because $\hat{\underline{\Theta}}_t$ is obtained by the RLS algorithm after Δe_{t+1} is observed.

Assumption 3.1 implies that in a certain horizon $\mathscr{A} = [1, t]$, $M \le t \le P$, $M \ll P < \infty$, the slowly varying augmented system dynamics can be regarded as a linear plant with constant pending parameters. Hence, based on the following assumption [109], the locally approximate convergence of the RLS algorithm is analyzed.

Assumption 3.3. *For the locally linear system* (3.33)*, in the local domain* $\mathscr{A}$*, the observed vectors* $\overline{x}_M, \cdots, \overline{x}_t$ *constitute the samples of an ergodic process, such that the time averages can be utilized. The unmodeled dynamics noises within one sliding window are formulated as a zero-mean white noise vector as:*

$$\Delta e_{t+1}^{\mathsf{T}} = \overline{x}_t^{\mathsf{T}}\underline{\Theta} + e_{o,t}, \tag{3.39}$$

where $e_{o,t}$ *is the equivalent plant noise independent of the samples* $\overline{x}_t$*.*

Theorem 3.2. *If Assumptions 3.1-3.3 hold, and the RLS algorithm is implemented using Eqs.* (3.35)-(3.38)*, the approximate augmented matrix* $\hat{\underline{\Theta}}_t$ *has the trend of converging to the locally optimal matrix* $\underline{\Theta}$*.*

Proof. Because the optimal augmented matrix $\underline{\Theta}$ is valid over $\mathscr{A}$, the previous observations can uniformly be written as:

$$\Delta E_{t+1}^{\mathsf{T}} = \overline{x}_t^{\mathsf{T}}\underline{\Theta} + E_{o,t}, \tag{3.40}$$

where $\Delta E_{t+1} = [\Delta e_{M+1}, \cdots, \Delta e_{t+1}]$, $\overline{x}_t = [\overline{x}_M, \cdots, \overline{x}_t]$, and $E_{o,t} = [e_{o,M}, \cdots, e_{o,t}]$. The PE condition is indispensable for convergence analysis, which guarantees $\overline{x}_t\overline{x}_t^{\mathsf{T}}$ is positive definite. According to [109], it can be obtained that $\underline{\mathrm{Cov}}_t^{-1} = \overline{x}_t\Gamma_t\overline{x}_t^{\mathsf{T}}$, where $\Gamma_t = \mathrm{diag}([\gamma_{\mathrm{RLS}}^{t-M}, \gamma_{\mathrm{RLS}}^{t-M-1}, \cdots, 1])$, and $\mathrm{diag}(\cdot)$ reshapes the vector to a diagonal matrix. Therefore, the approximate augmented matrix $\hat{\underline{\Theta}}_t$ can be represented as:

$$\hat{\underline{\Theta}}_t = \underline{\Theta} + \tilde{\underline{\Theta}}_t = \underline{\Theta} + \underline{\mathrm{Cov}}_t\overline{x}_t\Gamma_t E_{o,t}, \tag{3.41}$$

where $\tilde{\underline{\Theta}}_t$ is the approximate error vector.

Define the approximate error correlation matrix as:

$$\hat{L}_t = E(\underline{\tilde{\Theta}}_t \underline{\tilde{\Theta}}_t^{\mathrm{T}}), \tag{3.42}$$

where $E(\cdot)$ is the expectation operation. Substituting Eq. (3.41) into Eq. (3.42), and noticing both $\underline{\mathrm{Cov}}_t$ and Γ_t are symmetrical matrices, we can obtain that:

$$\hat{L}_t = E(\underline{\mathrm{Cov}}_t \overline{x}_t \Gamma_t E_{o,t} E_{o,t}^{\mathrm{T}} \Gamma_t \overline{x}_t^{\mathrm{T}} \underline{\mathrm{Cov}}_t). \tag{3.43}$$

Recalling the independence of $e_{o,t}$ and $\overline{x}_t$, and the white noise property of $e_{o,t}$ yields:

$$\hat{L}_t = E(\underline{\mathrm{Cov}}_t \overline{x}_t \Gamma_t E(E_{o,t} E_{o,t}^{\mathrm{T}}) \Gamma_t \overline{x}_t^{\mathrm{T}} \underline{\mathrm{Cov}}_t) = \sigma_o^2 E(\underline{\mathrm{Cov}}_t \underline{\mathrm{Cov}}_{2,t}^{-1} \underline{\mathrm{Cov}}_t), \tag{3.44}$$

where σ_o^2 is the variance of $e_{o,t}$, and $\underline{\mathrm{Cov}}_{2,t}^{-1} = \overline{x}_t \Gamma_t^2 \overline{x}_t^{\mathrm{T}}$.

Rigorous evaluation of Eq. (3.44) is intractable. Hence, Assumption 3.3 is utilized to facilitate an approximate evaluation of $\hat{L}_t$ [109]. It can be found that $\underline{\mathrm{Cov}}_t^{-1}$ is a weighted sum of the outer products $\overline{x}_t \overline{x}_t^{\mathrm{T}}$, $\cdots$, $\overline{x}_M \overline{x}_M^{\mathrm{T}}$. Therefore, based on Assumption 3.3, the following approximation holds:

$$\underline{\mathrm{Cov}}_t^{-1} \approx \frac{1 - \gamma_{\mathrm{RLS}}^{t-M+1}}{1 - \gamma_{\mathrm{RLS}}} E_o, \tag{3.45}$$

where $E_o = E(\overline{x}_t \overline{x}_t^{\mathrm{T}})$ is the correlation matrix of observations. If the PE condition is satisfied, $\overline{x}_t \overline{x}_t^{\mathrm{T}}$ is positive definite and E_o^{-1} can be expected.

Substituting Eq. (3.45) into Eq. (3.44) yields:

$$\hat{L}_t = \sigma_o^2 \left(\frac{1 - \gamma_{\mathrm{RLS}}}{1 - \gamma_{\mathrm{RLS}}^{t-M+1}} \right)^2 \cdot \frac{1 - \gamma_{\mathrm{RLS}}^{2(t-M+1)}}{1 - \gamma_{\mathrm{RLS}}^2} E_o^{-1} = \sigma_o^2 \frac{1 - \gamma_{\mathrm{RLS}}}{1 + \gamma_{\mathrm{RLS}}} \cdot \frac{1 + \gamma_{\mathrm{RLS}}^{t-M+1}}{1 - \gamma_{\mathrm{RLS}}^{t-M+1}} E_o^{-1}. \tag{3.46}$$

In the steady state, i.e., $t \longrightarrow P \longrightarrow \infty$, the following equation holds:

$$\hat{L}_P = \sigma_o^2 \frac{1 - \gamma_{\mathrm{RLS}}}{1 + \gamma_{\mathrm{RLS}}} E_o^{-1}. \tag{3.47}$$

It can be found that if γ_{RLS} is very close to 1, then $\hat{L}_P \longrightarrow 0$, which means the approximate augmented matrix converges to the optimal matrix. This completes the proof. □

3.4 The IGDHP Algorithm

Since the incremental model discretely identifies the system dynamics, it is also necessary to design the controller in a discrete manner. It can be found that the optimal cost-to-go (3.6) and optimal control law (3.7), which are presented in the continuous domain, have similar forms of discrete representation [21]. Letting the time horizon in Eqs. (3.6) and (3.7) be equivalent to the sampling time, i.e., $T = \Delta t$, we can discretize Eqs. (3.6) and (3.7) as:

$$J_t^* = \min_{u_t} \{ r_t + \gamma J_{t+1}^* \}, \tag{3.48}$$

and

$$\mu^*(e(t)) = \arg\min_{u_t} J_t^* = \arg\min_{u_t} \{ r_t + \gamma J_{t+1}^* \}, \tag{3.49}$$

where r_t is the one-step cost function of the discrete-time design and can still be formulated as a linear quadratic form:

$$r_t = e(t)^\mathsf{T} Q e(t) + u(t)^\mathsf{T} R u(t), \tag{3.50}$$

where both $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ are positive semi-definite. The weight matrix R is used to control the energy cost and note that it does not have to be positive definite.

With the incremental model technique, the IGDHP algorithm can iteratively solve this discrete-time optimal control problem with an actor-critic scheme. Based on current information, the actor network generates control inputs for both the real system and the plant model. The incremental model predicts the tracking errors at the next time instant, which are utilized by the critic network to approximate cost-to-go, whose derivatives are computed analytically. The structure of the IGDHP algorithm is illustrated in Fig. 3.1.

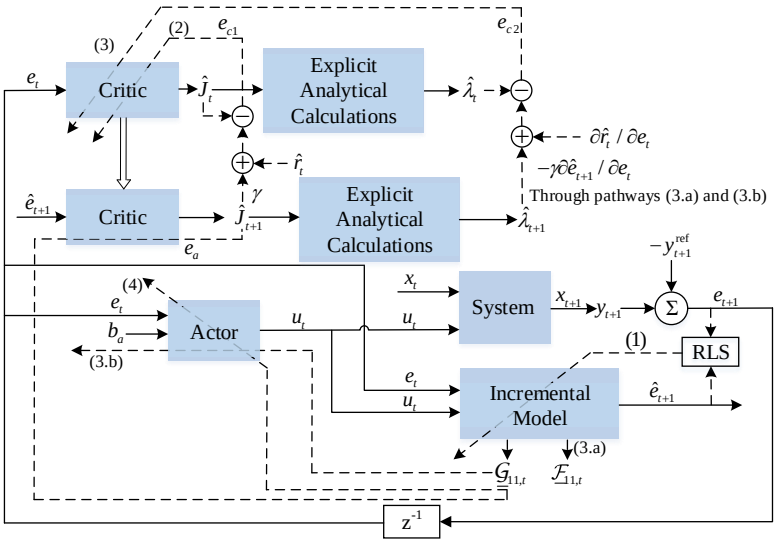


Figure 3.1: The architecture of the IGDHP algorithm with PO taken into consideration, where solid lines represent the feedforward flow of signals, dashed lines are backpropagation pathways, and the thick arrow represents the weight transmission.

For simplicity, the introduced ANNs in both the critic and actor networks are fully connected and feedforward, and consist of only three layers of nodes: an input layer, a hidden layer and an output layer. The activation function employed in the input layer is a unit-proportion linear function and in the hidden layer is a symmetrical sigmoid function, which is denoted by σ. In the following detailed implementations, the variables or pathways corresponding to the critic and actor networks and the incremental model are denoted by the subscripts c, a, and m, respectively.

3.4.1 The critic network

Although the one-step cost function consists of e_t and u_t, only e_t is set to be the input of the critic network. The main reason is that introducing u_t as an input will significantly

improve the complexity of the backpropagation. Besides, u_t is also derived from e_t by the actor network, and can still affect the approximation in an indirect way through the cost function. There is no bias term in the critic input, because the critic output is desired to be 0 when e_t is a zero vector in this case.

In the structure with explicit analytical calculations, the critic network only directly outputs the approximated cost-to-go $\hat{J}_t$, and its output layer utilizes a unit-proportion linear activation function:

$$\hat{J}_t = w_{c2,t}^{\mathsf{T}} \sigma(w_{c1,t}^{\mathsf{T}} e_t), \tag{3.51}$$

where $w_{c1,t}$ and $w_{c2,t}$ denote the critic weights between the input layer and the hidden layer, and the weights between the hidden layer and the output layer, respectively, and $\sigma(\cdot)$ denotes the activation function of the hidden layer. Then explicit analytical calculations [59] are carried out to compute $\hat{\lambda}_t$ directly using $\hat{J}_t$:

$$\hat{\lambda}_t = \frac{\partial \hat{J}_t}{\partial e_t} = w_{c1,t}(w_{c2,t} \odot \sigma'(w_{c1,t}^{\mathsf{T}} e_t)), \tag{3.52}$$

where $\odot$ is the Hadamard product, and $\sigma'(\cdot)$ denotes the first order derivative of function $\sigma(\cdot)$.

A TD method is introduced to iteratively update the critic network [16], whose principle is trying to minimize the TD error, the error between the current and successive estimates of the state value. Similar to the transformation of the optimal cost-to-go, Eq. (3.5) can also be transformed into a discrete form:

$$J_t = r_t + \gamma J_{t+1}, \tag{3.53}$$

Therefore the TD errors produced by the critic network are given as follows:

$$e_{c1,t} = \hat{J}_t - r_t - \gamma \hat{J}_{t+1}, \tag{3.54}$$

and

$$\mathrm{e}_{c2,t} = \frac{\partial [\hat{J}_t - r_t - \gamma \hat{J}_{t+1}]}{\partial e_t} = \hat{\lambda}_t - \frac{\partial r_t}{\partial e_t} - \gamma \frac{\partial \hat{e}_{t+1}}{\partial e_t} \hat{\lambda}_{t+1}, \tag{3.55}$$

where $e_{c1,t}$ denotes the TD error of the estimated cost-to-go $\hat{J}_t$ with current network weights, $\mathrm{e}_{c2,t}$ denotes the TD error of the computed derivatives $\hat{\lambda}_t$ with current network weights. It is noted that $\hat{J}_{t+1}$ and $\hat{\lambda}_{t+1}$ are predicted using the weights at time instant t.

The computation of items in Eq. (3.55) is worth analyzing. Considering Eq. (3.50), the second item on the right hand side of Eq. (3.55) can be computed by:

$$\frac{\partial r_t}{\partial e_t} = (Q^{\mathsf{T}} + Q) e_t + \frac{\partial u_t}{\partial e_t} \cdot ((R^{\mathsf{T}} + R) u_t), \tag{3.56}$$

where $\partial u_t / \partial e_t$ goes through the actor network, which will be introduced in the next subsection. Then $\partial \hat{e}_{t+1} / \partial e_t$ requires to be handled with care since there exist two pathways for e_t to influence $\hat{e}_{t+1}$. As shown in Fig. 3.1, one pathway goes through the incremental

model directly (pathway 3.a), while another pathway firstly passes through the actor network and then goes through the incremental model (pathway 3.b):

$$\frac{\partial \hat{e}_{t+1}}{\partial e_t} = \underbrace{\frac{\partial \hat{e}_{t+1}}{\partial e_t}|_m}_{\text{pathway (3.}a\text{)}} + \underbrace{\frac{\partial u_t}{\partial e_t}|_a \cdot \frac{\partial \hat{e}_{t+1}}{\partial u_t}|_m}_{\text{pathway (3.}b\text{)}}, \tag{3.57}$$

Based on the Markov theory, the states at the time instant $t+1$ only have a relationship with the states at the time instant t. Therefore, according to Eq. (3.33), Eq. (3.57) can be simplified by the incremental model as:

$$\frac{\partial \hat{e}_{t+1}}{\partial e_t} = (I_p + \underline{\hat{\mathcal{F}}}_{11,t}^{\mathsf{T}}) + \frac{\partial u_t}{\partial e_t} \cdot \underline{\hat{\mathcal{G}}}_{11,t}^{\mathsf{T}}, \tag{3.58}$$

Then, IGDHP combines both kinds of TD errors in an overall error function $E_{c,t}$:

$$E_{c,t} = \beta \frac{1}{2} e_{c1,t}^2 + (1-\beta)\frac{1}{2} \mathrm{e}_{c2,t}^{\mathsf{T}} \mathrm{e}_{c2,t}, \tag{3.59}$$

where β is a scalar within a range of $[0,1]$. If $\beta = 1$, then it becomes pure IHDP. If $\beta = 0$, then the tuning of weights merely depends on the TD error of computed derivatives $\hat{\lambda}_t$, and consequently it is equivalent to IDHP.

Given a learning rate η_c, the critic weights w_{ci}, where $i = 1,2$, are updated with a gradient-descent algorithm to minimize the overall error $E_{c,t}$:

$$w_{ci,t+1} = w_{ci,t} - \eta_c \cdot \frac{\partial E_{c,t}}{\partial w_{ci,t}}, \tag{3.60}$$

where

$$\frac{\partial E_{c,t}}{\partial w_{ci,t}} = \frac{\partial \hat{J}_t}{\partial w_{ci,t}} \cdot \frac{\partial E_{c,t}}{\partial \hat{J}_t} + \frac{\partial \hat{\lambda}_t}{\partial w_{ci,t}} \cdot \frac{\partial E_{c,t}}{\partial \hat{\lambda}_t} = \beta \frac{\partial \hat{J}_t}{\partial w_{ci,t}} \cdot e_{c1,t} + (1-\beta)\frac{\partial \hat{\lambda}_t}{\partial w_{ci,t}} \cdot \mathrm{e}_{c2,t}, \tag{3.61}$$

in which, $\partial \hat{\lambda}_t / \partial w_{ci,t}$ represents the second-order mixed gradient of the estimated cost-to-go $\hat{J}_t$, and how to compute it is given without derivation [59] as follows:

If $i = 2$, then

$$\frac{\partial \hat{\lambda}_t}{\partial w_{c2,t}} = \mathrm{diag}(\sigma'(w_{c1,t}^{\mathsf{T}} e_t))(I_1 \otimes w_{c1,t}^{\mathsf{T}}), \tag{3.62}$$

where $\otimes$ is the Kronecker product; and if $i = 1$, denote n_c as the number of neurons in the hidden layer, and then

$$\begin{aligned} \frac{\partial \hat{\lambda}_t}{\partial w_{c1,t}} =& (w_{c2,t} \odot \sigma'(w_{c1,t}^{\mathsf{T}} e_t)) \otimes I_p - \\ & K^{\mathsf{T}}(e_t \otimes I_{n_c})\mathrm{diag}(\sigma'(w_{c1,t}^{\mathsf{T}} e_t))\mathrm{diag}(\sigma(w_{c1,t}^{\mathsf{T}} e_t))\mathrm{diag}(\mathrm{vec}(w_{c2,t}))(I_1 \otimes w_{c1,t}^{\mathsf{T}}), \end{aligned} \tag{3.63}$$

where K is a commutation matrix of $w_{c1,t}$, and $\mathrm{vec}(\cdot)$ is a vector reshaping function. Note that the tensor operation can reduce the dimensionality of a matrix, and therefore dimensionality analysis is involved to reshape the results after computing $\partial \hat{\lambda}_t / \partial w_{ci,t}$.

3.4.2 The actor network

Although in [102] a single critic network structure is utilized, since the IGDHP algorithm is actually implemented in a discrete way, an actor network is required for approximating the optimal control. Safety is vital for real physical systems and thus some restrictions are usually added to system control. In this chapter, the output layer of the actor network employs a symmetrical sigmoid activation function, and is multiplied by an additive unchanged weight vector $u_b = [u_{b1}, \cdots, u_{bm}]^\mathsf{T}$, where $u_{bi} \geq 0$, for $i = 1, \cdots, m$, so that the system control $u_t = [u_{1,t}, \cdots, u_{m,t}]^\mathsf{T}$ outputted by the actor network is bounded by u_b, i.e., $|u_{i,t}| < u_{bi}$ for $i = 1, \cdots, m$, as shown in Fig. 3.2. Consequently, the actor network is presented as:

$$u_t = u_b \odot \sigma(w_{a2,t}^\mathsf{T} \sigma(w_{a1,t}^\mathsf{T} [e_t^\mathsf{T}, b_a]^\mathsf{T})), \tag{3.64}$$

where b_a is a constant bias term, which is introduced because the system control may not be a zero vector given zero tracking error, $w_{a1,t}$ and $w_{a2,t}$ are the weights of the actor network, and the way to define them is similar to that of $w_{c1,t}$ and $w_{c2,t}$.

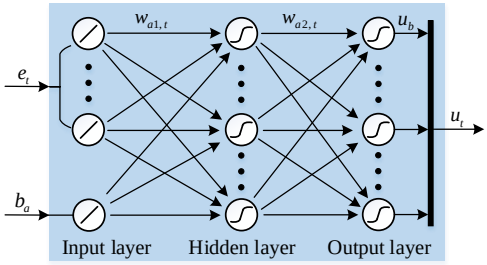


Figure 3.2: The structure of the actor network, where the input layer employs a unit-proportion linear activation function while the hidden and output layers exploit a symmetrical sigmoid activation function.

The purpose of the actor network is to generate a near-optimal control policy to minimize the future approximated cost-to-go $\hat{J}_{t+1}$:

$$u_t^* = \arg\min_{u_t} E_{a,t} = \arg\min_{u_t} \frac{1}{2} \mathrm{e}_{a,t}^2, \tag{3.65}$$

where $E_{a,t}$ denotes the overall error and $\mathrm{e}_{a,t}$ stands for the error between the approximated future cost-to-go $\hat{J}_{t+1}$ and the target 0 cost-to-go, i.e., $\mathrm{e}_{a,t} = \hat{J}_{t+1}$.

The system control u_t is also an input of the incremental model, so even though u_t does not appear in the reward function, i.e., R is zero, it has an influence on the critic output at the next time instant. Therefore, a gradient-descent algorithm can be implemented to iteratively solve Eq. (3.65) by the 4th pathway starting from $\hat{J}_{t+1}$ through $\hat{e}_{t+1}$ to u_t. Different from the straightforward form, whose back-propagation pathways of the actor network can start from either $\hat{J}_{t+1}$ or $\hat{\lambda}_{t+1}$, there is only one back-propagation pathway for IGDHP with explicit analytical calculations to update the actor weights. Nevertheless, this explains exactly why the structure with explicit analytical calculations surpasses the straightforward structure, in that it releases the the restriction of scalar β of being 0 or 1. Specifically, for the straightforward form, if $\beta = 0$, then the elements in w_{c2} linked to $\hat{J}_{t+1}$ will never be updated, and if the actor network is trained through the pathway leading from

$\hat{J}_{t+1}$, the back-propagation cannot be carried out. Similarly, if the back-propagation channel of the actor network starts from $\hat{\lambda}_{t+1}$, then $\beta \neq 1$ is mandatory for the straightforward form. On the contrary, the structure with explicit analytical calculations has no such limitations on β, because even though $\beta = 0$, the critic network can still be trained.

As presented in Fig. 3.1, the actor weights are updated along the 4th back-propagation pathway with a learning rate η_a:

$$w_{ai,t+1} = w_{ai,t} - \eta_a \cdot \frac{\partial E_{a,t}}{\partial w_{ai,t}}, \tag{3.66}$$

where $i = 1, 2$, and

$$\frac{\partial E_{a,t}}{\partial w_{ai,t}} = \frac{\partial u_t}{\partial w_{ai,t}} \cdot \frac{\partial \hat{e}_{t+1}}{\partial u_t} \cdot \frac{\partial \hat{J}_{t+1}}{\partial \hat{e}_{t+1}} \cdot \frac{\partial E_{a,t}}{\partial \hat{J}_{t+1}} = \frac{\partial u_t}{\partial w_{ai,t}} \cdot \hat{\underline{G}}_{11,t}^{\mathsf{T}} \cdot \hat{\lambda}_{t+1} \cdot \hat{J}_{t+1}. \tag{3.67}$$

So far the implementation of the proposed IGDHP with PO control scheme has been introduced. The procedure is briefly summarized in the following Algorithm 3.1, where line 6 is the dividing line between the current time instant and the next time step.

Algorithm 3.1: Design procedure of the IGDHP with PO control scheme.

1 **Initialization:** initialize system states, and parameters of the identifier, the critic network and the actor network;
2 **while** *terminal condition is not triggered* **do**
3 compute the control input u_t using the actor network (Eq. (3.64)) with the current observation;
4 predict the one-step observation $\hat{e}_{t+1}$ using the identifier (Eq. (3.32)) with the stored data;
5 evaluate the current control policy using the critic network (Eqs. (3.51) and (3.52));
6 sample the real one-step observation e_{t+1} by applying u_t to the real plant;
7 update the weights of the actor and critic networks via the backpropagation technique (Eqs. (3.60) and (3.66));
8 **if** *sufficient samples are stored in the sliding window* **then**
9 update the identifier using the RLS algorithm (Eqs. (3.36)-(3.38));
10 **else**
11 continue;
12 **end**
13 update the stored data in the sliding window;
14 **end**

Remark 3.2. The convergence analysis of the online identification has been presented in Section 3.3.3, and the convergence analysis of the ADP scheme has been investigated in [98, 99]. However, as stated in [59], it is currently unfeasible to theoretically prove the closed-loop stability of the IGDHP algorithm due to its completely online implementation. Accordingly, repeating numerical experiments are carried out in the next section for verification.

3.5 Numerical Experiments

This section assesses the developed IGDHP algorithm on a practical aerospace application. Firstly, traditional GDHP with PO (GDHP-PO), IGDHP with FSF (IGDHP-FSF) and IGDHP with PO (IGDHP-PO) are compared by applying them on an attitude tracking control task. Then the IGDHP-PO algorithm is adopted for an altitude control problem in combination with a hierarchy technique and PID controller.

3.5.1 Aerospace system model

A nonlinear model of aircraft is set up utilizing the public data [78] and only the longitudinal dynamics are taken into consideration. All parameters without special explanation in this chapter are determined by trimming the aerodynamic model in a steady wings-level flight condition at 15000 ft and 600 ft/s, which will be referred to as the benchmark condition.

In this longitudinal control problem, there are 6 states, namely altitude h, airspeed v, pitch angle θ, angle of attack (AOA) α, flight path angle γ_F and pitch rate q. Nevertheless, for a specific task, it is feasible to select only some main states to reduce computation. For instance, for the AOA tracking task, which is an attitude control problem, only pitch rate q, the basic state in the rate loop, is chosen as the additional feature state in [59], while other states are considered as parts of the black box to be identified.

For the longitudinal aerodynamic model, there are 3 control inputs, namely leading edge flap deflection δ_{lef}, engine thrust T and elevator deflection δ_e. The control surface of the leading edge flap cannot be directly changed by the pilot [78], and therefore is regarded as an inner unknown dynamics. Out of simplicity for implementation and convenience for analysis, a simple PID thrust control to maintain the airspeed is designed in a separate control loop [46], such that only one control input, elevator deflection δ_e, is taken into concern and a mapping between one control input and one final output can be constructed. Before the elevator deflection is practically adjusted, the system control u_t generated by the actor, or called elevator deflection command δ_e^c in this aerospace application, has to go through the actuator, which is modeled as a first-order system with rate and position constraints, as illustrated in Fig. 3.3 [59]. The bandwidth of actuator is 20.2 rad/s, deflection cannot exceed the range of $[-25\ \text{deg}, 25\ \text{deg}]$ and its changing rate is bounded in the range of $[-90\ \text{deg/s}, 90\ \text{deg/s}]$ [46, 78].

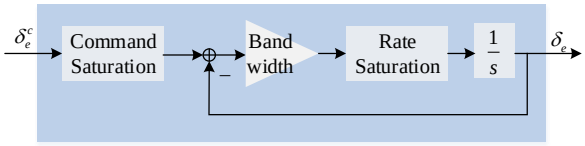


Figure 3.3: The dynamics of the actuator, a first-order system with rate and position limits [59].

In real-world applications, noise is unavoidable, and unforeseen changes in system dynamics or even sudden failures might be encountered. Consequently, there is a need to approximately model these uncertainties for verification of the developed method. Non-zero mean white measurement noise acting on the feedback signals and the actuator is taken into account, and the magnitude of real-world phenomena utilized for simulation is given in Table 3.1 [79]. This noise can have impacts on the performance of the controller,

but can also act as exploration noise to better satisfy the persistent excitation (PE) condition [59]. It is noted that only PO methods requires to consider measurement noise, while in the FSF condition, the acquired data is assumed noise-free.

Table 3.1: Magnitude of the noises having impacts on controller design (adapted from [79]).

	h [m]	θ, α, γ_F [deg]	q [deg/s]	δ_e [deg]
Mean	1	2.2×10^{-1}	1.7×10^{-3}	2.6×10^{-1}
Standard deviation	8×10^{-2}	1.8×10^{-3}	3×10^{-2}	4×10^{-2}

Sudden structural damages may be encountered during flight, which require fault-tolerant control (FTC) [87]. Fault diagnosis is a significant part of FTC, but will not be discussed in this chapter. Several kinds of sudden faults and their aftermath have been scrutinized in [46]. It has been investigated that moving mechanisms, such as actuators, are more likely to be affected by unexpected faults. There are four faults taken into consideration, namely the reduction of bandwidth, strengthening of rate limitation, output deviation and reduction of control effectiveness, while the last situation can also be triggered by the structural damages changing aerodynamics [59]. As to the faults of static structure, damage of the horizontal stabilizer is selected to be investigated. This damage has a significant impact on both static and dynamic stability for longitudinal dynamics, and mass loss is often encountered simultaneously, which will instantaneously change the center of gravity.

3.5.2 Simulation results

An AOA tracking problem is firstly taken into consideration. How to implement the GDHP method can be found in [59] and is omitted here. For better comparison, the settings in IGDHP-FSF, IGDHP-PO and GDHP-PO are as consistent as possible and the best possible parameters are ascertained by repeating the experiment. The actor, critic and model networks are all fully connected and consist of three layers of nodes. For balancing the accuracy with the computational efficiency, we experimentally set the number of hidden layer neurons in all three networks as 25 and a sigmoid function is adopted as the activation function. Both actor network and model network introduce a bias term as an input, and $b_a = b_m = 0.01$. Large random initial weights can significantly affect learning performance, whereas small ones will decrease initial exploration efficiency. Therefore, as a trade-off, All weights are randomly initialized within a small range of $[-0.1, 0.1]$, and bounded within the range of $[-20, 20]$. With the information of derivatives, the DHP technique generally surpasses HDP in tracking precision, convergence speed and success rate [50] and therefore β is chosen to be 0.1 to take advantage of the derivative information. We experimentally choose $Q = 8$, $R = 1$ and $\gamma = 0.99995$ to formulate the critic network. It should be noted that R can also be 0, but for the purpose of decreasing energy cost, it is set positive definite. As mentioned above, the system state only includes AOA and pitch angle, and therefore the minimum width of the sliding window is 3. To enhance the stability, the window width is set to 4 for IGDHP-PO and IGDHP-FSF, and 5 sets of data are utilized by GDHP-PO to ensure the same level of data use, in that both IGDHP-PO and IGDHP-FSF employ the increment information, which is acquired from 5 time instants. $\underline{\hat{\mathcal{F}}}_t$ is initialized to be

composed by 4 identity matrices and $\hat{\underline{G}}_t$ starts from a zero matrix. γ_{RLS} is chosen to be 0.99995. $\underline{\text{Cov}}_t$ originally is a diagonal matrix with all main diagonal elements chosen to be 10^7 [109], since the trace of $\underline{\text{Cov}}_t$ indicates the magnitude of the estimated errors, which can initially be infinite due to insufficient or incorrect knowledge of the system [110]. Besides, a descending method is applied to guarantee effective learning, which suggests that the learning rates are initially set to be large numbers and gradually decrease at each time step by being multiplied by a descending coefficient until the bound is reached, and the corresponding parameters are given in Table 3.2. All experiments are carried out with a sampling frequency of 1kHz and computed using Euler's method. In order to achieve the PE condition, a probing signal is introduced at the initial exploration stage. How to design a good probing signal is still an open problem, but for this chapter a 3211 disturbance signal that changes its sign over time as a particular proportion [47, 59] is introduced to excite the system modes.

Table 3.2: Parameters about the descending learning rates.

	η_a	η_{c1}	η_{c2}	η_m
Initial value	2	1	0.2	1
Descending coefficient	0.999995	0.9995	0.995	0.999995
Lower bound	10^{-2}	10^{-4}	10^{-4}	10^{-2}

Firstly, the system is supposed to track a human-designed AOA reference signal online using IGDHP-FSF, IGDHP-PO and GDHP-PO, respectively. The AOA reference signal varies around the trimmed condition, namely 2.6638 deg. The comparison of the performance is given in Fig. 3.4, where the system is initialized to be at the benchmark condition. If successfully implemented, all three methods can complete the tracking task. Among them, IGDHP-FSF shows the best performance, with a control policy that converges fast. In the PO condition, both IGDHP-PO and GDHP-PO methods can track the reference after a small period of vibration and get similar performances. The settling time of IGDHP-PO is slightly smaller than that of the GDHP-PO method but it can be ignored in the practical applications. Due to the influence of measurement noises, the tracking precision of these PO-based methods is lower than IGDHP-FSF. As presented in subfigure (b), the tracking errors (denoted by $\Delta\alpha$) of IGDHP-PO and GDHP-PO keep oscillating around the mean value of the AOA sensor noise, while the tracking error of IGDHP-FSF is approximately 0 at the stable stage.

Figure 3.5 illustrates the elevator deflection command produced by the IGDHP-PO method starting from the benchmark condition. From the inset, it can be seen that a 3211 disturbance signal, is applied to kick off the learning process. The control input command turns to be nearly 0 after the probing signal vanishes since the initial weights of the actor network are tiny random values, as presented in Fig. 3.6. Then both critic and actor networks are excited and the actor-critic scheme behaves as a high-gain controller during the exploration stage until the weights converge. The learning process is performed totally online and will continue even after the policy has converged.

In addition, the initial condition can have an impact on the controller while the proposed IGDHP-PO approach can deal with a wide range of initial states within [−10 deg, 15 deg]

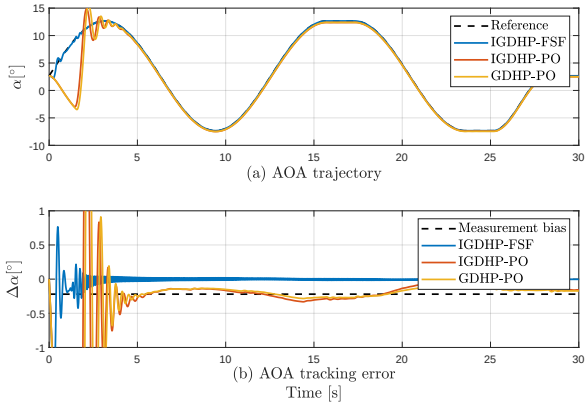


Figure 3.4: Online AOA tracking control with initial benchmark condition.

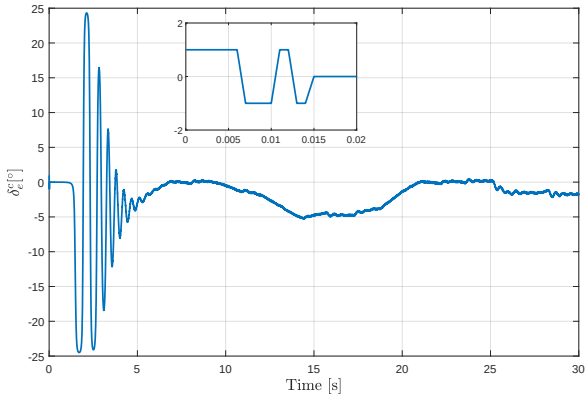


Figure 3.5: Elevator deflection command produced by the IGDHP-PO method starting from the benchmark condition.

without loss of precision. As presented in Fig. 3.7, the AOA can track the given reference signal α^{ref} in less than 2 seconds for all initial conditions using the IGDHP-PO approach, which is indicative of its competent adaptability and robustness.

Nevertheless, only when the task is successfully carried out, can the results presented above make sense. Random factors, such as initial weights of ANNs and measurement noises, can have impact on the performance and occasionally even trigger divergence and failure. To compare the robustness of these algorithms to various aspects, a concept of success ratio is introduced as a performance index. For a successful implementation, the amplitude of the AOA trajectory cannot transcend a range of $[-15\ \text{deg}, 20\ \text{deg}]$ over the whole control period, the system is supposed to be able to track the reference signal after 2π seconds, and the tracking errors will not exceed 1 deg hereafter. Remaining all parameters unchanged except for the probing signal, 1000 Monte Carlo simulations are implemented to assess the performance of these approaches.

The experiment is executed with 7 different initial AOAs and equal reference to evaluate

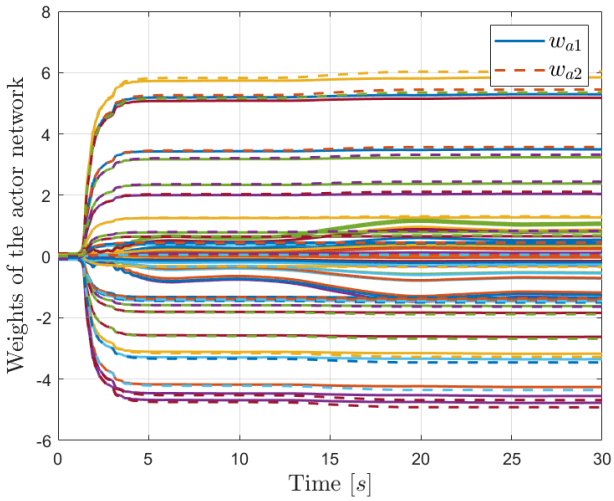


Figure 3.6: Convergence of the actor weights using the IGDHP-PO approach with initially benchmark condition.

the robustness of these approaches towards initial tracking errors and the results regarding success ratio are illustrated in Table 3.3. Both IGDHP-FSF and IGDHP-PO have a success ratio of 100% in the benchmark condition, which implies these approaches are stable for this tracking control problem. On the other hand, the success ratio of GDHP-PO in the benchmark condition is merely 51.2% and for all initial states, the success ratios of IGDHP-FSF and IGDHP-PO outclass those of GDHP-PO, demonstrating the incremental technique can significantly improve system identification compared to the traditional global ANN-based model in this online application. The success ratios of IGDHP-PO are generally smaller than those of IGDHP-FSF, which is intuitively predictable, while the differences are less than 1% for more than half of initial states and this shows that the IGDHP-PO can cope with PO situations. It is shown that the success ratios with $\alpha_0 = -5$ deg and $\alpha_0 = 0$ deg decrease by 35.5% and 21.5% for IGDHP-FSF and by 53.7% and 26.5% for IGDHP-PO compared to the benchmark condition, respectively. Besides the algorithms themselves, this reduction of success ratio can also be caused by the collective effect of the nominal reference signal and inherent dynamics of the system. Besides, it should also be noted that in multiple cases the success ratio is not 100%, which is 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 [59]. Although some cases can achieve a success ratio of 100%, random factors prevent the controller from consistently perfect tracking. Nevertheless, there is still the prospect of full success and the results presented in Table 3.3 are obtained based on current settings. The development of various aspects can benefit the stability and improve success ratio, such as sensor precision, probing signal, parameter initialization and learning rates. It is still an open problem to improve these factors and therefore this chapter concentrates on the assessment of robustness between IGHDP-FSF, IGDHP-PO and GDHP-PO.

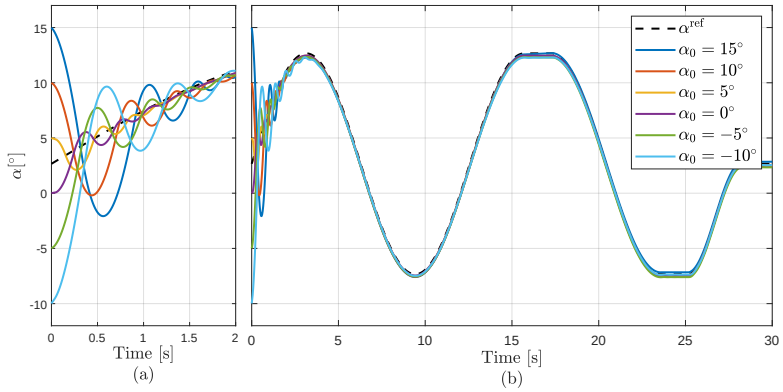


Figure 3.7: Online AOA tracking control with different initial states using the IGDHP-PO approach.

Table 3.3: Success ratio comparison for different initial states with 1000 times of Monte Carlo simulation.

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

[a]AOA value in the benchmark condition.

The adaptability is one of the most significant aspects through which ACD approaches show their superiority over other conventional optimal control methods. Sound policies can be learned automatically by updating the weights of the networks, which enables ACDs to be applied to FTC problems. Therefore, the following part investigates and compares the capability to adapt of IGDHP-FSF, IGDHP-PO and GDHP-PO by applying them to two fault scenarios, where the system is supposed to track a sinusoidal reference signal. Considering the moment when the faults happen can have an influence to the results, the faults are designed to take place at the instants of 4π seconds and 5π seconds, respectively.

The first fault scenario examined is that the elevator actuator is partially damaged. Specifically, the elevator suddenly loses 30% of its effectiveness and 50% of its bandwidth, and the bound of deflection rate degrades by 1/3 to [-60 deg/s, 60 deg/s]. Besides, there is a constant disturbance acting on the actuator, making the real deflection 2 deg larger than the produced commands. The results are illustrated in Figs. 3.8 and 3.9, where the malfunction of the actuator does not exceedingly affect the performance of these adaptive controllers, especially for IGDHP-FSF, which acts completely to normal after a small oscillation, without significant increase of tracking errors. The impacts of the sudden damage on IGDHP-PO and GDHP-PO are more obvious, which escalates their tracking errors to varying degrees. It can be seen that the tracking errors of both approaches increase after encountering instantaneous damage, while IGDHP-PO relatively surpasses GDHP-PO since the latter has larger errors. Nevertheless, all approaches are capable of handling this fault scenario.

The second fault scenario investigated is the damage of the left horizontal stabilizer. With an intact right horizontal stabilizer, the center of gravity inevitably shifts from the

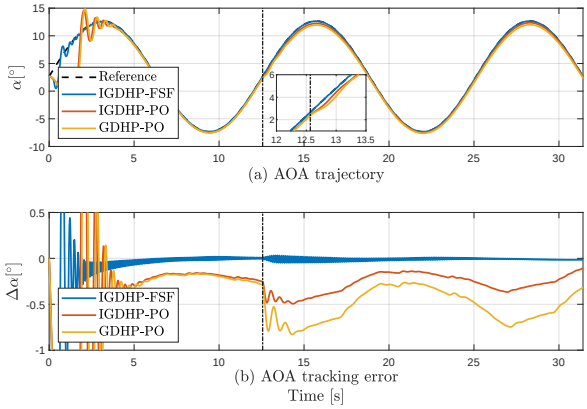


Figure 3.8: Online AOA tracking control with a malfunction of the actuator occurring at the 4π seconds.

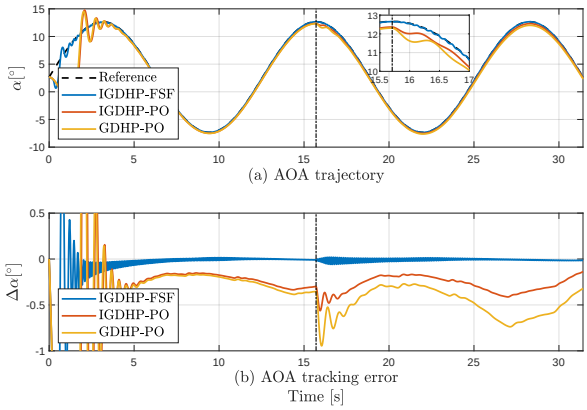


Figure 3.9: Online AOA tracking control with a malfunction of the actuator occurring at the 5π seconds.

normal position, which will lead to an increment of pitching moment. This structural damage can also affect the closed-loop performance by degrading longitudinal damping and stability margin. As presented in Figs. 3.10 and 3.11, the static structural fault caused by damage of the horizontal stabilizer demonstrates a bigger impact compared to the considered actuator fault. At large positive values of AOA, the tracking performance of all three approaches significantly degrades to different extents. On the whole, all approaches can adapt to this sudden fault in that their performance is improving with the time. IGDHP-PO still outperforms GDHP-PO in adaptation since after one period of reference signal, the tracking errors of IGDHP-PO has declined to less than 1.5 deg while tracking errors of GDHP are beyond 2 deg. Synthesizing two fault scenarios, it can be concluded that the incremental technique has an advantage of quick adaptation over the conventional ANN-based global model.

Then, more practical application scenarios are investigated, where the altitude control loop assisted with the PID controller is introduced to generate a more realistic reference

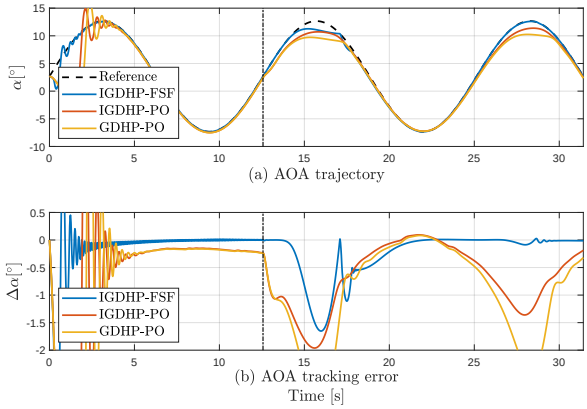


Figure 3.10: Online AOA tracking control with the left horizontal stabilizer damaged at the 4π seconds.

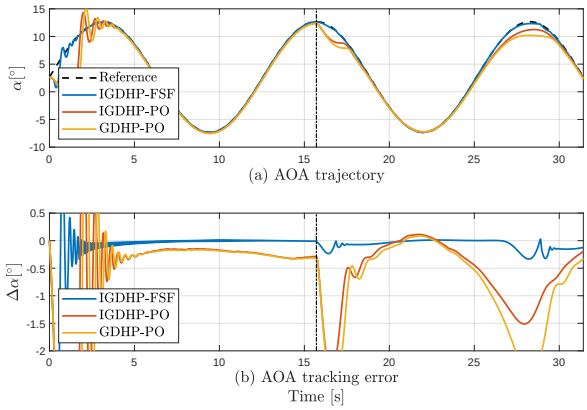


Figure 3.11: Online AOA tracking control with the left horizontal stabilizer damaged at the 5π seconds.

signal. The pitch angle of the system is selected as the controlled variable of the intermediate loop, i.e., from the outer loop to the inner loop, the controlled variables are altitude, pitch angle and AOA, respectively, as illustrated in Fig. 3.12. Other system states are regarded as the unmeasured inner states. Although flight path angle is more widely used as the intermediate variable, in this experiment, choosing pitch angle shows a better performance, which is also feasible in the real world. In practice, proportional control is often applied alone, and in this chapter, $k_h = 3$, $k_\theta = 5$, while all other parameters are kept unchanged.

Parameter variations can affect the dynamic response of the control system. Hence, different discounting factors are examined to further verify the robustness of the developed IGDHP-PO algorithm. Figures 3.13 and 3.14 demonstrate the tracking performance with different discounting factors, where the subscript 1, 2, and 3 respectively stands for the case of $\gamma = 0.99995$, $\gamma = 0.9$, and $\gamma = 0.85$. As can be seen, if successfully implemented, comparable performance can be obtained with different discounting factors. Because of the initially random policy and totally online learning, the tracking performance at

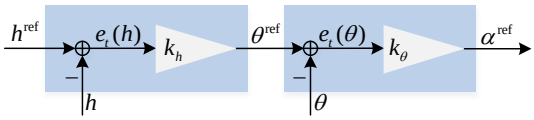


Figure 3.12: Altitude and pitch angle control loops used to generate AOA reference signal.

the beginning is also imperfect. Due to the measurement uncertainties and proportional controllers, the generated AOA reference oscillates over the altitude control task. Despite this, the developed approach manages to keep the tracking errors mostly within 1 deg and controls the system to track the designed altitude reference. Nevertheless, it is also observed that with other parameters unchanged, different discounting factors can lead to different success ratios, specifically 99.1% for $\gamma = 0.99995$, 97.3% for $\gamma = 0.9$, and 70.7% for $\gamma = 0.85$. This shows that the developed IGDHP-PO approach is robust to the forgetting factor to a certain extent.

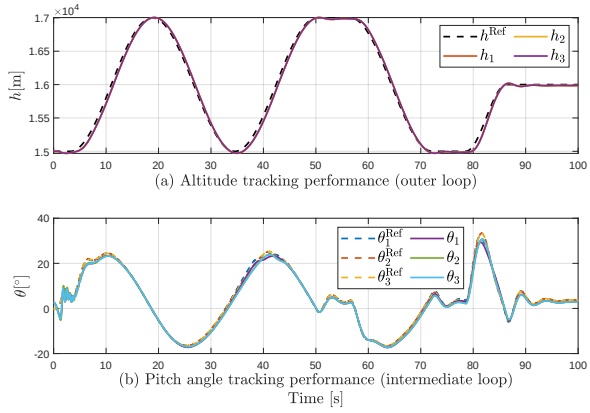


Figure 3.13: Altitude and pitch angle control loops used to generate AOA reference signal.

In addition, load disturbance is often encountered in flight control due to turbulence. Therefore, the tracking performance with the presence of sudden load disturbance is examined. Although the turbulence can act on the whole aircraft, out of the purpose of simplicity and reproducibility, the considered load disturbance is set as an equivalent square wave disturbance acting on the real elevator deflection, i.e. δ_e. As presented in Figs. 3.15 and 3.16, although sudden disturbance has an impact on tracking performance, the proposed IGHDP-PO approach can adapt with a fast-changing deflection command. The largest impact happens at the instant when the disturbance load appears for the first time. When an equivalent 5 deg deflection disturbance is encountered, the controller tries to generate an opposite action to stabilize it. Due to the overshoot, oscillations are initiated, but the AOA signal manages to track the reference. After the onset, the subsequent changing disturbance becomes less influential, which demonstrates the robustness and the online learning property of the proposed method.

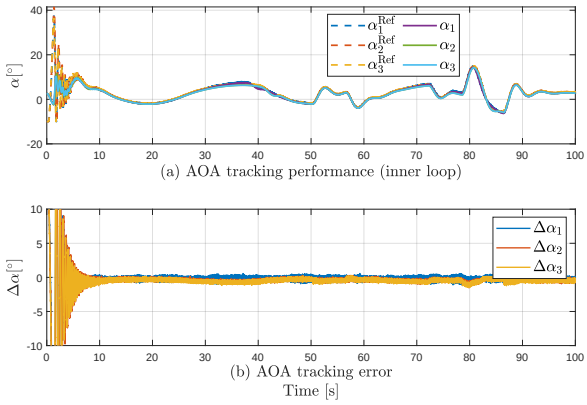


Figure 3.14: Online AOA tracking control with the reference provided by altitude tracking control task.

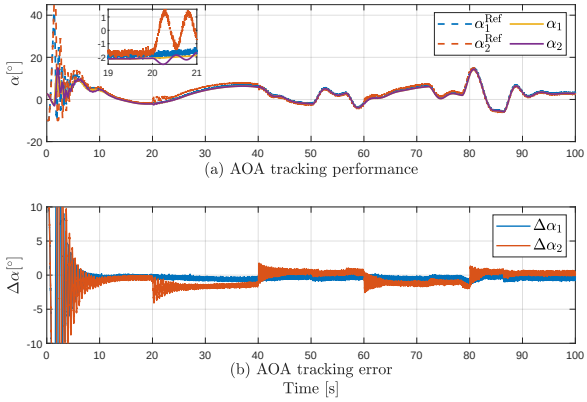


Figure 3.15: Online AOA tracking control with the presence of sudden load disturbance.

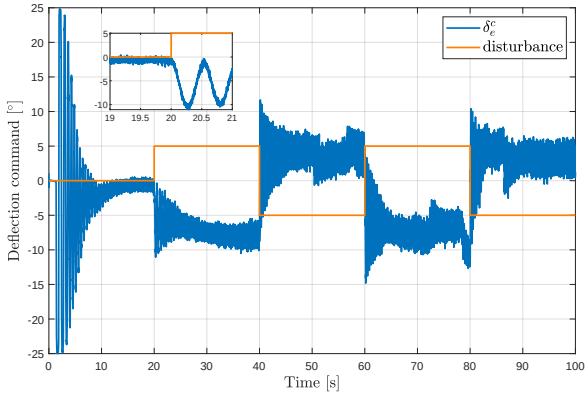


Figure 3.16: Evolution of the elevator deflection command and the sudden load disturbance.

3.6 Conclusion

this chapter develops an incremental model-based global dual heuristic programming (IGDHP) approach by combining global dual heuristic programming (GDHP) and augmented incremental techniques, which solves the partial observability problem. Moreover, the input saturation constraint is overcome by utilizing a symmetrical sigmoid function as the output layer activation function of the actor network, which frees the matrix R in the reward from having to be positive definite.

Various numerical simulation studies are conducted with an aerospace system, including attitude control problems with different initial states and sudden structural changes, and an altitude control problem combined with PID controller and hierarchy technique. The results uniformly demonstrate that the developed IGDHP algorithm can effectively deal with partial observability and surpass conventional GDHP in online stability, robustness to different initial states, and adaptability when encountering unforeseen faults. The applications to altitude control demonstrate that the developed IGDHP algorithm is robust to parameter variations and load disturbance, and has the potential to be applied to realistic complex scenarios combined with other techniques.

Future research should continue working on bridging the gap between the algorithms and real-world systems, and approaches and skills to better satisfy the persistence excitation (PE) condition are especially recommended.

4

Reinforcement Learning Control with Output Feedback and Input Constraints

Reinforcement-Learning-Based Adaptive Optimal Flight Control with Output Feedback and Input Constraints

Chapters 2 and 3 answered the research sub-question 1 in the sense of full-state feedback (FSF) and partial observability (PO). It was found in 3 that historical measurements can be used to identify system dynamics despite immeasurable inner states. This chapter will investigate the output feedback (OPFB) situation in which the system to be identified is deterministic. This chapter will build a bridge between research sub-questions 1 and 2, by taking control input constraints into consideration. A non-quadratic performance function, as well as a bounding actor network, will be introduced to handle input constraints. Finally, numerical simulations will be executed to verify the novel algorithm.

This chapter is based on the following article:
B. Sun and E. van Kampen, "Reinforcement-Learning-Based Adaptive Optimal Flight Control with Output Feedback and Input Constraints", *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 9, pp. 1685-1691, 2021. Doi: 10.2514/1.G005715 [111].

Abstract

This chapter aims at improving the present incremental model-based global dual heuristic programming algorithm proposed in the recent work [59] by taking the output-feedback situation and input constraints into consideration. Different from the common incremental model that is based on full-state feedback, an extended incremental model utilizing previous input/output data is introduced to identify locally linearized system dynamics for nonlinear systems. A non-quadratic performance function combined with a constrained-output actor network guarantees that the produced control input command satisfies actuator saturation constraints. Through numerical simulations, the effectiveness and the feasibility of the proposed method are verified.

4.1 Introduction

Reinforcement learning (RL) has become a promising tool for improving autonomy in various types of aerospace systems [36, 47, 50, 59, 112, 113] because of its self-learning property sprouting from psychological and neuroscientific perspectives on animal behaviour [16]. By interacting with the environment, RL can make a system learn optimal policies to achieve goals with limited or no priori knowledge of its dynamics or environment, and have the capability of adapting to changing situations [16, 112]. These advantages enable RL to provide a normative solution to adaptive optimal control [38].

One branch of RL is adaptive dynamic programming (ADP), which is developed from dynamic programming (DP), and it is performed in a forward-in-time way that allows for an online implementation [98]. ADP is often implemented with artificial neural networks (ANNs) and the actor-critic scheme [21], leading to adaptive critic design (ACD) [34], whose simple diagram is depicted in Fig. 4.1. ACDs not only handle the well-known "curse of dimensionality" [21], but also have a stronger generalization capability to deal with nonlinearity [27, 37, 92, 98, 114]. According to the information that the critic network approximates, ACDs can generally be categorized into three groups: heuristic dynamic programming (HDP) [39, 84], dual HDP (DHP) [34, 115], and global DHP (GDHP) [59, 63, 98]. Among them, GDHP combines the information utilized by HDP and DHP, i.e., the performance function and its derivatives. The conventional GDHP utilizes a straightforward form [63], where two kinds of outputs of the critic network share the same input and hidden layers, which brings in couplings. Furthermore, due to the approximating property, inconsistent errors exist between these outputs. Nevertheless, the structure with explicit analytical calculations proposed in [59] can overcome these limitations and therefore it is investigated in this chapter.

Although [38] claims that RL can perform in a *direct* manner with no need of identifying the system dynamics, as shown in Fig. 4.1, a third model module, next to the actor and critic networks, is often introduced to provide system transition information and thus to speed up learning and increase the success ratio [39]. By convention, ACDs rely on an ANN to approximate the global system dynamics [39, 98], which can be intractable to obtain for complex aerospace systems and can face difficulties when changing conditions are encountered [34, 59, 115]. Consequently, an incremental model (IM) is utilized in [34, 47, 50, 59, 115] to identify the locally linearized dynamics online so as to reduce the dependency on global models. As an improved version of GDHP, the IM-based GDHP

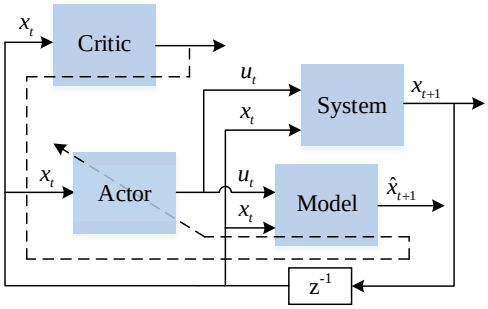


Figure 4.1: The simple diagram of ACDs, where the actor network generates the control policy that is evaluated by the critic network, and the model network is utilized to approximate system dynamics. Solid lines are the feedforward flow of signals, the dashed line denotes the updating pathway. x and u respectively denotes the system state and the control input, $\hat{x}_{t+1}$ is the estimated value of x_{t+1}, and the subscript denotes the time instant.

(IGDHP) has shown better performance in optimal tracking control problems (OTCPs) in the full-state feedback (FSF) condition [59].

Nevertheless, for real systems, sometimes not only the system dynamics, but also the measurements of some internal states are not available [84], which leads to output-feedback (OPFB) problems that cannot be tackled by the current IGDHP method. Although some ADP algorithms have been developed for OPFB [47, 50, 88], these algorithms are derived in a linear form with a quadratic performance function. These existing algorithms depend on solving a linear Riccati equation, and therefore are unable to handle complex nonlinear demands in the optimal control task, such as input constraints. However, handling input constraints is a common demand for control systems in many applications such as aerospace systems [116, 117]. A non-quadratic performance function is introduced to cope with actuator saturation constraints for optimal regulation control problems (ORCPs) [93], but cannot directly be applied to OTCPs [92]. Although an augmented system is proposed for HDP in [92, 114] to tackle this limitation, introducing the reference signal in addition to the tracking error into the actor and critic networks can slow down learning.

Motivated by overcoming the limitations existing in the current IGDHP method, this chapter first of all develops an extended incremental model to deal with OPFB problems. Then, a non-quadratic performance function as well as a bounding actor network is introduced to handle input constraints. Finally, numerical simulations are executed to verify the novel IGDHP method.

4.2 Incremental Model with Output Feedback

The situation investigated in this chapter is that the system dynamics are unknown and only input/output information can be acquired, so an incremental model is constructed to approximate the state transformation.

Consider an affine nonlinear discrete system represented by:

$$x_{t+1} = f(x_t) + g(x_t)u(x_t), \tag{4.1}$$

$$y_t = h(x_t), \tag{4.2}$$

where $x_t \in \mathbb{R}^n$, $u_t \in \mathbb{R}^m$ and $y_t \in \mathbb{R}^p$ are the system state vector, control input vector and measurable output state vector at the time instant t, respectively, $f(x_t) \in \mathbb{R}^n$, $g(x_t) \in \mathbb{R}^{n\times m}$ and $h(x_t)$ denote the drift dynamics, input dynamics and output dynamics of the system, respectively. It is assumed that $f(x_t)$, $g(x_t)$ and $h(x_t)$ are Lipschitz continuous on their domains. The nonlinear system is assumed to be both controllable and observable. For simplicity, $u(x_t)$ is represented by u_t in the rest of this paper.

According to [34, 59], if the system is full-state feedback, around time instant $t-1$, the nonlinear system (4.1) can approximately be written into the following linear equation by taking the first order Taylor series expansion and omitting second and higher-order terms:

$$\begin{aligned} x_{t+1} &\approx f(x_{t-1}) + F_{t-1}(x_t - x_{t-1}) + g(x_{t-1})u_{t-1} + G_{t-1}(u_t - u_{t-1}) \\ &= x_t + F_{t-1}(x_t - x_{t-1}) + G_{t-1}(u_t - u_{t-1}), \end{aligned} \tag{4.3}$$

where $F_{t-1} = \frac{\partial f^{\mathsf{T}}(x_{t-1})}{\partial x_{t-1}} \in \mathbb{R}^{n\times n}$ and $G_{t-1} = \frac{\partial g^{\mathsf{T}}(x_{t-1})}{\partial x_{t-1}} \in \mathbb{R}^{n\times m}$ are the state transition matrix and the input distribution matrix, respectively. F_{t-1} and G_{t-1} are bounded due to the Lipschitz continuity of $f(x_t)$ and $g(x_t)$ in Eq. (4.1).

Equation (4.3) can be rewritten as:

$$\Delta x_{t+1} \approx F_{t-1}\Delta x_t + G_{t-1}\Delta u_t. \tag{4.4}$$

Similarly, the system output equation (4.2) can also be linearized using Taylor expansion around time instant t:

$$y_{t+1} \approx y_t + H_t(x_{t+1} - x_t), \tag{4.5}$$

where $H_t = \frac{\partial h^{\mathsf{T}}(x_t)}{\partial \mathbf{x_t}} \in \mathbb{R}^{p\times n}$ denotes the observation matrix. Equation (4.5) can be rewritten as:

$$\Delta y_{t+1} \approx H_t \Delta x_{t+1}, \tag{4.6}$$

To identify and control the new incremental model presented by Eqs. (4.4) and (4.6), the following two assumptions are required.

Assumption 4.1. *The linearization does not change the property of controllability and observability of the original system given by Eqs.* (4.1) *and* (4.2)*, i.e.* (F_{t-1}, G_{t-1}) *is controllable and* (F_{t-1}, H_t) *is observable.*

Assumption 4.2. *The system is deterministic within the range of* M *steps, where* $M \geq n/p$.

Remark 4.1. Assumption 4.2 is the prerequisite that the system can be identified via input/output data. It has practical significance in that real systems are often influenced by stochastic factors such as measurement noises, unmodeled states and unforeseen disturbances, while in a small range of time horizon, the impact is small enough to be ignored. Assumption 4.2 can be satisfied when the sampling frequency is high enough.

It has been proved that for a deterministic observable system, the unmeasurable internal states (full states) can be reconstructed uniquely with adequate previous observations and control inputs [47, 50, 88]. Therefore, provided the input/output data over a sufficiently long time horizon, $[t-N+1, N]$, $n/p \leq N \leq M$, we can construct a new system called the

extended system. The extended system regards the previous increments of the input/output data as its system states. It can determine the next output increment Δy_{t+1} uniquely as follows:

$$\begin{aligned}\Delta y_{t+1} &\approx \underline{F}_t \overline{\Delta y}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N} \\ &= \underline{F}_{11,t}\Delta y_t + \underline{F}_{12,t}\overline{\Delta y}_{t-1,N-1} + \underline{G}_{11,t}\Delta u_t + \underline{G}_{12,t}\overline{\Delta u}_{t-1,N-1},\end{aligned} \tag{4.7}$$

where $\underline{F}_t \in \mathbb{R}^{p\times Np}$ and $\underline{G}_t \in \mathbb{R}^{p\times Nm}$ are the transition matrix and input distribution matrix of the extended discrete system, respectively, $\overline{\Delta u}_{t,N} = [\Delta u_t^\mathsf{T}, \Delta u_{t-1}^\mathsf{T}, \cdots, \Delta u_{t-N+1}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{Nm}$ and $\overline{\Delta y}_{t,N} = [\Delta y_t^\mathsf{T}, \Delta y_{t-1}^\mathsf{T}, \cdots, \Delta y_{t-N+1}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{Np}$ are the measured input/output data from N previous steps, respectively, $\underline{F}_{11,t} \in \mathbb{R}^{p\times p}$ and $\underline{F}_{12,t} \in \mathbb{R}^{p\times(N-1)p}$ are partitioned matrices from $\underline{F}_t$, and $\underline{G}_{11,t} \in \mathbb{R}^{p\times m}$ and $\underline{G}_{12,t} \in \mathbb{R}^{p\times(N-1)m}$ are partitioned matrices from $\underline{G}_t$. Assume that the generalised inverse of $\underline{G}_{11,t}$ exists such that $\underline{G}_{11,t}^{-1}\underline{G}_{11,t} = I_m \in \mathbb{R}^{m\times m}$, where I_m denotes the identity matrix and the subscript m gives the dimensionality.

In this way, the original nonlinear system is approximated by a locally linear incremental model and a direct mapping from the control input at the time instant t to the output at the time instant $t+1$ is built. Then, a recursive least squares (RLS) approach using a sliding window technique [84] is adopted to identify the matrices $\underline{F}_t$ and $\underline{G}_t$ online. Rewrite Eq. (4.7) in a vector form as:

$$\Delta y_{t+1} \approx \begin{bmatrix}\overline{\Delta y}_{t,N}^\mathsf{T} & \overline{\Delta u}_{t,N}^\mathsf{T}\end{bmatrix}\begin{bmatrix}\underline{F}_t^\mathsf{T} \\ \underline{G}_t^\mathsf{T}\end{bmatrix}. \tag{4.8}$$

Define $\overline{Y}_t = \left[\overline{\Delta y}_{t,N}^\mathsf{T}, \overline{\Delta u}_{t,N}^\mathsf{T}\right]^\mathsf{T} \in \mathbb{R}^{N(p+m)\times 1}$, which is the input information of the extended incremental model identification, and $\underline{\Theta}_t = [\underline{F}_t, \underline{G}_t]^\mathsf{T} \in \mathbb{R}^{N(p+m)\times p}$, which is the extended matrix to be determined using the RLS algorithm. Therefore, the output prediction equation can presented as follows:

$$\Delta\hat{y}_{t+1} = \overline{Y}_t^\mathsf{T} \cdot \hat{\underline{\Theta}}_t, \tag{4.9}$$

where $\hat{\underline{\Theta}}_t = \left[\hat{\underline{F}}_t, \hat{\underline{G}}_t\right]^\mathsf{T}$ is the approximated value of $\underline{\Theta}_t$, and the symbol $\hat{\cdot}$ denotes the approximated/estimated value.

The sliding window is utilized to store historical data $\overline{Y}_t$ for determining $\hat{\underline{\Theta}}_t$ in each time step, and the main procedure of the RLS approach is given as follows [84]:

$$\epsilon_t = \Delta y_{t+1}^\mathsf{T} - \Delta\hat{y}_{t+1}^\mathsf{T}, \tag{4.10}$$

$$\hat{\underline{\Theta}}_t = \hat{\underline{\Theta}}_{t-1} + \frac{\underline{\mathrm{Cov}}_{t-1}\overline{Y}_t}{\gamma_{\mathrm{RLS}} + \overline{Y}_t^\mathsf{T}\underline{\mathrm{Cov}}_{t-1}\overline{Y}_t}\epsilon_t, \tag{4.11}$$

$$\underline{\mathrm{Cov}}_t = \frac{1}{\gamma_{\mathrm{RLS}}}\left(\underline{\mathrm{Cov}}_{t-1} - \frac{\underline{\mathrm{Cov}}_{t-1}x_t x_t^\mathsf{T}\underline{\mathrm{Cov}}_{t-1}}{\gamma_{\mathrm{RLS}} + x_t^\mathsf{T}\underline{\mathrm{Cov}}_{t-1}x_t}\right), \tag{4.12}$$

where $\epsilon_t \in \mathbb{R}^p$ denotes the prediction error, $\underline{\mathrm{Cov}}_t \in \mathbb{R}^{(p+m)N\times(p+m)N}$ is the estimation covariance matrix , and γ_{RLS} denotes the forgetting factor. As to initialization settings of the RLS approach, we set $\hat{\underline{F}}_0 = [I_p, 0]$, and $\hat{\underline{G}}_0$ as a zero matrix. The covariance matrix is initialized as an identity matrix multiplied by a large positive value [50] and we choose 10^7 in this chapter, i.e., $\underline{\mathrm{Cov}}_0 = 10^7 I_{(p+m)N}$.

4.3 Optimal Tracking Control Problem (OTCP)

This section deals with the input constraints in the OTCP by designing a non-quadratic function. The OTCP aims to find the optimal control policy u_t^* such that the system described by (4.1) and (4.2) can track the reference trajectory $y_t^{\text{ref}} \in \mathbb{R}^p$ in an optimal manner by minimizing a predefined performance function. Furthermore, the control input must be constrained by a bound vector u_{b}, i.e. $|u_{i,t}| \leq u_{\text{b}i}$ for $\forall u_{\text{b}i} > 0$, $i = 1, \cdots, m$, where $u_{i,t}$ and u_{bi} denotes the elements of u_t and u_{b}, respectively.

To simplify the derivation and implementation of the algorithm, the reference trajectory y_t^{ref} is supposed to satisfy the following assumption [50]:

Assumption 4.3. *The reference signal is slow-varying in comparison to the system dynamics, such that the increment of the reference signal between two-time instants can be ignored.*

4

Accordingly, considering Eq. (4.7), the output tracking error at the time instant $t+1$ can be presented as:

$$\begin{aligned} e_{t+1} &= y_{t+1} - y_{t+1}^{\text{ref}} \\ &\approx y_t + \underline{F}_t \overline{\Delta y}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N} - (y_t^{\text{ref}} + \Delta y_{t+1}^{\text{ref}}) \\ &\approx e_t + \underline{F}_t \overline{\Delta y}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N} \\ &\approx e_t + \underline{F}_t \overline{\Delta e}_{t,N} + \underline{G}_t \overline{\Delta u}_{t,N}. \end{aligned} \tag{4.13}$$

Based on Assumption 4.3 and Eq. (4.13), the effect caused by the dynamics of the reference signal is approximately shielded between two sampling instants. Therefore, the original OTCP is transformed into an ORCP, so that the non-quadratic performance function used in [37, 92, 93, 114] can be adopted to generate constrained control input.

Then the following performance function is introduced for this new input-constrained ORCP:

$$J(e_t, u_t) = \sum_{l=t}^{\infty} \gamma^{l-t} c(e_l, u_l), \tag{4.14}$$

where γ is the discount factor with $0 < \gamma \leq 1$ and $c(e_l, u_l)$ is the one-step cost function that is defined as:

$$c(e_t, u_t) = e_t^{\mathsf{T}} Q e_t + Y(u_t), \tag{4.15}$$

where $Q \in \mathbb{R}^{p \times p}$ is positive semi-definite and is set to be a diagonal matrix in this chapter, and $Y(u_t)$ is a positive-definite integral function defined as:

$$Y(u_t) = 2 \sum_{i=1}^{m} \int_0^{u_{i,t}} u_{bi} \psi^{-1}(v_i / u_{bi}) R_i \, dv_i, \tag{4.16}$$

where $\psi(\cdot)$ is a bounded element-wise function satisfying $|\psi(\cdot)| \leq 1$, and is a monotonic odd function with its derivative bounded by a constant ψ_M, i.e. $\|d\psi(s)/ds\| \leq \psi_M$, $\forall s \in \mathbb{R}$, and R_i denotes the element of the positive definite weight matrix $R = \text{diag}([R_1, \cdots, R_m]) \in \mathbb{R}^{m \times m}$, in which diag$(\cdot)$ reshapes the vector to a diagonal matrix. Without loss of generality, the well-known hyperbolic tangent function $\psi(\cdot) = \tanh(\cdot)$ is chosen as the bounding function. Note that $Y(u_t)$ is positive definite since $\psi^{-1}(\cdot)$ is a monotonic odd function and R is positive definite. For simplicity, $J(e_t, u_t)$ is denoted by J_t and $c(e_t, u_t)$ is denoted by c_t hereafter.

According to Bellman's principle of optimality, the optimal performance function J_t^* is time-invariant and satisfies the discrete-time Hamilton-Jacobi-Bellman (DTHJB) equation:

$$J_t^* = \min_{u_t}(c_t + \gamma J_{t+1}^*). \tag{4.17}$$

Differentiate the right-hand side of Eq. (4.17) along the control input u_t and the following equation should be satisfied for the optimal control u_t^* [27]:

$$\frac{\partial \mathbf{c}_t}{\partial u_t} + \gamma \frac{\partial e_{t+1}^{\mathrm{T}}}{\partial u_t} \lambda_{t+1}^* = 0. \tag{4.18}$$

where $\lambda_{t+1}^* = \frac{\partial J_{t+1}^*}{\partial e_{t+1}}$. Then substituting Eqs. (4.15) and (4.16) in Eq. (4.18) yields [37, 92, 93]:

$$u_{i,t}^* = -u_{bi} \tanh(D_{i,t}^*), i = 1, \cdots, m, \tag{4.19}$$

and $D_{i,t}^*$ is given as:

$$D_{i,t}^* = \frac{\gamma}{2u_{bi}} R_i^{-1} \underline{g}_{i,11,t}^{\mathrm{T}} \lambda_{t+1}^*, \tag{4.20}$$

where $\underline{g}_{i,11,t}$ is the ith column vector of $\underline{G}_{11,t}$. The control input $u_{i,t}^*$ is bounded within its permitted range $[-u_{bi}, u_{bi}]$, $i = 1, \cdots, m$. The nonquadratic cost (4.16) for u_t^* is:

$$Y(u_t^*) = \sum_{i=1}^{m} [u_{bi} \gamma \lambda_{t+1}^{*\mathrm{T}} \underline{g}_{i,11,t} \tanh(D_{i,t}^*) + u_{bi}^2 R_i \ln(1 - \tanh^2(D_{i,t}^*))]. \tag{4.21}$$

By substituting Eq. (4.21) into Eq. (4.17), the DTHJB equation becomes:

$$J_t^* = e_t^{\mathrm{T}} Q e_t + \sum_{i=1}^{m} [u_{bi} \gamma \lambda_{t+1}^{*\mathrm{T}} \underline{g}_{i,11,t} \tanh(D_{i,t}^*) + u_{bi}^2 R_i \ln(1 - \tanh^2(D_{i,t}^*))] + \gamma J_{t+1}^*. \tag{4.22}$$

In this way, the original OTCP is recast as a new ORCP subject to input constraints. The DTHJB equation (4.22) cannot be solved analytically in the generally nonlinear cases, and therefore the IGDHP algorithm is introduced to iteratively solve the OTCP in the next section.

4.4 IGDHP Implementation

The IGDHP algorithm is introduced in this section with the IM and ANNs facilitating the implementation: the IM reflects the local dynamics of the nonlinear plant and the ANNs are utilized to build the critic network and the actor network. The architecture of the IGDHP algorithm is shown in Fig. 4.2. The reference signal at the time instant $t+1$, y_{t+1}^{ref}, is unavailable at the time instant t. To obviate the need for this information, the actor and critic networks are updated with the information from the time instants t and $t-1$ [59, 114].

4.4.1 The critic network

The IGDHP technique makes use of both the approximation of performance function $\hat{J}_t$ and its derivative with respect to the network input e_t, which is denoted by $\hat{\lambda}_t$. As shown

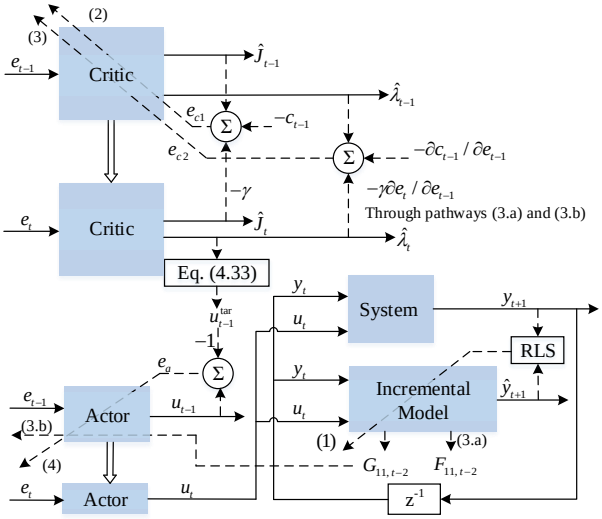


Figure 4.2: The architecture of the IGDHP algorithm, where solid lines represent the feedforward flow of signals, dashed lines are backpropagation pathways, and the thick arrows represent the weight transmission.

in Fig. 4.3, the critic network is utilized to approximate the performance function (4.14) with the facilitation of an ANN, which employs a feedforward structure with single hidden layer as follows:

$$\hat{J}_t = w_{c2,t}^{\mathsf{T}} \sigma(w_{c1,t}^{\mathsf{T}} e_t), \tag{4.23}$$

where $w_{c1,t}$ and $w_{c2,t}$ are weight matrices between different layers and the activation function σ is chosen to be a sigmoid function. By taking the explicit analytical calculations [59], $\hat{\lambda}_t$ is given as:

$$\hat{\lambda}_t = \frac{\partial \hat{J}_t}{\partial e_t} = w_{c1,t}(w_{c2,t} \odot \sigma'(w_{c1,t}^{\mathsf{T}} e_t)), \tag{4.24}$$

where $\odot$ is the Hadamard product, a.k.a. the element-wise product, and $\sigma'(\cdot)$ is the first order derivative of $\sigma(\cdot)$.

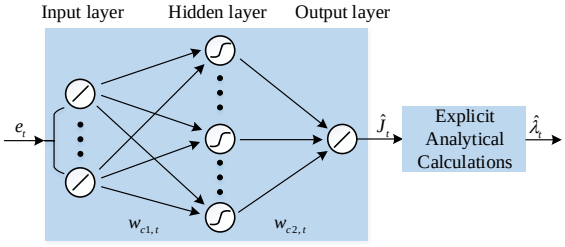


Figure 4.3: The architecture of the critic network, in which an ANN is utilized to approximate performance function and then explicit analytical calculations are taken to compute first-order derivatives.

Referring to the DTHJB equation (4.17), the approximation error of the performance function produced by the critic network is given as:

$$e_{c1,t} = \hat{J}_{t-1} - c_{t-1} - \gamma \hat{J}_t, \tag{4.25}$$

and the approximation error of the derivative is given as:

$$\mathrm{e}_{c2,t} = \frac{\partial(\hat{J}_{t-1} - c_{t-1} - \gamma \hat{J}_t)}{\partial e_{t-1}} = \hat{\lambda}_{t-1} - \frac{\partial c_{t-1}}{\partial e_{t-1}} - \gamma \frac{\partial e_t}{\partial e_{t-1}} \hat{\lambda}_t. \tag{4.26}$$

The second item on the right-hand side of Eq. (4.26) has an explicit calculation as follows:

$$\frac{\partial c_{t-1}}{\partial e_{t-1}} = 2Qe_{t-1} + 2\frac{\partial u_{t-1}}{\partial e_{t-1}}[R(\tanh^{-1}(u_{t-1} \odot u_{\mathrm{b}}^{\circ -1}) \odot u_{\mathrm{b}})], \tag{4.27}$$

where $\frac{\partial u_{t-1}}{\partial e_{t-1}}$ is derived by the actor network in the next subsection, and $u_{\mathrm{b}}^{\circ -1}$ stands for the element-wise inverse of the vector u_{b}. $\frac{\partial e_t}{\partial e_{t-1}}$ in the last item in Eq. (4.26) is composed of two parts [34, 59, 98]: one part is directly derived from the incremental model (pathway 3.a), whereas the other part starts from the incremental model and uses the control input u_{t-1} as the intermediate auxiliary (pathway 3.b):

$$\frac{\partial \hat{e}_t}{\partial e_{t-1}} = \underbrace{I_p + F_{11,t-2}^{\mathsf{T}}}_{\text{pathway (3.}a\text{)}} + \underbrace{\frac{\partial u_{t-1}}{\partial e_{t-1}} G_{11,t-2}^{\mathsf{T}}}_{\text{pathway (3.}b\text{)}}, \tag{4.28}$$

where $\underline{F}_{11,t-2} \in \mathbb{R}^{p \times p}$ is the upper-left partitioned matrix from $\underline{F}_{t-2}$ and $\underline{G}_{11,t-2} \in \mathbb{R}^{p \times m}$ is the upper partitioned matrix from $\underline{G}_{t-2}$.

Accordingly, the overall error of the critic network combines two kinds of approximation error as:

$$E_{c,t} = \beta \frac{1}{2} e_{c1,t}^2 + (1-\beta)\frac{1}{2} \mathrm{e}_{c2,t}^{\mathsf{T}} \mathrm{e}_{c2,t}, \tag{4.29}$$

where β is a scalar within a range of $[0, 1]$.

The weights of the critic network are updated by a gradient-descent algorithm with a learning rate η_c to minimize the overall error $E_{c,t}$:

$$w_{c,t+1} = w_{c,t} - \eta_c \frac{\partial E_{c,t}}{\partial w_{c,t}}, \tag{4.30}$$

and

$$\frac{\partial E_{c,t}}{\partial w_{c,t}} = \frac{\partial \hat{J}_t}{\partial w_{c,t}} \cdot \frac{\partial E_{c,t}}{\partial \hat{J}_t} + \frac{\partial \hat{\lambda}_t}{\partial w_{c,t}} \cdot \frac{\partial E_{c,t}}{\partial \hat{\lambda}_t} = \beta \frac{\partial \hat{J}_t}{\partial w_{c,t}} e_{c1,t} + (1-\beta)\frac{\partial \hat{\lambda}_t}{\partial w_{c,t}} \mathrm{e}_{c2,t}, \tag{4.31}$$

where $\partial \hat{\lambda}_{c,t} / \partial w_{c,t}$ is the second-order mixed gradient of $\hat{J}_t$, and the detailed explicit calculations can be found in [59].

4.4.2 The actor network

The pathway 3.b needs to compute $\frac{\partial u_{t-1}}{\partial e_{t-1}}$, which cannot be calculated exactly. Consequently, an actor network is introduced to produce the control input u_t and to facilitate backpropagation.

In this chapter, the output layer of the actor network employs a bounded element-wise function as Eq. (4.16) to be the activation function, and is multiplied by the bound vector u_b, so that the system control u_t output of the actor network is bounded within the constraints, as shown in Fig. 4.4. The actor network is also constructed as a single-hidden-layer feedforward ANN:

$$u_t = u_b \odot \psi(w_{a2,t}^{\mathsf{T}} \sigma(w_{a1,t}^{\mathsf{T}} e_t)), \tag{4.32}$$

where $w_{a1,t}$ and $w_{a2,t}$ are weight matrices between different layers of the actor network.

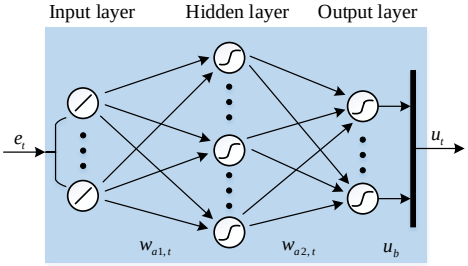


Figure 4.4: The architecture of the actor network, in which the output is constrained by the bounded activation function and the bound vector.

It is noted that the control input u_t outputted by the actor network is directly introduced to the IM and the real system, and that the actor network performs as a global approximation, so $\frac{\partial u_{t-1}}{\partial e_{t-1}} = \frac{\partial u_t}{\partial e_t}$ given the same actor weights. The actor network is supposed to approximate a target control input u_{t-1}^{tar} which is obtained by substituting $\hat{\lambda}_t$ into Eq. (4.20) and then Eq. (4.19) as follows:

$$u_{t-1}^{\text{tar}} = -u_{\text{b}} \odot \tanh(\hat{D}_{t-1}), \tag{4.33}$$

and

$$\hat{D}_{t-1} = \frac{\gamma}{2} u_{\text{b}}^{\circ -1} \odot (R^{-1} G_{11,t-2}^{\mathsf{T}} \hat{\lambda}_t). \tag{4.34}$$

It can be seen that both the target control input u_{t-1}^{tar} and the real control input u_t are bounded by u_{b}. Therefore, the actor network is aiming at minimizing the following error:

$$E_{a,t} = \frac{1}{2} e_{a,t}^{\mathsf{T}} e_{a,t}, \tag{4.35}$$

where

$$e_{a,t} = u_{t-1} - u_{t-1}^{\text{tar}}. \tag{4.36}$$

As illustrated in Fig. 4.2, the actor weights are updated along the 4th pathway with a learning rate η_a:

$$w_{a,t+1} = w_{a,t} - \eta_a \frac{\partial E_{a,t}}{\partial w_{a,t}} = w_{a,t} - \eta_a \frac{\partial u_t}{\partial w_{a,t}} e_{a,t} \tag{4.37}$$

4.5 Flight Control Simulation

In order to assess the performance of the developed novel IGDHP algorithm, the longitudinal dynamics of a nonlinear aircraft [59, 78, 118] are taken into account. The initial altitude and speed of the aircraft are set to be 15000 ft and 600 ft/s, respectively, based on which, the aerodynamic model is trimmed. To simply and clearly compare different methods, only short-period control is considered and the elevator deflection command is artificially bounded within $[-5\,\text{deg}, 5\,\text{deg}]$. The control system, discretized with a sampling frequency of 100 Hz, targets for controlling the angle of attack (AOA) of the aircraft to track a sinusoidal signal, namely $\alpha^{\text{ref}} = 10\sin(0.5t)$ deg.

Zero-mean white noises with a standard deviation of 4×10^{-2} deg and 1.8×10^{-3} deg are added onto the bounded elevator deflection command and measured AOA, respectively. A 3211 disturbance signal is employed to kick off the learning at the beginning so as to better satisfy the persistent excitation (PE) condition [59, 63]. Three methods are utilized for comparison, namely GDHP with input constraints, IGDHP with input constraints, and IGDHP without input constraints. All methods are implemented in the OPFB condition with the sliding window width of 3. GDHP employs a model network with previous states and control inputs as its inputs to approximate system dynamics, whereas IGDHP approaches utilize an incremental model. All ANNs adopt the fully connected feed-forward architecture with a single hidden layer. The number of hidden layer neurons is 10 for the actor and critic networks and 20 for the model network. All weights are initialized randomly within $[-0.1, 0.1]$ to decrease the impact of initialization. For IGDHP without input constraints, the performance function is set to a quadratic form and the output layer of the actor network employs a unit linear activation function. To compare the robustness of these approaches, Monte Carlo simulations are also conducted, and the randomness is introduced by aforementioned noises and initial weights. A concept of success ratio [34, 39, 59] is introduced to indicate the performance. A successful implementation in this chapter is defined by the tracking errors remaining within $[-4\,\text{deg}, 4\,\text{deg}]$ after the first 20 seconds. The simulations are carried out on an Intel Core i7-8550U @ 1.80 GHz processor, and 8 GB RAM.

The comparison of AOA trajectories and tracking error is illustrated in Fig. 4.5. It can be seen that although all methods can follow the reference after short online learning stage, GDHP with input constraints takes more time to get satisfying performance and the tracking error is the largest overall. The reason causing this phenomenon lies in that GDHP utilizes a model network to identify the global system model which requires more data to update weights. Without well-approximated dynamics, the control policy cannot be appropriately generated, which in turn can have impacts on the identification of the global model, i.e. Assumption 4.2 is not satisfied. A detailed view can be found in Fig. 4.6, which shows the weights between the input layer and hidden layer of the actor network w_{a1}. The update of w_{a1} requires information from the critic network, system dynamics and actor network, and therefore the trajectory of w_{a1} can be used to indicate the overall learning performance. It is clearly shown in Fig. 4.6 that the learning of GDHP is slower in comparison to the IGDHP methods, which results in a more conservative policy at the beginning stage, as presented in Fig. 4.7. Furthermore, due to the inaccurate information regarding system state transition, GDHP has the lowest success ratio for 1000 Monte Carlo simulations, which is merely 46.4%, compared with 99.4% and 98.0% for IGDHP with and

without input constraints, respectively.

As to IGDHP methods, after the weights have converged, both methods have a similar tracking performance, which is better than that of GDHP. Nevertheless, the developed IGDHP with input constraints has a slightly higher success ratio, and the benefit is brought by the collective effect of the non-quadratic performance function and the bounded actor network. With these measures, input constraints can be overcome. As shown in Fig. 4.6, it is clear that the weight update of IGDHP without input constraints can be more radical at the beginning when the policy has not converged yet. During this exploration stage, IGDHP without input constraints performs similar to "bang-bang" control, with larger control command that easily causes overshoot and oscillation. To further investigate the influence of the measures to deal with input constraints, the policies directly produced by the actor network are compared between the IGDHP methods with and without input constraints. As presented in Fig. 4.8, given the random data of AOA and its reference within the range of $[-2\,\text{deg}, 2\,\text{deg}]$, the methods can plot a mesh surface to illustrate their learned policy at 8s. Compared to IGDHP with input constraints thus having a smooth surface, IGDHP without input constraints has a sharper surface and tends to produce a large control command. The learned policy of GDHP with input constraints is similar to that of IGDHP with input constraints, and therefore its plot is omitted.

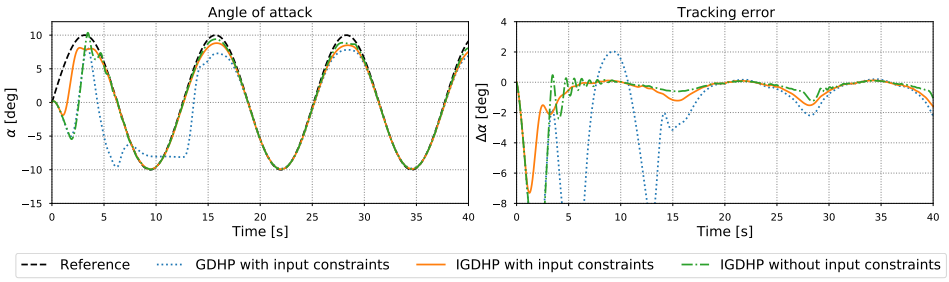


Figure 4.5: The tracking performance of the online AOA tracking control task. Three methods are compared and IGDHP with input constraints is the contribution of this chapter.

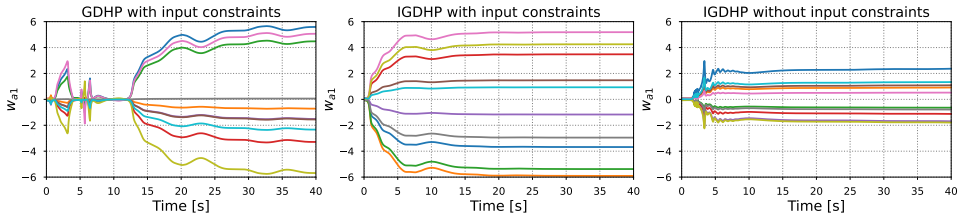


Figure 4.6: The weights between the input layer and hidden layer of the actor network, w_{a1}. The plots of three approaches are presented, and the middle one refers to the proposed approach which is the contribution of this chapter.

To further verify the robustness of the designed control approach when tracking fast-varying reference signals, a simulation experiment in which the frequency of the

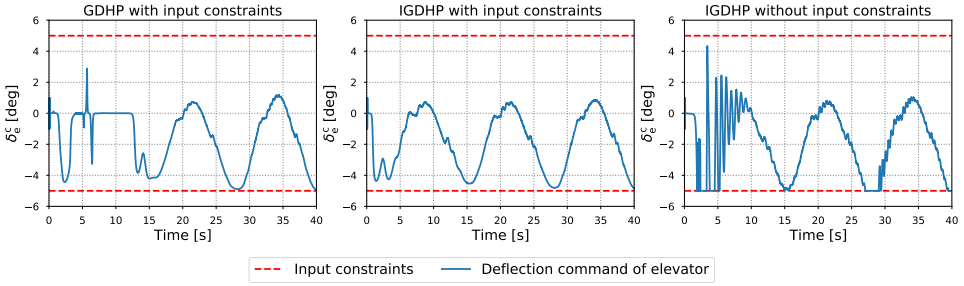


Figure 4.7: The deflection command of the elevator. The plots of three approaches are presented, and the middle one refers to the proposed approach which is the contribution of this chapter.

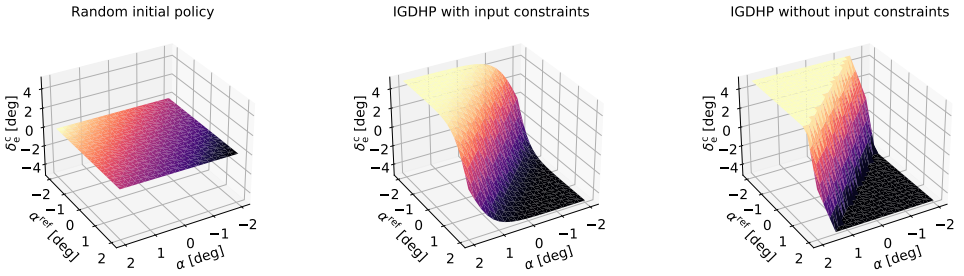


Figure 4.8: The change of the policy directly produced by the actor network from the initial stage to 8π s. The first subfigure is the random initial policy, and the middle subfigure refers to the proposed approach which is the contribution of this chapter.

reference signal keeps increasing is performed. The initial reference frequency is the same as the one in the above simulation experiments, and the control input is bounded within $[-25\,\text{deg}, 25\,\text{deg}]$. As illustrated in Fig. 4.9, after the initial exploration stage, the controlled AOA can track the given reference signal when it is slow-varying, and the tracking error is growing as the reference frequency is increasing. Specifically, when the reference frequency is around 5.3 times of the initial value, the tracking error for the first time exceeds 2 deg, and when the reference frequency is near 7.0 times of the initial value, the tracking error starts exceeding 4 deg. The results clarify the significance of Assumption 4.3 to a certain extent. Besides, it is noted that at the final stage of the simulation, the control input comes close to the input constraints but does not exceed the bound. Due to the existence of input constraints, when the reference frequency is too high, the aircraft cannot successfully complete the tracking task and the tracking error is large. Nevertheless, the simulation results demonstrate the developed IGDHP with input constraints is robust to the reference signal within a range of frequencies.

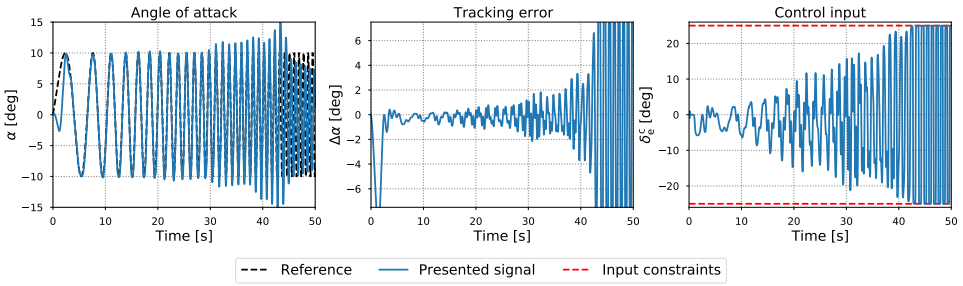


Figure 4.9: The tracking performance of the developed IGDHP with input constraints when tracking a reference signal with its frequency keeping increasing.

4.6 Conclusions

This chapter improves the current incremental model-based global dual heuristic programming (IGDHP) method for more complex application scenarios, including dealing with output feedback (OPFB) and input constraints. Different from the GDHP method that utilizes a neural network to identify the global system dynamics, the developed novel IGDHP method exploits an extended incremental model to approximate locally linear system dynamics via the previous input/output data at several previous time instants. The numerical simulation shows that the developed novel IGDHP method outperforms GDHP in the convergent speed of parameters, tracking precision, and success ratio. Moreover, the input saturation constraint is overcome by combining a non-quadratic performance function and bound activation function in the output layer of the actor network. The original IGDHP method employs a quadratic performance function and unit linear activation function, and compared to it, the developed novel IGDHP method has a smoother policy surface and slightly higher success ratio. In addition, through a simulation experiment, the robustness of the developed IGDHP with input constraints is verified for reference signals with a range of frequencies. The simulation results collectively demonstrate the effectiveness and the feasibility of the proposed method. Further research on better satisfying the persistent excitation (PE) condition so as to achieve a non-failure control is recommended.

5

Event-Triggered Constrained Control Using XGDHP for Discrete-Time Systems

Event-Triggered Constrained Control Using Explainable Global Dual Heuristic Programming for Nonlinear Discrete-Time Systems

In previous chapters, a novel global dual heuristic programming (GDHP) algorithm is developed with explicit analytical calculations to synthesize and improve current online adaptive critic designs (ACDs). This chapter will simplify these calculations by eliminating matrix dimensionality transformations, and propose the concept of explainable GDHP (XGDHP). In Chapter 4, symmetric input constraints were addressed by the combination of the non-quadratic cost function and the bounding layer of the actor network. This chapter will for the time overcome the asymmetric control input constraints for zero-equilibrium-point stabilization problems by involving a novel segmented utility function. Furthermore, to answer research sub-question 3, an event-triggered control (ETC) scheme will be introduced. The triggering condition will be derived and proved to be able to guarantee the input-to-state stability and asymptotic stability of the ETC system. The mechanism of multiple iterations at one time instant will be introduced to enhance the system stability using XGDHP. Finally, numerical simulations on two general nonlinear systems will be conducted to demonstrate the comparable performance of event-triggered XGDHP with the time-based approach.

This chapter is based on the following article:
B. Sun and E. van Kampen, "Event-Triggered Constrained Control Using Explainable Global Dual Heuristic Programming for Nonlinear Discrete-Time Systems", *Neurocomputing*, vol. 468, pp. 452-463, 2022. Doi: 10.1016/j.neucom.2021.10.046 [119].

Abstract

This chapter develops an event-triggered optimal control method that can deal with asymmetric input constraints for nonlinear discrete-time systems. The implementation is based on an explainable global dual heuristic programming (XGDHP) technique. Different from traditional GDHP, the required derivatives of the cost function in the proposed method are computed by explicit analytical calculations, which makes XGDHP more explainable. Besides, the challenge caused by the input constraints is overcome by the combination of a piece-wise utility function and a bounding layer of the actor network. Furthermore, an event-triggered mechanism is introduced to decrease the amount of computation, and the stability analysis is provided with fewer assumptions compared to most existing studies that investigate event-triggered discrete-time control using adaptive dynamic programming. Two simulation studies are carried out to demonstrate the applicability of the constructed approach. The results present that the developed event-triggered XGDHP algorithm can substantially reduce the computational load, while maintaining comparable performance with the time-based approach.

5

5.1 Introduction

Optimality is one of the most significant properties of a control system. The optimal control problem can be solved using the Hamilton-Jacobi-Bellman (HJB) equation. However, until now, there is no effective way to analytically solve the HJB equation for nonlinear systems [120, 121]. Nevertheless, adaptive dynamic programming (ADP) offers a promising tool to attain satisfying numerical solutions by incorporating artificial neural networks (ANNs), which has been applied to a wide range of nonlinear industrial applications [50, 59, 122–124]. As a branch of reinforcement learning (RL), ADP approximately addresses optimal control problems by iterations between policy improvement and policy evaluation [16, 125]. When dealing with the discrete-time (DT) optimal control problem, obtaining the current control policy usually relies on the control performance at the next time step [16, 50]. This bootstrapping property [16] can be addressed by the actor-critic scheme using two separate ANNs that respectively improves and evaluates the policy.

When multiple ANNs are involved, ADP is often called adaptive critic design (ACD) [50]. Based on the information utilized by the critic network, ACDs can be categorized into heuristic dynamic programming (HDP), dual HDP (DHP), and global DHP (GDHP) [50, 59]. GDHP combines the information of cost function and its derivatives, and recently has attained much attention [59, 69, 80, 99]. The most common architecture of GDHP is the straightforward form that approximates the cost function and its derivatives simultaneously [69, 99]. However, as claimed in [59], in this structure two kinds of outputs of the critic network share the same input and hidden layers, making them strongly coupled. Without analytical calculations, the approximated cost function and its derivatives can suffer from inconsistent errors. With the development of artificial intelligence (AI), there is an emerging need for understanding how strategies are made by AI methods, which arouses explainable AI (XAI) [126]. Following this idea, [59] introduces explicit analytical calculations to the GDHP technique, which makes it more explainable to designers because the approximate cost derivatives are explicitly computed from the approximate cost function. This explainable GDHP (XGDHP) algorithm has shown its applicability in aerospace control systems

[59, 80, 111]. However, matrix dimensionality transformations, a.k.a. tensor operations, are involved in these studies, making them complicated to implement.

Besides, in practical applications, due to physical limitations or safety considerations, handling input constraints is a common demand for control systems [127]. A classic approach is to design a non-quadratic cost function, such that the control inputs obtained by solving the HJB equation are limited by a symmetric bounded function [91]. However, although there are many researches aiming at dealing with symmetric input constraints in nonlinear optimal control problems [80, 91, 127–129], little attention has been paid to the situation subject to asymmetric input constraints. Motivated by the industrial need, Yang *et al* [130] managed to cope with the asymmetric input constraints by adjusting the cost function with the mean and range of the control input constraints. However, there are two limitations in their proposed method: 1) when the system states go to zero, the control inputs are still non-zero values, specifically, the mean values of the constraint range; 2) when the control inputs go to zero, the cost caused by the control inputs is not zero. Consequently, this approach is not applicable to the stabilization problem with an origin equilibrium point, which inspires our study.

Furthermore, in order to maintain the system stability, a significant number of iterations within a sampling interval are normally required for ACDs, which result in a high computational cost [131]. To enhance resource utilization and reduce the computational burden, event-triggered control (ETC) has evolved as an alternate control paradigm and acquired more attention in recent days [121, 131]. ETC is originally proposed in the networked system to deal with the limitation of communication bandwidth [56, 132–134]. These researches target solving communication issues such as synchronization, time delays and disturbances, rather than pursuing optimality or tackling control input constraints. A cross-fertilization of ETC and ACD leads to the event-triggered ACD that targets solving optimal control problems in an event-driven manner [121, 125]. Most event-triggered studies focus on continuous-time systems [135–137] and only a few articles discuss the DT system. The HDP algorithm is combined with the ETC in [131, 138–140] and [121, 129] describe the event-triggered DHP algorithm. Although [69] applies the event-triggered GDHP algorithm to a network control scenario, till now there is no related research on event-triggered XGDHP. Among them, only [129] attempts to deal with symmetric input constraints merely using the non-quadratic cost function, which however is not rigorous because the control input is directly generated by the actor network that is not bounded. Furthermore, the essence of the ETC scheme lies in that a task is executed only if a predefined triggered condition is satisfied. Therefore, defining a sound triggering condition is always the primary task for the ETC scheme. For the nonaffine system, the same triggering condition is employed in [69, 121, 129, 138] and among them [121, 129, 138] provide the stability analysis regarding the triggering condition. However, in [121] an extra assumption that the state norm is bounded by the supremum of control input norm is required, whereas in [129, 138] the input-to-state stability (ISS) Lyapunov function is directly assumed to exist without pointing out its specific form and additional hyperparameters are involved in [129]. These limitations prevent the proposed triggering condition from wider applications.

Motivated to tackle the limitations existing in the literature, we conduct this research by concentrating on the event-triggered XGDHP algorithm subject to asymmetric input constraints. The contributions are summarized as follows:

1. XGDHP is developed to solve optimal control problems online. Compared to [59], the XGDHP approach developed in this chapter simplifies the calculation by eliminating matrix dimensionality transformations.

2. To the best of our knowledge, it is the first time that the asymmetric control input constraints are overcome for zero-equilibrium-point stabilization problems. The combination of a novel segmented utility function and the bounding layer of the actor network guarantees strictly bounded inputs without affecting stability.

3. An event-triggered mechanism is introduced to save computational and communication load. It is the first time that ETC has been combined with XGDHP for DT systems. Compared to existing literature, fewer assumptions are required to guarantee the stability of the triggering condition and a more specific proof is provided, which demonstrates the advantage for wider applications.

The remainder of this chapter is organized as follows: Section 5.2 states the event-triggered optimal control problem with asymmetric input constraints for the general nonlinear DT system. The triggering condition and the stability analysis of the system are provided in Section 5.3. Section 5.4 introduces the iterative XGDHP algorithm with the facilitation of three ANNs. The simulation verification is presented in Section 5.5 by applying the proposed approach to two nonlinear DT systems and Section 5.6 summarizes this chapter and discusses further research.

5.2 Problem Description

Consider a general nonlinear DT system described by:

$$x_{t+1} = f(x_t, u_t),\ t \in \mathbb{N}, \tag{5.1}$$

where t denotes the time instant, $x_t \in \Omega \subset \mathbb{R}^n$ is the state vector, and $u_t \in \Omega_u$ is the control input vector. $\Omega_u = \{u | u \in \mathbb{R}^m, u_{\min} < u_i < u_{\max}, i = 1, \ldots, m\}$, with $u_{\min} < 0$ and $u_{\max} > 0$ denoting the minimum and maximum constraint of u_i, respectively. $|u_{\min}| \neq |u_{\max}|$, i.e., the input constraints are asymmetric.

Assumption 5.1. *System* (5.1) *is controllable and observable.* $f : \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}^n$ *is a Lipschitz continuous function and assumed unknown. The origin* $x_t = 0$ *is the unique equilibrium point of the system* (5.1) *under* u_t*, i.e.,* $f(0, 0) = 0$.

Assumption 5.1 implies that there exists a continuous state feedback control policy $u_t = \mu(x_t)$, $\mu : \Omega \longrightarrow \Omega_u$ that can stabilize system (5.1) to the equilibrium point.

Considering the event-triggered scheme, we define a sequence of triggering instants $\{s_k\}_{k=0}^{\infty}$, with s_k satisfying $s_k < s_{k+1}$, $k \in \mathbb{N}$. The control input is only updated at the triggering instant when a certain triggering condition is satisfied, and remains constant during the time interval $[s_k, s_{k+1})$ by involving a zero-order hold (ZOH) [138, 139]. Therefore, a gap function can be defined using the event error:

$$e_t = x_{s_k} - x_t, \forall t \in [s_k, s_{k+1}), \tag{5.2}$$

where x_t is the current state and x_{s_k} is the triggering state held by the ZOH. Subsequently, the feedback control policy can be represented as:

$$u_t = \mu(x_{s_k}) = \mu(e_t + x_t). \tag{5.3}$$

Accordingly, system (5.1) takes the form:

$$x_{t+1} = f(x_t, \mu(e_t + x_t)). \tag{5.4}$$

Considering the characteristics of system (5.4), we introduce a discounted cost formulated as:

$$J(x_t) = \sum_{l=t}^{\infty} \gamma^{l-t} U(x_l, \mu(x_{s_k})), \tag{5.5}$$

where $\gamma \in (0,1]$ is the discount factor, and $U(x_t, \mu(x_{s_k}))$ is the utility function. For the regulation task, $U(x_t, \mu(x_{s_k}))$ is supposed to satisfy $U(x,\mu) \geq 0$ and $U(0,0) = 0$. Therefore, we define $U(x_t, \mu(x_{s_k}))$ as followings:

$$U(x_t, \mu(x_{s_k})) = x_t^{\mathsf{T}} Q x_t + Y(\mu(x_{s_k})), \tag{5.6}$$

where $Q \in \mathbb{R}^{n\times n}$ is a symmetrical positive definite matrix, and $Y(\mu(x_{s_k}))$ is a positive semi-definite function that satisfies $Y(\mu(x_{s_k})) \geq 0$.

Remark 5.1. The discount factor γ indicates the extent to which the short-term cost or long-term cost is concerned [59, 80]. For the regulation task, given Assumption 5.1, $\gamma \leq 1$ can hold because of the origin equilibrium point, whereas for tasks where the equilibrium point is not the origin, $\gamma < 1$ must be satisfied to guarantee the cost function is finite [114, 141].

The input constraints are asymmetric, which cannot be handled by the integrand function utilized in [91, 93, 127], and for the regulation task, the modified function proposed in [130] is not applicable. Inspired by these studies, we design $Y(\mu(x_{s_k})))$ as the following novel piece-wise integrand function:

$$Y = \begin{cases} 2u_{\max} \displaystyle\int_0^{\mu(x_{s_k})} \tanh^{-\mathsf{T}}(v/u_{\max}) R dv, \ \mu(x_{s_k}) > 0, \\ 2|u_{\min}| \displaystyle\int_0^{\mu(x_{s_k})} \tanh^{-\mathsf{T}}(v/|u_{\min}|) R dv, \ \mu(x_{s_k}) \leq 0, \end{cases} \tag{5.7}$$

where $\tanh^{-\mathsf{T}}(\cdot)$ stands for $(\tanh^{-1}(\cdot))^{\mathsf{T}}$, and $\tanh^{-1}(\cdot)$ is the inverse function of the hyperbolic tangent function $\tanh(\cdot)$, both of which are monotonic odd. $R = \mathrm{diag}([r_1, \cdots, r_m]) \in \mathbb{R}^{m\times m}$ is a positive definite weight matrix, where $\mathrm{diag}(\cdot)$ reshapes the vector to a diagonal matrix. It is worth mentioning that although $Y(\mu(x_{s_k}))$ is piece-wise, it is at least second-order continuous with respect to $\mu(x_{s_k})$, and that only when $\mu(x_{s_k}) = 0$, $Y(\mu(x_{s_k})) = 0$.

Our target is to search for a feedback control law μ to minimize the designed discounted cost function (5.5). On the basis of Bellman's principle of optimality [27], the optimal cost function $J^*(x_t)$ conforms to the DTHJB equation:

$$J^*(x_t) = \min_{\mu(x_{s_k})} \{U(x_t, \mu(x_{s_k})) + \gamma J^*(x_{t+1})\}. \tag{5.8}$$

The optimal control law $\mu^*(x_{s_k})$ at time instant t is accordingly defined as:

$$\mu^*(x_{s_k}) = \arg\min_{\mu(x_{s_k})} \{U(x_t, \mu(x_{s_k})) + \gamma J^*(x_{t+1})\}. \tag{5.9}$$

It is worth mentioning that $\mu^*(x_{s_k})$ is the optimal feedback control law for the sampled state x_{s_k} at the triggering instant s_k, instead of the current state x_t. To obtain appropriate triggering instants for system (5.4), we define a triggering condition as follows:

$$||e_t|| > e_{\text{Thr}}, \tag{5.10}$$

where e_{Thr} is the threshold to be determined. Therefore, it is a primary task for event-triggered control to design a sound threshold, which will be discussed in the next section.

5.3 Event-Triggered System Analysis

In this section, the triggering condition for the DT system is developed and the ISS analysis is carried out. First of all, the following assumption is necessary [121, 138]:

5

Assumption 5.2. *For system* (5.4)*, there exists a positive constant* $C \in (0, 0.5)$ *guaranteeing the following equation:*

$$||f(x_t, \mu(e_t + x_t))|| \leq C||x_t|| + C||e_t||, \tag{5.11}$$

and $||e_t||$ *satisfies* $||e_t|| \leq ||x_t||$.

Lemma 5.1. *If Assumption 5.2 holds, the triggering condition can be defined as follows:*

$$||e_t|| > e_{\text{Thr}} = C\frac{1-(2C)^{t-s_k}}{1-2C}||x_{s_k}||. \tag{5.12}$$

Proof. Regard s_k as the last triggered instant. According to Assumption 5.2, for each $t \in [s_k, s_{k+1})$, we have:

$$||e_{t+1}|| = ||x_{s_k} - x_{t+1}|| \leq ||x_{t+1}||. \tag{5.13}$$

Substituting Eq. (5.11) into Eq. (5.13) yields:

$$||e_{t+1}|| \leq C||x_t|| + C||e_t||. \tag{5.14}$$

With Eq. (5.2), Eq. (5.14) can be rewritten as:

$$||e_{t+1}|| \leq 2C||e_t|| + C||x_{s_k}||. \tag{5.15}$$

Therefore, by conducting back-forward recursion, we obtain the following inequality:

$$||e_t|| \leq 2C||e_{t-1}|| + C||x_{s_k}|| + \cdots \leq (2C)^{t-s_k}||e_{s_k}|| + (2C)^{t-s_k-1}C||x_{s_k}|| + \cdots + (2C)C||x_{s_k}|| + C||x_{s_k}||. \tag{5.16}$$

By solving Eq. (5.16) with initial condition $e_{s_k} = 0$, we attain:

$$||e_t|| \leq C\frac{1-(2C)^{t-s_k}}{1-2C}||x_{s_k}||. \tag{5.17}$$

If Eq. (5.17) is violated, i.e., Eq. (5.12) is satisfied, the event is triggered. This completes the proof. □

It is noted that the threshold value e_{Thr} is not unique since it is influenced by the triggered state x_{s_k} and the designed constant C that is usually chosen experimentally. Subsequently, inspired by [140], we proceed to prove the system (5.4) is asymptotically stable under the triggering condition (5.12).

Definition 5.1. [121] A continuous function $V : \mathbb{R}^n \longrightarrow \mathbb{R}_+$ is called an ISS Lyapunov function for the system (5.4), if there exist $\mathcal{K}_\infty$ functions α_1, α_2, and α_3, and a $\mathcal{K}$ function ρ, such that:

$$\alpha_1(\|x_t\|) \leq V(x_t) \leq \alpha_2(\|x_t\|), \tag{5.18}$$

$$V(f(x_t, \mu(e_t + x_t))) - V(x_t) \leq -\alpha_3(\|x_t\|) + \rho(\|e_t\|), \tag{5.19}$$

hold for all $x_t \in \mathbb{R}^n$ and $e_t \in \mathbb{R}^n$.

Theorem 5.1. *With Assumption 5.2 and the triggering condition* (5.12)*, the event-triggered system is input-to-state stable and is asymptotically stable.*

Proof. The following proof only takes the situation that the event is not triggered at the time instant $t+1$ into consideration, because when the event is triggered, the control input will be updated, and it will be equivalent to the time-based control at $t+1$. According to the optimal control theory, stability can be guaranteed at this single instant.

We firstly define a Lyapunov function as follows:

$$V(x_t) = x_t^\mathsf{T} Q x_t + Y(\mu(x_t)). \tag{5.20}$$

Then, we define a series of functions as follows:

$$\alpha_1(\|x_t\|) = x_t^\mathsf{T} Q_1 x_t + Y(\mu(x_t)), \tag{5.21}$$

$$\alpha_2(\|x_t\|) = x_t^\mathsf{T} Q_2 x_t + Y(\mu(x_t)), \tag{5.22}$$

$$\alpha_3(\|x_t\|) = \|q\|^2(1 - 2C^2)\|x_t\|^2, \tag{5.23}$$

$$\rho(\|e_t\|) = 2C^2\|q\|^2\|e_t\|^2, \tag{5.24}$$

where Q_1 and Q_2 can be selected to satisfy Eq. (5.18), and the vector q in Eqs. (5.23) and (5.24) can be determined from Eq. (5.6) as $Q = qq^\mathsf{T}$. Therefore, $x_t^\mathsf{T} Q x_t = x_t^\mathsf{T} q q^\mathsf{T} x = \|x^\mathsf{T} q\|^2$.

Subsequently, the proof is conducted by presenting that Eq. (5.20) is an ISS Lyapunov function and it is non-increasing, i.e., $\Delta V = V(x_{t+1}) - V(x_t) \leq 0$.

For all $t \in [s_k, s_{k+1})$, according to the ETC mechanism, $\mu(x_{t+1}) = \mu(x_t) = \mu(x_{s_k})$, and therefore, we have

$$\Delta V = x_{t+1}^\mathsf{T} Q x_{t+1} + Y(\mu(x_{t+1})) - (x_t^\mathsf{T} Q x_t + Y(\mu(x_t))) = x_{t+1}^\mathsf{T} Q x_{t+1} - x_t^\mathsf{T} Q x_t. \tag{5.25}$$

Substituting Eq. (5.11) into Eq. (5.25) yields:

$$\Delta V \leq \|q\|^2((C\|x_t\| + C\|e_t\|)^2 - \|x_t\|^2). \tag{5.26}$$

According to the Cauchy-Schwarz inequality, Eq. (5.26) becomes:

$$\begin{aligned} \Delta V \leq & \|q\|^2(2C^2\|x_t\|^2 + 2C^2\|e_t\|^2 - \|x_t\|^2) \\ = & (2C^2 - 1)\|q\|^2\|x_t\|^2 + 2C^2\|q\|^2\|e_t\|^2 \\ = & -\alpha_3(\|x_t\|) + \rho(\|e_t\|). \end{aligned} \tag{5.27}$$

Consequently, referring to Definition 5.1, Eq. (5.20) is an ISS Lyapunov function. According to [121, 129], a system is input-to-state stable if it admits a smooth ISS Lyapunov function.

Then, considering Eqs. (5.2) and (5.17), Eq. (5.27) continues as:

$$\begin{aligned}\Delta V \leq & (4C^2-1)\|q\|^2\|e_t\|^2+(2C^2-1)\|q\|^2\|x_{s_k}\|^2 \\ \leq & \left[(4C^2-1)C\frac{1-(2C)^{t-s_k}}{1-2C}+(2C^2-1)\right]\|q\|^2\|x_{s_k}\|^2,\end{aligned} \tag{5.28}$$

Since $C < 0.5$ and $4C^2 - 1 = (2C-1)(2C+1)$, the last inequality in Eq. (5.28) can be rewritten as:

$$\Delta V = -[(2C+1)C(1-(2C)^{t-s_k})+(1-2C^2)]\|q\|^2\|x_{s_k}\|^2 \leq 0, \tag{5.29}$$

where $\Delta V = 0$ if and only if $\|x_{s_k}\| = 0$, which implies that the system has been stabilized since the time instant s_k.

Overall, we can conclude that the event-triggered system (5.4) is input-to-state stable and is asymptotically stable with the triggering condition (5.12), which completes the proof. □

Remark 5.2. The triggering condition (5.12) has the same form as that in some existing literature [69, 121, 129, 138]. Nevertheless, different from them, fewer assumptions are required to guarantee asymptotic stability. Furthermore, [129, 138] assume the existence of an ISS Lyapunov function without providing its specific formula, whereas in this chapter the ISS Lyapunov function is specifically defined by Eqs. (5.20) - (5.24).

The simple diagram of the ETC scheme is illustrated in Fig. 5.1. Only when an event is triggered, will the XGDHP algorithm be activated and the control input be updated. In the next section, the detailed implementation of the XGDHP algorithm will be presented.

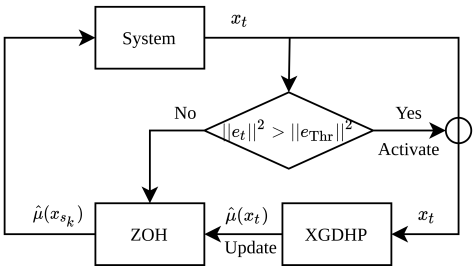


Figure 5.1: Simple diagram of the ETC scheme incorporating the XGDHP algorithm.

5.4 Event-Triggered Iterative ACD with XGDHP

In this section, according to the universal approximation property of ANNs [131], we first construct a model network, represented by subscript m, to identify the system dynamics. Then, the event-triggered iterative adaptive critic algorithm is introduced, and the actor and critic networks, respectively represented by subscript a and c, are built to facilitate

the implementation. The XGDHP technique is developed based on explicit analytical computations in the critic network, and the asymmetric input constraints are addressed by modifying the output layer of the actor network.

All ANNs are constructed with the full-connected feed-forward architecture, and their hidden layers respectively has l_m, l_a, and l_c neurons, all of which adopt a sigmoid function as the activation function:

$$\sigma(\tau) = \frac{1 - e^{-\tau}}{1 + e^{-\tau}}, \tag{5.30}$$

whose derivative is $\sigma'(\tau) = 0.5(1 - (\sigma(\tau))^2)$.

5.4.1 The model network

Since the system dynamics is unknown, a model network is built and trained in advance before implementing the XGDHP technique. The model network is constructed offline to identify the dynamics and predict the next state as follows:

$$x_{t+1} = w_{m2}^{\mathsf{T}} \sigma\left(\begin{bmatrix} w_{m1,x} \\ w_{m1,u} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} x_t \\ u_t \end{bmatrix} \right) + \varepsilon_{m,t}, \tag{5.31}$$

in which $w_{m2} \in \mathbb{R}^{l_m \times n}$, $w_{m1,x} \in \mathbb{R}^{n \times l_m}$, and $w_{m1,u} \in \mathbb{R}^{m \times l_m}$ are ideal weight matrices of the model network, and $\varepsilon_{m,t} \in \mathbb{R}^n$ is the reconstruction error. Subsequently, by defining $w_{m1} = [w_{m1,x}^{\mathsf{T}}, w_{m1,u}^{\mathsf{T}}]^{\mathsf{T}}$, the identification scheme is described as:

$$\hat{x}_{t+1} = \hat{w}_{m2}^{\mathsf{T}} \sigma(\hat{w}_{m1}^{\mathsf{T}} [x_t^{\mathsf{T}}, u_t^{\mathsf{T}}]^{\mathsf{T}}), \tag{5.32}$$

where $\hat{x}_{t+1}$, $\hat{w}_{m1}$, and $\hat{w}_{m2}$ are the estimations of x_{t+1}, w_{m1}, and w_{m2}, respectively.

The model network is supposed to minimize the identification error $\tilde{x}_{m,t+1} = \hat{x}_{t+1} - x_{t+1}$, and therefore the target performance measure is defined as:

$$E_{m,t} = \frac{1}{2} \tilde{x}_{m,t+1}^{\mathsf{T}} \tilde{x}_{m,t+1}. \tag{5.33}$$

The weight tuning law is designed to obey a gradient-descent algorithm:

$$\begin{aligned} \Delta \hat{w}_{m2} &= -\eta_m \frac{\partial \hat{x}_{t+1}}{\partial \hat{w}_{m2,t}} \frac{\partial E_{m,t}}{\partial \hat{x}_{t+1}}, \\ \Delta \hat{w}_{m1} &= -\eta_m \frac{\partial \hat{x}_{t+1}}{\partial \hat{w}_{m1,t}} \frac{\partial E_{m,t}}{\partial \hat{x}_{t+1}}, \end{aligned} \tag{5.34}$$

where $\eta_m > 0$ is the learning rate, and $\Delta \hat{w}_{m1}$ and $\Delta \hat{w}_{m2}$ are the differences of two subsequent updating steps.

After a sufficient training session, the model network can achieve a satisfying precision, with the weight matrix converging to a constant value. It is important to note that after training, the model weight matrix is kept unchanged for controller design. With the model network, the necessary partial derivative information can be obtained for training critic and actor networks.

Considering the event-triggered framework, by replacing u_t with $\mu(x_{s_k})$ in Eq. (5.32) and taking the partial derivative with respect to x_t and $\mu(x_{s_k})$, respectively, we get:

$$\frac{\partial \hat{x}_{t+1}}{\partial x_t} = \hat{w}_{m1,x}(\sigma'(\hat{w}_{m1}^{\mathsf{T}}[x_t^{\mathsf{T}}, \mu^{\mathsf{T}}(x_{s_k})]^{\mathsf{T}}) \odot \hat{w}_{m2}), \tag{5.35}$$

$$\frac{\partial \hat{x}_{t+1}}{\partial \mu(x_{s_k})} = \hat{w}_{m1,u}(\sigma'(\hat{w}_{m1}^{\mathsf{T}}[x_t^{\mathsf{T}}, \mu^{\mathsf{T}}(x_{s_k})]^{\mathsf{T}}) \odot \hat{w}_{m2}). \tag{5.36}$$

By denoting $F_t = \partial \hat{x}_{t+1}^{\mathsf{T}}/\partial x_t$ and $G_t = \partial \hat{x}_{t+1}^{\mathsf{T}}/\partial \mu(x_{s_k})$, we can approximate Eq. (5.4) as a new affine system:

$$\hat{x}_{t+1} = F_t x_t + G_t \mu(x_{s_k}). \tag{5.37}$$

With Eq. (5.37), the optimal event-triggered control law $\mu^*(x_{s_k})$ can accordingly be approximated as:

$$\hat{\mu}^*(x_{s_k}) = \phi(\hat{D}^*(x_{s_k})), \tag{5.38}$$

where $\phi(\cdot)$ is a one-to-one piece-wise function defined as:

$$\phi(\tau) = \begin{cases} u_{\max} \tanh(\tau/u_{\max}), & \tau > 0, \\ |u_{\min}| \tanh(\tau/u_{\min}), & \tau \le 0, \end{cases} \tag{5.39}$$

and $\hat{D}^*(x_{s_k})$ is described by:

$$\hat{D}^*(x_{s_k}) = -\frac{\gamma}{2} R^{-1} G_{s_k}^{\mathsf{T}} \hat{\lambda}^*(\hat{x}_{s_k+1}), \tag{5.40}$$

in which $\hat{\lambda}^*(\hat{x}_{s_k+1}) = \partial \hat{J}^*(\hat{x}_{s_k+1})/\partial \hat{x}_{s_k+1}$ is the costate function. It can be found that $\phi(\cdot)$ is at least second-order continuous. Accordingly, the approximate DTHJB equation takes the form:

$$\hat{J}^*(x_t) = U(x_t, \hat{\mu}^*(x_{s_k})) + \gamma \hat{J}^*(F_t x_t + G_t \hat{\mu}^*(x_{s_k})). \tag{5.41}$$

5.4.2 Iterative adaptive critic algorithm

Through Eqs. (5.38) and (5.41), it can be found that the computation of $\hat{J}^*(x_t)$ and $\hat{\mu}^*(x_{s_k})$ requires the future information. Clearly, although the system dynamics has been identified, this bootstrapping phenomenon [16] makes it intractable or impossible to obtain the analytical solution of the DTHJB equation for nonlinear systems. Consequently, we introduce an iterative adaptive critic algorithm with the XGDHP technique to iteratively solve it.

The procedure of the DT iterative adaptive critic algorithm is briefly depicted in Algorithm 5.1, where $b_{\Delta J} > 0$ is a designed threshold and $i \in \mathbb{N}$ denotes the iteration index. It is worth mentioning that only the situation that $\|e_t\| > \|e_{\text{Thr}}\|$, i.e., $t = s_k$, is considered, because the control input is updated only at the triggered instant. The main idea is to construct two iterative sequences $\{J^{(i)}(x_{s_k})\}$ and $\{\mu^{(i)}(x_{s_k})\}$ to perform the value iteration process so as to achieve approximately optimal values [16, 121].

The convergence analysis of the DT iterative adaptive critic algorithm has been carried out in [27, 128, 142] and thus is omitted here. The core procedure is to prove that $\{J^{(i)}(x_{s_k})\}$ is a non-decreasing sequence with an upper bound b_J, i.e.,

$$J^{(0)}(\cdot) \le J^{(1)}(\cdot) \le \cdots \le J^{(\infty)}(\cdot) \le b_J. \tag{5.44}$$

Algorithm 5.1: Iterative Adaptive Critic Algorithm

Initialization: Choose $b_{\Delta J}$, and set $i = 0$ and $J^{(0)}(\cdot) = 0$;
while $|J^{(i+1)}(x_{s_k}) - J^{(i+1)}(x_{s_k})| > b_{\Delta J}$ **do**
  compute the iterative control input as

$$\mu^{(i)}(x_{s_k}) = \arg\min_{\mu(x_{s_k})} \{U(x_{s_k}, \mu(x_{s_k})) + \gamma J^{(i)}(\hat{x}_{s_k+1})\} = \phi(\hat{D}_{s_k}^{(i)}(x_{s_k})) \tag{5.42}$$

  update the iterative cost function as

$$\begin{aligned} J^{(i+1)}(x_{s_k}) &= \min_{\mu(x_{s_k})} \{U(x_{s_k}, \mu^{(i)}(x_{s_k})) + \gamma J^{(i)}(\hat{x}_{s_k+1})\} \\ &= U(x_{s_k}, \mu^{(i)}(x_{s_k})) + \gamma J^{(i)}(F_{s_k} x_{s_k} + G_{s_k} \mu(x_{s_k})); \end{aligned} \tag{5.43}$$

  i=i+1;
end
Results: Obtain the near optimal control law $\hat{\mu}^*(x_{s_k})$.

Accordingly, we can further derive that the iteration between the sequences (5.42) and (5.43) guarantees the convergence to the optimal values for both sequences, i.e., $J^{(i)}(x_{s_k}) \longrightarrow J^{(\infty)}(x_{s_k}) = \hat{J}^*(x_{s_k})$ and $\mu^{(i)}(x_{s_k}) \longrightarrow \hat{\mu}^*(x_{s_k})$ as $i \longrightarrow \infty$ [121, 143].

Remark 5.3. Since $J^{(i)}(x_{s_k}) \longrightarrow \hat{J}^*(x_{s_k})$ as $i \longrightarrow \infty$, by denoting $\lambda^{(i)}(x_{s_k}) = \partial J^{(i)}(x_{s_k})/\partial x_{s_k}$, we can conclude that the costate function sequence $\{\lambda^{(i)}(x_{s_k})\}$ is also convergent with $\lambda^{(i)}(x_{s_k}) \longrightarrow \hat{\lambda}^*(x_{s_k})$ as $i \longrightarrow \infty$. Nevertheless, in practical implementation, the satisfying convergent results can already be observed when the iteration index i is sufficiently large, rather than infinite.

Subsequently, for carrying out the iterative adaptive critic algorithm, the actor and critic networks are constructed to respectively approximate the control law and the cost function in the following subsections. The derivation presents the calculations in one iteration step and therefore the superscript is omitted for simplicity.

5.4.3 The actor network

For building a direct differentiable mapping from the state to the control input, the actor network is constructed, whose output is directly introduced to the model network and the real system. Inspired by [80, 111, 114], to guarantee the asymmetric input constraints, a bounding layer is connected to the original output layer of the three-layer network. In this bounding layer, the aforementioned function $\phi(\cdot)$ that is defined in Eq. (5.39) is adopted as the activation function. Figure 5.2 illustrates the architecture of the actor network, and its output is presented as:

$$\hat{\mu}(x_{s_k}) = \phi(\hat{w}_{a2}^{\mathrm{T}} \sigma(\hat{w}_{a1}^{\mathrm{T}} x_{s_k})), \tag{5.45}$$

where $\hat{w}_{a1} \in \mathbb{R}^{n \times l_a}$ and $\hat{w}_{a2} \in \mathbb{R}^{l_a \times m}$ are the estimations of the ideal weight matrices $w_{a1} \in \mathbb{R}^{n \times l_a}$ and $w_{a2} \in \mathbb{R}^{l_a \times m}$, respectively.

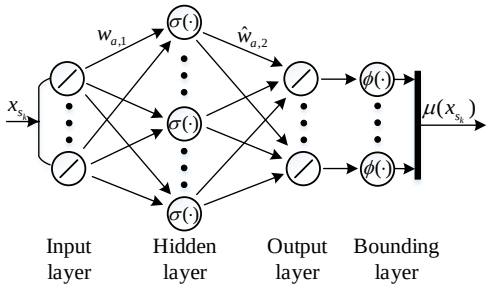


Figure 5.2: The structure of the actor network, where the input layer and the output layer employ a unit-proportion linear activation function, while the hidden layer and the bounding layer exploit aforementioned $\sigma(\cdot)$ and $\phi(\cdot)$ as their activation functions, respectively.

Based on Eqs. (5.42) and (5.45), the performance to be minimized for the actor network can be defined as:

$$E_{a,s_k} = \frac{1}{2}[\hat{\mu}(x_{s_k}) - \mu(x_{s_k})]^{\mathsf{T}}[\hat{\mu}(x_{s_k}) - \mu(x_{s_k})]. \tag{5.46}$$

Similarly, with a learning rate $\eta_a > 0$, the weight matrices are updated by:

$$\begin{aligned} \Delta\hat{w}_{a2} &= -\eta_a \frac{\partial\hat{\mu}(x_{s_k})}{\partial\hat{w}_{a2}} \frac{\partial E_{a,s_k}}{\partial\hat{\mu}(x_{s_k})}, \\ \Delta\hat{w}_{a1} &= -\eta_a \frac{\partial\hat{\mu}(x_{s_k})}{\partial\hat{w}_{a1}} \frac{\partial E_{a,s_k}}{\partial\hat{\mu}(x_{s_k})}. \end{aligned} \tag{5.47}$$

Remark 5.4. The combination of the segmented utility function and the bounding layer of the actor network is one of the highlights of this chapter. With the segmented utility function, a target policy within the designed asymmetric range is provided to the actor network to learn. Besides, the bounding layer is necessary because the signal $\hat{\mu}(x_{s_k})$, which is an output of the actor network, is directly utilized to control the system.

5.4.4 The critic network

For the conventional GDHP technique, the critic network outputs the approximation of cost function and its derivatives simultaneously [69, 99], whose description is as follows:

$$\begin{bmatrix} \hat{J}(x_{s_k}) \\ \hat{\lambda}(x_{s_k}) \end{bmatrix} = \begin{bmatrix} \hat{w}_{c2,J} \\ \hat{w}_{c2,\lambda} \end{bmatrix}^{\mathsf{T}} \sigma(\hat{w}_{c1}^{\mathsf{T}} x_{s_k}), \tag{5.48}$$

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 denotes the estimation of the ideal weights $w_{c1} \in \mathbb{R}^{n\times l_c}$ $w_{c2,J} \in \mathbb{R}^{l_c}$, and $w_{c2,\lambda} \in \mathbb{R}^{l_c\times n}$. However, due to the inevitable approximation error, $\hat{J}(x_{s_k})$ and $\hat{\lambda}(x_{s_k})$ approximated in this way cannot exactly provide the derivative relationship, which is called suffering from the inconsistency error [59].

Therefore, inspired by [59, 80], a novel XGDHP technique that takes advantage of explicit analytical calculations is developed, with the critic network only approximating the cost function as follows:

$$\hat{J}(x_{s_k}) = \hat{w}_{c2}^{\mathsf{T}} \sigma(\hat{w}_{c1}^{\mathsf{T}} x_{s_k}), \tag{5.49}$$

where $\hat{w}_{c2} \in \mathbb{R}^{l_c}$ is the estimation of the ideal weight matrix $w_{c2} \in \mathbb{R}^{l_c}$. By taking the explicit analytical calculations, we obtain $\hat{\lambda}(x_{s_k})$ as:

$$\hat{\lambda}(x_{s_k}) = \frac{\partial \hat{J}(x_{s_k})}{\partial x_{s_k}} = w_{c1}(\hat{w}_{c2} \odot \sigma'(w_{c1}^{\mathsf{T}} x_{s_k})), \tag{5.50}$$

where $\odot$ is the Hadamard product.

XGDHP makes use of the cost function and its derivative information, so recalling Eq. (5.43), the critic network is expected to minimize the following performance measure:

$$e_{c1,s_k} = \hat{J}(x_{s_k}) - U(x_{s_k}, \hat{\mu}(x_{s_k})) - \gamma \hat{J}(\hat{x}_{s_k+1}), \tag{5.51}$$

$$e_{c2,s_k} = \frac{\partial [\hat{J}(x_{s_k}) - U(x_{s_k}, \hat{\mu}(x_{s_k})) - \gamma \hat{J}(\hat{x}_{s_k+1})]}{\partial x_{s_k}}, \tag{5.52}$$

$$E_{c,s_k} = \beta \frac{1}{2} e_{c1,s_k}^2 + (1-\beta) \frac{1}{2} e_{c2,s_k}^{\mathsf{T}} e_{c2,s_k}, \tag{5.53}$$

where β is a scalar within a range of $[0,1]$. If $\beta = 1$, it becomes pure HDP, whereas if $\beta = 0$, then the weight matrix is tuned merely based on the computed derivatives $\hat{\lambda}(x_{s_k})$, and consequently it is equivalent to DHP [80].

Different from [69, 121], we also take the partial derivative of $\hat{\mu}(x_{s_k})$ with respect to x_{s_k} into consideration in the critic network updating procedure for more precise calculations. According to the chain rule, Eq. (5.52) can further be derived as:

$$e_{c2,s_k} = \hat{\lambda}(x_{s_k}) - 2Qx_{s_k} - \frac{\partial \hat{\mu}(x_{s_k})}{\partial x_{s_k}} \frac{\partial Y(\hat{\mu}(x_{s_k}))}{\partial \hat{\mu}(x_{s_k})} - \gamma \left(\frac{\partial \hat{x}_{s_k+1}}{\partial x_{s_k}} + \frac{\partial \hat{\mu}(x_{s_k})}{\partial x_{s_k}} \frac{\partial \hat{x}_{s_k+1}}{\partial \hat{\mu}(x_{s_k})} \right) \hat{\lambda}(x_{s_k+1}), \tag{5.54}$$

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

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

$$\Delta \hat{w}_{c2} = -\eta_c \frac{\partial E_{c,s_k}}{\partial \hat{w}_{c2}}, \ \Delta \hat{w}_{c1} = -\eta_c \frac{\partial E_{c,s_k}}{\partial \hat{w}_{c1}}, \tag{5.55}$$

and

$$\begin{aligned} \frac{\partial E_{c,s_k}}{\partial \hat{w}_{c2}} &= \beta \frac{\partial \hat{J}(x_{s_k})}{\partial \hat{w}_{c2}} e_{c1,s_k} + (1-\beta) \frac{\partial \hat{\lambda}(x_{s_k})}{\partial \hat{w}_{c2}} \mathrm{e}_{c2,s_k}, \\ \frac{\partial E_{c,s_k}}{\partial \hat{w}_{c1}} &= \beta \frac{\partial \hat{J}(x_{s_k})}{\partial \hat{w}_{c1}} e_{c1,s_k} + (1-\beta) \frac{\partial \hat{\lambda}(x_{s_k})}{\partial \hat{w}_{c1}} \mathrm{e}_{c2,s_k}, \end{aligned} \tag{5.56}$$

where $\partial \hat{\lambda}(x_{s_k}) / \partial \hat{w}_{c2}$ and $\partial \hat{\lambda}(x_{s_k}) / \partial \hat{w}_{c1}$ are the second-order mixed gradients of the cost function $\hat{J}(x_{s_k})$. To compute $\partial \hat{\lambda}(x_{s_k}) / \partial \hat{w}_{c2}$ and $\partial \hat{\lambda}(x_{s_k}) / \partial \hat{w}_{c1}$, Kronecker product and thus tensor operations are involved in [59, 80, 111], which result in the need for matrix dimensionality transformation. In this chapter, we develop a simpler computation method as follows:

$$\begin{aligned} \frac{\partial \hat{\lambda}(x_{s_k})}{\partial \hat{w}_{c2}} e_{c2,s_k} &= (\hat{w}_{c1}^{\mathsf{T}} \mathrm{e}_{c2,s_k}) \odot \sigma'(\hat{w}_{c1}^{\mathsf{T}} x_{s_k}), \\ \frac{\partial \hat{\lambda}(x_{s_k})}{\partial \hat{w}_{c1}} e_{c2,s_k} &= e_{c2,s_k} (\hat{w}_{c2} \odot \sigma'(\hat{w}_{c1}^{\mathsf{T}} x_{s_k}))^{\mathsf{T}} - x_{s_k} (\hat{w}_{c1}^{\mathsf{T}} x_{s_k} \odot \hat{w}_{c2} \odot \sigma(\hat{w}_{c1}^{\mathsf{T}} x_{s_k}) \odot \sigma'(\hat{w}_{c1}^{\mathsf{T}} x_{s_k}))^{\mathsf{T}}. \end{aligned} \tag{5.57}$$

Through mathematical derivation, it can be found that Eq. (5.57) is equivalent to the method proposed in [59].

The closed-loop stability and the convergence of weights of ANNs can be found in [138]. Note that the weights between the input layer and the hidden layer of these networks are also updated, which is the same as in [69, 99, 114, 121, 129] but different from [138, 139] where they are fixed after the initialization. Nevertheless, the update behaviours in these methods obey the same gradient descent logic and similar rules. The update is proved successful through the simulation studies in this chapter and others [69, 99, 114, 121, 129]. In practice, one can manually set a constraint for the weights to guarantee boundedness and safety.

Overall, the structural diagram of the present XGDHP implementation is depicted in Fig. 5.3 to clarify the design procedure, where DER is given by:

$$\text{DER} = \frac{\partial \hat{x}_{s_k+1}}{\partial x_{s_k}} + \frac{\partial \hat{\mu}(x_{s_k})}{\partial x_{s_k}} \frac{\partial \hat{x}_{s_k+1}}{\partial \hat{\mu}(x_{s_k})}. \tag{5.58}$$

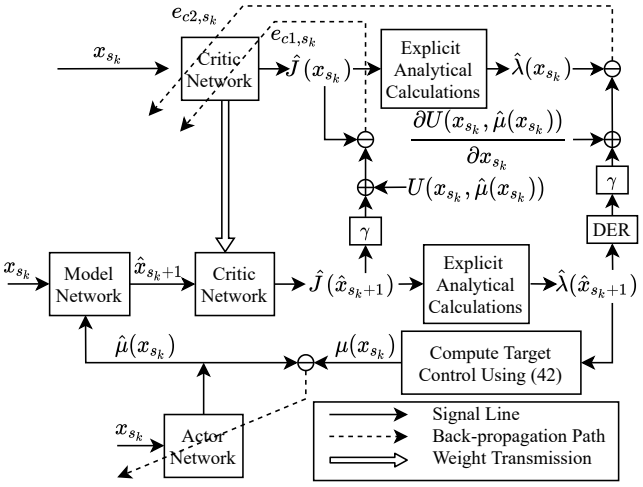


Figure 5.3: Structural diagram of the developed XGDHP algorithm.

5.5 Simulation Studies

In this section, two simulation studies are carried out to illustrate the feasibility of the developed approach and compare the performance of the event-triggered XGDHP with the time-based approach.

5.5.1 Example 1

Consider the following nonlinear affine mass-spring system [128]:

$$\begin{cases} x_{1,t+1} = x_{1,t} + 0.05 x_{2,t}, \\ x_{2,t+1} = -0.0005 x_{1,t} - 0.0335 x_{1,t}^3 + x_{2,t} + 0.05 u_t, \end{cases} \tag{5.59}$$

where $x_t = [x_{1,t}, x_{2,t}]^\mathsf{T} \in \mathbb{R}^2$ and $u_t \in \Omega_u = \{u_t | u_t \in \mathbb{R}, -0.5 < u_t < 0.2\}$. The parameters in the utility function are selected as $Q = I_2$ and $R = 1$, and the forgetting factor is chosen as $\gamma = 0.995$.

In what follows, we perform the proposed event-triggered XGDHP algorithm with the facilitation of ANNs. All ANNs are constructed with 8 hidden neurons, i.e., $l_m = l_c = l_a = 8$. Their weight matrices between the input layer and the hidden layer are initialized within $[-1, 1]$, and the weights between the hidden layer and the output layer are randomly initialized with the uniform distribution within $[-0.01, 0.01]$. The initial weights of the actor and the critic networks utilized to present results are provided in Appendix 5.A. The learning rates are experimentally set as $\eta_m = \eta_c = \eta_a = 0.01$.

First of all, we employ 500 data samples to train the model network for 500 times, and then utilize another 500 data samples for testing. The identification errors of the model network of testing samples are illustrated in Fig. 5.4, from which, we can see that the mean sum of squares of the identification errors is below 1.4×10^{-3}. Therefore, we can say that the model network with high accuracy has been obtained. After training, the weights of the model network are kept unchanged for controller design.

Next, we start the controller design procedure. It is noted that the simulation of the control algorithm is conducted in an online manner, which means that the control policy improves as it is applied to the real system. Through setting $C = 0.12$, we can accordingly obtain the triggering threshold e_{Thr} as:

$$e_{\text{Thr}} = 0.12 \cdot \frac{1 - (0.24)^{t - s_k}}{1 - 0.24} \|x_{s_k}\|. \tag{5.60}$$

If the condition $\|e_t\| > \|e_{\text{Thr}}\|$ is satisfied, the controller will be updated, and the triggering state x_{s_k} is reset with the current state. Before the occurring the next triggering event, the control input u_t is remained by ZOH as u_{s_k}. Different from all other works that apply ETC to DT systems using ADP algorithms [69, 121, 129, 138–140], the actor and critic networks are not updated until an event is triggered in this chapter so as to further reduce computational burden. For the XGDHP technique, we set $\beta = 0.5$ to combine the information of the cost function and its derivatives. For ensuring sufficient learning, the prespecified accuracy $b_{\Delta J}$ is set to be 10^{-4}, and during each time step that is triggered, at most 1000 internal cycles for training the critic and actor networks are included to achieve satisfying performance.

With the initial state chosen as $x_0 = [1, -1]^\mathsf{T}$, we conduct the proposed event-triggered XGDHP algorithm in comparison to the time-based XGDHP algorithm. Both control algorithms share the same settings and parameters except for the triggering mechanism. The simulation results corresponding to the systems state and the control input are depicted in Figs. 5.5 and 5.6, respectively. Due to the event-triggered mechanism, the event-triggered XGDHP algorithm presents a stair-steeping control input signal. With the piece-wise integral function (5.7) and the bounding layer in the actor network, it can be observed that the control input is bounded within the range of $(-0.5, 0.2)$. Therefore, we can say that the asymmetric control input constraints have been addressed. The evolution of the weights of ANNs is depicted in Fig. 5.7, where solid lines denote the weights between the input layer and the hidden layer while the weights between the hidden layer and the output layer are represented by dashed lines. It can be observed that all weights eventually converge to constant values during the online learning process. The evolution of the one-step cost and

the accumulative cost in the learning process is demonstrated in Fig. 5.8. Utilizing fewer data samples, the event-triggered XGDHP requires 3 more steps to control the system achieving the stage where the one-step cost is kept below 0.05, and 7 more steps below 0.01. Because of the delayed control, the event-triggered XGDHP shows a greater overshot, which results in a larger accumulative cost. Nevertheless, in many practical scenarios where saving computational resources is preferred, this depletion of the control effectiveness is acceptable.

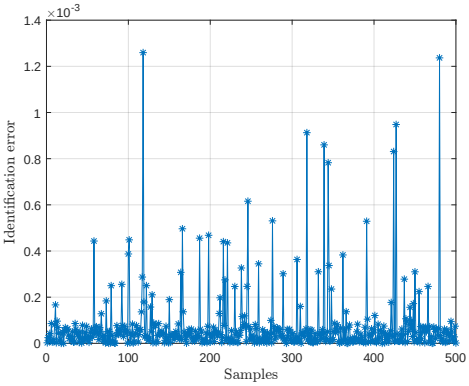


Figure 5.4: The mean sum of squares of the identification errors for Example 1.

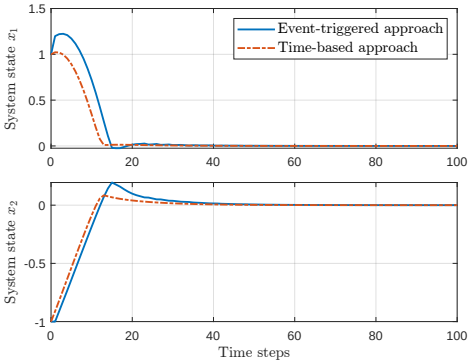


Figure 5.5: Evolution of the system state in the online learning process for Example 1. Controllers aim at stabilizing system states initially from $x_0 = [1, -1]^T$ to 0.

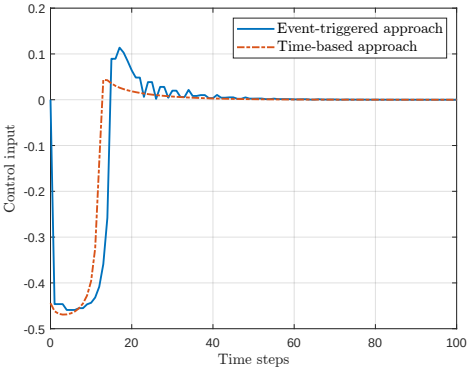


Figure 5.6: Evolution of the control input in the online learning process for Example 1.

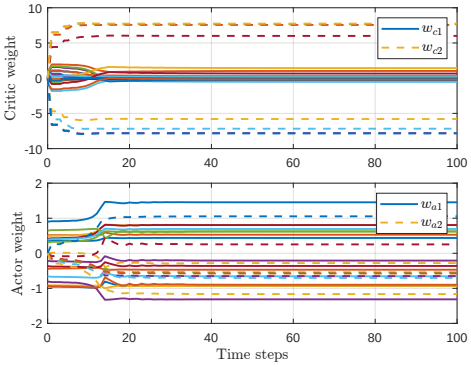


Figure 5.7: Evolution of the weights of ANNs in the online earning process for Example 1. Subscripts c and a denote the critic and actor networks, respectively. Subscripts 1 and 2 denote the weights between the input and the hidden layers and the weights between the hidden and output layers, respectively.

In addition, the evolution curve of the triggering threshold is depicted in Fig. 5.9, which converges to around zero along with the event error. The inter-execution time is

illustrated in Fig. 5.10, which presents the time interval between two triggered instants. It is worth mentioning that the time-based controller requires every sample in this 100-step task, whereas the proposed event-triggered approach only utilizes 60 samples. Since at each triggered instant, the critic and actor networks are trained for 1000 steps, the event-triggered approach greatly reduces the computational burden up to 40%.

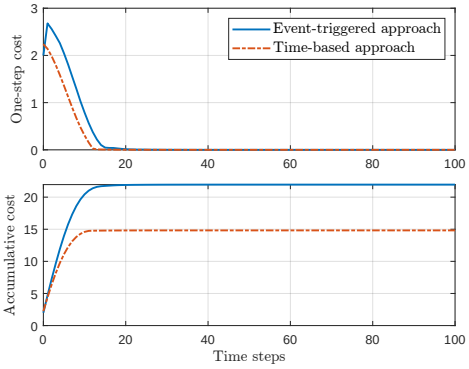


Figure 5.8: Evolution of the one-step and accumulative cost in the online learning process for Example 1.

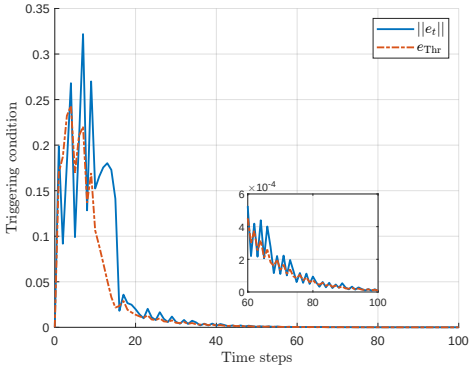


Figure 5.9: Evolution of the triggering condition in the online learning process for Example 1.

Remark 5.5. The simulation results show that the system is gradually stabilized. Nevertheless, it is noted that due to the asymptotic stability property, the system states may not exactly converge to zero, which makes the ETC scheme keep working during the whole presented time range, as depicted in Figs. 5.9 and 5.10. This phenomenon is because the triggering condition described by Eq. (5.12) is dependent on the system states. As the stabilization continues, the triggering threshold will accordingly adapt to a stricter value to guarantee precision. In practice, a threshold can be set for the controller, such that, when the system states reach a certain range, the controller can be deactivated to further save resources.

5.5.2 Example 2

The second numerical example considered is a nonlinear multiple-input-multiple-output nonaffine system [121] described by:

$$\begin{cases} x_{1,t+1} = x_{1,t} + 0.1x_{2,t}, \\ x_{2,t+1} = -0.17\sin(x_{1,t}) + 0.98x_{2,t} + 0.1u_{1,t}, \\ x_{3,t+1} = 0.1x_{1,t} + 0.2x_{2,t} + x_{3,t}\cos(u_{2,t}), \end{cases} \tag{5.61}$$

where $x_t = [x_{1,t}, x_{2,t}, x_{3,t}]^{\mathsf{T}} \in \mathbb{R}^3$ and $u_t \in \Omega_u = \{u_t | u_t \in \mathbb{R}^2, -4 < u_{i,t} < 2, i = 1, 2\}$.

The settings for the second system are similar to those in example 1. The parameters in the utility function are chosen as $Q = I_3$ and $R = 0.01I_2$, and the forgetting factor is set as $\gamma = 0.95$. According to the dimensions of x_t and u_t, the model network is established with the structure of 5-10-3 while both of the critic and actor networks are built as 3-10-2, i.e., $l_m = l_c = l_a = 10$. The weights of all three networks are initialized within $[-0.1, 0.1]$. The

5

initial weights of the actor and the critic networks utilized to present results are provided in Appendix 5.A. Letting $\eta_m = 0.01$, we train the model network for 1000 times using 1000 data samples and examine its performance on a testing data set of another 500 samples. From Fig. 5.11, it is evidently observed that the mean sum of squares of the identification errors has been decreased to less than 6×10^{-4}, which indicates the high accuracy of identification.

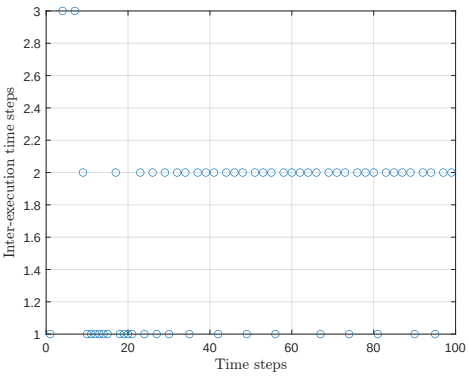


Figure 5.10: Evolution of the inter-execution time in the online learning process for Example 1.

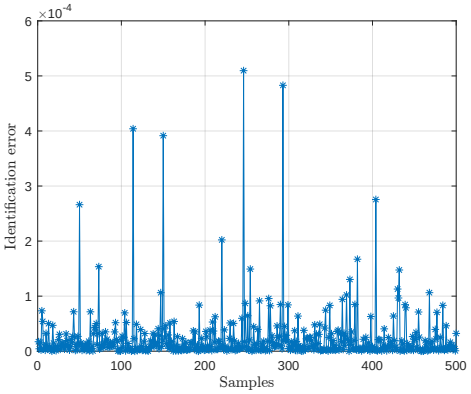


Figure 5.11: The mean sum of squares of the identification errors for Example 2.

For the implementation of the XGDHP technique, we set $\beta = 0.5$ and $b_{\Delta J} = 5 \times 10^{-6}$, and train the critic and actor networks for at most 1000 steps with the learning rates of $\eta_c = \eta_a = 0.001$ at the triggered instant. The triggering threshold is obtained by setting $C = 0.15$ as:

$$e_{\text{Thr}} = 0.15 \cdot \frac{1-(0.3)^{t-s_k}}{1-0.3} ||x_{s_k}||. \tag{5.62}$$

By initializing the system state as $x_0 = [0.5, 0.5, 0.5]^{\mathsf{T}}$, we carry out the online control simulation to verify the performance of the proposed event-triggered XGDHP algorithm. The state trajectories of the event-triggered approach and the time-based approach are displayed in Fig. 5.12. Comparing the event-triggered and time-based approaches, we can observe that, although the event-triggered XGDHP algorithm involves fewer calculations, the state eventually converges to the equilibrium point without obviously deteriorating the converge rate. The control inputs are bounded within $[-4, 2]$, whose curves are depicted in Fig. 5.13. As depicted in Fig. 5.14, the weights of both critic and actor networks are initialized randomly and updated as the controller works, and all weights eventually converge to constant values.

As to the optimal control performance, the event-triggered XGDHP takes 10 more steps to keep the one-step cost below 0.1 and 7 more steps below 0.01. Different from that in Example 1, although the time-based approach converges faster, the accumulative cost of the event-triggered XGDHP is less than the time-based approach. This phenomenon is because the second example requires more oscillations before be stabilized. Since the time-based approach exerts control in each time step, due to the near-optimal property, it shows more aggressive strategies and leads to larger accumulative cost, as illustrated in Fig. 5.15. Remarkably, the control input is only updated 64 times in a total of 200 simulation

steps with the event-triggered approach, saving up to 68% of computational load, which improves the resource utilization. The evolution curves of the triggering threshold and the inter-execution time are illustrated in Figs. 5.16 and 5.17, respectively. All the simulation results uniformly verify the effectiveness of the event-triggered XGDHP control algorithm proposed in this chapter.

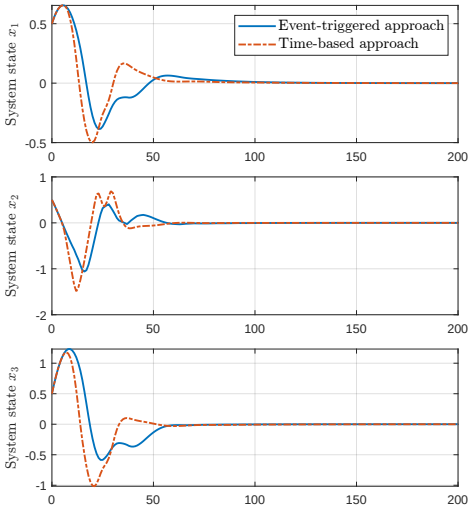


Figure 5.12: Evolution of the system state in the online learning process for Example 2. Controllers aim at stabilizing system states initially from $x_0 = [0.5, 0.5, 0.5]^\mathsf{T}$ to 0.

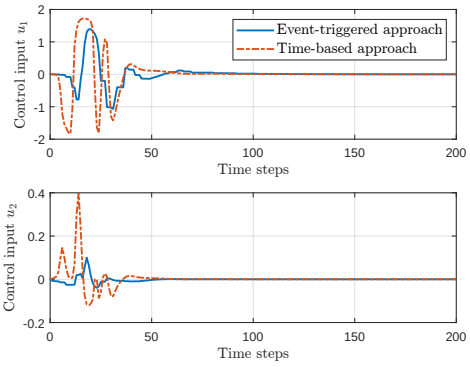


Figure 5.13: Evolution of the control input in the online learning process for Example 2.

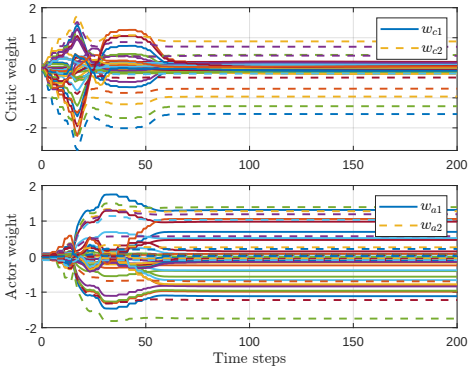


Figure 5.14: Evolution of the weights of ANNs in the online earning process for Example 2. Subscripts c and a denote the critic and actor networks, respectively. Subscripts 1 and 2 denote the weights between the input and the hidden layers and the weights between the hidden and output layers, respectively.

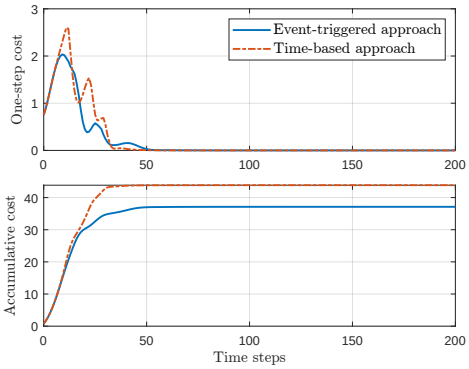


Figure 5.15: Evolution of the one-step and accumulative cost in the online learning process for Example 2.

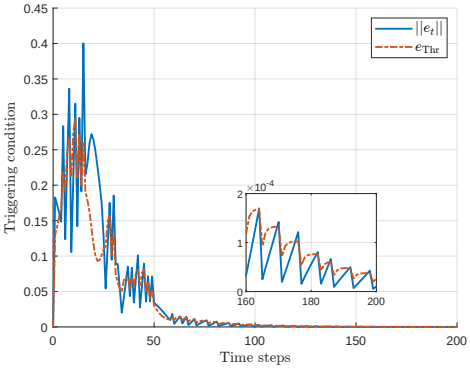


Figure 5.16: Evolution of the triggering condition in the online learning process for Example 2.

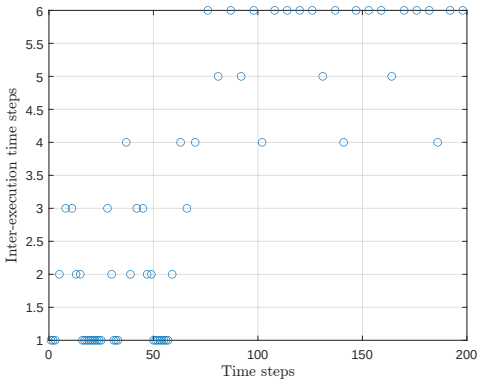


Figure 5.17: Evolution of the inter-execution time in the online learning process for Example 2.

5.6 Conclusion

In this chapter, we develop an event-triggered optimal control algorithm that can deal with asymmetric input constraints for unknown nonlinear discrete-time systems. The stability of the event-triggered control system is analyzed based on the triggering condition with fewer assumptions than existing literature. Besides, the asymmetric input constraints are coped with by the combination of a piece-wise integral function and a bounding layer of the actor network. In addition, with the facilitation of artificial neural networks, the explainable global dual heuristic programming (XGDHP) algorithm is developed to online solve the nonlinear optimal control problem, and the calculations for the derivative of the cost function are simplified without matrix dimensionality transformations.

Two numerical studies are included to illustrate the feasibility and effectiveness of the proposed method. The experimental results present that the nonlinear system can successfully be stabilized with the asymmetric input constraints handled. Furthermore, compared to the conventional time-based approach, the developed event-triggered approach can stabilize these nonlinear systems with at most 10 time steps delayed, while significantly

reducing computational burden up to 40% and 68%, respectively. The communication load between the controller and the plant can also be saved. The results collectively demonstrate the applicability of the proposed approach.

This chapter utilizes a triggering condition that is derived based on the state feedback scheme. However, in many practical systems, full-state feedback is infeasible. Therefore, further investigation into output-feedback control approaches is highly recommended.

5.A Initial Weights

The initial weights for Example 1 are as follows:

$$
\begin{aligned}
w_{a1} &= \begin{bmatrix} 0.3575, 0.4863, 0.3110, 0.4121, -0.4462, -0.8057, 0.3897, 0.9004 \\ 0.5155, -0.2155, -0.6576, -0.9363, -0.9077, 0.6469, -0.3658, -0.9311 \end{bmatrix}, \\
w_{a2} &= [-0.0012, -0.0024, 0.0053, 0.0059, -0.0063, -0.0002, -0.0011, 0.0029]^{\mathsf{T}}, \\
w_{c1} &= \begin{bmatrix} 0.6294, -0.7460, 0.2647, -0.4430, 0.9150, -0.6848, 0.9143, 0.6006 \\ 0.8116, 0.8268, -0.8049, 0.0938, 0.9298, 0.9412, -0.0292, -0.7162 \end{bmatrix}, \\
w_{c2} &= [-0.0016, -0.0083, 0.0058, 0.0092, 0.0031, -0.0093, -0.0070, 0.0087]^{\mathsf{T}}.
\end{aligned}
$$

The initial weights for Example 2 are as follows:

$$
\begin{aligned}
w_{a1} &= \begin{bmatrix} -0.0154, -0.0058, 0.0277, -0.0361, -0.0185, 0.0937, -0.0789, -0.0153, -0.0693, 0.0054 \\ -0.0812, 0.0392, -0.0933, 0.0062, 0.0640, 0.0063, 0.0222, -0.0818, -0.0438, -0.0085 \\ 0.0197, 0.0400, -0.0862, 0.0309, 0.0437, -0.0350, 0.0058, -0.0467, -0.0120, 0.0751 \end{bmatrix}, \\
w_{a2} &= \begin{bmatrix} 0.0036, 0.0887, 0.0275, 0.0915, -0.0519, 0.0352, -0.0422, 0.0344, 0.0390, -0.0864 \\ -0.0490, -0.0552, 0.0336, 0.0689, -0.0311, 0.0561, 0.0351, -0.0987, 0.0204, -0.0226 \end{bmatrix}^{\mathsf{T}}, \\
w_{c1} &= \begin{bmatrix} -0.0935, 0.0338, -0.0079, 0.0711, -0.0618, -0.0759, -0.0231, -0.0419, 0.0649, -0.0312 \\ 0.0122, -0.0619, 0.0963, 0.0290, -0.0143, 0.0179, 0.0166, 0.0234, 0.0965, 0.0168 \\ 0.0764, -0.0262, -0.0687, -0.0247, -0.0036, -0.0548, -0.0496, -0.0469, 0.0460, -0.0784 \end{bmatrix}, \\
w_{c2} &= [0.0813, 0.0759, 0.0636, -0.0479, 0.0189, -0.0955, -0.0149, -0.0375, -0.0677, -0.0642]^{\mathsf{T}}.
\end{aligned}
$$

6

Event-Triggered Intelligent Critic Control with Input Constraints

Event-Triggered Intelligent Critic Control with Input Constraints Applied to A Nonlinear Aeroelastic System

In previous chapters, research sub-questions 2 and 3 were answered with respect to discrete-time (DT) systems. The optimality and stability were persuaded while dealing with the control input constraints of aerial vehicles, and event-triggered control (ETC) was introduced to save computational and communication load. In this chapter, the progress will be extended to the continuous-time (CT) system. A single-critic-network adaptive dynamic programming (ADP) algorithm will be introduced. Different from DT systems where the ETC loop is designed separately, in CT systems ETC is incorporated into ADP. Therefore, this chapter will design a novel triggering condition with control input constraints considered and the infamous Zeno phenomenon in the ETC field will be analysed and avoided. To further enhance the system stability, an improved weight updating criterion will be designed to eliminate the requirement of initial admissible control. Then, the Lyapunov stability of the closed-loop system will be analysed. Finally, the proposed method will be applied to an aeroelastic wing section for verification.

This chapter is based on the following article:
B. Sun, X. Wang and E. van Kampen, "Event-Triggered Intelligent Critic Control with Input Constraints Applied to A Nonlinear Aeroelastic System", *Aerospace Science and Technology*, vol. 120, pp. 107279, 2022. Doi: 10.1016/j.ast.2021.107279 [144].

Abstract

In this chapter, we establish an event-triggered intelligent control scheme with a single critic network, to cope with the optimal stabilization problem of nonlinear aeroelastic systems. The main contribution lies in the design of a novel triggering condition with input constraints, avoiding the Lipschitz assumption on the inverse hyperbolic tangent function. Based on an improved weight updating criterion that eliminates the requirement of initial admissible control, the control law is obtained approximately by online training of a single critic network. The Lyapunov stability and the Zeno phenomenon of the closed-loop system are analysed. The feasibility of the established algorithm is verified by applying it to an optimal stabilization task of a nonlinear aeroelastic system. The results reveal that the developed approach can handle input-constrained optimal control problems, with performance comparable to the time-based method that updates control inputs at each instant, while reducing the computational and communication load.

6.1 Introduction

Aeroelastic systems exhibit a variety of unstable phenomena, such as flutter and limit-cycle oscillations (LCOs), which can significantly degrade the flight performance of an aircraft [145–147]. For this reason, stable controller design for aeroelastic systems has been receiving considerable attention for decades in aerospace engineering research groups [146, 148–150]. Most current controllers are designed based on feedback linearization approaches [148]. However, with the rapid development of aviation technologies, these traditional methods show their limitations in dealing with stronger nonlinearities. Input nonlinearities such as saturation constraints commonly exist in real systems [37, 111], but they are rarely investigated for aeroelastic systems in the existing literature. Furthermore, complex systems usually involve multiple control loops closed through some communication mediums, which brings a growing interest in enhancing resource utilization [151]. Motivated by the demands for tackling these challenges, this chapter aims to develop a constrained-input optimal control approach with reduced computational and communication costs for nonlinear aeroelastic systems.

Optimal control problems pursue optimal control policies for dynamical systems by maximizing/minimizing a pre-defined performance function that captures desired objectives [125]. When dealing with optimal control problems, it is common to solve the Hamilton-Jacobi-Bellman (HJB) equation, but there are few effective approaches to obtain its analytical solutions for nonlinear systems [125, 152]. Adaptive dynamic programming (ADP) provides a promising method to acquire numerical solutions of general HJB equations. By incorporating artificial neural networks (ANNs), ADP acquires a more powerful generalization capability and has been successfully applied to a variety of aerospace systems [50, 59, 80, 105, 153, 154]. As a branch of reinforcement learning (RL), the principle of ADP lies in the effective iterations between policy improvement and policy evaluation [16, 125], which are sometimes approximated by an actor network and a critic network, respectively [50, 59, 105]. However, for continuous-time (CT) systems, by solving the HJB equation, the single critic network (SCN) architecture is able to perform ADP with lower computational cost and eliminate the approximation error introduced by the actor network [37, 155]. Different from the actor-critic architecture, where the input saturation constraints are

addressed by the bounded output neurons of the actor network [59], the SCN structure ordinarily utilizes a non-quadratic cost function, such that the control inputs derived from the solution of HJB equation can be bounded by a hyperbolic tangent function [37, 91, 156].

Although time-based ADP approaches provide a mature and normative solution to nonlinear optimal control problems, the need for reducing transmitted data is not fully satisfied. Arising from networked control systems [55, 157], event-triggered control (ETC) has attained a lot of attention in recent years because of its ability to reduce computational and communication load [125]. A cross-fertilization of ETC and ADP produces event-triggered ADP, which has successfully been implemented for optimal stabilization of both discrete-time systems [119, 131] and CT systems [158, 159]. The key attribute of the event-triggered mechanism lies in that the control signals are updated only when a certain condition is triggered [119]. Therefore, designing a sound triggering condition is the principal task of ETC. 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. The related analysis is conducted in [136, 160] without incorporating ANNs and in [151] without taking the input constraints into account. Based on these studies, the closed-loop analysis is carried out to ensure that the Zeno phenomenon is inapplicable to the proposed method.

Developing from the time-based ADP, event-triggered ADP methods inherit and continue most of the properties and techniques of the time-based ADP, including the technique for handling input constraints with a non-quadratic cost function [136, 161]. However, in most existing literature, the triggering condition is derived by involving a Lipschitz constant of the inverse hyperbolic tangent function without effectively narrowing its domain. Although satisfying experimental results can be obtained in certain circumstances, this derivation is not mathematically rigorous. According to [162], in which the actor-critic structure is adopted, this chapter replaces this Lipschitz constant using meticulously mathematical transformations with the SCN architecture.

In addition, the initial admissible control is a requirement for both time-based and event-triggered ADP methods, which weakens their application, especially for closed-loop online learning control. Inspired by [136, 151, 163], an improved weight updating rule is designed by adding a stabilizing term based on the Lyapunov stability theory, such that the requirement of initial admissible control is eliminated.

The contributions of this chapter are summarized as follows:

1. It is the first time that an ADP-based controller is developed for a nonlinear aeroelastic system. This chapter develops a general control method that can be applied directly without making coordinate transformations.

2. A novel triggering condition incorporating input constraints is derived without requiring the Lipschitz assumption on the inverse hyperbolic tangent function.

3. The demand for the initial admissible control is relaxed by an improved critic weight updating criterion.

4. The Zeno phenomenon is analysed and avoided regarding the closed-loop system with the event-triggered control strategy.

The remainder of this chapter is organized as follows: Section 6.2 states the constrained-input optimal control problem for a CT nonlinear aeroelastic system under the event-triggered framework. Section 6.3 provides the implementation of the event-triggered controller using ANN, and analyses the closed-loop stability as well as avoids the Zeno phenomenon. The simulation verification is presented in Section 6.4, and Section 6.5 summarizes this chapter and states further research.

The main notations used in what follows are listed. $\mathbb{N}$ is the set of all natural numbers. $\mathbb{R}$ denotes the set of all real numbers. $\mathbb{R}^n$ indicates the Euclidean space of all n-dimensional real vectors. $\mathbb{R}^{n\times m}$ is the space of all $n\times m$ real matrices. $|\cdot|$ is the scalar absolute value and $\|\cdot\|$ is the norm of the corresponding vector or matrix. $(\cdot)^-$ denotes the left continuity and $(\cdot)^{\mathrm{T}}$ represents the transpose operation. I_n denotes the $n\times n$ identity matrix and $\underline{\mathbf{1}}$ is a column vector with all elements equal to one. $\overline{\lambda}(\cdot)$ and $\underline{\lambda}(\cdot)$ respectively represents the maximal and minimal eigenvalues of a matrix. Denote Ω as a compact subset of $\mathbb{R}^n$, Ω_u as a compact subset of $\mathbb{R}^m$, and $\mathscr{A}(\Omega)$ as the set of admissible controllers on Ω. The symbol $\nabla(\cdot)\triangleq\partial(\cdot)/\partial x$ stands for the gradient operator.

6.2 Problem Description

A typical aeroelastic wing section plant with two degrees of freedom is modeled in this section. Then, we describe the constrained-input optimal control problem of general nonlinear systems, and present the event-triggered control mechanism.

6.2.1 Aeroelastic wing section model

With the wide usage of composite materials, high aspect-ratio aircraft wings can suffer from aeroelastic instability phenomena, including the LCOs [147, 164]. If not suppressed by active control, LCOs can lead to structural failure and even flight accidents [146]. The schematic of an aeroelastic wing section controlled by a single trailing-edge flap is illustrated in Fig. 6.1 [148], 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 the angle of attack α. Consequently, the governing expressions of motion are presented as [148, 150]:

$$\begin{bmatrix} m_t & m_w x_\alpha b \\ m_w x_\alpha b & I_\alpha \end{bmatrix}\begin{bmatrix}\ddot{h}\\ \ddot{\alpha}\end{bmatrix}+\begin{bmatrix} c_h & 0\\ 0 & c_\alpha\end{bmatrix}\begin{bmatrix}\dot{h}\\ \dot{\alpha}\end{bmatrix}+\begin{bmatrix} k_h(h) & 0\\ 0 & k_\alpha(\alpha)\end{bmatrix}\begin{bmatrix}h\\ \alpha\end{bmatrix}=\begin{bmatrix}-L\\ M\end{bmatrix}, \tag{6.1}$$

where $k_h(h)$ and $k_\alpha(\alpha)$ respectively represents the plunge and pitch stiffness, which can be formulated by nonlinear polynomials as [146]:

$$\begin{aligned} k_h &= 2844(1+0.9h^2)\\ k_\alpha &= 2.82(1-22.1\alpha+1315.5\alpha^2-8580\alpha^3+17289.7\alpha^4), \end{aligned} \tag{6.2}$$

and the remaining constant parameters are listed in Table 6.1; L and M respectively denotes the aerodynamic force and moment, which are formulated in a quasi-steady form as [148]:

$$\begin{aligned} L &= \rho U^2 b c_{L\alpha}\left(\alpha + \frac{\dot{h}}{U} + \overline{a}b\frac{\dot{\alpha}}{U}\right) + \rho U^2 b c_{L\beta}\beta, \\ M &= \rho U^2 b^2 c_{m\alpha}\left(\alpha + \frac{\dot{h}}{U} + \overline{a}b\frac{\dot{\alpha}}{U}\right) + \rho U^2 b^2 c_{m\beta}\beta, \end{aligned} \tag{6.3}$$

where $\overline{a} = 0.5 - a$, and β is the control surface deflection. As presented by (6.1) - (6.3), the motion dynamics of the aeroelastic system are nonlinear. To describe the system more profoundly, the properties of a simplified linear system are provided. By neglecting the nonlinear terms in (6.2), the flutter speed of the resulting linear aeroelastic system is 12.41 m/s. The natural frequencies of the corresponding linear undamped aeroelastic system are 9.11 rad/s and 13.28 rad/s.

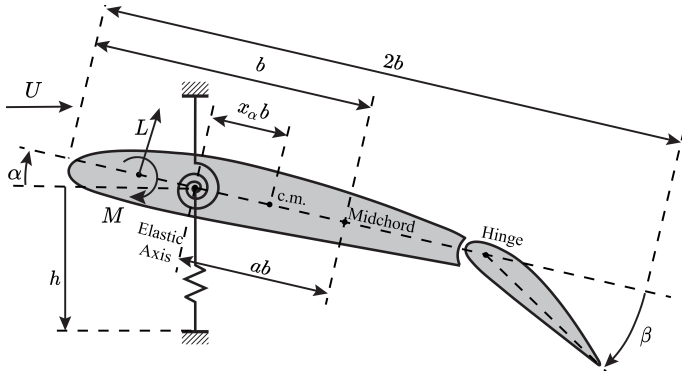


Figure 6.1: An two-degree-of-freedom aeroelastic system with one control surface (adapted from [148]).

The complete flight control system often involves the actuator, which can be described as a first-order component [59]:

$$\dot{\beta} - k_\beta \beta = k_\beta \beta_c, \tag{6.4}$$

where β_c is the deflection command directly generated by the controller. In this case, the complete state vector is $x = [x_1, x_2, x_3, x_4, x_5]^\mathsf{T} = [h, \alpha, \dot{h}, \dot{\alpha}, \beta]^\mathsf{T}$, and the control input is $u = \beta_c$. Besides, due to mechanical limitations, the control surface deflection always has constraints, which should be taken into consideration in the controller design process.

6.2.2 Optimal control design with input constraints

To provide a general description, we consider a class of nonlinear CT systems formulated by:

$$\dot{x}(t) = f(x(t)) + g(x(t))u(x(t)), \tag{6.5}$$

where $x(t) \in \Omega \subset \mathbb{R}^n$ is the state vector and $u(x(t)) \in \Omega_u$ is the control signal vector, and $\Omega_u = \{u | u \in \mathbb{R}^m, |u_i| < u_b, i = 1, \ldots, m\}$, in which u_b is the control saturating bound. $f(\cdot)$ is Lipschitz continuous in Ω satisfying $f(0) = 0$. The initial state at $t = 0$ is $x(0) = x_0$, and $x = 0$

Table 6.1: CONSTANT PARAMETERS FOR THE AEROELASTIC SYSTEM

Symbol	Meaning	Value
a	nondimensional distance from the midchord to the elastic axis	-0.6847
b	wing semichord	0.135 m
m_t	total mass of the wing and the flap	12.387 kg
m_w	mass of the wing	2.049 kg
x_α	nondimensional distance from the elastic axis to the c.m.	$(0.0873-(b+ab))/b$
I_α	moment of inertia	$m_w x_\alpha^2 b^2 + 0.0517$ kg $\cdot$ m
c_h	plunge damping coefficient	27.43 kg/s
c_α	pitch damping coefficient	0.036 kg $\cdot$ m^2/s
$c_{L\alpha}$	lift coefficient per α	6.28
$c_{m\alpha}$	moment coefficient per α	$(0.5+a)c_{L\alpha}$
$c_{L\beta}$	lift coefficient per β	3.358
$c_{m\beta}$	moment coefficient per β	-1.94
ρ	density of air	1.225 kg/m^3
U	freestream velocity	15m/s

is the equilibrium point of the system. System (6.5) is generally assumed to be controllable. For simplicity, we denote $x(t)$ by x hereafter.

For system (6.5), an infinite-horizon cost function can be defined as:

$$J(x) = \int_t^\infty x^\mathsf{T} Q x + Y(u)\mathrm{d}\tau, \tag{6.6}$$

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 denote $U(x,u(x)) = x^\mathsf{T} Q x + Y(u)$ as the utility function, and $U(x,u(x))$ satisfies $U(x,u) \geq 0$ and $U(0,0) = 0$. Inspired by [37, 91], we define $Y(u)$ as:

$$Y(u) = 2u_b \int_0^u \tanh^{-\mathsf{T}}(v/u_b) R \mathrm{d}v = 2u_b \sum_{i=1}^m \int_0^{u_i} \tanh^{-\mathsf{T}}(v_i/u_b) r_i \mathrm{d}v_i, \tag{6.7}$$

where $\tanh^{-\mathsf{T}}(\cdot)$ stands for $(\tanh^{-1}(\cdot))^\mathsf{T}$, and $\tanh^{-1}(\cdot)$ is 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. $Y(u)$ satisfies $Y(u) \geq 0$, and only when $u = 0$, $Y(u) = 0$.

Admissible control is a prerequisite of optimal feedback stabilization, such that the cost function $J(x)$ is guaranteed to be finite. Choosing an admissible control law $u(x) \in \mathscr{A}(\Omega)$, and accordingly the Hamiltonian is defined as:

$$H(x,u(x),\nabla J(x)) = U(x,u(x)) + \nabla J^\mathsf{T}(x)[f(x)+g(x)u(x)]. \tag{6.8}$$

The optimal value of the cost function given in (6.6) is:

$$J^*(x) = \min_{u(x)\in\mathscr{A}(\Omega)} \int_t^\infty U(x(\tau),u(x(\tau)))\mathrm{d}\tau, \tag{6.9}$$

and it satisfies the HJB equation:

$$0 = \min_{u(x)\in\mathscr{A}(\Omega)} H(x, u(x), \nabla J^*(x)). \tag{6.10}$$

By making $\partial H(x, u(x), \nabla J(x))/\partial u(x) = 0$, the corresponding optimal feedback control solution is derived by:

$$\begin{aligned} u^*(x) &= \arg\min_{u(x)\in\mathscr{A}(\Omega)} H(x, u(x), \nabla J^*(x)) \\ &= -u_b \tanh(D^*), \end{aligned} \tag{6.11}$$

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

$$D^* = \frac{1}{2u_b} R^{-1} g^\mathsf{T}(x) \nabla J^*(x). \tag{6.12}$$

The control input u^* is bounded by u_b, and the nonquadratic cost (6.7) regarding u^* is:

$$Y(u^*(x)) = u_b \nabla J^{*\mathsf{T}}(x) g(x) \tanh(D^*) + u_b^2 \underline{R} \ln(\underline{\mathbf{1}} - \tanh^2(D^*)), \tag{6.13}$$

where $\nabla J^{*\mathsf{T}}(x)$ denotes $(\nabla J^*(x))^\mathsf{T}$ and $\underline{R} = [r_1, \cdots, r_m]^\mathsf{T}$. Substituting (6.11) and (6.12) into the HJB equation produces:

$$0 = x^\mathsf{T} Q x + u_b^2 \underline{R} \ln(\underline{\mathbf{1}} - \tanh^2(D^*)) + \nabla J^{*\mathsf{T}}(x) f(x), \tag{6.14}$$

with $J^*(0) = 0$ that leads to $H(x, u^*(x), \nabla J^*(x)) = 0$.

6.2.3 Event-triggered scheme design

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:

$$e_k(t) = x_k - x, \forall t \in [s_k, s_{k+1}). \tag{6.15}$$

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 piecewise 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 (6.5) takes the form:

$$\dot{x} = f(x) + g(x) u(x + e_k), \forall t \in [s_k, s_{k+1}). \tag{6.16}$$

Considering the event-triggered framework, combined with (6.12), the feedback control function (6.11) becomes:

$$u^*(x_k) = -u_b \tanh(D_k^*), \tag{6.17}$$

where D_k^* is given as:

$$D_k^* = \frac{1}{2u_b} R^{-1} g^\mathsf{T}(x_k) \nabla J^*(x_k). \tag{6.18}$$

For system (6.5), with the infinite-horizon cost function represented by (6.6), we define a triggering condition as follows:

$$\|e_k\|^2 > \|e_T\|^2, \tag{6.19}$$

where e_T is the threshold to be determined. We say the event is triggered if (6.19) is satisfied, and in the following section, we present the details of how to determine the threshold.

6.3 Intelligent Critic Control Implementation

Since (6.14) is a nonlinear partial differential equation intractable to be solved analytically, in this section an ANN with an improved updating rule is used to approximate the optimal control policy. Then, the system stability is analysed and the Zeno phenomenon is avoided regarding the closed-loop system.

6.3.1 Improved neural control implementation

In light of the powerful generalization property of ANNs, the optimal cost function can be reconstructed as follows:

$$J^*(x) = w_c^\top \sigma_c(x) + \varepsilon_c(x), \tag{6.20}$$

where $w_c \in \mathbb{R}^{l_c}$ stands for the ideal weight, l_c is the number of neurons, $\sigma(x)_c \in \mathbb{R}^{l_c}$ denotes the activation function, and $\varepsilon_c(x) \in \mathbb{R}$ represents the neural approximation error. Accordingly, the gradient vector of the optimal cost is:

$$\nabla J^*(x) = \nabla \sigma_c^\top(x) w_c + \nabla \varepsilon_c(x). \tag{6.21}$$

Since the ideal weight vector is unavailable in advance, a critic network is constructed to approximate the cost function with an estimated weight vector $\hat{w}_c \in \mathbb{R}^{l_c}$ such that:

$$\hat{J}^*(x) = \hat{w}_c^\top \sigma_c(x). \tag{6.22}$$

Similarly, we determine:

$$\nabla \hat{J}^*(x) = \nabla \sigma_c^\top(x) \hat{w}_c. \tag{6.23}$$

Considering the ANN formulation (6.21), (6.18) can be rewritten as:

$$D_k^* = \frac{1}{2u_b}[R^{-1} g^\top(x_k)(\nabla \sigma_c^\top(x_k) w_c + \nabla \varepsilon_c(x))]. \tag{6.24}$$

Based on the mean-value theorem [91], we can accordingly rewrite (6.17) as:

$$u^*(x_k) = -u_b \tanh(D_k) + \varepsilon_{u_k^*}, \tag{6.25}$$

where $D_k = 1/(2u_b)(R^{-1} g^\top(x_k) \nabla \sigma_c^\top(x_k) w_c)$, and $\varepsilon_{u_k^*} = -1/2(\underline{1} - \tanh^2(\xi)) R^{-1} g^\top(x_k) \nabla \varepsilon$, where $\xi \in \mathbb{R}^m$ is selected between D_k and D_k^*.

Hence, according to (6.17) and (6.23), the event-triggered approximate optimal policy can be formulated as:

$$\hat{u}(x_k) = -u_b \tanh(\hat{D}_k), \tag{6.26}$$

and $\hat{D}_k$ is modulated as:

$$\hat{D}_k = \frac{1}{2u_b}(R^{-1}g^{\mathsf{T}}(x_k)\nabla\sigma_c^{\mathsf{T}}(x_k)\hat{w}_c). \tag{6.27}$$

Substituting (6.25) into the Hamiltonian (6.8) yields that:

$$H(x, u^*(x_k), w_c) = w_c^{\mathsf{T}}\nabla\sigma_c(x)[f(x) + g(x)u^*(x_k)] + x^{\mathsf{T}}Qx + Y(u^*(x_k)) \triangleq e_{cH}, \tag{6.28}$$

where $u^*(x_k) = u_b \tanh(D_k^*)$, and $e_{cH} = -\nabla\varepsilon_c^{\mathsf{T}}(x)[f(x)+g(x)u^*(x_k)]$ is the residual error brought by the ANN. Utilizing (6.23), the approximate Hamiltonian is presented as:

$$\hat{H}(x, u^*(x_k), \hat{w}_c) = \hat{w}_c^{\mathsf{T}}\nabla\sigma_c(x)[f(x) + g(x)u^*(x_k)] + x^{\mathsf{T}}Qx + Y(u^*(x_k)) \triangleq e_c. \tag{6.29}$$

Defining the critic error vector as $\tilde{w}_c = w_c - \hat{w}_c$ and combining (6.28) with (6.29), we obtain an equivalent expression of e_c:

$$e_c = e_{cH} - \tilde{w}_c^{\mathsf{T}}\nabla\sigma_c(x)[f(x) + g(x)u^*(x_k)]. \tag{6.30}$$

Hence, the aim of training the critic network is to obtain an appropriate weight vector $\hat{w}_c$ such that the objective function $E_c = (1/2)e_c^{\mathsf{T}}e_c$ is minimum. It is worth mentioning that the actual control law utilized during the learning process is the approximated control (6.26). In [161], a direct gradient-descent method is applied to adjust the critic weight vector:

$$\dot{\hat{w}}_{c,\text{trad}} = -\eta_c\frac{\partial e_c}{\partial\hat{w}_c}e_c = -\eta_c\phi e_c, \tag{6.31}$$

where $\eta_c > 0$ is the learning rate parameter, and $\phi = \nabla\sigma_c(x)(f(x) + g(x)\hat{u}(x_k))$.

The admissible control is essential for the general ADP-based optimal control design but is intractable to obtain in advance. To overcome this challenge, inspired by [136, 151, 163], we bring in an extra stabilizing term to improve the direct gradient-descent method and adopt it to enhance the ANN weight updating. Similar to [91, 151, 163], we make the following assumption:

Assumption 6.1. *Consider system* (6.5) *with the cost function* (6.6) *and its closed-loop form governed by the event-triggered optimal controller* (6.17) *and* (6.24). *Let $J_s(x)$ be a continuously differentiable Lyapunov function candidate satisfying:*

$$\dot{J}_s^*(x) = \nabla J_s^{\mathsf{T}}(x)[f(x) + g(x)u^*(x_k)] < 0.$$

Then, there exists a positive definite matrix $M \in \mathbb{R}^{n\times n}$ such that the following inequality holds:

$$\dot{J}_s^*(x) = \nabla J_s^{\mathsf{T}}(x)M\nabla J_s(x) \le -\underline{\lambda}(M)\|\nabla J_s(x)\|^2. \tag{6.32}$$

When adopting the event-triggered approximate optimal control (6.26), we should exclude the following case to guarantee the system stability:

$$\dot{J}_s(x) = \nabla J_s^{\mathsf{T}}(x)[f(x) + g(x)\hat{u}(x_k)] > 0.$$

Hence, the learning performance is reinforced by adjusting the time derivative of $J_s(x)$ along the direction of the negative gradient, which is modulated as follows:

$$\begin{aligned}\dot{\hat{w}}_{c,\text{stab}} &= -\eta_s \frac{\partial \nabla J_s^{\mathrm{T}}(x)[f(x)+g(x)\hat{u}(x_k)]}{\partial \hat{w}_c} \\ &= \frac{1}{2}\eta_s \nabla\sigma(x_k)g(x_k)R^{-1}(\underline{1}-\tanh^2(\hat{D}_k))g^{\mathrm{T}}(x)\nabla J_s(x),\end{aligned} \tag{6.33}$$

where $\eta_s > 0$ is the designed learning rate. By combining the stabilizing term (6.33) and the traditional rule (6.31), we established the improved ANN learning criterion as follows:

$$\dot{\hat{w}}_c = \dot{\hat{w}}_{c,\text{trad}} + \Xi(x,\hat{u}(x_k))\dot{\hat{w}}_{c,\text{stab}}, \tag{6.34}$$

where $\Xi(x,\hat{u}(x_k))$ is a sign function utilized to eliminate the effect of the reinforced term when the system is already stable, which is defined as:

$$\Xi(x,\hat{u}(x_k)) = \begin{cases} 0, \text{when } \dot{J}_s(x) < 0, \\ 1, \text{elsewhere.} \end{cases} \tag{6.35}$$

Remark 6.1. The improved updating rule (6.34) with the reinforced term relaxes the demand for initial admissible control, which implies that the critic weight vector can initially be set as any random vector.

6

6.3.2 Closed-loop stability analysis

We firstly construct the error dynamics of the critic network by defining $\tilde{w}_c = w_c - \hat{w}_c$ and finding that $\dot{\tilde{w}}_c = -\dot{\hat{w}}_c$. Consequently, the critic error dynamics is presented as:

$$\dot{\tilde{w}}_c = -\eta_c\phi(\phi^{\mathrm{T}}\tilde{w}_c - e_{cH}) - \frac{1}{2}\eta_s\Xi(x,\hat{u}(x_k))\nabla\sigma(x_k)g(x_k)R^{-1}(\underline{1}-\tanh^2(\hat{D}_k))g^{\mathrm{T}}(x)\nabla J_s(x). \tag{6.36}$$

Remark 6.2. The persistent excitation (PE) assumption is required. If the PE condition holds, we easily derive $\underline{\lambda}(\phi\phi^{\mathrm{T}}) > 0$ [158], which is of great significance for stability analysis. A common approach to achieve PE is introducing a probing noise to excite the system [59, 136, 163].

Subsequently, we study the closed-loop stability based on the approximate event-triggered feedback control incorporating the weight estimation dynamics. Before proceeding, the following assumptions are required, which are commonly employed in ADP literature, such as [91, 136, 151, 157].

Assumption 6.2. *$g(x)$ is Lipschitz continuous rendering $\|g(x)-g(x_k)\| \le L_g\|e_k\|$, and is upper bounded as $\|g(x)\| \le b_g$, where L_g are b_g are positive real constants.*

Assumption 6.3. *Denote $L_{\nabla\sigma_c}$, $b_{\nabla\sigma_c}$, $b_{\nabla\varepsilon_c}$, $b_{\varepsilon_{u^*}}$, and $b_{e_{cH}}$ as positive real constants. $\nabla\sigma_c(x)$ is Lipschitz continuous guaranteeing $\|\nabla\sigma_c(x)-\nabla\sigma_c(x_k)\| \le L_{\nabla\sigma_c}\|e_k\|$. $\nabla\sigma_c(x)$, $\nabla\varepsilon(x)$, $\varepsilon_{u_k^*}$, and e_{cH} are all upper bounded, such that $\|\nabla\sigma_c(x)\| \le b_{\nabla\sigma_c}$, $\|\nabla\varepsilon(x)\| \le b_{\nabla\varepsilon_c}$, $\|\varepsilon_{u_k^*}\| \le b_{\varepsilon_{u^*}}$, and $|e_{cH}| \le b_{e_{cH}}$.*

Theorem 6.1. *Considering Assumptions 6.1-6.3, utilizing the formula* (6.34) *to update the critic network, and with the event-triggered approximate optimal control policy* (6.26)*, the*

closed-loop system is asymptotically stable while the weight error dynamics is ultimately uniformly bounded (UUB) if

$$\|e_k\|^2 \leq \frac{(1-\eta)\underline{\lambda}(Q)\|x\|^2 + Y(u^*(x_k))}{C_1\|\hat{w}_c\|^2} \triangleq \|\hat{e}_T\|^2, \tag{6.37}$$

and $\|\tilde{w}_c\|^2 \geq \mathcal{W}$*, where* $\eta \in (0,1)$*,* C_1 *and* $\mathcal{W}$ *are given in the proof.*

Proof. We construct a Lyapunov function candidate as:

$$\mathcal{L} = L_x + L_{x_k} + L_{\tilde{w}_c} + L_{J_s}, \tag{6.38}$$

where $L_x = J^*(x)$, $L_{x_k} = J^*(x_k)$, $L_{\tilde{w}_c} = (1/2)\tilde{w}_c^{\mathsf{T}}\tilde{w}_c$, and $L_{J_s} = \eta_s J_s(x)$. The proof consists of two situations conforming to whether the event is triggered or not.

Situation 1: the events are not triggered, i.e., $\forall t \in [s_k, s_{k+1})$. Computing the time derivative of the Lyapunov function, the second term is $\dot{L}_{x_k} = 0$.

Considering the closed loop system using the approximate feedback control (6.26), and the optimal HJB equation (6.14), the first term can be derived as:

$$\begin{aligned}\dot{L}_x =& \dot{J}^*(x) = \nabla J^{*\mathsf{T}}(x)[f(x) + g(x)\hat{u}(x_k)] \\ =& -x^{\mathsf{T}}Qx - u_b^2\underline{R}\ln(\underline{\mathbf{1}} - \tanh^2(D^*)) + \nabla J^{*\mathsf{T}}(x)g(x)\hat{u}(x_k).\end{aligned} \tag{6.39}$$

According to the definition of the utility function, the second term of (6.39) is converted into:

$$u_b^2\bar{R}\ln(\underline{\mathbf{1}} - \tanh^2(D^*)) = \int_{\hat{u}(x_k)}^{u^*(x)} 2u_b\tanh^{-\mathsf{T}}(v/u_b)R\mathrm{d}v + Y(\hat{u}(x_k)) - u_b\nabla J^{*\mathsf{T}}(x)g(x)\tanh(D^*). \tag{6.40}$$

Besides, (6.11) and (6.12) imply that:

$$\nabla J^{*\mathsf{T}}(x_k)g(x) = -2u_b\tanh^{-\mathsf{T}}(u^*(x)/u_b)R. \tag{6.41}$$

Therefore, the last term in (6.39) can be rewritten as:

$$\nabla J^{*\mathsf{T}}(x)g(x)\hat{u}(x_k) = \int_{u^*(x)}^{\hat{u}(x_k)} 2u_b D^{*\mathsf{T}}(x)R\mathrm{d}v - u_b\nabla J^{*\mathsf{T}}(x)g(x)\tanh(D^*(x)). \tag{6.42}$$

Substituting (6.40) and (6.42) into (6.39) yields:

$$\dot{\mathcal{L}}_x = -x^{\mathsf{T}}Qx - Y(\hat{u}(x_k)) + \int_{u^*(x)}^{\hat{u}(x_k)} 2u_b[\tanh^{-1}(v/u_b) + D^*]^{\mathsf{T}}R\mathrm{d}v. \tag{6.43}$$

Letting $v = -u_b\tanh(\omega)$, the last term in (6.43) is written as:

$$\begin{aligned}\int_{u^*(x)}^{\hat{u}(x_k)} 2u_b[\tanh^{-1}(v/u_b) + D^*]^{\mathsf{T}}R\mathrm{d}v &= \int_{D^*}^{\hat{D}_k} 2u_b^2(\omega - D^*)^{\mathsf{T}}R[1-\tanh^2(\omega)]\mathrm{d}\omega \\ &\leq u_b^2(\hat{D}_k - D^*)^{\mathsf{T}}R(\hat{D}_k - D^*).\end{aligned} \tag{6.44}$$

Therefore, (6.43) satisfies:

$$\dot{L}_x \le -x^{\mathrm{T}}Qx - Y(\hat{u}(x_k)) + \frac{1}{4}\underline{\lambda}(R)\|g^{\mathrm{T}}(x_k)\nabla\hat{J}(x_k) - g^{\mathrm{T}}(x)\nabla J^*(x)\|^2, \tag{6.45}$$

where $\nabla\hat{J}(x_k) = \nabla\sigma^{\mathrm{T}}(x_k)\hat{w}_c$. According to Assumptions 6.2 and 6.3, we obtain:

$$\begin{aligned}
&\|g^{\mathrm{T}}(x_k)\nabla\hat{J}(x_k) - g^{\mathrm{T}}(x)\nabla J^*(x)\|^2 \\
&=\|(g^{\mathrm{T}}(x)\nabla\sigma_c^{\mathrm{T}}(x) - g^{\mathrm{T}}(x_k)\nabla\sigma_c^{\mathrm{T}}(x_k))\hat{w}_c + g^{\mathrm{T}}(x)(\nabla\sigma_c^{\mathrm{T}}(x)\tilde{w}_c + \nabla\varepsilon_c(x))\|^2 \\
&\le 2\|\nabla\sigma_c(x)g(x) - \nabla\sigma_c(x_k)g(x_k)\|^2\|\hat{w}_c\|^2 + 2b_g^2(b_{\nabla\sigma_c}^2\|\tilde{w}_c\|^2 + b_{\nabla\varepsilon_c}^2),
\end{aligned} \tag{6.46}$$

in which

$$\begin{aligned}
&\|\nabla\sigma_c(x)g(x) - \nabla\sigma_c(x_k)g(x_k)\|^2 \\
&\le 2\|\nabla\sigma_c(x)(g(x) - g(x_k))\|^2 + 2\|(\nabla\sigma_c(x) - \nabla\sigma_c(x_k))g(x_k)\|^2 \\
&\le 2(b_{\nabla\sigma_c}^2 L_g^2 + L_{\nabla\sigma_c}^2 b_g^2)\|e_k\|^2.
\end{aligned} \tag{6.47}$$

Therefore (6.45) continues as:

$$\dot{L}_x \le C_1\|\hat{w}_c\|^2\|e_k\|^2 + C_2\|\tilde{w}_c\|^2 + C_3 - Y(\hat{u}(x_k)) - \eta\underline{\lambda}(Q)\|x\|^2 - (1-\eta)\underline{\lambda}(Q)\|x\|^2, \tag{6.48}$$

where $C_1 = \underline{\lambda}(R)(b_{\nabla\sigma_c}^2 L_g^2 + L_{\nabla\sigma_c}^2 b_g^2)$, $C_2 = \frac{1}{2}\underline{\lambda}(R)b_g^2 b_{\nabla\sigma_c}^2$, and $C_3 = \frac{1}{2}\underline{\lambda}(R)b_g^2 b_{\nabla\varepsilon_c}^2$.

Then, we investigate the last two terms in (6.38). By taking the definition of $\Xi(x, \hat{u}(x_k))$ into consideration, two scenarios are examined separately.

I: $\Xi(x, \hat{u}(x_k)) = 0$. We have $\dot{L}_{J_s} < 0$ and

$$\dot{L}_{\tilde{w}_c} = -\eta_c\tilde{w}_c^{\mathrm{T}}\phi(\phi^{\mathrm{T}}\tilde{w}_c + e_{cH}) \le -C_4\|\tilde{w}_c\|^2 + C_5, \tag{6.49}$$

where $C_4 = \frac{1}{2}\eta_c\underline{\lambda}(\phi\phi^{\mathrm{T}})$ and $C_5 = \frac{1}{2}\eta_c e_{cH}^2$. Combining (6.48) and (6.49), we observe:

$$\begin{aligned}
\dot{\mathcal{L}} =&\dot{L}_x + \dot{L}_{\tilde{w}_c} \\
\le& C_1\|\hat{w}_c\|^2\|e_k\|^2 + C_2\|\tilde{w}_c\|^2 + C_3 - C_4\|\tilde{w}_c\|^2 + C_5 - \eta\underline{\lambda}(Q)\|x\|^2 - (1-\eta)\underline{\lambda}(Q)\|x\|^2 - Y(\hat{u}(x_k)).
\end{aligned} \tag{6.50}$$

By choosing an appropriate η_c, we can achieve $C_4 > C_2$. Therefore, when $\|\tilde{w}_c\|^2 \ge (C_3 + C_5)/(C_4 - C_2) \triangleq \mathcal{W}_1$ and $\|e_k\|^2 < \|\hat{e}_T\|^2$, we have $\dot{\mathcal{L}} < 0$, $\forall x \ne 0$.

II: $\Xi(x, \hat{u}(x_k)) = 1$. We combine $\dot{L}_{J_s}$ with the stabilization term of $\dot{L}_{\tilde{w}_c}$ as:

$$\begin{aligned}
\dot{L}' =&\dot{L}_{J_s} + \dot{L}'_{\tilde{w}_c} \\
=&\eta_s\nabla J_s^{\mathrm{T}}(x)[f(x) + g(x)\hat{u}(x_k)] - \frac{1}{2}\eta_s\tilde{w}_c^{\mathrm{T}}\nabla\sigma(x_k)g(x_k)R^{-1}(\underline{1} - \tanh^2(\hat{D}_k))g^{\mathrm{T}}(x)\nabla J_s(x) \\
=&\eta_s\nabla J_s^{\mathrm{T}}(x)f(x) - \eta_s\nabla J_s^{\mathrm{T}}(x)g(x)[u_b\tanh(\hat{D}_k) + \frac{1}{2}(\underline{1} - \tanh^2(\hat{D}_k))R^{-1}g^{\mathrm{T}}(x_k)\sigma^{\mathrm{T}}(x_k)\tilde{w}_c].
\end{aligned} \tag{6.51}$$

By taking the first-order Taylor series expansion of $\tanh(D_k)$, we obtain:

$$\tanh(D_k) = \frac{1}{2u_b}(\underline{1} - \tanh^2(\hat{D}_k))R^{-1}g^{\mathrm{T}}(x_k)\sigma^{\mathrm{T}}(x_k)\tilde{w}_c + \tanh(\hat{D}_k) + o[(D_k - \hat{D}_k)^2], \tag{6.52}$$

where $o[(D_k - \hat{D}_k)^2]$ has a bound, which is denoted by b_{oD} [91]. Substituting (6.52) into (6.25) and then (6.51) yields:

$$\begin{aligned}\dot{L}' =&\eta_s \dot{J}_s^*(x) - \eta_s \nabla J_s^{\mathrm{T}}(x) g(x) \varepsilon_{u_k^*} + \eta_s \nabla J_s^{\mathrm{T}}(x) g(x) u_b o[(D_k - \hat{D}_k)^2] \\ \leq& -\eta_s \underline{\lambda}(M) \|\nabla J_s(x)\|^2 + \eta_s C_6 \|\nabla J_s(x)\| \\ =& -\eta_s \underline{\lambda}(M) \left(\|\nabla J_s(x)\| - \frac{C_6}{2\underline{\lambda}(M)} \right)^2 + C_7,\end{aligned} \tag{6.53}$$

where $C_6 = b_g(b_{\varepsilon_{u^*}} + u_b b_{oD})$, and $C_7 = \eta_s C_6^2/(4\underline{\lambda}(M))$. Accordingly, we obtain:

$$\begin{aligned}\dot{\mathcal{L}} =&\dot{L}_x + \dot{L}_{\tilde{w}_c} + \dot{L}_{J_s} \\ \leq& -\eta \underline{\lambda}(Q) \|x\|^2 - (1-\eta) \underline{\lambda}(Q) \|x\|^2 - Y(\hat{u}(x_k)) \\ &+ C_1 \|\hat{w}_c\|^2 \|e_k\|^2 + C_2 \|\tilde{w}_c\|^2 + C_3 - C_4 \|\tilde{w}_c\|^2 \\ &+ C_5 - \eta_s \underline{\lambda}(M) \left(\|\nabla J_s(x)\| - \frac{C_6}{2\underline{\lambda}(M)} \right)^2 + C_7.\end{aligned} \tag{6.54}$$

Therefore, when $\|\tilde{w}_c\|^2 \geq (C_3 + C_5 + C_7)/(C_4 - C_2) \triangleq \mathcal{W}_2$ and $\|e_k\|^2 < \|\hat{e}_T\|^2$, we have $\dot{\mathcal{L}} < 0$, $\forall x \neq 0$.

Consequently, combining I and II, we can conclude that, if the event is not triggered, i.e., $\|e_k\|^2 < \|\hat{e}_T\|^2$, if $\|\tilde{w}_c\|^2 \geq \max\{\mathcal{W}_1, \mathcal{W}_2\} \triangleq \mathcal{W}$ is satisfied, then we have $\dot{\mathcal{L}} < 0$, $\forall x \neq 0$.

Situation 2: the events are triggered, i.e., $\forall t = s_{k+1}$. Utilizing (6.38), the difference terms are derived as:

$$\begin{aligned}\Delta\mathcal{L} =&\Delta L_x + \Delta L_{x_k} + \Delta L_{\tilde{w}_c} + \Delta L_{J_s} \\ =&J^*(x_{k+1}) - J^*(x(s_{k+1}^-)) + J^*(x_{k+1}) - J^*(x_k) \\ &+ \frac{1}{2} \tilde{w}_c^{\mathrm{T}}(x_{k+1}) \tilde{w}_c(x_{k+1}) - \frac{1}{2} \tilde{w}_c^{\mathrm{T}}(x(s_{k+1}^-)) \tilde{w}_c(x(s_{k+1}^-)) + \eta_s J_s(x_{k+1}) - \eta_s J_s(x(s_{k+1}^-)).\end{aligned} \tag{6.55}$$

From Situation 1, we find that if $\|e_k\|^2 \leq \|\hat{e}_T\|^2$ and $\|\tilde{w}_c\|^2 \geq \mathcal{W}$, $\dot{\mathcal{L}} < 0$, $\forall t \in [s_k, s_{k+1})$. Since the states and cost function are both continuous, according to the property of the limit, we have $\Delta L_x + \Delta L_{\tilde{w}_c} + \Delta L_{J_s} < 0$ and then obtain $\Delta\mathcal{L} < \Delta L_{x_k} \leq -\kappa(\|e_{k+1} - e_k\|)$, in which $\kappa(\cdot)$ is a class-κ function [165]. Hence, (6.38) is still decreasing when $\forall t = s_{k+1}$.

In summary, if $\|e_k\|^2 \leq \|\hat{e}_T\|^2$ and $\|\tilde{w}_c\|^2 \geq \mathcal{W}$ hold, we can reach the conclusion that the closed-loop state is asymptotically stable while the critic error dynamics is UUB, which ends the proof. □

6.3.3 Analysis of Zeno phenomenon in the closed-loop system

For nonlinear CT systems with event-triggered control inputs, the inter-execution time is denoted as $\Delta s = s_{k+1} - s_k$, and the minimal inter-execution time $\Delta s_{\min} = \min_{k \in \mathbb{N}}\{s_{k+1} - s_k\}$ might be zero, which can lead to the accumulation of event times, a.k.a., the Zeno phenomenon. Hence, the condition of $\Delta s > 0$ should be guaranteed such that the undesired Zeno phenomenon is avoided.

Theorem 6.2. *Considering the closed-loop form of the nonlinear system* (6.5) *governed by the event-triggered approximate optimal control* (6.26)*, the k-th inter-execution time* Δs_k *determined by* (6.37) *has a lower bound as:*

$$\Delta s_k \geq \frac{1}{k_f} \ln \left(\frac{k_f \|\hat{e}_T\|}{k_f \|e_k\| + b_g u_b} + 1 \right) > 0, \ k \in \mathbb{N}, \tag{6.56}$$

where k_f *is a positive constant.*

Proof. We apply the approximate optimal control (6.26) to formulate the closed-loop dynamics as follows:

$$\dot{x} = f(x) - g(x)\hat{u}(x_k). \tag{6.57}$$

By noticing the fact that $\hat{u}(x_k)$ is upper bounded by u_b, and according to Assumption 6.2, we can derive that:

$$\|\dot{x}\| = \|f(x) - g(x)\hat{u}(x_k)\| \leq k_f \|x\| + b_g u_b. \tag{6.58}$$

Considering (6.15), we can further derive that:

$$\|\dot{e}_k\| \leq k_f \|x_k\| + k_f \|e_k\| + b_g u_b, \ \forall t \in [s_k, s_{k+1}). \tag{6.59}$$

Since $e_k(s_k) = x_k - x(s_k) = 0$, by employing the comparison lemma [151, 165] to solve (6.59), for any $t \in [s_k, s_{k+1})$, we have:

$$\|e_k\| \leq \frac{k_f \|e_k\| + b_g u_b}{k_f} \left(e^{k_f (t - s_k)} - 1 \right). \tag{6.60}$$

Therefore, the k-th inter-execution time Δs_k satisfies:

$$\Delta s_k = s_{k+1} - s_k \geq \frac{1}{k_f} \ln \left(\frac{k_f \|\hat{e}_T\|}{k_f \|e_k\| + b_g u_b} + 1 \right). \tag{6.61}$$

According to (6.37), $\|\hat{e}_T\| > 0$. In summary, $\Delta s_k > 0$ for any $x_k \neq 0$, i.e., $\Delta s_{\min} > 0$, which ends the proof.

□

Overall, the structural diagram of the present control implementation is depicted in Fig. 6.2 to clarify the design procedure.

6.4 Simulation Study

Finally, we verify the effectiveness of the proposed control approach through the numerical simulation experiments based on the nonlinear aeroelastic system demonstrated above. Considering the cost function (6.6) from $t = 0$, we choose $Q = I_5$ and $R = 1$ as a trade-off between fast stabilizing and avoiding aggressive control, and set the deflection constraint as $u_b = 10$ deg. For the purpose of simulation, we set the simulation frequency as 1 kHz whereas the sensing frequency as 100 Hz. Let the initial state vector be $x_0 = [-0.01 \text{ m}, 10 \text{ deg}, 0, 0, 0]^\mathsf{T}$. In what follows, we implement the event-triggered intelligent critic control with the facilitation of an ANN.

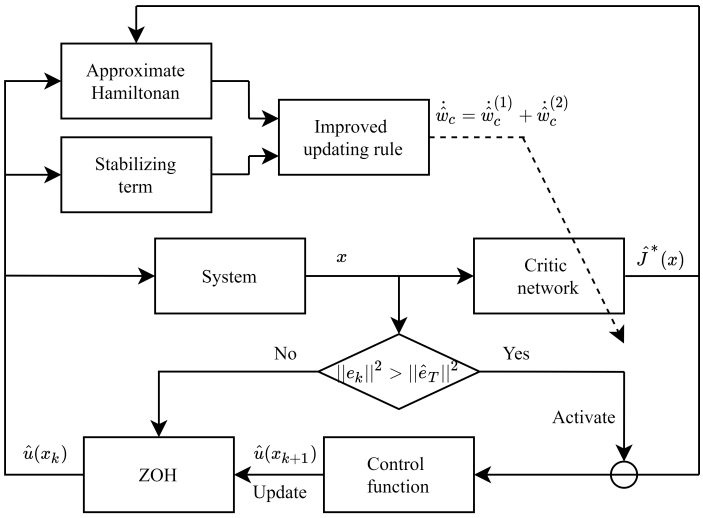


Figure 6.2: Structural diagram of the developed event-triggered ADP control method, where solid lines represent the feed-forward flow of signals and the dashed line is the back-propagation path.

A critic network is constructed to approximate the optimal cost function. The number of neurons and the nonlinearity of the activation function positively correlate with the approximation precision. However, more neurons with higher nonlinearities can also increase the computational load and cause overfitting that harms the control robustness [59]. For balancing control accuracy and computational complexity, we choose the activation function as $\sigma_c(x) = [x_1x_2, x_1x_3, x_1x_4, x_1x_5, x_2x_3, x_2x_4, x_2x_5, x_3x_4, x_3x_5, x_4x_5]^\mathsf{T}$. We choose $J_s(x) = 0.5x^\mathsf{T}x$ to enhance stability and experimentally set $\eta_c = 0.05$, $\eta_s = 0.001$, $\eta = 0.1$, and $C_1 = 250$. As claimed in Remark 6.2, an exploration noise u_e is introduced to satisfy the PE condition. The probing noise is designed as a composition of decaying sinusoidal functions, whose formula is $u_e = -0.05e^{-20t}(\sin^2(100t)\cos(100t) + \sin^2(2t)\cos(0.1t) + \sin^2(1.2t)\cos(0.5t) + \sin^5(t))$ deg. Only at the instant when it is triggered, will u_e really be added to the control command. Since the critic network has different hidden neurons, the weights can initially be set as zero. Recalling the triggering condition in (6.37), we find that $\|\hat{w}_c\|$ appears in the denominator, which can cause a large time interval of control at the beginning. Therefore, we manually set an upper bound for the inter-execution time as $\Delta s_{\max} = 0.1$ s. This configuration is set in the engineering sense for safety guarantee, and does not affect the theoretical completeness.

The simulation is conducted in an online manner, which means that the control policy improves in a closed-loop way. For presenting the advantage of the ETC scheme, a time-based approach is adopted for comparison, whose settings are exactly the 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. We can observe from Fig. 6.3 that the convergence of the weight vector occurs around 1 s. Subsequently, we display the trajectory of the approximated cost function in Fig. 6.4, which presents the direct performance of the controller. Due to the initial zero values of the weight vector, the initial approximate cost is zero, and subsequently grows as the learning continues. Then

because of the convergence of the weight vector, the approximate cost function swiftly decreases to a low level. Furthermore, the triggering threshold trajectory is displayed in Fig. 6.5, which presents a trend to zero along with the event error. The inter-execution time is depicted in Fig. 6.6. 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.

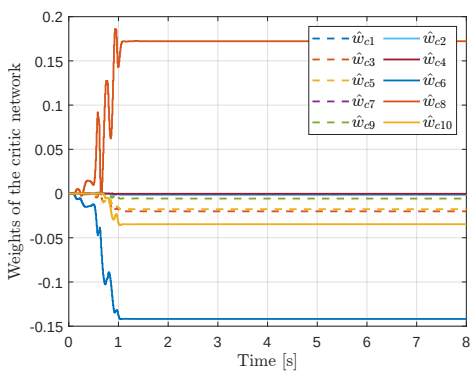


Figure 6.3: Convergence process of the critic weights.

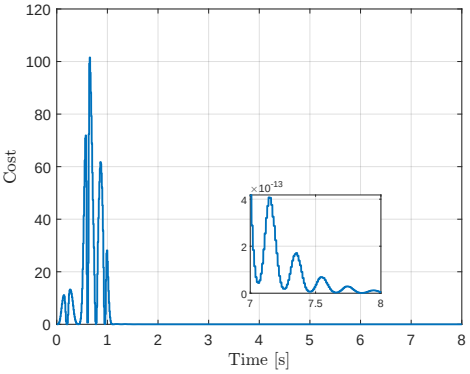


Figure 6.4: Evolution of the approximate cost function.

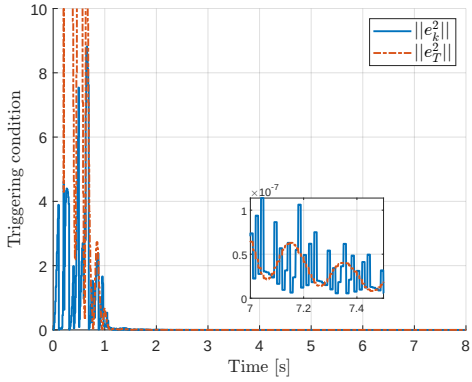


Figure 6.5: Evolution of the triggering condition.

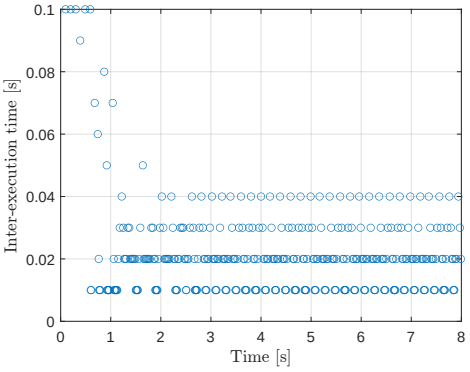


Figure 6.6: Evolution of the inter-execution time.

Figs. 6.7 and 6.8 present the aeroelastic system states trajectory divided into plunge and pitch motion, respectively. We compare the results between the event-triggered and time-based approaches, and observe 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. Figs. 6.9 and 6.10 respectively presents the control command directly generated by the controller, u (β_c), and the real deflection of the control surface, x_5 (β). 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 stepwise. Nevertheless, the control command signal has to go through an actuator and the real deflection is adequately smooth for the wing surface control. Furthermore, we observe that the control command (incorporating exploration noise) is bounded by the pre-designed saturation constraints, i.e., $|u| < u_b$. Therefore, we conclude that the control input constraints problem has been overcome.

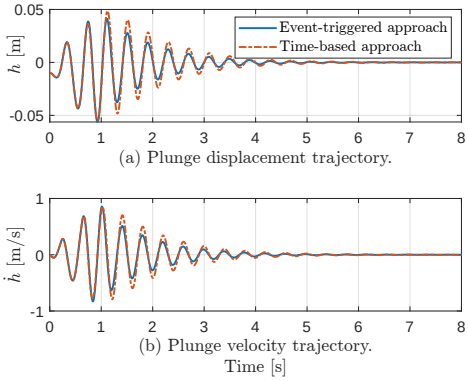


Figure 6.7: Evolution of the plunge motion states.

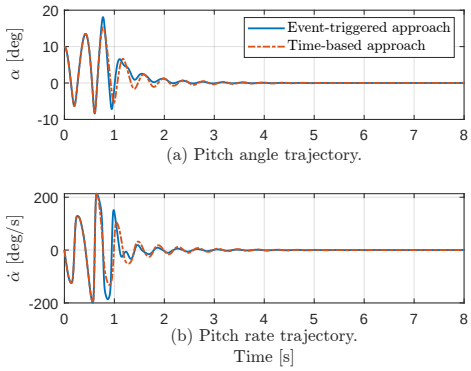


Figure 6.8: Evolution of the pitch motion states.

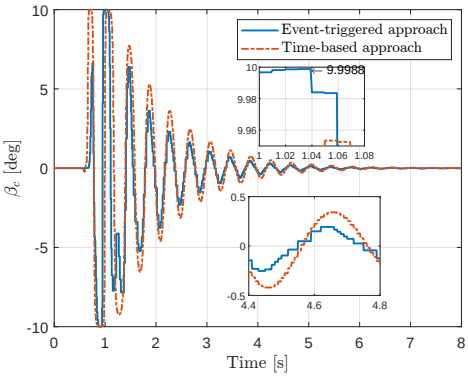


Figure 6.9: Control command generated by the controller.

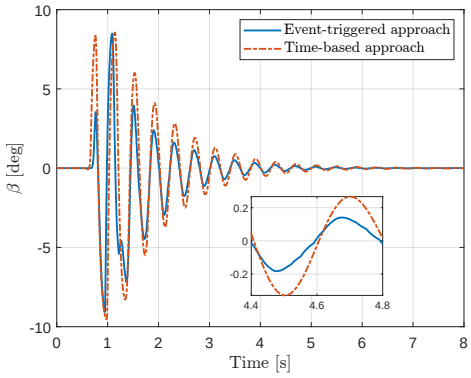


Figure 6.10: Real deflection of the control surface.

The phase portraits of plunge and pitch motions are illustrated in Figs. 6.11 and 6.12, respectively. As can be observed, 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.

To further verify its performance, robustness tests are carried out with different freestream velocities using the proposed event-triggered intelligent optimal control strategy. In addition to nominal velocities, the freestream is also assumed to be disturbed. The uncertain freestream is modeled as a composition of white noise with unit variance and

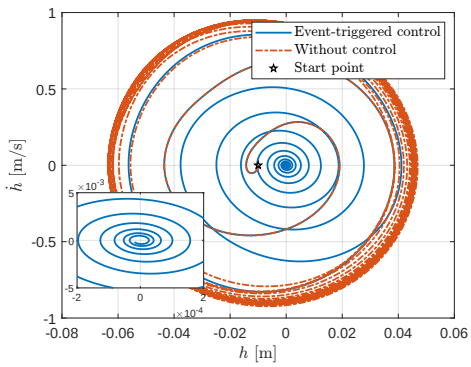


Figure 6.11: Phase portrait of the plunge motion states.

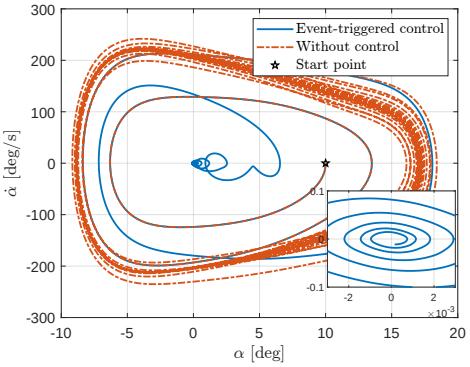


Figure 6.12: Phase portrait of the pitch motion states.

a sinusoidal gust $0.5\cos(4\pi t + 0.3\pi)$ m/s. With other settings unchanged, the results are depicted in Figs. 6.13 - 6.15, in which the nominal freestream velocity U respectively equals to 12 m/s, 15 m/s and 27 m/s.

For simplicity, the plunge displacement is selected as a representative to show the state evolution, as illustrated in Fig. 6.13. It can be observed that in all conditions, the controller manages to stabilize the plunge displacement within 8 s. The situations with and without uncertainties demonstrate similar performance for U = 15 m/s and U = 27 m/s, whereas for U = 12 m/s, the control performance is even better with uncertainties involved. The reason behind this phenomenon lies in that when U = 12 m/s the flutter frequency is low and it is difficult to fully excite the system. The velocity uncertainties disturb the system but meanwhile provide stronger excitation and thus speed up the learning process. In fact, this acceleration phenomenon also takes place in the other two situations though it is more obvious with lower freestream velocity, which illustrates the robustness of the proposed control method to uncertainties. With lower incoming flow speed (U = 12 m/s), the control effectiveness is also lower, and therefore it requires a larger deflection of the control surface to generate sufficient control torque, leading to the aggressive but saturated control command shown in Fig. 6.14 (a). When the freestream velocity is higher (U = 27 m/s), the flutter frequency is higher, which provides more excitations at the initial stage. Therefore, it can be seen from Fig. 6.14 (c) that the control command becomes effective earlier than for the other two conditions. However, it is remarkable that if U < 12 m/s or U > 27 m/s, the controller is capable of stabilizing the system within 8 s with current settings due to the insufficient control effectiveness or the excessive flutter frequency, respectively. Nevertheless, this can be improved by adapting hyperparameters.

Fig. 6.15 compares the root means square (RMS) of critic weights with 3 different freestream velocities in the presence of uncertainties. Consistent with the above, the convergence speed is faster when the freestream velocity is higher because of the stronger excitation. These curves are different in that the control policy is learned online, and therefore the controller adapts to different conditions in real time, which validates the adaptability of the proposed control approach. The simulation results collectively verify the feasibility and effectiveness of the event-triggered intelligent optimal control approach developed in this chapter.

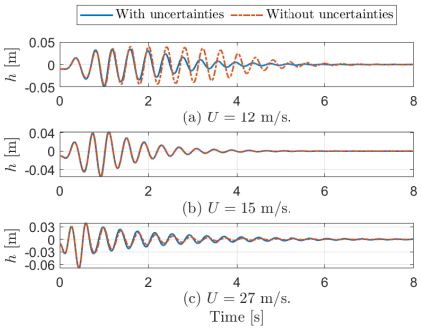


Figure 6.13: Evolution of plunge displacements with different freestream velocities.

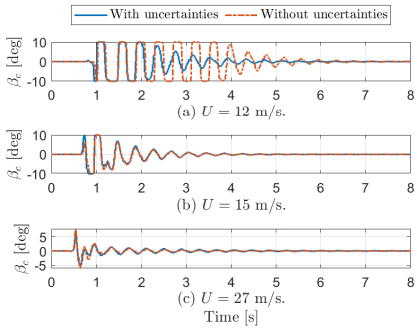


Figure 6.14: Control commands generated by the controller with different freestream velocities.

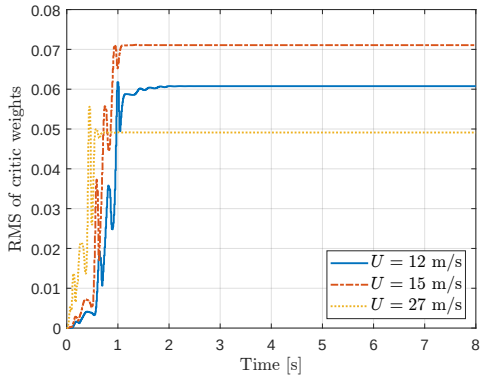


Figure 6.15: Convergence process of the RMS of critic weights with different freestream velocities.

6.5 Conclusion

In this chapter, we develop an event-triggered intelligent optimal control scheme, and apply it to an aeroelastic system control problem. Taking the input constraints into account, we derive a novel triggering condition without making the Lipschitz assumption on the inverse hyperbolic tangent function. The controller is conducted by adopting the adaptive dynamic programming (ADP) technique with a single critic network.

The theoretical analysis of the closed-loop system shows that, with the derived event-triggered controller, the system states can be guaranteed asymptotically stable, while the Zeno phenomenon is also avoided during the learning phase. The simulation results demonstrate that the nonlinear aeroelastic system is successfully stabilized with input saturation constraints handled. Besides, compared to the conventional time-based ADP method, the present event-triggered ADP method can achieve comparable performance with reduced control updates, which presents the advantages of the developed method in saving the computational and communication load. Furthermore, the robustness tests demonstrate that the designed controller is able to adapt online to different situations and has robustness to uncertainties to some extent.

At the current stage, we concentrate on the control algorithm development based on the known system dynamics and perfect measurements. Due to the fact that uncertainties generally exist in the real world, further investigation into robust control methods is recommended. Besides, due to the limitation of actuator power, the deflection rate of the control surface should also be constrained, which can be studied in the future.

7

Conclusions and Recommendations

This dissertation has been dedicated to enhancing the autonomy of flight control systems using reinforcement learning (RL) methods. At the beginning of the dissertation, the development of autonomous control in the aerial vehicle was reviewed, the basic principles and promising applications of RL and adaptive dynamic programming (ADP) were introduced, and some challenges in ADP for flight control were accordingly posed: learning efficiency, control stability and computational load. In view of these challenges, the main research goal of the dissertation was formulated as:

Research goal

To improve the adaptability and online learning capability of flight control systems by designing nonlinear ADP approaches.

To achieve this research goal, a main research question and three sub-questions were raised in Chapter 1. These research questions have been investigated in Chapters 2-6 and their answers will be presented in Sub-section 7.1.1. Based on the findings through implementations, the final conclusions will be drawn in Sub-section 7.1.2. Finally, several recommendations for future work and my personal prospects for the RL research in the flight control field are depicted in Section 7.2.

7.1 Findings and Conclusions

This section summarizes the research findings through implementation and answers the raised research questions, and then the final conclusions of this dissertation are reached.

7.1.1 Answers to research questions

The task of achieving efficient learning presents a challenge for the flight control of unknown, nonlinear systems, particularly in cases where the system is time-varying,

disturbed, or imperfectly measurable. Although there exists some work on adaptive critic designs (ACDs) concerning these aspects, there is no paradigm that synthesizes them, which comes up with the first research sub-question:

> **RQ1**: How to synthesize and improve current online ACDs to cope with dynamic and measurement uncertainties, partial observability, external disturbances and sudden faults?

This research question is addressed by developing the *global dual heuristic programming (GDHP)* approach, incorporating *explicit analytical calculations* and the *incremental model*. ACDs are typically classified as heuristic dynamic programming (HDP), dual heuristic programming (DHP), and GDHP, based on the information used by the critic network [26]. Among them, GDHP combines the information utilized by HDP and DHP with a β parameter to indicate the composition ratio and therefore takes advantage of both approaches. Conventionally, the critic network of GDHP adopts a straightforward form that approximates the cost function and its derivatives simultaneously at its outputs [43, 63]. However, this structure can introduce undesired inconsistent errors because two kinds of outputs share the same network except for the output layer. Furthermore, the back-propagation path of the actor network starting from the approximated cost function is different from that starts from cost derivatives. Consequently, this dissertation focuses on the development of a novel GDHP approach to synthesize current online ACDs.

Chapter 2 proposes an online incremental model-based GDHP with explicit analytical calculations, called IGDHP, to synthesize current online ACDs and enable an online learning controller without prior knowledge of the system dynamics. The critic network of this method only directly approximates the cost function, based on which its derivatives are calculated through the critic network in an explicit analytical way. Therefore, the approximated error of cost derivatives is consistent with the approximated cost error. Furthermore, the back-propagation path of the actor network in this approach always starts from the cost function, but the information on derivatives can also be fully utilized. If $\beta = 1$, it represents pure HDP, whereas if $\beta = 0$, the adjustment of weights relies solely on the computed derivatives, which therefore is equivalent to DHP. This property outperforms the conventional straight form where if β is incompatible with the back-propagation path outset, for example, if $\beta = 0$ and the back-propagation path originates from the cost function, the back-propagation channel of the actor network will be invalid. The incremental model also plays an important role in accelerating online learning by approximating the locally linearized system dynamics. The simulation results of IGDHP on an aerial vehicle attitude tracking control problem demonstrate that compared to GDHP, IGDHP has higher precision, online learning speed, robustness to different initial states and adaptability for fault-tolerant control problems.

Chapter 3 extends the IGDHP method proposed in Chapter 2 by augmenting the incremental model with historical data sets to deal with the partial observability (PO) issue that involves stochastic or time-varying dynamics [49, 50]. Compared to full-state feedback (FSF) and output feedback (OPFB), PO provides a more strict demand for the controller because of the stochastic factors. In the control problem of this chapter, the only observation is the tracking error that incorporates a changing and immeasurable reference signal, which

makes the system partially observable. The incremental model is augmented to build a new system with the observations as the system states. In this way, historical information can be utilized to provide observation transitions. The precision and convergence of the online identification method using the incremental model have also been theoretically proven. Chapter 4 continues the approach of extending the incremental model to cope with an OPFB tracking control problem with input constraints well handled. The presented results indicate that the methods proposed adeptly handle imperfect measurements with precision and adaptability while retaining efficient control, despite unknown and time-varying dynamics, different initial conditions and even sudden faults.

Overall, GDHP with explicit analytical calculations proposed in Chapter 2 is able to synthesize current online ACDs in one paradigm, and the augmented incremental model developed in Chapter 3 and Chapter 4 can adaptively provide precise system transition information to enable an online control. Chapter 4 also performs as a bridge connecting RQ1 and RQ2.

RQ2: How to achieve optimality and stability of ADP for discrete-time (DT) and continuous-time (CT) systems, while dealing with input constraints of aerial vehicles?

This research question is answered through the construction of the enhanced *non-quadratic integral utility function* and the *bounding layer* in the actor network. In Chapter 4 an enhanced utility function is developed, in which the traditional quadratic utility function of the control input part is replaced with a non-quadratic integral function, whose upper limit is the current control input. The integrand is also particularly designed with the facilitation of the inverse of an identity bounding function, such as the hyperbolic tangent function. By solving Hamilton-Jacobi-Bellman (HJB) equation, an analytical solution of control input that is constrained by the bounding function can accordingly be obtained. Due to the bootstrap, an actor network is required by the DT system and the analytical computed control input is supposed to perform as the target control to update the actor network. Therefore, a bounding layer is designed in the actor network before the signal goes to the output, in which the activation function is chosen as the bounding function same as the one used for bounding the target input. The weights between the bounding layer and the output layer are element-wise based on given constraints. The proposed method is eligible for dealing with different constraints for each control channel. The incremental model and the improved GDHP technique support quick online learning and promote the success ratio to more than 99%.

Chapter 5 improves the techniques developed in Chapter 4 by extending the non-quadratic integral utility function to a piece-wise one, and the activation function in the bounding layer is also piece-wise based on the constraint distribution on both sides of the origin. In this way, the proposed algorithm is able to handle asymmetric control input constraints. In Chapter 5, to enhance the control stability, the iterative adaptive critic algorithm is introduced so that multiple iterations are conducted at each time instance to achieve a satisfying performance. The convergence of the iterative adaptive critic algorithm can be guaranteed as long as the iteration is sufficient, which is different from the one update in Chapter 2 - Chapter 4. Furthermore, the calculation procedure of the

cost derivatives in the improved GDHP has been simplified in this chapter, and the concept of explainable GDHP (XGDHP) is for the first time proposed.

Chapter 6 investigates a CT nonlinear system subject to input constraints. By assuming that the measurement is also continuous in time, the bootstrapping property in the DT system is not an issue anymore, and therefore a single critic network is adequate. Consequently, the target control input in Chapters 3 and 4 can directly be introduced to the real system with no need for an actor network, which simplifies the algorithm. An improved update criterion of the critic network is proposed, such that the requirement of the initial admissible control is eliminated. Specifically, in addition to the conventional error function, a second Lyapunov function is defined and utilized for enhancing the computation of the weight gradient based on the update direction. Furthermore, the closed-loop stability incorporating the network approximation error is analyzed and guaranteed in the sense of Lyapunov. The application regarding wing section stabilization demonstrates the proposed method can achieve both optimality and stability with input constraints well handled.

Overall, to deal with input constraints, Chapter 4 provides a paradigm that includes the enhanced non-quadratic integral utility function and the bounding layer in the actor network. Especially, for CT systems only the integral utility function is required as presented in Chapter 6. The combination of incremental model and improved GDHP can promote the success ratio and the iterative adaptive critic algorithm with multiple iterations in Chapter 5 can further enhance the stability. For the CT system, the closed-loop stability is theoretically proven and the improved update criterion developed in Chapter 6 is adopted to enhance the network update.

7

RQ3: How to save computational and communication load by event-triggered ADP without significantly affecting overall performance?

In Chapter 5 it is the first time that ETC is combined with XGDHP for DT systems. ETC performs as the outer loop of the XGDHP, and therefore the networks are not updated until an event is triggered. Although for DT systems, the triggering condition usually takes the same formulation, Chapter 5 decreases the amount of the required assumptions to guarantee the stability of the triggering condition and provides a more specific proof compared to existing literature. The proposed method has been verified on general nonlinear systems to demonstrate its advantage of saving computation and communication load.

Chapter 6 introduces the ETC scheme into CT systems in combination with the ADP algorithm. Different from DT systems, ETC is incorporated in ADP for CT systems. In Chapter 6 a novel triggering condition considering input constraints is derived without requiring the Lipschitz assumption on the inverse hyperbolic tangent function. ETC for CT systems may suffer from the Zeno phenomenon where the triggering interval is no more than 0 and infinite events can be encountered. Therefore, Chapter 6 conducts the rigorous theoretical analysis of the Zeno phenomenon and avoids it concerning the derived triggering condition. Furthermore, the closed-loop stability of the ETC scheme is theoretically proven and verified through simulation on an aeroelastic system. The results demonstrate that event-triggered ADP can produce stair-wise control commands, which however are smoothed by the filter-like actuator.

Overall, the ETC scheme respectively developed for DT and CT systems in Chapters 5 and 6 can save computational and communication load and meanwhile can achieve comparable control performance with common ADP approaches.

7.1.2 Final conclusions

In conclusion, several ADP approaches have been developed to improve the autonomy and online learning of aerial vehicles. Therefore, the research goal is achieved by answering the main research question from 3 different aspects:

Main research question

How can aerial vehicles benefit from ADP to improve autonomy and online learning in an uncertain environment, while satisfying requirements on input constraints, adaptability, stability and computational efficiency?

Improve the leaning efficiency

- The GDHP approach is improved with explicit analytical calculations, which can take away the inconsistencies in the existing literature and make the method more explainable.
- The incremental model is used to online identify the locally linear dynamics despite output feedback (OPFB) and partial observability (PO), which provides precise state transitions.

Enhance the control stability

- The developed algorithms generally outperform existing incremental model-based ACDs for flight control, significantly improving the success ratio. In the specific research cases of this dissertation, the success ratio has been enhanced to over 99%, compared to 86.7% for IHDP [42] and 95.5% for IDHP [34].
- Two approaches without online identification are proposed in Chapters 5 and 6. These approaches ensure stability in the sense of Lyapunov despite input constraints and event-triggered applications, as guaranteed by theoretical analysis.
- An improved update criterion is developed to eliminate the requirement for initial admissible control while facilitating the weight update.

Reduce computational load

- The ETC scheme is introduced to DT and CT ADP approaches, and the triggering condition is designed, respectively. For CT systems, the Zeno phenomenon is also theoretically avoided.

These benefits can support ADP flight controllers to achieve higher autonomy despite uncertainties and constraints. Moreover, this dissertation bridges the gap between method development and application scenarios in the ADP field and builds a link between the aerospace community and the intelligent control community.

7.2 Recommendations and Outlook

As an active research area, online intelligent flight control keeps attracting attention in the future and reinforcement learning will play an important role. Consequently, this section provides some potential areas for exploration and gives some personal prospects for this area.

7.2.1 Recommendations for future study

Although aerial vehicles can benefit from the proposed methods to improve autonomy and online learning in an uncertain environment, while taking input constraints, adaptability, stability and computational efficiency into consideration, there is still much space for improvement before a realistic flight. The developed approaches are also not limited to the application scenarios in this dissertation. Hereby the following insights for further research are provided:

- This dissertation concentrates on the online learning ability of aerial vehicles. At each run, the control policy is learnt from scratch. Nevertheless, offline learning is often more stable and sometimes can provide initial admissible control for online learning, which can accelerate online convergence. Consequently, the combination with offline training techniques such as imitation learning [166] is recommended.

- Similar to other intelligent algorithms, the approaches developed in this dissertation require well-tuned hyperparameters to achieve satisfying performance. In this dissertation, both grid search and manual search are utilized. However, manual work requires a significant amount of effort in the development process. Therefore, exploring autonomous hyperparameter-tuning techniques, such as using tools like random search [167], Bayesian optimisation [168], genetic algorithms [169], or automated machine learning (AutoML) platforms [170], could be an interesting avenue for further research.

- In this dissertation ADP serves as the main and only control algorithm. It is also possible to combine ADP with other classic control approaches, such as PID [171], MPC [172] and NDI [173] because these well-known approaches have demonstrated their stability and swift action in practical applications.

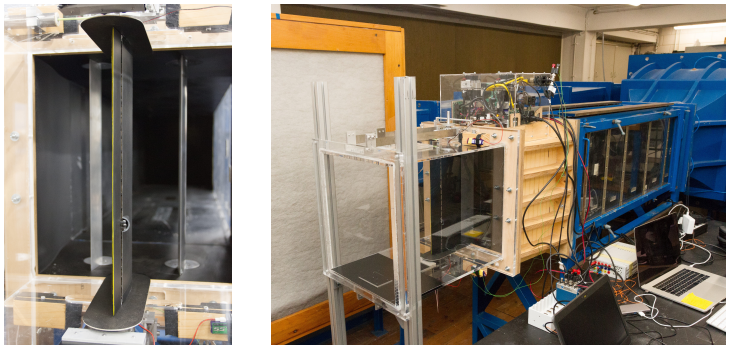
- In this dissertation, only low-level control problems are investigated because the low-level control capability lays a foundation for high-level guidance and navigation of aerial vehicles. However, reinforcement learning and ADP are not limited to low-level control. In combination with various artificial neural networks (ANNs), RL and ADP have the potential to deal with complex high-level tasks, especially for off-policy RL methods, such as soft actor-critic (SAC) [174, 175] and deep deterministic policy gradient (DDPG) [176].

- A cascaded actor network is adopted in Chapter 2 to make use of the physical information. This hierarchical idea can be further extended to multiple levels of guidance, navigation and control (GNC) in aerial vehicles.

- The efficacy of the incremental model has been demonstrated through an experiment [3] on the morphing wing shown in Fig. 1.1d. In this experiment, a NDI controller is employed. To validate the performance of ADP methods proposed in this dissertation, two experiments have been carried out on a wing section with the Low-Turbulence Tunnel at TU Delft, and the experimental equipment is shown in Fig. 7.1. Although the experimental results met expectations, we noticed that due to the weight and structure limitations, the power of current ADP methods has been limited. However, the most promising theoretical and technical solution for autonomous flight GNC may rely on complicated ANNs that can capture complex features. Therefore, the integration of more sampling and computation-efficient approaches may move RL for aerial vehicles from the lab to the real world.

- Only a single agent is considered in this dissertation. In fact, RL and ADP are eligible for solving multi-agent problems, which sometimes can incorporate game theory. Therefore, further research on information control or swarm using the developed approaches is recommended.

- Although the approaches in this dissertation are particularly developed for aerial vehicles, they are all general approaches that can be applied to general nonlinear systems without complicated transformations. It is therefore suggested to apply them to other physical systems for an extension.



(a) Aeroelastic system (b) Whole experimental equipment

Figure 7.1: Wing section (aeroelastic system) and Low-Turbulence Tunnel. All computations of control law and data transmissions are carried out on an embedded microcontroller attached to the wing section, which shows limited computational capacity. In these experiments, both DT and CT ADP approaches proposed in this dissertation have been tested. Publications about these experiments are expected.

7.2.2 Prospects for Research Field

As we embark on a journey towards the future of Online Intelligent Flight Control using Reinforcement Learning (RL), my vision is one where advanced technology and human expertise converge to create safer, more efficient, and sustainable aerial vehicles.

I see a future where intelligent flight control systems are not only able to analyze vast amounts of data but also interpret and act on it in real time. These systems will be equipped with advanced algorithms and sensors that can detect and respond to any changes in the flight environment, making aerial vehicles more reliable. In this vision, RL plays a crucial role in training intelligent flight control systems and allows them to learn from experience and adapt to new situations. With this approach, flight control systems can make more informed decisions about how to control aircraft.

In addition, I expect that autonomous flight control systems will be able to collaborate with each other in real time. These systems will be able to learn from each other's experiences and adapt their behavior accordingly. By incorporating RL algorithms into the decision-making of multiple agents, we can improve the efficiency of aerial vehicle flow and reduce congestion.

As we move forward, I also anticipate that autonomous flight control systems will extend beyond individual aircraft and into other areas of aviation such as air traffic control, aircraft maintenance or specific practical tasks including inspection and rescue. This will require new training programs and protocols for aviation professionals that incorporate intelligent flight control systems, ensuring that they are able to work effectively alongside these systems. I also expect the integration of explainable AI will be essential to ensure transparency and understanding of the decisions made by these advanced systems.

Overall, I believe that the future of Online Intelligent Flight Control using Reinforcement Learning holds tremendous potential. With continued investment in research and development, we can unlock the full potential of RL and will see significant advancements in this field in the years to come. My hope is that these advancements will lead to safer, more efficient, and environmentally conscious aerial vehicles that benefit society as a whole.

Bibliography

References

[1] M. Karásek, F. T. Muijres, C. De Wagter, B. D. Remes, and G. C. de Croon, "A tailless aerial robotic flapper reveals that flies use torque coupling in rapid banked turns," *Science*, vol. 361, no. 6407, pp. 1089–1094, 2018.

[2] T. Mkhoyan, N. R. Thakrar, R. De Breuker, and J. Sodja, "Design of a smart morphing wing using integrated and distributed trailing edge camber morphing," in *Smart Materials, Adaptive Structures and Intelligent Systems*, vol. 84027, p. V001T04A023, American Society of Mechanical Engineers, 2020.

[3] B. Sun, T. Mkhoyan, E.-J. van Kampen, R. De Breuker, and X. Wang, "Vision-based nonlinear incremental control for a morphing wing with mechanical imperfections," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 6, pp. 5506–5518, 2022.

[4] Electric VTOL News, "Airbus CityAirbus." https://evtol.news/airbus-helicopters/. Accessed: 20-06-2022.

[5] Y. Chen, H. Wang, E. F. Helbling, N. T. Jafferis, R. Zufferey, A. Ong, K. Ma, N. Gravish, P. Chirarattananon, M. Kovac, *et al.*, "A biologically inspired, flapping-wing, hybrid aerial-aquatic microrobot," *Science robotics*, vol. 2, no. 11, p. eaao5619, 2017.

[6] L. Burrows, "New robobee flies, dives, swims, and explodes out the of water." https://wyss.harvard.edu/news/new-robobee-flies-dives-swims-and-explodes-out-the-of-water. Accessed: 22-06-2022.

[7] Y. Zhou, *Online reinforcement learning control for aerospace systems.* PhD thesis, Delft University of Technology, 2018.

[8] X. Wang, S. Sun, E.-J. van Kampen, and Q. P. Chu, "Quadrotor fault tolerant incremental sliding mode control driven by sliding mode disturbance observers," *Aerospace Science and Technology*, vol. 87, pp. 417–430, 2019.

[9] W. Su and C. E. Cesnik, "Dynamic response of highly flexible flying wings," *AIAA Journal*, vol. 49, no. 2, pp. 324–339, 2011.

[10] T. E. Noll, J. M. Brown, M. E. Perez-Davis, S. D. Ishmael, G. C. Tiffany, and M. Gaier, "Investigation of the helios prototype aircraft mishap volume I mishap report," tech. rep., NASA Langley Research Center, 2004.

[11] M. Pachter and P. R. Chandler, "Challenges of autonomous control," *IEEE Control Systems Magazine*, vol. 18, no. 4, pp. 92–97, 1998.

[12] H. Chen, X.-m. Wang, and Y. Li, "A survey of autonomous control for uav," in *2009 International Conference on Artificial Intelligence and Computational Intelligence*, vol. 2, pp. 267–271, IEEE, 2009.

[13] J. D. Lee, "Perspectives on automotive automation and autonomy," *Journal of Cognitive Engineering and Decision Making*, vol. 12, no. 1, pp. 53–57, 2018.

[14] J. Junell, *An Empirical Approach to Reinforcement Learning for Micro Aerial Vehicles*. PhD thesis, Delft University of Technology, 2018.

[15] European Union Aviation Safety Agency (EASA), "Artificial intelligence roadmap: A human-centric approach to ai in aviation," tech. rep., European Union Aviation Safety Agency (EASA), 2020.

[16] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2 ed., 2018.

[17] P. J. Antsaklis, K. M. Passino, and S. J. Wang, "An introduction to autonomous control systems," *IEEE Control Systems Magazine*, vol. 11, no. 4, pp. 5–13, 1991.

[18] D. Bertsekas, *Dynamic programming and optimal control*. Athena scientific, 4 ed., 2012.

[19] R. Bellman, *Dynamic Programming*. Dover Publications Inc., 2003.

[20] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[21] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE circuits and systems magazine*, vol. 9, no. 3, pp. 32–50, 2009.

[22] S. G. Khan, G. Herrmann, F. L. Lewis, T. Pipe, and C. Melhuish, "Reinforcement learning and optimal adaptive control: An overview and implementation examples," *Annual reviews in control*, vol. 36, no. 1, pp. 42–59, 2012.

[23] F. L. Lewis and D. Liu, *Reinforcement learning and approximate dynamic programming for feedback control*. John Wiley & Sons, 2013.

[24] W. B. Powell, *Approximate Dynamic Programming: Solving the curses of dimensionality*, vol. 703. John Wiley & Sons, 2007.

[25] D. Wang, H. He, and D. Liu, "Adaptive critic nonlinear robust control: A survey," *IEEE transactions on cybernetics*, vol. 47, no. 10, pp. 3429–3451, 2017.

[26] D. V. Prokhorov and D. C. Wunsch, "Adaptive critic designs," *IEEE transactions on Neural Networks*, vol. 8, no. 5, pp. 997–1007, 1997.

[27] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear hjb solution using approximate dynamic programming: Convergence proof," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 4, pp. 943–949, 2008.

[28] D. P. Bertsekas and J. N. Tsitsiklis, "Neuro-dynamic programming: an overview," in *Proceedings of 1995 34th IEEE conference on decision and control*, vol. 1, pp. 560–564, IEEE, 1995.

[29] R. Enns and J. Si, "Neuro-dynamic programming applied to helicopter flight control," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, p. 4280, AIAA, 2000.

[30] B. Lincoln and A. Rantzer, "Relaxing dynamic programming," *IEEE Transactions on Automatic Control*, vol. 51, no. 8, pp. 1249–1260, 2006.

[31] I. Grondman, *Online model learning algorithms for actor-critic control.* PhD thesis, Delft University of Technology, 2015.

[32] J. Si, A. G. Barto, W. B. Powell, and D. Wunsch, *Handbook of learning and approximate dynamic programming*, vol. 2. John Wiley & Sons, 2004.

[33] A. P. Valadbeigi, A. K. Sedigh, and F. L. Lewis, "H∞ static output-feedback control design for discrete-time systems using reinforcement learning," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 2, pp. 396–406, 2019.

[34] Y. Zhou, E.-J. van Kampen, and Q. P. Chu, "Incremental model based online dual heuristic programming for nonlinear adaptive control," *Control Engineering Practice*, vol. 73, pp. 13–25, 2018.

[35] T. Mannucci, *Safe Online Robust Exploration for Reinforcement Learning Control of Unmanned Aerial Vehicles.* PhD thesis, Delft University of Technology, 2017.

[36] S. Ferrari and R. F. Stengel, "Online adaptive critic flight control," *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 5, pp. 777–786, 2004.

[37] A. Heydari and S. N. Balakrishnan, "Finite-horizon control-constrained nonlinear optimal control using single network adaptive critics," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 1, pp. 145–157, 2013.

[38] R. S. Sutton, A. G. Barto, and R. J. Williams, "Reinforcement learning is direct adaptive optimal control," *IEEE control systems magazine*, vol. 12, no. 2, pp. 19–22, 1992.

[39] E.-J. van Kampen, Q. P. Chu, and J. Mulder, "Continuous adaptive critic flight control aided with approximated plant dynamics," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, p. 6429, 2006.

[40] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous deep q-learning with model-based acceleration," in *International conference on machine learning*, pp. 2829–2838, PMLR, 2016.

[41] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[42] Y. Zhou, E.-J. van Kampen, and Q. P. Chu, "Launch vehicle adaptive flight control with incremental model based heuristic dynamic programming," in *68th International Astronautical Congress (IAC), Adelaide, Australia*, 2017.

[43] D. Liu, D. Wang, D. Zhao, Q. Wei, and N. Jin, "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*, vol. 9, pp. 628–634, July 2012.

[44] S. Bhasin, R. Kamalapurkar, M. Johnson, K. G. Vamvoudakis, F. L. Lewis, and W. E. Dixon, "A novel actor–critic–identifier architecture for approximate optimal control of uncertain nonlinear systems," *Automatica*, vol. 49, no. 1, pp. 82–92, 2013.

[45] D. Liu, Y. Huang, D. Wang, and Q. Wei, "Neural-network-observer-based optimal control for unknown nonlinear systems using adaptive dynamic programming," *International Journal of Control*, vol. 86, no. 9, pp. 1554–1566, 2013.

[46] X. Wang, E.-J. van Kampen, Q. P. Chu, and P. Lu, "Incremental sliding-mode fault-tolerant flight control," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 2, pp. 244–259, 2018.

[47] Y. Zhou, E.-J. van Kampen, and Q. P. Chu, "Nonlinear adaptive flight control using incremental approximate dynamic programming and output feedback," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 2, pp. 493–496, 2016.

[48] Z. Wang, R. Lu, F. Gao, and D. Liu, "An indirect data-driven method for trajectory tracking control of a class of nonlinear discrete-time systems," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 5, pp. 4121–4129, 2017.

[49] S. Ragi and E. K. Chong, "UAV path planning in a dynamic environment via partially observable Markov decision process," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 4, pp. 2397–2412, 2013.

[50] Y. Zhou, E.-J. van Kampen, and Q. P. Chu, "Incremental approximate dynamic programming for nonlinear adaptive tracking control with partial observability," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 12, pp. 2554–2567, 2018.

[51] H. Kim, M. Jordan, S. Sastry, and A. Ng, "Autonomous helicopter flight via reinforcement learning," *Advances in neural information processing systems*, vol. 16, 2003.

[52] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a quadrotor with reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2096–2103, 2017.

[53] J. Xu, T. Du, M. Foshey, B. Li, B. Zhu, A. Schulz, and W. Matusik, "Learning to fly: computational controller design for hybrid uavs with reinforcement learning," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.

[54] Y. Yu, "Towards sample efficient reinforcement learning.," in *IJCAI*, pp. 5739–5743, 2018.

[55] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1291–1297, 2012.

[56] J. Wang, X.-M. Zhang, and Q.-L. Han, "Event-triggered generalized dissipativity filtering for neural networks with time-varying delays," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 1, pp. 77–88, 2016.

[57] Q. Liu, M. Liu, Y. Shi, and J. Yu, "Event-triggered adaptive attitude control for flexible spacecraft with actuator nonlinearity," *Aerospace Science and Technology*, vol. 106, p. 106111, 2020.

[58] M. Poozesh and S. Mobayen, "Event-triggered fractional-order sliding mode control technique for stabilization of disturbed quadrotor unmanned aerial vehicles," *Aerospace Science and Technology*, p. 107337, 2022.

[59] B. Sun and E.-J. van Kampen, "Incremental model-based global dual heuristic programming with explicit analytical calculations applied to flight control," *Engineering Applications of Artificial Intelligence*, vol. 89, p. 103425, 2020.

[60] M. Lungu and R. Lungu, "Neural network based adaptive control of airplane's lateral-directional motion during final approach phase of landing," *Engineering Applications of Artificial Intelligence*, vol. 74, pp. 322–335, 2018.

[61] A. Coates, P. Abbeel, and A. Y. Ng, "Autonomous helicopter flight using reinforcement learning," *Encyclopedia of Machine Learning and Data Mining*, pp. 75–85, 2017.

[62] Y. Zhou, E. van Kampen, and Q. P. Chu, "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*, 2016.

[63] B. Sun and E.-J. van Kampen, "Incremental model-based global dual heuristic programming for flight control," *IFAC-PapersOnLine*, vol. 52, no. 29, pp. 7–12, 2019.

[64] D. Wang, "Intelligent critic control with robustness guarantee of disturbed nonlinear plants," *IEEE transactions on cybernetics*, vol. 50, no. 6, pp. 2740–2748, 2019.

[65] D. Wang, "Robust policy learning control of nonlinear plants with case studies for a power system application," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 1733–1741, 2019.

[66] E. F. Ferreira, P. H. Rêgo, and J. V. Neto, "Numerical stability improvements of state-value function approximations based on rls learning for online hdp-dlqr control system design," *Engineering Applications of Artificial Intelligence*, vol. 63, pp. 1–19, 2017.

[67] D. Wang, M. Ha, and J. Qiao, "Self-learning optimal regulation for discrete-time nonlinear systems under event-driven formulation," *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 1272–1279, 2019.

[68] G. K. Venayagamoorthy, R. G. Harley, and D. C. Wunsch, "Comparison of heuristic dynamic programming and dual heuristic programming adaptive critics for neurocontrol of a turbogenerator," *IEEE Transactions on Neural Networks*, vol. 13, no. 3, pp. 764–773, 2002.

[69] J. Yi, S. Chen, X. Zhong, W. Zhou, and H. He, "Event-triggered globalized dual heuristic programming and its application to networked control systems," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1383–1392, 2019.

[70] M. Fairbank, E. Alonso, and D. Prokhorov, "Simple and fast calculation of the second-order gradients for globalized dual heuristic dynamic programming in neural networks," *IEEE transactions on neural networks and learning systems*, vol. 23, no. 10, pp. 1671–1676, 2012.

[71] J. R. Magnus and H. Neudecker, *Matrix differential calculus with applications in statistics and econometrics*. John Wiley & Sons, 2019.

[72] M. Abouheaf, W. Gueaieb, and F. Lewis, "Model-free gradient-based adaptive learning controller for an unmanned flexible wing aircraft," *Robotics*, vol. 7, no. 4, p. 66, 2018.

[73] K. G. Vamvoudakis and H. Ferraz, "Model-free event-triggered control algorithm for continuous-time linear systems with optimal performance," *Automatica*, vol. 87, pp. 412–420, 2018.

[74] H. Lin, Q. Wei, and D. Liu, "Online identifier–actor–critic algorithm for optimal control of nonlinear systems," *Optimal Control Applications and Methods*, vol. 38, no. 3, pp. 317–335, 2017.

[75] X. Wang, E. van Kampen, Q. P. Chu, and P. Lu, "Stability analysis for incremental nonlinear dynamic inversion control," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 5, pp. 1116–1129, 2019.

[76] X. Wang and E. van Kampen, "Incremental backstepping sliding mode fault-tolerant flight control," in *AIAA Scitech 2019 Forum*, p. 0110, 2019.

[77] A. A. Abdullah, P. A. Ioannou, and G. Samaras, "Control design for f-16 longitudinal motion," *IFAC Proceedings Volumes*, vol. 37, no. 6, pp. 529–534, 2004.

[78] L. Nguyen, M. Ogburn, W. Gilbert, K. Kibler, P. Brown, and P. Deal, "Nasa technical paper 1538-simulator study of stall/post-stall characteristics of a fighter airplane with relaxed longitudinal static stability," *Tech. rep., NASA*, 1979.

[79] R. Van't Veld, E. van Kampen, and Q. P. Chu, "Stability and robustness analysis and improvements for incremental nonlinear dynamic inversion control," in *2018 AIAA Guidance, Navigation, and Control Conference*, p. 1127, 2018.

[80] B. Sun and E.-J. van Kampen, "Intelligent adaptive optimal control using incremental model-based global dual heuristic programming subject to partial observability," *Applied Soft Computing*, vol. 103, p. 107153, 2021.

[81] S. Suresh and N. Kannan, "Direct adaptive neural flight control system for an unstable unmanned aircraft," *Applied Soft Computing*, vol. 8, no. 2, pp. 937–948, 2008.

[82] D. P. Bertsekas, *Dynamic programming and optimal control*, vol. 1. Athena scientific Belmont, MA, 1995.

[83] H. J. Kappen, *Optimal control theory and the linear Bellman equation*. Cambridge University Press, 2011.

[84] B. Sun and E.-J. van Kampen, "Incremental model-based heuristic dynamic programming with output feedback applied to aerospace system identification and control," in *2020 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 366–371, IEEE, 2020.

[85] P. Hušek and K. Narenathreyas, "Aircraft longitudinal motion control based on takagi–sugeno fuzzy model," *Applied Soft Computing*, vol. 49, pp. 269–278, 2016.

[86] S. A. A. Rizvi and Z. Lin, "Output feedback Q-learning control for the discrete-time linear quadratic regulator problem," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 5, pp. 1523–1536, 2018.

[87] X. Liu, B. Zhao, and D. Liu, "Fault tolerant tracking control for nonlinear systems with actuator failures through particle swarm optimization-based adaptive dynamic programming," *Applied Soft Computing*, vol. 97, p. 106766, 2020.

[88] F. L. Lewis and K. G. Vamvoudakis, "Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 41, no. 1, pp. 14–25, 2010.

[89] B. Kiumarsi, F. L. Lewis, M.-B. Naghibi-Sistani, and A. Karimpour, "Optimal tracking control of unknown discrete-time linear systems using input-output measured data," *IEEE transactions on cybernetics*, vol. 45, no. 12, pp. 2770–2779, 2015.

[90] A. Keshavarz and S. Boyd, "Quadratic approximate dynamic programming for input-affine systems," *International Journal of Robust and Nonlinear Control*, vol. 24, no. 3, pp. 432–449, 2014.

[91] D. Liu, X. Yang, D. Wang, and Q. Wei, "Reinforcement-learning-based robust controller design for continuous-time uncertain nonlinear systems subject to input constraints," *IEEE transactions on cybernetics*, vol. 45, no. 7, pp. 1372–1385, 2015.

[92] H. Modares and F. L. Lewis, "Optimal tracking control of nonlinear partially-unknown constrained-input systems using integral reinforcement learning," *Automatica*, vol. 50, no. 7, pp. 1780–1792, 2014.

[93] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, vol. 50, no. 1, pp. 193–202, 2014.

[94] F. A. Yaghmaie and D. J. Braun, "Reinforcement learning for a class of continuous-time input constrained optimal control problems," *Automatica*, vol. 99, pp. 221–227, 2019.

[95] K. Zhang, H. Zhang, X. Liang, and Z. Wang, "Neurodynamic programming and tracking control scheme of constrained-input systems via a novel event-triggered pi algorithm," *Applied Soft Computing*, vol. 83, p. 105629, 2019.

[96] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2042–2062, 2017.

[97] B. Depraetere, M. Liu, G. Pinte, I. Grondman, and R. Babuška, "Comparison of model-free and model-based methods for time optimal hit control of a badminton robot," *Mechatronics*, vol. 24, no. 8, pp. 1021–1030, 2014.

[98] B. Sun and E.-J. van Kampen, "Launch vehicle discrete-time optimal tracking control using global dual heuristic programming," in *2020 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 162–167, IEEE, 2020.

[99] D. Liu, D. Wang, D. Zhao, Q. Wei, and N. Jin, "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*, vol. 9, no. 3, pp. 628–634, 2012.

[100] D. Wang, D. Liu, Q. Wei, D. Zhao, and N. Jin, "Optimal control of unknown non-affine nonlinear discrete-time systems based on adaptive dynamic programming," *Automatica*, vol. 48, no. 8, pp. 1825–1832, 2012.

[101] D. Vrabie and F. Lewis, "Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems," *Neural Networks*, vol. 22, no. 3, pp. 237–246, 2009.

[102] J. Na and G. Herrmann, "Online adaptive approximate optimal tracking control with simplified dual approximation structure for continuous-time unknown nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 1, no. 4, pp. 412–422, 2014.

[103] I. Grondman, M. Vaandrager, L. Busoniu, R. Babuska, and E. Schuitema, "Efficient model learning methods for actor–critic control," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 3, pp. 591–602, 2012.

[104] Y. Huang, D. M. Pool, O. Stroosma, and Q. Chu, "Long-stroke hydraulic robot motion control with incremental nonlinear dynamic inversion," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 1, pp. 304–314, 2019.

[105] Y. Zhou, E.-J. Van Kampen, and Q. Chu, "Incremental model based online heuristic dynamic programming for nonlinear adaptive tracking control with partial observability," *Aerospace Science and Technology*, vol. 105, p. 106013, 2020.

[106] N. Szanto, V. Narayanan, and S. Jagannathan, "Event-sampled direct adaptive NN output- and state-feedback control of uncertain strict-feedback system," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 5, pp. 1850–1863, 2017.

[107] Y. Zhou, E.-J. van Kampen, and Q. P. Chu, "Hybrid hierarchical reinforcement learning for online guidance and navigation with partial observability," *Neurocomputing*, vol. 331, pp. 443–457, 2019.

[108] F. Grondman, G. Looye, R. O. Kuchar, Q. P. Chu, and E.-J. van Kampen, "Design and flight testing of incremental nonlinear dynamic inversion-based control laws for a passenger aircraft," in *2018 AIAA Guidance, Navigation, and Control Conference*, p. 0385, 2018.

[109] B. Farhang-Boroujeny, *Adaptive filters: theory and applications*. John Wiley & Sons, 2 ed., 2013.

[110] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[111] B. Sun and E.-J. van Kampen, "Reinforcement learning-based adaptive optimal flight control with output feedback and input constraints," *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 9, pp. 1685–1691, 2021.

[112] J. Junell, T. Mannucci, Y. Zhou, and E. van Kampen, "Self-tuning gains of a quadrotor using a simple model for policy gradient reinforcement learning," in *AIAA Guidance, Navigation, and Control Conference*, p. 1387, 2016.

[113] A. Heydari and S. Balakrishnan, "Adaptive critic-based solution to an orbital rendezvous problem," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 1, pp. 344–350, 2014.

[114] B. Kiumarsi and F. L. Lewis, "Actor–critic-based optimal tracking for partially unknown nonlinear discrete-time systems," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 1, pp. 140–151, 2015.

[115] H. Li, L. Sun, W. Tan, B. Jia, and X. Liu, "Switching flight control for incremental model-based dual heuristic dynamic programming," *Journal of Guidance, Control, and Dynamics*, pp. 1–7, 2020.

[116] M. D. Tandale and J. Valasek, "Adaptive dynamic inversion control with actuator saturation constraints applied to tracking spacecraft maneuvers," *Journal of the Astronautical Sciences*, vol. 52, no. 4, pp. 517–530, 2004.

[117] L. Sonneveldt, Q. Chu, and J. Mulder, "Nonlinear flight control design using constrained adaptive backstepping," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 2, pp. 322–336, 2007.

[118] L. Sonneveldt, E. Van Oort, Q. Chu, and J. Mulder, "Nonlinear adaptive trajectory control applied to an f-16 model," *Journal of Guidance, control, and Dynamics*, vol. 32, no. 1, pp. 25–39, 2009.

[119] B. Sun and E.-J. van Kampen, "Event-triggered constrained control using explainable global dual heuristic programming for nonlinear discrete-time systems," *Neurocomputing*, vol. 468, no. 1, pp. 452–463, 2022.

[120] J. Zhao, J. Na, and G. Gao, "Adaptive dynamic programming based robust control of nonlinear systems with unmatched uncertainties," *Neurocomputing*, vol. 395, pp. 56–65, 2020.

[121] D. Wang, M. Ha, and J. Qiao, "Self-learning optimal regulation for discrete-time nonlinear systems under event-driven formulation," *IEEE Transactions on Automatic Control*, vol. 65, no. 3, pp. 1272–1279, 2020.

[122] Q. Liu, T. Li, Q. Shan, R. Yu, and X. Gao, "Virtual guide automatic berthing control of marine ships based on heuristic dynamic programming iteration method," *Neurocomputing*, vol. 437, pp. 289–299, 2021.

[123] L. Kong, S. Zhang, and X. Yu, "Approximate optimal control for an uncertain robot based on adaptive dynamic programming," *Neurocomputing*, vol. 423, pp. 308–317, 2021.

[124] G. Che and Z. Yu, "Neural-network estimators based fault-tolerant tracking control for AUV via ADP with rudders faults and ocean current disturbance," *Neurocomputing*, vol. 411, pp. 442–454, 2020.

[125] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2042–2062, 2018.

[126] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, *et al.*, "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[127] M. Abu-Khalaf and F. L. Lewis, "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach," *Automatica*, vol. 41, no. 5, pp. 779–791, 2005.

[128] H. Zhang, Y. Luo, and D. Liu, "Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints," *IEEE Transactions on Neural Networks*, vol. 20, no. 9, pp. 1490–1503, 2009.

[129] M. Ha, D. Wang, and D. Liu, "Event-triggered constrained control with DHP implementation for nonaffine discrete-time systems," *Information Sciences*, vol. 519, pp. 110–123, 2020.

[130] X. Yang and Q. Wei, "Adaptive critic learning for constrained optimal event-triggered control with discounted cost," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 91–104, 2021.

[131] A. Sahoo, H. Xu, and S. Jagannathan, "Near optimal event-triggered control of nonlinear discrete-time systems using neurodynamic programming," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 9, pp. 1801–1815, 2016.

[132] M. Dai, J. Xia, H. Xia, and H. Shen, "Event-triggered passive synchronization for markov jump neural networks subject to randomly occurring gain variations," *Neurocomputing*, vol. 331, pp. 403–411, 2019.

[133] H. Zhang, Z. Qiu, J. Cao, M. Abdel-Aty, and L. Xiong, "Event-triggered synchronization for neutral-type semi-markovian neural networks with partial mode-dependent time-varying delays," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 11, pp. 4437–4450, 2020.

[134] S. Wang, Y. Cao, T. Huang, Y. Chen, and S. Wen, "Event-triggered distributed control for synchronization of multiple memristive neural networks under cyber-physical attacks," *Information Sciences*, vol. 518, pp. 361–375, 2020.

[135] S. Zhang, B. Zhao, and Y. Zhang, "Event-triggered control for input constrained nonaffine nonlinear systems based on neuro-dynamic programming," *Neurocomputing*, vol. 440, pp. 175–184, 2021.

[136] S. Xue, B. Luo, D. Liu, and Y. Li, "Adaptive dynamic programming based event-triggered control for unknown continuous-time nonlinear systems with input constraints," *Neurocomputing*, vol. 396, pp. 191–200, 2020.

[137] R. Song and L. Liu, "Event-triggered constrained robust control for partly-unknown nonlinear systems via adp," *Neurocomputing*, vol. 404, pp. 294–303, 2020.

[138] L. Dong, X. Zhong, C. Sun, and H. He, "Adaptive event-triggered control based on heuristic dynamic programming for nonlinear discrete-time systems," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 7, pp. 1594–1605, 2017.

[139] M. Ha, D. Wang, and D. Liu, "Event-triggered adaptive critic control design for discrete-time constrained nonlinear systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 9, pp. 3158 – 3168, 2020.

[140] Z. Wang, Q. Wei, and D. Liu, "A novel triggering condition of event-triggered control based on heuristic dynamic programming for discrete-time systems," *Optimal Control Applications and Methods*, vol. 39, no. 4, pp. 1467–1478, 2018.

[141] L. Liu, Z. Wang, and H. Zhang, "Neural-network-based robust optimal tracking control for MIMO discrete-time systems with unknown uncertainty using adaptive critic design," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 4, pp. 1239–1251, 2018.

[142] Q. Wei, D. Liu, and H. Lin, "Value iteration adaptive dynamic programming for optimal control of discrete-time nonlinear systems," *IEEE Transactions on cybernetics*, vol. 46, no. 3, pp. 840–853, 2016.

[143] T. Dierks, B. T. Thumati, and S. Jagannathan, "Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence," *Neural Networks*, vol. 22, no. 5-6, pp. 851–860, 2009.

[144] B. Sun, X. Wang, and E.-J. van Kampen, "Event-triggered intelligent critic control with input constraints applied to a nonlinear aeroelastic system," *Aerospace Science and Technology*, vol. 120, p. 107279, 2022.

[145] J. Yuan, N. Qi, Z. Qiu, and F. Wang, "Adaptive rbf observer-sliding mode controller design for a two dimensional aeroelastic system with unsteady aerodynamics," *Aerospace Science and Technology*, vol. 80, pp. 482–495, 2018.

[146] K. W. Lee and S. N. Singh, "Robust finite-time continuous control of an unsteady aeroelastic system," *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 4, pp. 978–986, 2018.

[147] D. Li, J. Xiang, and S. Guo, "Adaptive control of a nonlinear aeroelastic system," *Aerospace Science and Technology*, vol. 15, no. 5, pp. 343–352, 2011.

[148] H. Lhachemi, Y. Chu, D. Saussié, and G. Zhu, "Flutter suppression for underactuated aeroelastic wing section: Nonlinear gain-scheduling approach," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 8, pp. 2102–2109, 2017.

[149] W. Su and W. Song, "A real-time hybrid aeroelastic simulation platform for flexible wings," *Aerospace Science and Technology*, vol. 95, p. 105513, 2019.

[150] T. W. Strganac, J. Ko, D. E. Thompson, and A. J. Kurdila, "Identification and control of limit cycle oscillations in aeroelastic systems," *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 6, pp. 1127–1133, 2000.

[151] D. Wang, H. He, and D. Liu, "Improving the critic learning for event-based nonlinear $H\infty$ control design," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3417–3428, 2017.

[152] C. Peng and J. Ma, "Online integral reinforcement learning control for an uncertain highly flexible aircraft using state and output feedback," *Aerospace Science and Technology*, p. 106442, 2020.

[153] X. Han, Z. Zheng, L. Liu, B. Wang, Z. Cheng, H. Fan, and Y. Wang, "Online policy iteration adp-based attitude-tracking control for hypersonic vehicles," *Aerospace Science and Technology*, vol. 106, p. 106233, 2020.

[154] C. Liu, C. Dong, Z. Zhou, and Z. Wang, "Barrier lyapunov function based reinforcement learning control for air-breathing hypersonic vehicle with variable geometry inlet," *Aerospace Science and Technology*, vol. 96, p. 105537, 2020.

[155] H. Dong, X. Zhao, and B. Luo, "Optimal tracking control for uncertain nonlinear systems with prescribed performance via critic-only adp," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 1, pp. 561–573, 2020.

[156] J. Na, B. Wang, G. Li, S. Zhan, and W. He, "Nonlinear constrained optimal control of wave energy converters with adaptive dynamic programming," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 10, pp. 7904–7915, 2018.

[157] V. Narayanan, H. Modares, S. Jagannathan, and F. L. Lewis, "Event-driven off-policy reinforcement learning for control of interconnected systems," *IEEE transactions on cybernetics*, vol. 52, no. 3, pp. 1936–1946, 2020.

[158] K. G. Vamvoudakis, "Event-triggered optimal adaptive control algorithm for continuous-time nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 1, no. 3, pp. 282–293, 2014.

[159] A. Sahoo, H. Xu, and S. Jagannathan, "Approximate optimal control of affine nonlinear continuous-time systems using event-sampled neurodynamic programming," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 3, pp. 639–652, 2016.

[160] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1680–1685, 2007.

[161] D. Wang, C. Mu, X. Yang, and D. Liu, "Event-based constrained robust control of affine systems incorporating an adaptive critic mechanism," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1602–1612, 2017.

[162] Y. Zhu, D. Zhao, H. He, and J. Ji, "Event-triggered optimal control for partially unknown constrained-input systems via adaptive dynamic programming," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 5, pp. 4101–4109, 2017.

[163] D. Wang, H. He, and D. Liu, "Intelligent optimal control with critic learning for a nonlinear overhead crane system," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 2932–2940, 2018.

[164] L. Sanches, T. A. Guimarães, and F. D. Marques, "Aeroelastic tailoring of nonlinear typical section using the method of multiple scales to predict post-flutter stable lcos," *Aerospace Science and Technology*, vol. 90, pp. 157–168, 2019.

[165] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*. Prentice hall Upper Saddle River, NJ, 3 ed., 2002.

[166] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5628–5635, IEEE, 2018.

[167] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization.," *Journal of machine learning research*, vol. 13, no. 2, 2012.

[168] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, 2012.

[169] H. Alibrahim and S. A. Ludwig, "Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1551–1559, IEEE, 2021.

[170] J. Waring, C. Lindvall, and R. Umeton, "Automated machine learning: Review of the state-of-the-art and opportunities for healthcare," *Artificial intelligence in medicine*, vol. 104, p. 101822, 2020.

[171] S. Heyer, D. Kroezen, and E.-J. Van Kampen, "Online adaptive incremental reinforcement learning flight control for a cs-25 class aircraft," in *AIAA Scitech 2020 Forum*, p. 1844, 2020.

[172] L. Dong, J. Yan, X. Yuan, H. He, and C. Sun, "Functional nonlinear model predictive control based on adaptive dynamic programming," *IEEE transactions on cybernetics*, vol. 49, no. 12, pp. 4206–4218, 2018.

[173] G. S. Lakshmikanth, R. Padhi, J. M. Watkins, and J. E. Steck, "Single network adaptive critic aided nonlinear dynamic inversion for suboptimal command tracking," in *2011 IEEE International Symposium on Intelligent Control*, pp. 1347–1352, IEEE, 2011.

[174] C. Teirlinck, "Reinforcement learning for flight control: Hybrid offline-online learning for robust and adaptive fault-tolerance," tech. rep., Delft University of Technology, 2022.

[175] L. Vieira dos Santos, "Safe & intelligent control: Fault-tolerant flight control with distributional and hybrid reinforcement learning using dsac and idhp," tech. rep., Delft University of Technology, 2023.

[176] A. De Marco, a. M. D'Onza, and S. Manfredi, "A deep reinforcement learning control approach for high-performance aircraft," *Nonlinear Dynamics*, vol. 111, p. 17037–17077, 2023.

Acknowledgments

Everyone who lived through the pandemic can surely relate to the feeling of time just slipping away. Thinking back on my PhD journey, I can't help but notice how the unique challenges of this time have shaped my academic adventure. Over this wild period we've all been through, marked by uncertainty and unprecedented challenges, eventually completing a PhD has become more than an academic accomplishment; it's a nod to the grit, flexibility, and incredible support I've received from so much wonderful people.

First of all, I would like to extend my heartfelt gratitude to my promotor, Prof. Guido de Croon, for your invaluable guidance and support in various aspects. Your kindness and dedication to both people and work have also left a lasting impression on me. I found it impressive that you not only had great insight and offered invaluable suggestions on the overarching direction in nearly every meeting, but also could provide meticulous feedback on every piece of scientific writing. Without your unwavering support, my dissertation wouldn't be in its current form.

To Dr. Erik-Jan van Kampen, my co-promoter and daily supervisor, I sincerely thank you for your patient guidance and effective mentorship throughout my research journey. You are always willing to share valuable insights and provide constructive suggestions, which has greatly enriched my work. Engaging in discussions with you has consistently proven to be beneficial, which not only contributes significantly to my research but also broadens my horizons in culture and life. Moreover, I appreciate the freedom you granted me to explore my research independently and explore my career outside the university.

I also want to convey my gratitude to Prof. Max Mulder, who was my former promoter for the first two years. The scene where you and Prof. Guido de Croon interviewed me is still vivid in my mind. Thank you for granting me the opportunity to pursue a PhD here. It marks the starting point of a memorable journey. Your support and guidance have been instrumental in shaping my academic endeavors, and I am truly thankful for the positive impact you have had on my research work.

Walking together allows for a smoother journey. Many thanks to my main cooperator Dr. Xuerui Wang during this PhD period. Thank you for connecting me with many passionate researchers, so that my theoretical research methods came into practice. Your enthusiasm for academia always encourages me, enabling me to persevere during challenging times in the laboratory. To our "dàgē" (gang leader & elder brother) Dr. Tigran Mkhoyan, I will always remember those days we spent together in the OJF when we enjoyed pizza on the grass after a test together with Xuerui, Iren and Oscar, accompanied by a gentle breeze, and when the security came to escort us home in the late evening. To Dr. Jing Chang, I think back with joy on that busy summer we spent together in the high-speed wind tunnel lab. Your solid professional ability is a crucial guarantee for achieving commendable experimental results. To Killian Dally, thank you for unreservedly sharing your knowledge and experience in building an impressive RL simulation.

Furthermore, I would like to express my gratitude to all the PhD colleagues in the C&S department for the enjoyable moments during our breaks, including coffee breaks, lunchtime gatherings, BBQs, PhD drinks, and various PhD events. We had a great time, fostering a supportive environment where we not only had fun but also encouraged one another by suggesting potential solutions for each other's problems both in work and life. During these years of pursuing a PhD, it has been a period of significant transformation for my worldview, and every interaction with all of you has expanded my perspective. My thanks go to: Shuo Li, Shushuai Li, Yingfu Xu, Dyah Jatiningrum, Annemarie Landman, Tom van Dijk, Ying Yu, Cheng Liu, Kimberly McGuire, Sihao Sun, Daniel Friesen, Jerom Maas, Diana Olejnik, Jelmer Reitsma, Dirk van Baelen, Kirk Scheper, Federico Paredes-Vallés, Sarah Barendswaard, Emmanuel Sunil, Isabel Metz, Jaime Junell, Kasper van der El, Mario Coppola, Malik Doole, Noor Nabi, Yingzhi Huang, Ye Zhang, Wei Fu, Ewoud Smeur, Jesse Hagenaars, Yifei Li, Ziqing Ma, Sophie Armanini, Junzi Sun, Rowenna Wijlens, Gijs de Rooij, Sunyou Hwang, Sven Pfeiffer, Sunyi Wang, Marta Ribeiro, Paolo Scaramuzzino, Stein Stroobants, Hang Yu, Jiayu Chen, Yilun Wu, Yiyuan Zou, Alessandro Mancinelli, Julia Rudnyk, Liming Zheng, Tiago Monteiro Nunes. My special thanks to Shuo who gave me a welcome when I arrived here and to Shushuai who helped me with this dissertation writing.

In addition to my fellow PhD candidate colleagues, I am also fortunate to work together with all staff members in the C&S section, including professors, researchers, support, and administration members. I enjoyed the time I asked questions, requested your help, and had lunch and coffee breaks together. My thanks go to Qiping Chu, Coen De Visser, Matěj Karásek, René van Paassen, Daan Pool, Clark Borst, Joost Ellerbroek, Olaf Stroosma, Julien Dupeyroux, Anahita Jamshidnejad, Andries Muis, Salua Hamaza, Erik van der Horst, Christophe De Wagter. And thank you, Bertine Markus, for your help on the administration stuff. My thanks also go to student friends: Lorenzo Martini, Nilay Sheth, José Ignacio de Alvear Cardenas.

I wish to thank my friends outside the C&S section. They are: Jinke He, Rong Wan, Qisong Yang, Junhan Wen, Zhou Nie, Wei Yuan, Xueer Hu, Zihan Liu, Pang-Chieh Lin, Zhiyu Shi, Zilong Zhao, Fenghua Wang, Jing Xu, Lin Hao, Xiao Cui, Li Zou, Yaiza Fei Raigoso. I sincerely want to express my gratitude to Dr. Chunyan Ling. It is your understanding and support that gave me the determination to embark on the journey of pursuing a doctoral degree. I am grateful to Yudi Ma for pulling me out of the quagmire of isolation and depression. Wishing Bo Zhang all the best fills me with joy and honor, knowing that I have the privilege to be your guardian star. I also express my gratitude to my colleagues and friends at ING Bank where I completed the final stretch of my PhD journey.

Before the end, my deepest gratitude goes to my parents, Yaotang Sun and Guiming Chen. Thank you for your unconditional and unwavering. No matter how far I go, I am always aware of where my home is.

"Time flies", as we always sigh. The names mentioned above cannot cover all the people to whom my gratitude goes. May those who have met be able to meet again. It is the time to close the book and look outside of the window, where a path winds and leads into the distance. Thank you, everyone. Thank you, the alluring world.

Bo Sun
Delft, June 2024

Curriculum Vitæ

Bo Sun

10-08-1995 Born in Chifeng, Nei Mongol, China

Education

2018-2022 Ph.D. in Aerospace Engineering
Delft University of Technology, the Netherlands
Thesis: Online Intelligent Optimal Flight Control
Promotor: Prof. dr. G.C.H.E. de Croon

2016-2019 M.Sc. in Aerospace Engineering
Northwestern Polytechnical University, China

2012-2016 B.Sc. in Aerospace Engineering
Northwestern Polytechnical University, China

Awards

2016 Distinguished Bachelor degree dissertation

2014 First prize in China Robot Competition & RoboCup China Open

Academic activities

Reviewer for IEEE Transactions on Neural Networks and Learning Systems

Reviewer for IEEE Transactions on Aerospace and Electronic Systems

Reviewer for Engineering Applications of Artificial Intelligence

Reviewer for IEEE Conference on Control Technology and Applications (CCTA)

Publications during Master Stage

5. C. Ling, Z. Lu, **B. Sun** and M. Wang, "An Efficient Method Combining Active Learning Kriging and Monte Carlo Simulation for Profust Failure Probability", *Fuzzy Sets and Systems*, vol. 387, pp. 89-107, 2020. Doi: 10.1016/j.fss.2019.02.003.

4. C. Ling, Z. Lu, K. Feng and **B. Sun**, "An Efficient Method for Estimating Global Reliability Sensitivity Indices", *Probabilistic Engineering Mechanics*, vol. 56, pp. 35-49, 2019. Doi: 10.1016/j.probengmech.2019.04.003.

3. C. Ling, Z. Lu, K. Feng and **B. Sun**, "Efficient Numerical Simulation Methods for Estimating Fuzzy Failure Probability Based Importance Measure Indices", *Structural and Multidisciplinary Optimization*, vol. 59, no. 2, pp. 577-593, 2019. Doi: 10.1007/s00158-018-2085-1.

2. X. Chang, **B. Sun**, J. Yan, and W. Fu, "3-Dimensional Nonlinear Differential Game-based Guidance Law Against High-speed Maneuvering Target", *Dandao Xuebao / Journal Of Ballistics*, vol. 30, no. 3, pp. 1-6, 2018. Doi: 10.12115/j.issn.1004-499X(2018)03-001. In Chinese.

1. X. Chang, **B. Sun**, J. Yan, and W. Fu, "A Game-Based Guidance Law Against Higher-Speed Maneuvering Penetrator Using Model Predictive Method", in *9th International Conference on Mechanical and Aerospace Engineering (ICMAE)*, IEEE, Budapest, Hungary, pp. 183-189, 2018. Doi: 10.1109/ICMAE.2018.8467620.

List of Publications

Journals

7. **B. Sun**, E. van Kampen, J. Sodja, R. De Breuker and X. Wang, "Event-Triggered Adaptive Dynamic Programming for an Aeroservoelastic System: An Experimental Study", 2024. To be submitted.

6. **B. Sun**, T. Mkhoyan, E. van Kampen, R. De Breuker and X. Wang, "Vision-Based Nonlinear Incremental Control for A Morphing Wing with Mechanical Imperfections", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 6, pp. 5506–5518, 2022. Doi: 10.1109/TAES.2022.3175679.

5. **B. Sun**, X. Wang and E. van Kampen, "Event-Triggered Intelligent Critic Control with Input Constraints Applied to A Nonlinear Aeroelastic System", *Aerospace Science and Technology*, vol. 120, pp. 107279, 2022. Doi: 10.1016/j.ast.2021.107279.

4. **B. Sun** and E. van Kampen, "Event-Triggered Constrained Control Using Explainable Global Dual Heuristic Programming for Nonlinear Discrete-Time Systems", *Neurocomputing*, vol. 468, pp. 452-463, 2022. Doi: 10.1016/j.neucom.2021.10.046.

3. **B. Sun** and E. van Kampen, "Reinforcement-Learning-Based Adaptive Optimal Flight Control with Output Feedback and Input Constraints", *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 9, pp. 1685-1691, 2021. Doi: 10.2514/1.G005715.

2. **B. Sun** and E. van Kampen, "Intelligent Adaptive Optimal Control Using Incremental Model-Based Global Dual Heuristic Programming Subject to Partial Observability", *Applied Soft Computing*, vol. 103, pp. 107153, 2021. Doi: 10.1016/j.asoc.2021.107153.

1. **B. Sun** and E. van Kampen, "Incremental Model-Based Global Dual Heuristic Programming with Explicit Analytical Calculations Applied to Flight Control", *Engineering Applications of Artificial Intelligence*, vol. 89, pp. 103425, 2020. Doi: 10.1016/j.engappai.2019.103425.

Conference Proceedings

5. **B. Sun**, C. Liu, K. Dally and E. van Kampen, "Intelligent Aircraft Stabilization Control with Event-Triggered Scheme", in *6th CEAS Conference on Guidance, Navigation and Control (EuroGNC)*, CEAS, Berlin, Germany, 2022. CEAS-GNC-2022-075.

4. J. I. de Alvear Cardenas, **B. Sun** and E. van Kampen, "Intelligent Adaptive Control Using LADP and IADP Applied to F-16 Aircraft with Imperfect Measurements", in *AIAA Scitech 2021 Forum*, AIAA, US, 2021. Doi: 10.2514/6.2021-1119. Virtual Event.

3. **B. Sun** and E. van Kampen, "Launch Vehicle Discrete-Time Optimal Tracking Control Using Global Dual Heuristic Programming", in *4th IEEE Conference on Control Technology and Applications (CCTA)*, IEEE, Montreal, Canada, pp. 162-167, 2020. Doi: 10.1109/CCTA41146.2020.9206252. Virtual Event.

2. **B. Sun** and E. van Kampen, "Incremental Model-Based Heuristic Dynamic Programming with Output Feedback Applied to Aerospace System Identification and Control", in *4th IEEE Conference on Control Technology and Applications (CCTA)*, IEEE, Montreal, Canada, pp. 366-371, 2020. Doi: 10.1109/CCTA41146.2020.9206261. Virtual Event.

1. **B. Sun** and E. van Kampen, "Incremental model-based global dual heuristic programming for flight control.", in *13th IFAC Workshop on Adaptive and Learning Control Systems (ALCOS)*, IFAC-PapersOnLine, Winchester, UK, vol. 52, no. 29, pp. 7-12, 2019. Doi: 0.1016/j.ifacol.2019.12.613.

Book Chapters

1. E. van Kampen and **B. Sun**, "Adaptive Dynamic Programming for Flight Control", in *Control of Autonomous Aerial Vehicles: Advances in Autopilot Design for Civilian UAVs*, edited by A. L'Afflitto, G. Inalhan and H.-S. Shin, Springer, Cham, Switzerland. pp. 269-292, 2023. Doi: 10.1007/978-3-031-39767-7_10.