arXiv:2206.05596v2 [eess.SY] 14 Jun 2022

Neural Network-based Flight Control Systems: Present and Future

Seyyed Ali Emami[a], Paolo Castaldi[b], Afshin Banazadeh[a,*]

[a] *Department of Aerospace Engineering, Sharif University of Technology, Tehran, Iran*
[b] *Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi", University of Bologna, Via Dell'Universit'a 50, Cesena, Italy*

Abstract

As the first review in this field, this paper presents an in-depth mathematical view of Intelligent Flight Control Systems (IFCSs), particularly those based on artificial neural networks. The rapid evolution of IFCSs in the last two decades in both the methodological and technical aspects necessitates a comprehensive view of them to better demonstrate the current stage and the crucial remaining steps towards developing a truly intelligent flight management unit. To this end, in this paper, we will provide a detailed mathematical view of Neural Network (NN)-based flight control systems and the challenging problems that still remain. The paper will cover both the model-based and model-free IFCSs. The model-based methods consist of the basic feedback error learning scheme, the pseudocontrol strategy, and the neural backstepping method. Besides, different approaches to analyze the closed-loop stability in IFCSs, their requirements, and their limitations will be discussed in detail. Various supplementary features, which can be integrated with a basic IFCS such as the fault-tolerance capability, the consideration of system constraints, and the combination of NNs with other robust and adaptive elements like disturbance observers, would be covered, as well. On the other hand, concerning model-free flight controllers, both the indirect and direct adaptive control systems including indirect adaptive control using NN-based system identification, the approximate dynamic programming

*Corresponding author
Email address: banazadeh@sharif.edu (Afshin Banazadeh)

using NN, and the reinforcement learning-based adaptive optimal control will be carefully addressed. Finally, by demonstrating a well-organized view of the current stage in the development of IFCSs, the challenging issues, which are critical to be addressed in the future, are thoroughly identified. As a result, this paper can be considered as a comprehensive road map for all researchers interested in the design and development of intelligent control systems, particularly in the field of aerospace applications.

Keywords: Flight control, Intelligent control, Neural networks, Reinforcement learning

---

Contents

1 Introduction **4**

1.1 Intelligent control systems 4

1.2 Direct versus indirect adaptive control 8

1.3 Model-based versus Model-free control 9

1.3.1 Model-based approach 9

1.3.2 Model-free approach 10

2 Foundations of model-based intelligent control **12**

2.1 Feedback Error Learning 13

2.2 Pseudocontrol strategy 21

2.3 Neural backstepping control 26

2.3.1 Dynamic surface control 28

2.3.2 Command filtered backstepping 29

2.3.3 Backstepping augmented by the First-Order Sliding Mode Differentiators (FOSMD) 31

2.3.4 Direct neural-backstepping control 32

2.4 How to analyze the closed-loop stability? 34

2.4.1 Asymptotic stability 35

2.4.2 Exponential stability 37

2.4.3 Finite-time stability 38

3 Supplementary features in model-based IFCSs 40

3.1 Output Feedback (OFB) control 40

3.2 Minimal-learning parameter 42

3.3 Systems with unknown control direction 46

3.4 Neural networks and Disturbance Observers (DOs) 48

3.4.1 Neural disturbance observer 48

3.4.2 Combination of NN function approximation and DOs . . 50

3.5 Fault-tolerant control 54

3.5.1 FEL-based fault identification 55

3.5.2 Using a separate Neural fault detection and identification (FDI) block 57

3.5.3 Multimodel approaches 59

3.6 Consideration of input constraints 60

3.6.1 Pseudocontrol Hedging (PCH) 60

3.6.2 Employment of a modified tracking error 61

3.6.3 Neuro-predictive control 63

3.6.4 Using Nussbaum function 65

3.7 Consideration of state/output constraints 66

3.8 Self-organizing neural networks 68

3.9 Concerns with air vehicle's characteristics 72

4 Towards truly model-free control systems 75

4.1 Neural network-based system identification 75

4.1.1 Single-hidden-layer neural networks 77

4.1.2 Deep neural networks 82

4.2 Neuroadaptive optimal control 83

4.2.1 Optimal control formulation (HJB vs. HJI equations) . . 83

4.2.2 Approximate dynamic programming (continuous-time systems) 85

4.3 Direct adaptive control using Reinforcement learning (RL) 92
4.3.1 Approximate dynamic programming (discrete-time systems) 94
4.3.2 Direct policy updating 99

5 Concluding remarks and future directions 115

1. Introduction

1.1. Intelligent control systems

Although the words *Intelligence* and *Autonomy* have been widely employed interchangeably, there is an essential conceptual difference between them [1]. Different definitions have been given for both concepts in the literature [2, 3]. However, in a general view, the intelligence may be defined as *a very general mental capability that involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly and learn from experience* [3]. On the other hand, *the ability to generate one's own purposes without any instruction from outside* can be interpreted as the autonomy of a system [1]. Nevertheless, concerning these two general definitions, in some cases, distinguishing the high level of autonomy from the low level of intelligence is not trivial at all. Within the framework of the control theory, typically, the final purpose is to develop an autonomous system (rather than an intelligent system), which can fulfill a set of predefined missions in a satisfactory manner. However, like the common literature, one may interpret the high level of autonomy of some Unmanned Aerial Vehicles (UAVs) as intelligence. Accordingly, despite the conceptual difference of these two words, in the current study, (like existing literature,) we will use the words *Intelligence* and *Autonomy*, interchangeably.

Different metrics have been provided in literature to specify the Level of Autonomy (LoA) of a system [4], particularly a UAV. Despite the lack of a unique definition categorization [5], a beneficial division has been given in [1, 6] in the case of UAVs. Considering it, the highest LoA for a single UAV (level 4) indicates the self-accomplishment of an assigned tactical plan, where it is capable of on-board trajectory replanning, event-driven self resource management, and

compensating for most faults and disturbances in different flight conditions. This, in turn, requires different self-adaptive mechanisms in the entire control system including the entire range from low-level control such as attitude control to high-level control such as path planning. Such a scheme will be known as an intelligent control system in the rest of the paper.

Nowadays, there are different types of Intelligent Flight Control Systems (IFCSs) in the literature that have been designed by employing neural networks, fuzzy systems [7], behavior tree [8, 9], reinforcement learning [10], different data-driven approaches [11, 12], evolutionary algorithms [13, 14], etc. Such a widespread and scattered use of IFCSs in the literature necessitates a comprehensive survey, which can clearly demonstrate the evolution of IFCSs in both theoretical and practical aspects in recent years. As the first review in this field, authors in [15], have addressed various technical and practical aspects of IFCSs, where different approaches including fuzzy inference systems, Neural networks, genetic algorithms, swarm intelligence, and hybrid evolutionary systems have been discussed in the paper. However, due to the breadth of the subject, it could not provide an in-depth theoretical view of IFCSs. To deal with such an issue, in the current survey, we will mainly focus on a specific type of IFCSs, namely the Neural Network (NN)-based flight control systems as the most commonly used approach in the literature within recent years. NNs have been satisfactorily employed in both the dynamic model identification and controller design process. Due to their universal approximation property, they can be employed to estimate different nonlinearities in dynamic systems. In addition, unlike the basic fuzzy control schemes, which highly depend on expert knowledge and pure experiments to construct the fuzzy rule base [16], NNs can effectively learn the system or controller dynamics with no prior information about the system. They can also be integrated with different learning-based methodologies and have been successfully utilized in both direct and indirect control structures, which are discussed in the following. Further, due to their inherent property of parallel processing, neural networks can be suitably employed in real-time implementations [17, 18].

Historically, the concept of IFCS was introduced in the 1990s by adopting NNs in the structure of flight control systems as a learning element to adapt to unexpected fault and flight conditions [19]. However, although the beginning of using NNs in flight control systems dates back to early 1990s, due to both technological and methodological limitations, dynamic NNs were not employed in practical flight control problems until 2001 [20]. As the largest project in this field, the *IFCS program* has been conducted in a collaboration between NASA and Boeing between 1999 to 2009 [21, 18, 22]. This program consists of two main phases. The first phase of the program focused on the development of an indirect adaptive flight control system. The first set of flights using a highly modified F-15B prototype occurred in 1999, where the stability and control derivatives of the air vehicle have been estimated using pre-trained NNs. Subsequently, dynamic cell structure NNs have been adopted in the control scheme for online modification of the estimated derivatives. The second set of flights using such an online identification scheme have been performed in 2003 [23]. Although the obtained results in the closed-loop simulation were reported as a *promising* achievement, due to the fact that the online identified model was not utilized in the control structure in real flights, the control scheme was not yet really adaptive [24]. On the other hand, the second phase of the program dealt with developing a direct adaptive control architecture in which a dynamic inversion block was augmented by an online NN [25]. Flight tests of the second phase began in 2006 and continued into 2008. Flight tests consisted of performance evaluation, with and without dynamic NN augmentation, in the presence of structural damages and control surface faults. The evaluation was based on performance measurements and pilot ratings. As reported in [26, 27], for structural damages, NN augmentation was generally found to provide significant improvements in overall pitch performance. However, control surface faults led to mixed results from slight improvements in pitch rate response to a propensity for roll pilot-induced oscillation. A modification was also introduced in the designed control scheme employed in the second phase of the program. The utilized modifications included the use of alternate NN inputs in the de-

signed framework which can satisfactorily tackle high-correlation and high-gain problems in the basic design, the adoption of a weight decay term (in updating the NN weights) to avoid the overfitting problem, and using *scalar* dead-zones in adaptation laws for simplicity. The results obtained by the modified control scheme in 2009 indicated a significant improvement over the basic design [27]. With the retirement of the F-15B air vehicle in January 2009, the IFCS program was finally finished [22].

The lesson learned from the IFCS program demonstrated that the high complexity of the control design, as well as the unpredictable behavior of the control scheme in the presence of unexpected flight and fault conditions, could be serious concerns in utilizing adaptive control approaches in real applications, particularly manned aircraft [28, 29]. Another program has been accordingly launched by NASA in 2009, namely the Integrated Resilient Aircraft Control (IRAC) project, where one of its main objectives was to investigate simple, yet effective, adaptive control methods to address the issue of verification and validation of adaptive flight controls to a safety-critical level. Addressing this project in more detail is beyond the scope of this paper. Motivated by the above discussion, in this survey, we will address both indirect and direct NN-based adaptive control schemes and their evolution towards more reliable approaches with less computational complexity, in detail. However, as will be discussed in the following, although the indirect and direct adaptive control approaches arise from two different points of view, they can be formulated within a similar mathematical framework with the same updating rules for dynamic NNs in the case of model-based approaches as in the IFCS project of NASA. Accordingly, we will deal with IFCSs in a different primary categorization, i.e. the model-based and model-free NN-based control methods. In addition, in the current research, we will address a variety of model-free flight control systems that have been built upon some other machine learning approaches such as Reinforcement Learning (RL).

1.2. Direct versus indirect adaptive control

There are a variety of flight control systems in the literature in which neural networks have been utilized to solve an online optimization problem within the control block [30, 31] or to mimic the behavior of a classical controller [32]. In this paper, we will focus on control methods where neural networks have been directly adopted in the control design procedure as *intelligent elements* to bring a degree of intelligence into the closed-loop behavior. This can be performed in different manners: Various studies have attempted to employ NNs to estimate model uncertainties, which are subsequently utilized in designing the control command. Such a framework is known as *indirect* neural control. The training process of NNs can be performed online using well-known learning algorithms such as feedback error learning or offline to provide a pre-trained dynamic model of the system. On the other hand, in the *direct* neural controller, NNs have been utilized to directly construct the control command.

To be more specific, in the case of model-based control approaches, if the system dynamics can be formulated as $\dot{x} = f(x)+g(x)u$, where $f(x)$ and $g(x)$ denote unknown nonlinear functions of system states, we have two general choices to design the control command u. In the first approach, we can estimate both the unknown functions f and g using NNs and then employ them in the designed control command. In the second approach, however, we attempt to directly design the control command by estimating $g^{-1}(\dot{x}_d - f)$, which is required in the control command, using a NN (x_d represents the reference trajectory). In the literature, the first method is known as an indirect adaptive control, while the second approach corresponds to a direct adaptive control scheme [33, 34]. However, as will be discussed in Section 2, concerning the mathematical formulation and the structure of updating rules of NNs, there is no fundamental difference between these two model-based control approaches.

On the other hand, in the case of model-free adaptive control methods, it is not easy to provide a general view of different types of indirect and direct adaptive control schemes employed in the literature. As one of the most commonly used model-free indirect IFCSs, (different types of) NNs are used to identify

the entire system dynamics, and subsequently an adaptive control scheme is designed based on the online identified model. In addition, the direct model-free control could be originated from various design methodologies, while in the current survey, two more common schemes, i.e. the adaptive optimal and the RL-based control methods will be discussed in detail.

Although due to the simpler structure and less computational complexity [35], direct adaptive control has been widely employed in different applications, there are various concerns regarding its applicability in serious missions. Briefly, increasing the learning rate in direct adaptive control, known as aggressive learning, is a typical approach to rapid reduction of the dynamic inversion error [36]. In this regard, high-gain control due to aggressive learning in direct adaptive control is a problematic issue which can lead to actuator saturation, the excitation of unmodeled dynamics, and other well-known problems of high learning rates [37]. Besides, in the case of a damaged aircraft, the system dynamics can significantly change, and the lack of reliable knowledge about the current system dynamics may result in inefficient control commands, particularly, when the control system consists of a *nominal* controller augmented by an adaptive NN-based control command [38].

1.3. Model-based versus Model-free control

In this section, we present a more detailed classification of NN-based flight control systems to be used in the remainder of the paper. Neural networks have been extensively employed in the structure of flight control systems for the past three decades [39]. They can be generally studied in two fundamentally different categories, i.e. the model-base and the model-free control approaches.

1.3.1. Model-based approach

The model-based neural control, which utilizes a nominal model of the system in the control design process, has significantly evolved during the last two decades. Feedback Error Learning (FEL) is the most popular learning scheme, which has been widely incorporated in intelligent control systems. By employing

the tracking error of the system, the prediction error of the model, the output of a baseline controller, or a combination of them, an unsupervised learning approach is developed in such a way that both the tracking error of the system and the estimation error of the neural networks remain bounded (unsupervised learning occurs when the NN is trained to respond to a certain pattern in the absence of output examples [40]). Several variations of FEL-based IFCSs have been proposed by researchers in recent years. In this method, the neural network attempts either to estimate the model uncertainties (or/and external disturbances) or to determine the control command. The first approach leads to an indirect control structure, while the second one results in a direct adaptive control scheme. In addition, different types of feedforward or recurrent neural networks including Radial Basis Function (RBF) neural networks, multilayer perceptron, High-Order NNs (HONNs) [41], Extreme Learning Machine (ELM) [42], Elman NN, etc have been employed in FEL-based control methods. The FEL scheme, its characteristics, and different variants within the flight control framework will be intently studied in Section 2.

One of the main drawbacks of FEL-based control systems is that all the uncertain terms in the model are typically estimated as a single term using NNs. This may result in poor training performance, particularly in the lack of Persistent Exciting (PE) signals. In addition, in most of the FEL-based neural controllers, a baseline control, which is designed based on a nominal model of the system, is employed where the NN acts as an aid to the controller. This may cause large control actions under severe structural damages or dynamic changes. Such concerns and other design considerations result in incorporating several features in the basic FEL-based control methods, which will be addressed in Section 3.

1.3.2. Model-free approach

On the other hand, the model-free scheme does not require any prior information about the system dynamics to be used in the control design procedure. As a traditional model-free approach, the *entire* controller was modeled by a

single NN [43, 44], where the error back-propagation technique was typically utilized to train the NN. Although such a control scheme, in some cases, could provide an acceptable response even under severe external disturbances [45], the stability of the closed-loop system could not be mathematically analyzed [46]. In addition, such a training method may occasionally converge to local minima. Thus, this control approach could not be safely utilized in important missions.

In recent years, a class of model-free intelligent control systems has been proposed in the literature using the concept of Approximate Dynamic Programming (ADP) and Reinforcement Learning (RL). Indeed, although the introduced schemes have originated from different scientific points of view (from the control theory to machine learning and information theory), the principal methodology employed by them are fundamentally similar. More specifically, in many of such control structures, an actor-critic framework is defined (where NNs may be used to estimate both the actor and critic functions). In this regard, the critic corresponds to the cost-to-go function (or the value function), while the actor determines the control input applied to the system. With a focus on the RL framework, the entire control design process is typically transformed to a Markov Decision Process (MDP). Accordingly, the value function demonstrates an accumulative discounted reward function obtained by the system (from the present time) using the current policy. The control objective is then to endeavor to maximize the value function by changing the policy function. This can be performed using the conventional policy-gradient method. Owing to the advancement of the computational power of processors, different RL-based IFCSs have been proposed in few recent years, where the control design process has been completely performed in the simulation environment and subsequently, the designed controller is satisfactorily applied to a real application. Such a control methodology will be discussed in detail in Sections 4.2 and 4.3.

There is also a variety of model-free indirect IFCSs in literature in which a separate identification process has been defined in the control design procedure. Different types of neural networks including nonlinear autoregressive with exogenous inputs (NARX) network [43], Elman networks (as recurrent NN), Con-

volutional NN, Wavelet NN, ELMs [47], and Fuzzy NNs [48] have been utilized in the identification step to identify different unmodeled dynamics. The identified models can also be updated online in order to adapt to dynamic changes in the system. Subsequently, the control system is designed based on the identified NN. Although the analysis of the closed-loop stability in such a multi-step control design process may be more challenging, this can result in a more efficient control system in comparison with FEL-based control methods, especially in the presence of severe dynamic changes. We will address this type of indirect IFCS in Section 4.1.

Finally, a set of concluding remarks and possible future directions for NN-based IFCSs will be provided in Section 5, which indeed attempts to illustrate the main existing gaps in the framework of IFCSs to be employed in serious missions as a reliable, effective, and really intelligent control scheme with acceptable computational cost.

2. Foundations of model-based intelligent control

In this section, we deal with model-based NN-based flight control systems. Regarding the dynamic model of air vehicles, they can be categorized in different manners. As a general classification, an air vehicle can be modeled as a nonlinear affine or nonaffine dynamic model. Concerning the affine model, we will pay more attention to two types of more popular dynamic models: dynamic models with $dim(x) = dim(u)$ (Section 2.1) and the dynamic models in the strict feedback form (Section 2.3). Most of the current approaches in the literature to control the aerial vehicles attempt to transform the system dynamics into one of the above-mentioned variants by defining intermediate control variables, designing multi-loop control systems, etc, where such techniques will also be briefly discussed. It is notable that, both the continuous-time and discrete-time models will be covered, as well. In addition, the control of nonaffine systems, mainly using the pseudocontrol strategy or similar methods, will be discussed in Section 2.2.

Besides, regarding the consideration of model uncertainties, internal faults, and external disturbances, it should be noted that they can be modeled using either additive or multiplicative uncertain terms. Although both schemes have been utilized in the literature, the employment of additive uncertain terms is more general than the multiplicative case [49, 50]. Indeed, multiplicative uncertainties can also be modeled using additive terms, though the unknown terms may become a function of both the system inputs and states [51]. Accordingly, in the following, we mainly focus on the control of dynamic systems with additive uncertain terms and will transform the possible multiplicative uncertainties into additive terms. In addition, here, we will consider uncertain terms in the dynamic model as a lumped disturbance, which will be estimated by a NN. However, dealing with different types of uncertain dynamics such as model uncertainties, atmospheric disturbances, and operational faults may necessitate different learning strategies with their own requirements, which are addressed in the following section. More specifically, combined approaches that utilize a combination of NNs, disturbance observers, and/or state estimators to tackle different uncertain terms in the system dynamics will be discussed in Sections 3.4 and 3.5. Further, as the consideration of the multiplicative representation of uncertain terms can be more beneficial in case of identification of unknown gains corresponding to *actuator faults*, we will also address this type of model uncertainties in the framework of NN-based Fault-Tolerant Control (FTC) systems in Section 3.5.

2.1. Feedback Error Learning

Here, we will introduce the fundamental theoretical basis for the most commonly used approach to incorporate NNs within the adaptive control design process, i.e. the FEL method, while the application of such an algorithm in flight control systems would be addressed in the following subsections. FEL can effectively integrate the control design procedure and the online updating law for the parameters of the NN, which is utilized to compensate for model uncertainties and disturbances. Accordingly, in a general view, the control block

can consist of a conventional controller in the inner loop to stabilize the system dynamics, and the neural controller acts as an aid to the controller to compensate for model uncertainties. Thus, employing a composite Lyapunov function including both the tracking error and the estimation error of NN parameters, the closed-loop system can satisfy the Bounded-Input–Bounded-Output (BIBO) stability requirement in the presence of model uncertainties and external disturbances. To be more precise, consider the dynamic model of an aircraft (in the affine form) as follows:

$$\dot{x} = F(x) + B(x)u + \Delta, \tag{1}$$

where $x, u \in \mathbb{R}^n$, and Δ stands for model uncertainties and external disturbances. Defining the desired trajectory as x_d, the tracking error is obtained as $e = x - x_d$. Now, the control command can be defined as

$$u = B^{-1}\left(-F(x) - \Delta + \dot{x}_d - k_1 e\right), \tag{2}$$

where k_1 is a positive-definite matrix. However, the vector Δ is unknown. Thus, it is approximated by a NN (such as RBFNN or multilayer perceptron) as $\hat{\Delta} = \hat{W}^T \mu(x)$, where μ represents the vector of basis functions (corresponding to hidden layers of the NN) and W indicates the matrix of unknown weights which should be identified. Such a formulation can be used to represent different feedforward and recurrent NNs. Here, we will use such a general formulation, and for brevity, do not address different possible network structures and their advantages and disadvantages in flight control systems (for more details, see [52]). Accordingly, due to the universal approximations property of NNs, we have:

$$\Delta = {W^*}^T \mu(x) + \varepsilon, \tag{3}$$

where W^* denotes the (unknown) optimal weight and ε indicates the *bounded* approximation error ($\|\varepsilon\| \leq \varepsilon_M$). The control command can now be constructed as follows:

$$u = B^{-1}\left(-F(x) - \hat{\Delta}(x) + \dot{x}_d - k_1 e\right). \tag{4}$$

Now, consider a Lyapunov function as

$$V = \frac{1}{2}e^T e + \frac{1}{2}tr\left(\tilde{W}^T\Gamma^{-1}\tilde{W}\right), \tag{5}$$

where $\tilde{W} = \hat{W} - W^*$ and Γ is a positive definite matrix. Next, we have

$$\begin{aligned}\dot{V} &= e^T\dot{e} + tr\left(\tilde{W}^T\Gamma^{-1}\dot{\tilde{W}}\right)\\ &= -e^T\left(\tilde{\Delta} + k_1 e\right) + tr\left(\tilde{W}^T\Gamma^{-1}\dot{\tilde{W}}\right),\end{aligned} \tag{6}$$

where, $\tilde{\Delta} = \hat{\Delta} - \Delta$. Accordingly, defining

$$\dot{\hat{W}} = \Gamma\mu e^T, \tag{7}$$

and considering $\dot{\tilde{W}} \approx \dot{\hat{W}}$ (as a consequence of assuming a *constant* optimal weight W^*, while such an assumption is reasonable even in the case of a time-dependent uncertain term $\Delta = W^{*T}(t)\mu(x) + \varepsilon$ with $\dot{W}^* \ll \dot{\hat{W}}$), we have

$$\begin{aligned}\dot{V} &= -e^T\left(\tilde{\Delta} + k_1 e\right) + tr\left(\tilde{W}^T\Gamma^{-1}\dot{\tilde{W}}\right)\\ &= -e^T\left(\tilde{\Delta} + k_1 e\right) + tr\left(\tilde{W}^T\mu e^T\right)\\ &= -e^T\left(\tilde{\Delta} + k_1 e\right) + e^T\tilde{W}^T\mu\\ &= -k_1 e^T e + e^T\varepsilon = -e^T(k_1 e - \varepsilon),\end{aligned} \tag{8}$$

which leads to $\dot{V} < 0$ for $\|k_1 e\| > \|\varepsilon\|$ thereby guaranteeing the bounded tracking error. The determination of optimal design parameters such as k_1 and Γ is not an easy task, where it is typically done by trial and error. It is also possible to define an optimization problem in terms of these parameters and solve it using well-known optimization methods (such as evolutionary algorithms) to determine their optimal values according to predefined criteria [53]. In cases where the matrix B is also uncertain, we have:

$$\dot{x} = F(x) + (B + \Delta B)u + \Delta, \tag{9}$$

where B represents the nominal part. Thus, it is possible to define

$$\bar{\Delta} := \Delta B u + \Delta \tag{10}$$

and estimate it using an NN as $\hat{\Delta}(x,u) = \hat{W}^T \mu(x,u)$. Consequently, the control command can be calculated as follows:

$$u = B^{-1}\left(-F(x) - \hat{\Delta}(x,u) + \dot{x}_d - k_1 e\right). \tag{11}$$

As seen, the control command results in an equation as $u = h(.,u)$. The existence and uniqueness of a solution for u require a contraction assumption [54]. Sufficient conditions for satisfying this assumption are given in [55]. Notably, this assumption implicitly requires the sign of the control gain function to be known [56]. Note that, it is also possible to update the weights of the hidden layer to provide more effective learning. This can be performed using a similar Lyapunov stability analysis by taking advantage of the Taylor expansion of the hidden layer output ($\mu(x)$) [57]. However, due to the more complicated formulation and excessive computational burden, in this paper, we will only update the output weights, $\hat{W}$, and the other parts of the NN remain unchanged.

Besides, one can replace the tracking error e in (5) and (7) by a filtered tracking error as $s = e + \lambda \int e \; dt$ (with λ denotes a positive constant or a positive definite matrix) to compensate for the steady-state tracking error [58]. Further, the introduced FEL neural control scheme can be applied to a second-order system, i.e. $\ddot{x} = F(x) + B(x)u + \Delta$ by substituting e by a filtered tracking error as $s = \dot{e} + \lambda e$ [59, 60].

On the other hand, as the designed controller should be programmed on a digital processor in real applications, the development of a control system in the discrete-time domain makes more sense. Using a discrete-time controller, the dependence of the closed-loop *performance* on the sampling rate can also be eliminated. This would be more beneficial in the case of NN-based control systems in which the differential equations for updating the NN weights change to difference equations. Furthermore, in the case of discrete-time controllers, the NN weights' updating rate that guarantees the convergence of the training rule can be computed analytically [61]. To illustrate the fundamental structure of a discrete-time FEL scheme, consider the equivalent discrete-time model of

(1) as follows:

$$x(k+1) = F_d(x(k)) + B_d(x(k))u(k) + \Delta_d(k). \tag{12}$$

Defining

$$u = B(k)^{-1}\left(-F_d(k) - \hat{W}^T\mu(k) + ce(k) + x_d(k+1)\right), \tag{13}$$

$$\Delta_d(k) = W^{*T}\mu(x(k)) + \varepsilon, \tag{14}$$

$$e(k) = x(k) - x_d(k), \tag{15}$$

where $0 < c < 1$ leads to the following equation.

$$e(k+1) = ce(k) - \tilde{W}^T\mu(k) + \varepsilon. \tag{16}$$

By multiplying both side of (16) by $e^T(k+1)$, we have

$$e^T(k+1)\tilde{W}^T\mu(k) = ce^T(k+1)e(k) - e^T(k+1)e(k+1) + e^T(k+1)\varepsilon. \tag{17}$$

Using Cauchy–Schwarz and Young's inequalities, it is obtained that

$$e^T(k+1)\tilde{W}^T\mu(k) \leq \|e(k+1)\|^2\Big(-1+\rho_1+\rho_2\Big) + \frac{c^2}{4\rho_1}\|e(k)\|^2 + \frac{1}{4\rho_2}\varepsilon_M^2, \tag{18}$$

with $\rho_1, \rho_2 > 0$. Thus, if a Lyapunov function is defined as (5) (without the coefficient 1/2), using the following updating rule,

$$\hat{W}(k+1) = \hat{W}(k) + \Gamma\mu(k)e^T(k+1), \tag{19}$$

the first difference of $V(k)$ is obtained as follows:

$$\begin{aligned}\Delta V(k) = V(k+1) - V(k) &= \|e(k+1)\|^2 - \|e(k)\|^2 \\ &\quad + tr\left(\tilde{W}^T(k+1)\Gamma^{-1}\tilde{W}^T(k+1) - \tilde{W}^T(k)\Gamma^{-1}\tilde{W}^T(k)\right) \\ &\leq \|e(k+1)\|^2\Big(1+2(-1+\rho_1+\rho_2)\Big) + \|e(k)\|^2\Big(\frac{c^2}{4\rho_1} - 1\Big) \\ &\quad + \frac{1}{4\rho_2}\varepsilon_M^2 + \|\mu\|_\Gamma^2\|e(k+1)\|^2.\end{aligned} \tag{20}$$

Accordingly, we have

$$\Delta V(k) \leq -k_1\|e(k+1)\|^2 - k_2\|e(k)\|^2 + c_1, \tag{21}$$

where,

$$k_1 = 1 - 2\rho_1 - 2\rho_2 - \|\mu\|_\Gamma^2, \tag{22}$$

$$k_2 = 1 - \frac{c^2}{4\rho_1}, \tag{23}$$

$$c_1 = \frac{1}{4\rho_2}\varepsilon_M^2. \tag{24}$$

Thus, assuming the boundedness of $\mu(x)$, it is possible to determine ρ_1, ρ_2, c, and Γ such that $k_1, k_2 > 0$ thereby guaranteeing $\Delta V(k) < 0$ for $\|e(k+1)\|^2 > c_1/k_1$. As seen, although the updating rule (19) is similar to that of continuous-time systems, i.e. (7), the stability analysis of the discrete-time FEL neural control is quite different and more complex compared to that of continuous-time systems. Consequently, in the following, we mainly focus on the continuous-time formulation of control systems, while their discrete-time equivalent would be obtained in a similar manner as discussed above.

It should be noted that, in the introduced (continuous/discrete) adaptive control scheme, the convergence of the NN weights to their ideal values is not trivial, and it requires a Persistent Excitation (PE) [18]. More precisely, in the absence of persistent exciting input signals, the NN weight estimates might drift to very large values, which will result in a variation of high-gain control [62, 63]. Different approaches have been proposed in the literature to prevent parameter drift in such conditions. Some of the more common methods are briefly introduced in the following.

1. Dead-zone: In this straightforward method, the previously mentioned updating rule is only used when the tracking error exceeds a predefined threshold [64]. Otherwise, the NN weights remain constant. Although such a method can successfully prevent parameter drift, as discussed in [65, 66], the determination of an appropriate threshold requires the bounds of the control gain function and the NN estimation error (ε_M), which may not be generally known.
2. Projection: The second simple method is to limit the NN weights to a predefined interval. It means that the time derivative of the parameters

is set to zero when they reach the given bounds [67]. The main drawback of this method is the requirement of the lower and upper bounds of the NN parameters.

3. Sigma-modification: The third method, which has been introduced by Ioannou and Kokotovic [68, 69] is a more useful approach. In this method, a modification term is incorporated in the updating rule of the NN parameters as $\dot{\hat{W}} = \Gamma\left(\mu e^T - \sigma W\right)$, where σ is a positive constant [70]. Such an approach has been employed in many NN-based flight control systems such as [71, 63, 72, 73].
4. e-modification: Another popular approach has been introduced in [74], where the constant parameter σ in the previous technique is replaced by a term proportional to $|e|$ [62, 75]. The boundedness of the NN parameters using e-modification has been presented in [76]. Further, as a major advantage of the e-modification technique over the σ-modification, such a modification term is effectively attenuated by approaching the tracking error to zero, and (in the lack of the estimation error ε,) this method does not affect the convergence of the NN weights to their ideal values in the presence of persistent exciting training signals.
5. Alternate weights: This approach has been first proposed in [77]. The e-modification method may not achieve acceptable performance in the presence of large oscillatory disturbances [78]. The basic idea of this method is that different sets of NN weights are capable of uniformly approximating the same nonlinear function. An alternate set of weights with a smaller magnitude than $\hat{W}$ can be used to improve the training. By keeping the NN weights close to the smaller alternate weights, it is possible to provide a more efficient compromise between the approximation performance and keeping the NN weights bounded, while there is a need for two distinct sets of NN weights and their corresponding updating rules. This method has been employed in [79] to design a flight control system for a quadrotor air vehicle under the wind buffeting.

Although the aforementioned approaches result in bounded NN parameters, satisfactorily, they do not ensure the convergence of the NN weights to their ideal values. Recently, a variety of modified learning approaches have been proposed in the literature, which causes the improvement in training the NN parameters. Some of the more attractive methods are as follows:

1. Composite learning: Different composite learning approaches have been introduced in the literature where their fundamental idea is to include the estimation performance into the updating law [80, 81, 82]. This can lead to faster learning speed as well as higher precision [83]. More specifically, the state estimation can be constructed as $\dot{\hat{x}} = F(x) + B(x)u + \hat{W}^T\mu(x) - \beta\tilde{x}$, where $\tilde{x} = \hat{x} - x$, and β is a positive constant. Thus, the updating rule (7) can be modified as $\dot{\hat{W}} = \Gamma\mu(e^T - \Gamma_1\tilde{x}^T)$ where Γ_1 is a positive definite matrix [83]. An improved learning method has been presented in [34], where the basic updating rule (7) is augmented by a novel prediction error signal constructed using online recorded data within a time interval $[t-\tau, t]$, which is equal to $\tilde{W}^T \int_{t-\tau}^{t} \mu(x) + \int_{t-\tau}^{t} \varepsilon$. As shown in [34], the proposed approach, which has been applied to the longitudinal model of a hypersonic aircraft, can lead to better tracking with less chattering.
2. Concurrent learning: A beneficial approach, which has been introduced in [84, 56], utilizes a set of recorded data points concurrently with instantaneous data to improve the convergence of both parameter and tracking errors. The main benefit of the concurrent learning method is that PE or high adaptation gains are not required. More precisely, in the case of nonlinear systems with *parametric* uncertainty (which can be modeled as $\Delta(x) = W^{*T}\mu(x)$), it has been proved that, if the training input signal is exciting in a sufficiently large finite time interval, both tracking error (e) and NN weight's estimation error ($\tilde{W}$) converge exponentially to zero. However, this approach requires precise estimation of the time derivative of the system states, which may be impractical in some cases.
3. Reinforced learning: Another approach to improve the training perfor-

mance is to reinforce the learning signal [85, 86]. This can be done by modifying the training rule (7) using the output of another NN (commonly known as the critic network) as $\dot{\hat{W}} = \Gamma\mu\left(e + \|e\|\hat{W}_c^T\mu_c\right)^T$ where $\hat{W}_c$ represents the output weights of the critic network, which is tuned in such a way that guarantees the closed-loop stability [87]. The provided learning signal is more informative than the basic training rule (7) thereby strengthening the control performance [88].

The above-mentioned formulation corresponds to the *indirect* FEL-based control where a NN attempts to identify model uncertainties, and then the control command is constructed using the estimated uncertainty [89]. FEL can also be satisfactorily employed in the framework of direct adaptive control systems. In this regard, it is possible to directly estimate the entire control command u or the uncertain term $B^{-1}\Delta$ in (2) by a NN. As a result, the updating rule of $\hat{W}$ will include the control gain matrix B. However, there is no fundamental difference between the direct and indirect approaches regarding the formulation of the updating rules and the stability analysis of the closed-loop system. Direct methods may also be preferred in cases where the control gain function is entirely unknown (see Section 2.3.4).

2.2. Pseudocontrol strategy

In a somewhat similar manner to the introduced direct FEL-based neural control, in the case of *nonaffine* models, traditionally, the output of a baseline controller (such as a PID controller) may be used to train a NN, which augments the output of the baseline controller to learn the inverse dynamics of the system, while there is considerable complexity in the closed-loop stability analysis [90]. Accordingly, an auto-landing scheme has been proposed in [91] for an aircraft under external disturbances using a FEL-based neural aided H_∞ control. Similarly, a combination of a classical trajectory tracking control (using the loop shaping technique) with a FEL-based neural controller has been employed in [92] as a fault-tolerant auto-landing control method. Such a method has also

been adopted in [93] to control the attitude of a simplified model of a fighter aircraft using fully-tuned growing RBFNNs.

By incorporating a similar framework, type-2 Fuzzy Neural Networks (T2-FNN) have been employed in [94, 95] to augment a classical PD controller in the case of a set of SISO systems. FEL algorithm has been adopted, where the updating rule corresponding to the consequent part of the T2-FNN has been derived by minimizing $\int (\dot{e} + \lambda e)^2 \, dt$. Although in [94], it has been assumed that the intended system has a second-order stabilizable dynamic model, the stability of the closed-loop system has been analyzed without making any assumption on the system characteristics (even the system's stabilizability!). Apparently, this is a consequence of estimating an *explicit function of time* by a neural network as ${W^*}^T \mu(x)$, which is not generally feasible. This is a common issue in NN-based identification schemes (see Section 3.4). The proposed control system has been applied to the trajectory tracking control of a quadrotor UAV. A self-organizing neuro-fuzzy-based control has been introduced in [96] in which the consequent part of the fuzzy rules has been trained using a similar FEL scheme, where the designed controller has been applied to a hexacopter and a flapping-wing Micro Aerial Vehicle (MAV) to control the altitude and the attitude of the system. It has been claimed that the controller's performance does not depend on *any* features for the system. This is clearly an exaggerated statement since the most obvious feature required in a controlled system is the system's controllability. Again, it seems that the generality of the stability analysis is due to the aforementioned concern regarding the NN-based identification schemes. Different from [96], in [97], the stability analysis has been provided for a nth order, affine, SISO model. A FEL scheme has been used to train the consequent parameters of a neuro-fuzzy control system, which augments a PID controller, while the updating rule is subject to the parameter drift issue.

On the other hand, a simpler and popular approach to FEL-based direct adaptive control of nonaffine systems, known as the *pseudocontrol* strategy, has been widely employed in IFCSs. Generally speaking, in this approach, the control command is determined using a model inversion block where a neural

network is utilized to cancel out the inversion error [98]. To be more precise, consider a generic nonaffine nonlinear model of the system as follows:

$$\dot{x} = F(x, u). \tag{25}$$

As seen, unlike the previous subsection, here, there is no need for an affine model of the controlled system. Despite the possible complexities in the control of nonaffine systems, the following design would be more effective in the case of nonconventional air vehicles with highly nonlinear dynamics, which could not be modeled as an affine model, satisfactorily. In particular, such a method could be an optimal choice in the case of an HFV, which possesses a completely nonaffine model [99]. Indeed, although, using some simplifications, HFVs are typically modeled as an approximate affine model (and the remaining nonlinear terms are treated as model uncertainty) to facilitate the control design, such an approach results in a conservative control system. It should be noted that, in the case of a flight control problem, the dynamic model of the system is typically formulated as $\ddot{x} = F(\dot{x}, x, u)$. However, the following design can be applied to such a control problem, as well, using a simple change of variables and employing a composite error function consisting of both e and $\dot{e}$. Now, assuming the availability of an approximate inversion model, the control command can be computed as follows:

$$u = \hat{F}^{-1}(x, \nu), \tag{26}$$

where ν denotes the pseudocontrol input, which should be designed. Notice that, although there is no need for an accurate inversion model, the chosen inversion model should capture the control assignment structures. It means that, for example, the inversion model should include the fact that the elevator deflection affects the pitch rate. In addition, it is assumed that $\hat{F}^{-1}(x, \nu)$ is a one-to-one function. This assumption can be realized if $dim(u) = dim(x)$ [100], which is reasonable in a typical flight control problem. Accordingly, the pseudocontrol input ν can be designed as follows [101]:

$$\nu = \dot{x}_d - ke - \nu_{ad}, \tag{27}$$

where k is a positive constant and ν_{ad} denotes an additional command to alleviate the inversion error. More precisely, defining $\Delta(x,u) = F(x,u) - \hat{F}(x,u)$, we have:

$$\dot{e} = -ke - \nu_{ad} + \Delta(x,u). \tag{28}$$

Thus, if it is possible to have $\nu_{ad} = \Delta(x,u)$, the tracking error converges asymptotically to zero. However, $\Delta(x,u)$ is unknown. So, we estimate it using the feedback error learning scheme. In this regard, using a NN to identify $\Delta(x,u)$, we have $\Delta(x,u) = W^{*T}\mu(x,u) + \varepsilon$. Subsequently, ν_{ad} can be determined as $\nu_{ad} = \hat{W}^T\mu(x,u)$. Introducing a Lyapunov function V as

$$V = \frac{1}{2}e^T e + \frac{1}{2}tr\left(\tilde{W}^T\Gamma^{-1}\tilde{W}\right), \tag{29}$$

and using the updating rule $\dot{\tilde{W}} \approx \dot{\hat{W}} = \Gamma\mu(x,u)e^T$, the time derivative of V is obtained as the following equation.

$$\begin{aligned} \dot{V} &= -e^T\left(ke + \tilde{W}^T\mu(x,u) - \varepsilon\right) + tr\left(\tilde{W}^T\Gamma^{-1}\dot{\tilde{W}}\right) \\ &= -e^T\left(ke - \varepsilon\right). \end{aligned} \tag{30}$$

Accordingly, the introduced control strategy can satisfactorily ensure the bounded tracking error. Again, one of the above-mentioned modification techniques can be adopted in the proposed updating rule to prevent parameter drift. Notice that the proposed control framework results in a control law as $u = \hat{F}^{-1}\left(x, \dot{x}_d - ke - \hat{W}^T\mu(x,u)\right)$, thereby requiring the contraction assumption. A modification to the introduced strategy has been given in [99] by taking advantage of the Mean Value theorem to relax this assumption, while the sign of $\partial F/\partial u$ should be known, and there are some concerns with the provided stability analysis. Besides, authors in [99] have employed the pseudocontrol approach in the case of a SISO system in the normal feedback form with $dim(x) > dim(u)$, while there is a requirement for the availability of all the system states in the proposed control scheme. To this end, if we have $z_1 = e = x_1 - x_{1d}$, $z_2 = \dot{z}_1$, and $\ddot{x}_1 = f(x,u)$, one can define a filtered tracking error $s = \dot{e} + \lambda e$, which results in $\dot{s} = f(x,u) - \ddot{x}_{1d} + \lambda\dot{e}$. Thus, by replacing the real tracking error

e with the filtered tracking error s in the introduced method, it is possible to design a similar pseudocontrol framework.

The pseudocontrol strategy has been employed in different flight control systems [102] such as the attitude control of a tailless fighter aircraft [103, 104, 105, 106], the trajectory tracking control of a helicopter [101], the attitude control of a tilt-rotor aircraft [75], etc. A similar direct adaptive control has been utilized in [107, 38] to control the trajectory of a conventional fixed-wing aircraft under structural damages. A hybrid direct-indirect adaptive control has also been developed in them in which parallel FEL algorithms attempted to provide both the control augmentation signal and estimated uncertain dynamics. In [100], an inner loop attitude control block based on the pseudocontrol strategy has been employed within a fault-tolerant guidance and control system for a conventional fixed-wing air vehicle. An acceleration (outer loop) guidance loop has been designed, which attempts to provide feasible acceleration command in the presence of structural damages and actuator faults. The performance of the proposed approach was verified in the presence of severe structural and actuator damages.

As an alternative to the above-mentioned approach to control nonaffine systems, authors in [108] attempted to directly estimate the desired control command rather than the inversion model error using a NN (in the case of a SISO system with stable zero dynamics). Under conservative assumptions on the value of $\partial F/\partial u$ (and its time derivative) and employing the implicit function theorem, one can assume that there is an ideal control command u^* which ensures the closed-loop system stability, i.e. $F(x, u^*) = \dot{x}_d - ke$. Subsequently, in lieu of utilizing an approximate inversion model, the Mean Value theorem has been adopted to provide an expression for $F(x, u)$ in terms of $F(x, u^*)$. Using such a formulation, a NN with a typical FEL scheme can be employed to estimate u^*. Although in this method, there is no need for an approximate inversion model of the system, different restrictive assumptions are required in the control design, which may not be satisfied in a practical flight control problem. A somewhat similar approach has been utilized in [109] in the framework

of indirect adaptive control, where the singular perturbation theory has been adopted to move u towards u^* as $\epsilon\dot{u} = F(x, u^*) - F(x, u)$ with ϵ denotes a small positive constant. Such a method has been employed to control the longitudinal model of an HFV, where a set of NNs has been incorporated to estimate unknown dynamics. In this regard, there is no need for a strict feedback model and a backstepping design, while, again, restrictive assumptions should be made to ensure closed-loop stability.

2.3. Neural backstepping control

In the basic FEL scheme, it was assumed that $dim(x) = dim(u)$. Also, in both the above-mentioned control structures, the entire dynamic model of the system is assumed invertible. However, in many cases, the dimension of the system inputs is less than that of the system states. The backstepping control method can be effectively employed in such circumstances when the dynamic model can be formulated in a strict feedback form. For simplicity, consider an uncertain nonlinear SISO system as follows:

$$\dot{x}_i = f_i(\bar{x}_i) + g_i(\bar{x}_i)x_{i+1} + \bar{\Delta}_i + d_i, \quad 1 \leq i \leq n-1, \tag{31}$$

$$\dot{x}_n = f_n(\bar{x}_n) + g_n(\bar{x}_n)u + \bar{\Delta}_n + d_n, \tag{32}$$

$$y = x_1, \tag{33}$$

where, $\bar{\Delta}_i$ and d_i stand, respectively, for model uncertainties and external disturbances and $\bar{x}_i = [x_1, \ldots, x_i]^T$. Without loss of generality, in the following, we assume that $n = 2$. The introduced control method can be simply applied to higher-order systems. Defining $\Delta_i = \bar{\Delta}_i + d_i$ and the desired output as y_d, we have:

$$\dot{e}_1 = \dot{y} - \dot{y}_d = f_1(\bar{x}_1) + g_1(\bar{x}_1)x_2 + \Delta_1 - \dot{y}_d. \tag{34}$$

Thus, a virtual control can be defined for x_2 as

$$x_{2d} = g_1^{-1}(\bar{x}_1)\left(\dot{y}_d - k_1 e_1 - f_1(\bar{x}_1) - \hat{\Delta}_1\right), \tag{35}$$

where $\hat{\Delta}_1$ represent the estimation of Δ_1 and k_1 is a positive constant. Therefore, defining $e_2 = x_2 - x_{2d}$, we have:

$$\dot{e}_2 = \dot{x}_2 - \dot{x}_{2d} = f_2(\bar{x}_2) + g_2(\bar{x}_2)u + \Delta_2 - \dot{x}_{2d}. \tag{36}$$

Finally, the control command can be defined as

$$u = g_2^{-1}(\bar{x}_2)\left(\dot{x}_{2d} - g_1(\bar{x}_1)e_1 - k_2 e_2 - f_2(\bar{x}_2) - \hat{\Delta}_2\right), \tag{37}$$

where $\hat{\Delta}_2$ denotes the estimation of Δ_2 and k_2 is a positive constant. Using feedforward NNs to estimate Δ_is, it is obtained that:

$$\Delta_i = W_i^{*T}\mu_i(\bar{x}_i) + \varepsilon_i, \tag{38}$$

such that $\|\varepsilon_i\| \leq \varepsilon_{M_i}$. To derive the updating rules of the NNs' parameters, one can define a Lyapunov function as follows:

$$V = \frac{1}{2}\left(e_1^2 + e_2^2 + \tilde{W}_1^T\Gamma_1^{-1}\tilde{W}_1 + \tilde{W}_2^T\Gamma_2^{-1}\tilde{W}_2\right). \tag{39}$$

The time derivative of V is obtained as

$$\begin{aligned}\dot{V} =& e_1\dot{e}_1 + e_2\dot{e}_2 + \tilde{W}_1^T\Gamma_1^{-1}\dot{\tilde{W}}_1 + \tilde{W}_2^T\Gamma_2^{-1}\dot{\tilde{W}}_2 \\ =& e_1\left(g_1(\bar{x}_1)e_2 - \tilde{\Delta}_1 - k_1 e_1\right) + e_2\left(-g_1(\bar{x}_1)e_1\right. \\ &\left.-\tilde{\Delta}_2 - k_2 e_2\right) + \tilde{W}_1^T\Gamma_1^{-1}\dot{\tilde{W}}_1 + \tilde{W}_2^T\Gamma_2^{-1}\dot{\tilde{W}}_2 \\ =& -k_1 e_1^2 - k_2 e_2^2 + \tilde{W}_1^T\left(\Gamma_1^{-1}\dot{\tilde{W}}_1 - \mu_1(x_1)e_1\right) \\ &+ \tilde{W}_2^T\left(\Gamma_2^{-1}\dot{\tilde{W}}_2 - \mu_2(\bar{x}_2)e_2\right) + e_1\varepsilon_1 + e_2\varepsilon_2.\end{aligned} \tag{40}$$

Thus, assuming $\dot{\tilde{W}}_i = \dot{\hat{W}}_i$, the updating rules of W_is can be defined as follows:

$$\dot{\hat{W}}_i = \Gamma_i\left(\mu_i(\bar{x}_i)e_i - \sigma_i\hat{W}_i\right), \tag{41}$$

where the second term on the right-hand side of the equation corresponds to the σ-modification. Using the updating rules (41), it is easy to show that $\dot{V} \leq -kV + C$ with k and C denote positive constant. As will be discussed in Section 2.4, this can ensure that all signals in the closed-loop system are uniformly ultimately bounded.

Such a control method can similarly be employed in cases where x_is are some vectors rather than scalar functions. A neural backstepping controller for an uncertain MIMO dynamic model of a helicopter has been introduced in [110] to control the attitude of the vehicle considering actuator dynamics, while each step of the design process deals with the control of a four-dimensional state vector. A neural backstepping controller has been adopted in [111] to control a planar VTOL air vehicle, where a gradient descent training algorithm has been replaced the updating rule (41). Although it has been claimed that such a training method results in better control performance, there is a need for the exact value of the uncertain term that is estimated by the NN, while in this paper, it has been computed by approximating the time derivatives of the system states and using the dynamic equations of the air vehicle.

It should be noted that although the proposed adaptive backstepping control leads to bounded tracking error in the presence of model uncertainties and external disturbances, it suffers from the *explosion of terms*. More precisely, the control command (37) includes the time derivative of $\dot{x}_{2d}$, which requires the time derivative of $g_1(x_1)$, $f_1(x_1)$, and $\hat{\Delta}_1$. This issue becomes more problematic by increasing the relative degree of the system.

2.3.1. Dynamic surface control

To solve the above-mentioned issue, Dynamic Surface Control (DSC) has been introduced in [112] in which the virtual control is passed through a first-order filter. More precisely, if x_{2c} is defined by (35), then the desired value of x_2 is obtained as

$$\tau \dot{x}_{2d} + x_{2d} = x_{2c}, \quad x_{2d}(0) = x_{2c}(0), \tag{42}$$

where τ represents the filter time constant. Subsequently, the filtering error is also incorporated in the Lyapunov function of the system to be compensated by the designed control commands. Using such a technique, the problem of the explosion of terms in the traditional backstepping control can be effectively avoided, though at the cost of reducing the *global* stability of the system obtained

using the backstepping control to the *semi-global* stability in the case of DSC [112].

Several NN-based DSC methods have been introduced in the literature for different aerial vehicles [113, 114, 115, 116]. Such an approach has been proposed in [117] to control the flight path angle and velocity of a flexible HFV, where the employment of the integral of the tracking error in the control law improves the tracking performance. DSC has been employed in [118] to control the attitude of a Near-Space Vehicle (NSV) in which recurrent wavelet NNs have been utilized at each step and trained using a composite learning method to compensate for external disturbances and model uncertainties. Also, such a scheme has been adopted in [119] to control the longitudinal dynamics of an air-breathing HFV considering model uncertainties and external disturbances compensated by fully tuned RBFNNs. In addition, DSC has been applied to the longitudinal mode of an HFV in [33]. In comparison with conventional DSC, which results in a semi-globally uniformly ultimately bounded stability, global tracking has been achieved through aggregating the neural function approximation and a robust term (using a switching function), which brings the system states into the neural approximation domain from outside. The robust term has been designed to estimate the upper bound of uncertain terms in a similar way as discussed in Section 3.4.2. However, the determination of the *active region* of NNs (which is required in designing the switching functions [82, 120]) is not trivial.

2.3.2. Command filtered backstepping

To simplify the stability analysis of DSC, a command filtered backstepping has been proposed in [121] for a nonlinear system without uncertainty. The introduced method attempts to eliminate the filter effects using a set of compensating signals. This idea has been extended to nonlinear systems with parametric uncertainties in [122]. To clarify the main idea, consider again the aforementioned control problem. Assuming that the virtual control signals x_{2c} and x_{2d} are defined, respectively, by (35) and (42), and by defining the auxiliary

variable ξ_1 as

$$\dot{\xi}_1 = -k_1\xi_1 + g_1(\bar{x}_1)\left(x_{2d} - x_{2c}\right), \quad \xi_1(0) = 0, \tag{43}$$

a compensated tracking error can be defined as $\epsilon_1 = y - y_d - \xi_1$. Thus, we have:

$$\begin{aligned}\dot{\epsilon}_1 &= \dot{y} - \dot{y}_d - \dot{\xi}_1 \\ &= f_1(\bar{x}_1) + g_1(\bar{x}_1)x_2 + \Delta_1 - \dot{y}_d + k_1\xi_1 - g_1(\bar{x}_1)\left(x_{2d} - x_{2c}\right) \\ &= -k_1\epsilon_1 + g_1(\bar{x}_1)e_2 - \tilde{\Delta}_1.\end{aligned} \tag{44}$$

Accordingly, the control command can be defined as

$$u = g_2^{-1}(\bar{x}_2)\left(\dot{x}_{2d} - g_1(\bar{x}_1)\epsilon_1 - k_2e_2 - f_2(\bar{x}_2) - \hat{\Delta}_2\right), \tag{45}$$

which leads to

$$\dot{e}_2 = \dot{x}_2 - \dot{x}_{2d} = -g_1(\bar{x}_1)\epsilon_1 - k_2e_2 - \tilde{\Delta}_2. \tag{46}$$

Thus, using the following updating rules

$$\dot{\hat{W}}_1 = \Gamma_1\left(\mu_1(\bar{x}_1)\epsilon_1 - \sigma_1\hat{W}_1\right), \tag{47}$$

$$\dot{\hat{W}}_2 = \Gamma_2\left(\mu_2(\bar{x}_2)e_2 - \sigma_2\hat{W}_2\right), \tag{48}$$

and defining a Lyapunov function as

$$V = \frac{1}{2}\left(\epsilon_1^2 + e_2^2 + \tilde{W}_1^T\Gamma_1^{-1}\tilde{W}_1 + \tilde{W}_2^T\Gamma_2^{-1}\tilde{W}_2\right), \tag{49}$$

it can be shown that $\dot{V} \leq -kV + C$, where k and C are positive constants. This results in bounded ϵ_1 and e_2. As discussed in [122, 123], assuming that $g_1(x_1)$ is bounded, it can be simply proved that ξ_1 is also bounded, thereby resulting in a bounded tracking error. A command filtered backstepping control has been designed in [123] for the longitudinal dynamics of an HFV considering input constraints and additive actuator faults. The control gain functions (g_i) have also been considered unknown, where it has been assumed that model uncertainties, as well as the control gain functions, can be written into a parametric form with partially unknown parameters. Considering the neural network-based

representation, the above assumption means that the residual terms ε_i in (38) are equal to zero, while in the case of complex air vehicles with nonparametric uncertainties, such an assumption becomes infeasible. Similarly, a command filtered backstepping control has been adopted in [124, 125] to control the trajectory of an F-16 fighter aircraft model with parametric uncertainties, where second-order filters have been used to impose both the magnitude and rate limits on the system states (see Section 3.6.2). Another analogous formulation has been presented in [80] in which the time derivative of ξ_i consists of ξ_{i+1}, where i represents the step of the backstepping control design process. Using this formulation, the control command u can be written in terms of e_1 rather than ϵ_1.

2.3.3. Backstepping augmented by the First-Order Sliding Mode Differentiators (FOSMD)

Another improved approach to approximate the time derivative of the virtual control signal x_{2d} is to employ a first-order sliding mode differentiator rather than employing a first-order filter. Using the FOSMD, the differentiation error tends to zero or a compact neighborhood of zero (depending on the signal's characteristics) after a finite-time transient process [126]. Considering a known function $l(t)$, the FOSMD formulation is obtained as follows:

$$\dot{\varsigma}_0 = -\varrho_0 \left|\varsigma_0 - l(t)\right|^{0.5} sign\left(\varsigma_0 - l(t)\right) + \varsigma_1, \tag{50}$$

$$\dot{\varsigma}_1 = -\varrho_1 sign\left(\varsigma_1 - \dot{\varsigma}_0\right), \tag{51}$$

where ς_0 and ς_1 represent the states of the differentiator, and ϱ_0 and ϱ_1 denote design parameters. Therefore, $\dot{\varsigma}_0 - \dot{l}(t)$ remains bounded if $\dot{\varsigma}_0(0) - \dot{l}(0)$ and $\varsigma_0(0) - l(0)$ are bounded.

This approach has been adopted in the backstepping control design in [34, 83] to control the longitudinal mode of an HFV. As shown in [34], using the FOSMD, the stability analysis is more concise compared to the traditional backstepping, DSC, and command filtered design. Besides, a neural backstepping control approach using FOSMD has been proposed in [127] for the longitudinal dynamic

model of a sweep-back wings morphing aircraft subject to input–output constraints. It is notable that higher-order sliding mode differentiators (HOSMD), which result in superior performance compared to FOSMDs [128], can also be employed in the structure of the neural backstepping scheme [129].

2.3.4. Direct neural-backstepping control

In addition to the above techniques, there are a variety of direct adaptive backstepping flight control systems in the literature, which can satisfactorily prevent the problem of the explosion of terms. To be more precise, consider again the nonlinear model (31)-(33) with $n = 2$. Defining

$$x_{2d}^* = g_1^{-1}(\bar{x}_1)\left(\dot{y}_d - k_1 e_1 - f_1(\bar{x}_1) - \Delta_1\right), \tag{52}$$

$$u^* = g_2^{-1}(\bar{x}_2)\left(\dot{x}_{2d} - g_1(\bar{x}_1)e_1 - k_2 e_2 - f_2(\bar{x}_2) - \Delta_2\right), \tag{53}$$

and using two distinct neural networks to identify them as $x_{2d}^* = W_1^{*T}\mu_1(\bar{x}_1) + \varepsilon_1$ and $u^* = W_2^{*T}\mu_2(\bar{x}_2) + \varepsilon_2$, we have:

$$x_{2d} = \hat{W}_1^T \mu_1(\bar{x}_1) = x_{2d}^* - \varepsilon_1 + \tilde{W}_1^T \mu_1(\bar{x}_1), \tag{54}$$

$$u = \hat{W}_2^T \mu_2(\bar{x}_2) = u^* - \varepsilon_2 + \tilde{W}_2^T \mu_2(\bar{x}_2). \tag{55}$$

Thus, considering a Lyapunov function candidate as (39), we have:

$$\begin{aligned}
\dot{V} = \;& e_1\dot{e}_1 + e_2\dot{e}_2 + \tilde{W}_1^T\Gamma_1^{-1}\dot{\hat{W}}_1 + \tilde{W}_2^T\Gamma_2^{-1}\dot{\hat{W}}_2 \\
= \;& e_1\left(g_1(\bar{x}_1)\left(e_2 - \varepsilon_1 + \tilde{W}_1^T\mu_1(\bar{x}_1)\right) - k_1 e_1\right) + \\
& e_2\left(-g_1(\bar{x}_1)e_1 - k_2 e_2 + g_2(\bar{x}_2)\left(\tilde{W}_2^T\mu_2(\bar{x}_2) - \varepsilon_2\right)\right) \\
& + \tilde{W}_1^T\Gamma_1^{-1}\dot{\hat{W}}_1 + \tilde{W}_2^T\Gamma_2^{-1}\dot{\hat{W}}_2 \\
= & - k_1 e_1\left(e_1 + \frac{g_1(\bar{x}_1)}{k_1}\varepsilon_1\right) - k_2 e_2\left(e_2 + \frac{g_2(\bar{x}_2)}{k_2}\varepsilon_2\right) \\
& + \tilde{W}_1^T\left(\Gamma_1^{-1}\dot{\hat{W}}_1 + \mu_1(x_1)g_1(\bar{x}_1)e_1\right) \\
& + \tilde{W}_2^T\left(\Gamma_2^{-1}\dot{\hat{W}}_2 + \mu_2(\bar{x}_2)g_2(\bar{x}_2)e_2\right).
\end{aligned} \tag{56}$$

Accordingly, by introducing the following updating rules,

$$\dot{\hat{W}}_i = -\Gamma_i\left(\mu_i(\bar{x}_i)g_i(\bar{x}_i)e_i + \sigma_i\hat{W}_i\right), \quad i = 1, 2, \tag{57}$$

and assuming that g_is are nonzero and *bounded*, again, it can be concluded that $\dot{V} < -kV + C$, which ensures that all signals in the closed-loop system remain bounded (see Section 2.4). As seen, despite the simpler formulation of the direct method compared to previously proposed indirect backstepping schemes, the boundedness of control gain functions (g_is) is necessary to guarantee the closed-loop stability.

It is notable that, using the aforementioned direct neural backstepping scheme, the control *singularity* problem in the control of dynamic systems with unknown g_is (induced by approaching $\hat{g}_i$s to zero) is also avoided [130]. A direct neural backstepping control has been designed in [130] to control the longitudinal mode of an air-breathing HFV with unknown g_is, where it is necessary to have $\overline{g}_i \geq g_i > 0$ ($\overline{g}_i$ denotes a positive constant). To this end, concerning the appropriate Lyapunov function candidate, $1/g_i$ is multiplied by $1/2e_i^2$ to eliminate the requirement for g_i in updating rules, and extra terms in $\dot{V}$ raised by this reformulation have been compensated by defining an appropriate ideal control command (u^*). Besides, a filtered tracking error including the integral of the error has been considered as the error function to remove the steady-state error, while the tracking error corresponding to the second step of the backstepping design was not considered in the first step.

On the other hand, by introducing an output feedback form and utilizing High Gain Observers (HGOs) to estimate the time derivatives of the system output (Section 3.1), it is possible to derive the control command with no requirement for a backstepping scheme. In this regard, a filtered tracking error is defined (as discussed in Section 2.2) to provide a unified error dynamic model. Such a method has been adopted in [131, 132] to control an HFV, where only one neural network is required in the altitude control block to determine the actual control command. In a somewhat similar manner, a direct neuroadaptive control scheme has been integrated with the funnel control method in [133] to control an air-breathing HFV considering non-affine dynamics. The altitude subsystem has been transformed into a simplified normal output feedback model, where only one NN is required to determine the control command. Fur-

ther, the non-affine dynamics of the vehicle have been handled by incorporating a low-pass filter in the last step of the design to define a new virtual control input in affine form.

In addition to the above-mentioned continuous-time backstepping control methods, several studies in the literature have addressed the design of a discrete-time neural backstepping controller. As discussed in Section 2.1, the general formulation of the NN updating rules in the discrete-time domain is similar to that of the continuous-time controller, while the stability analysis of the closed-loop system is quite different [134]. A discrete-time direct neural backstepping control has been proposed in [135] by incorporating HONNs to estimate uncertain terms in control commands. A similar control formulation has been given in [42] using Extreme Learning Machines (ELMs). To simplify the control structure, in [136, 137], dynamic equations corresponding to the altitude dynamics of an HFV have been aggregated into a prediction model as $x_1(k+n) = \bar{f}(x)+\bar{g}(x)u$, where $\bar{f}$ and $\bar{g}$ represent uncertain nonlinear functions and n denotes the system order. Subsequently, a single NN has been employed to tackle uncertain terms in the control command. Such a method can be considered as the discrete-time equivalent of the above-mentioned approach to control output feedback models in the continuous-time domain. Alternatively, an equivalent prediction model of an HFV has been defined in [41] in which $x_i(k+n-i+1)$ is obtained as a function of $x_{i+1}(k+n-i)$. Using such a change in the formulation of system dynamics, all the information of the desired trajectory in future n steps is involved in designing the controller, thereby improving the closed-loop performance. The designed controller in all these papers has been applied to the longitudinal mode of an HFV model.

2.4. How to analyze the closed-loop stability?

As mentioned before, most of the current feedback error learning-neural control schemes in the literature can only guarantee the Uniformly Ultimately Bounded (UUB) stability of the closed-loop system. To be more precise, it is not typically possible to prove the negative definiteness of the time derivative

of the Lyapunov function, but it can be proved that

$$\dot{V} \leq -kV + C, \tag{58}$$

where k and C denote positive constants. As a result, it is obtained that [138, 110]

$$V(t) \leq \left(V(0) - \frac{C}{k}\right) e^{-kt} + \frac{C}{k}. \tag{59}$$

In this regard, many studies in the literature attempted to propose a flight control system, which satisfies the local [57, 70], semi-global [131, 139, 140, 110, 141], or global [33] UUB tracking. There are fewer works that have addressed more stringent stability criteria including asymptotic, exponential, or finite-time stability.

2.4.1. Asymptotic stability

A variety of flight control systems are given in the literature, which can prove the convergence of the tracking error to zero as time tends to infinity. As a straightforward approach, if we can assume that the estimation error ε in (8) is zero (or negligible), it is obtained that $\dot{V} = -k_1 e^T e$, which guarantees the asymptotic convergence of the tracking error to zero. Such an assumption is reasonable in the case of dynamic systems with *parametric* uncertainties. More precisely, in such a circumstance, it is possible to estimate uncertain terms as $\Delta(x) = W^{*T}\mu(x)$, where $\mu(x)$ and W^* represent the vector of appropriate basis functions and the matrix of unknown weights, respectively. According to this, a constrained adaptive backstepping control scheme has been proposed in [124] for a strict feedback system with parametric uncertainties. A set of modified tracking errors ($\bar{z}_i$) have been defined (Section 3.6.2), and it has been proved that the time derivative of the Lyapunov function is obtained as $\dot{V} = -\sum c_i \bar{z}_i^2$ with c_i denotes positive constants. This leads to convergence of the modified tracking errors to zero as time tends to infinity [142], while the actual tracking error may increase if the control inputs are saturated. Finally, the proposed control scheme was employed in [124] to control the attitude of a simplified model of an F-16 aircraft with multi-axis thrust vectoring, considering actuator

faults and symmetric structural damages (only in the simulation phase). A similar approach has been presented in [125] to provide a trajectory tracking control for an F-16 fighter aircraft under parametric uncertainties and system constraints. An adaptive backstepping controller has been introduced in [143] for the attitude control of an NSV in the presence of model uncertainties and multiplicative actuator faults. For this purpose, first, an adaptive neural state observer has been proposed (Section 3.4), and subsequently, the estimated states have been utilized in a backstepping control scheme. The asymptotic stability of the system has been proved assuming that the NN estimation error is negligible. Obviously, such control approaches can not ensure the asymptotic stability of the system in the presence of *nonparametric* uncertainties and external disturbances, which are usually present in practical flight control problems.

Several studies have been reported in the literature in which NNs are combined with *discontinuous* feedback control methods such as variable structure or Sliding Mode Controllers (SMC) to guarantee the asymptotic stability of the closed-loop system. The fundamental idea of the combination of NN function approximation and robust terms (such as in SMC), which can result in the asymptotic stability of the closed-loop system, is given in Section 3.4.2. A robust output feedback control with neural network function approximation has been designed in [139] for the attitude and altitude control of a quadrotor UAV in the presence of model uncertainties and external disturbances, where the attitude dynamics are constructed in terms of the unit quaternion. Although the asymptotic stability of the closed-loop system has been proved, the proposed control command leads to high-amplitude and oscillatory thrust forces, and the chattering phenomenon due to the employment of the signum function in the control command. Indeed, such discontinuous controllers suffer from well-known limitations including a requirement for an infinite control bandwidth and chattering. Unfortunately, ad hoc fixes for these effects result in a loss of asymptotic stability [144]. An adaptive SMC has been proposed in [145], where the parameters of the sliding surface were trained by NNs through error back-propagation learning. The hyperbolic tangent function has replaced the signum function to

eliminate the chattering phenomenon, while the asymptotic stability of the system has been achieved by neglecting model uncertainties in the control design process.

To overcome the above-mentioned issues, a trajectory tracking control system has been introduced in [146] for a rotorcraft UAV using Robust Integral of the Signum of the Error (RISE) feedback [144], where a NN has been adopted to compensate for uncertain dynamics. The RISE control scheme is a differentiable control method that can compensate for additive disturbances and parametric uncertainties. By combining it with an NN-based FEL method, there is no need for linearity in the parameters to ensure the asymptotic stability of the system. The proposed approach was employed in [146] in a multi-loop control structure, where the desired attitude is determined in the outer loop and the attitude tracking control has been addressed in the inner loop. The semi-global asymptotic stability of the inner loop in tracking the desired attitude has been proved assuming that the first four time derivatives of the reference trajectory are bounded. Another alternative has been given in [138, 147], where the signum function of e has been substituted by $e/\left(\sqrt{e^2+\omega^2}\right)$ with $\omega(t)$ denotes a vanishing positive function satisfying $\int_0^\infty \omega^2(t)dt < \infty$. Accordingly, asymptotic tracking can be achieved, while the updating rules are subject to possible parameter drift. A class of NN-based optimal control methods with guaranteed asymptotic stability has also been introduced in the literature, which will be addressed in Section 4.2.

2.4.2. Exponential stability

As stated in [148], in the case of a flight control problem, few studies have claimed to achieve exponential convergence [149]. This is even less so regarding uncertain dynamic systems. As mentioned earlier, an improved learning method has been introduced in [56], which can ensure the exponential parameter and tracking error convergence for a specific class of single-input nonlinear systems with parametric uncertainties assuming that a precise estimation of the time derivative of system states is available.

2.4.3. Finite-time stability

Moreover, a variety of indirect NN-based flight control systems has been developed in the literature employing SMC-based methods, which can guarantee the finite-time or *practical finite-time* stability of the closed-loop system. The practical finite-time stability means that the tracking error converges to a small neighborhood of the origin in finite time [150]. To clarify the principal idea of the mentioned control methods, consider again the first control problem given in Section 2.1. Let's define a sliding manifold as

$$s = e + \eta \int_0^t sig^r(e)dt, \tag{60}$$

where $sig^r(e) = [sign(e_1)|e_1|^r, \cdots, sign(e_n)|e_n|^r]^T$, $\eta > 0$, and $0 < r < 1$. Now, if the control command is computed as

$$u = B^{-1}\big(\dot{x}_d - F(x) - \eta sig^r(e) - \hat{\Delta}(x) - k_1 s - k_2 sig^r(s)\big), \tag{61}$$

where the same NN function approximation as (3) with an updating rule as $\dot{\hat{W}} = \Gamma\left(\mu s^T - \sigma\hat{W}\right)$ is incorporated, using a Lyapunov function as

$$V = \frac{1}{2}s^T s + \frac{1}{2}tr\left(\tilde{W}^T\Gamma^{-1}\tilde{W}\right), \tag{62}$$

it is easy to prove the satisfaction of (58), thereby guaranteeing the boundedness of all signals in the closed-loop system. As a consequence, we have

$$\dot{s} \leq -k_1 s - k_2 sig^r(s) + \rho_M I \tag{63}$$

where I and ρ_M denote the identity matrix and a positive constant satisfying

$$\left\|\varepsilon - \tilde{W}^T\mu(x)\right\|_\infty \leq \rho_M. \tag{64}$$

Thus, by appropriate choice of k_1 and k_2, one can simply show the convergence of s to a compact neighborhood of the origin in finite time [151]. Such a method is extensively discussed within the framework of adaptive Terminal SMC (TSMC). Further, by incorporating a robust term into the control command (in a similar manner as discussed in Section 3.4.2) to compensate for ρ_M,

and replacing the first term in the Lyapunov function (62) by $\|s\|$, it is possible to guarantee the finite-time stability of the closed-loop system [152]. An FTC has been introduced in [152] to control the longitudinal mode of a conventional aircraft using a similar SMC, which ensures the finite-time convergence of s to zero under appropriate control gains. Self-constructing fuzzy neural networks have been utilized to estimate the bound of uncertain dynamics caused by actuator faults and model uncertainties, while the minimum estimation error ε under optimal network weights (W^*) has been neglected. A TSMC augmented by neural approximation and disturbance observers (as given in Section 3.4.2) has been presented in [151] for the trajectory tracking control of a quadrotor aerial robot under model uncertainties, input dead-zone, and external disturbances, where the practical finite-time stability of the system has been ensured. A sliding surface, which is equal to the time derivative of (60) due to the second-order dynamics of the controlled system, has been utilized in the proposed design. A formation flight control problem for a group of helicopter UAVs has been dealt with in [153] using an analogous control scheme. TSMC combined with neural approximation and the above-mentioned robust term has been adopted in both the position and attitude control loops, which can ensure, respectively, the finite-time and practical finite-time stability of the position and attitude tracking error, while the control loops have been decoupled based on the multiple time-scale assumption. The same problem has been addressed in [154] using TSMC in the inner control loop and an adaptive NN-based control scheme in the outer loop, though, again, the control loops have been analyzed separately. The inter-vehicle collision avoidance has also been solved by incorporating an exponential potential function into the design process. The proposed control structure can guarantee the practical finite-time stability of the closed-loop system, while it requires only the *relative* position of UAVs to their adjacent.

In addition to the aforementioned indirect adaptive control schemes, Direct T2-FNNs have been employed in [155, 48, 156] as an augmentation for a PD controller in the case of SISO nonlinear systems, where the network parameters are updated by an SMC-based algorithm (with a sliding surface as $s = \dot{e} + \eta e$)

using the output of the PD controller as the learning signal. The practical finite-time stability of the closed-loop system has been proved using a simple Lyapunov function as $V = 1/2s^2$ assuming that a PD controller can stabilize the system [48]. Six T2-FNNs have been used in [156] to control the trajectory of a 6-DOF quadrotor air vehicle, where each T2-FNN corresponds to a distinct system state. Due to the complexity of computing the gradient of the cost function with respect to the antecedent parameters of the FNN, particle swarm optimization has been adopted as a gradient-free approach to train them, while the consequent parameters have been updated using the mentioned SMC-based algorithm. Notice that, despite the superior convergence properties of the above-mentioned approaches, as discussed earlier, they are subject to considerable limitations of discontinuous control systems.

3. Supplementary features in model-based IFCSs

Several additional features may be required in different flight control systems due to different additional requirements. In this section, we will address various supplementary features, which have been widely incorporated in IFCSs. Again, the focus of this section is on model-based IFCSs, while some of the introduced elements such as self-organizing NNs can be effectively employed in model-free approaches, as well.

3.1. Output Feedback (OFB) control

The basic feedback error learning method has been developed assuming all the system states are measurable. This assumption is not feasible in many applications. Accordingly, different modifications have been introduced in the literature to effectively control an uncertain nonlinear system using only the system inputs-outputs. Such an intention is typically fulfilled by employing *state observers* [157], where a composite Lyapunov function is subsequently incorporated to compensate for both the tracking error and the state estimation error.

A common assumption in the design of OFB control methods is that the system is input-output linearizable with a specified relative degree [108]. More precisely, consider a SISO dynamic system as $\dot{x} = f(x)+g(x)u$, $y = h(x)$. Thus, we have:

$$\dot{y} = \frac{\partial h}{\partial x}(f(x) + g(x)u) = L_f h(x) + L_g h(x)u, \tag{65}$$

where $L_f h(x) = \frac{\partial h}{\partial x} f(x)$ is the Lie derivative of h along f [158]. Assuming the system has a relative degree ρ, we have $L_g L_f^{\rho-1} h(x) \neq 0$. Therefore, defining

$$u = \frac{1}{L_g L_f^{\rho-1} h(x)}(-L_f^{\rho} h(x) + \nu), \tag{66}$$

the dynamic model reduces to $y^{(\rho)} = \nu$ [159] (a similar definition can also be provided for a nonaffine system). Using such an assumption and assuming that the system is globally exponentially minimum phase, an adaptive OFB control has been introduced in [72] for a nonaffine SISO system with an unknown (but bounded) dimension. A linear observer of dimension $2\rho - 1$ has been developed to estimate the time derivatives of the tracking error signal. The estimated vector is then used as the training signal for a Single-Hidden-Layer (SHL) NN that attempts to compensate for the model inversion error in the framework of the pseudocontrol strategy. Under the same assumption, a backstepping control scheme has been designed in [108], where the time derivatives of y have been estimated using High Gain Observers (HGOs). Also, an adaptive neural network has been developed to construct the control command in the presence of model uncertainties. Finally, the proposed control scheme has been applied to a helicopter model to control the altitude of the vehicle in vertical flight. Similarly, HGOs have been used in [160] to provide the estimation of the time derivatives of Euler angles, which are required in the attitude control of a flapping-wing micro aerial vehicle. Analogously, a beneficial approach to control the longitudinal model of HFVs has been introduced in [131, 132]. As discussed earlier, typically, a backstepping control scheme is designed for an HFV in which the longitudinal dynamics are transformed into the strict feedback form. Alternatively, a new formulation has been given in [131, 132] to transform the altitude subsystem into

a normal output feedback form. More precisely, considering the longitudinal dynamics of an HFV and defining $z_1 = y = \gamma$, $z_2 = \dot{z_1}$, and $z_3 = \dot{z_2}$, we have

$$\dot{z_1} = z_2, \quad \dot{z_2} = z_3, \quad \dot{z_3} = a(X) + b(X)\delta_e, \quad y = z_1, \tag{67}$$

where $X = [\gamma, \theta, q]^T$ and a and b are unknown. Here, γ, θ, and q represent the flight path angle, the pitch angle, and the pitch rate, respectively. Accordingly, z_2 and z_3 are the time derivatives of the system output and are unknown. Utilizing an HGO, the system states $Z = [z_1, z_2, z_3]^T$ can be estimated by $\hat{Z} = [z_1, \frac{\xi_2}{\varepsilon}, \frac{\xi_3}{\varepsilon^2}]^T$, where

$$\dot{\xi_1} = \frac{\xi_2}{\varepsilon}, \tag{68}$$

$$\dot{\xi_2} = \frac{\xi_3}{\varepsilon}, \tag{69}$$

$$\dot{\xi_3} = \frac{-d_1\xi_3 - d_2\xi_2 - \xi_1 + y(t)}{\varepsilon}, \tag{70}$$

and ε is a small design constant and d_1 and d_2 are chosen such that $s^3 + d_1 s^2 + d_2 s + 1$ is Hurwitz. Consequently, there exist positive constants h_s and t_s such that $\forall t > t_s$, we have $|\hat{Z} - Z| \leq \varepsilon h_s$ [161]. Afterward, an NN-based control command has been developed in [131, 132] to ensure the convergence of a filtered tracking error to a small neighborhood of zero.

An NN-based observer (see Section 3.4) has been designed in [162] to estimate the angular and translational velocities of a quadrotor air vehicle, which are subsequently utilized in the control loop. On the other hand, non-model-based filters have been employed in [139] to provide an estimation of the unknown angular velocity, which was required in the proposed OFB control.

3.2. Minimal-learning parameter

One of the major drawbacks of neural networks in the structure of the feedback error learning scheme is the excessive computational burden of the training process due to the high number of parameters that should be identified. An efficient identification technique with significantly fewer training parameters, called the *Minimal-Learning Parameter* (MLP), has been widely employed by

researchers in recent years. MLP has been first introduced by Yang and his colleagues and employed in the traditional backstepping control combined with T-S fuzzy systems [163, 164] or RBFNNs [165]. Subsequently, it was effectively integrated with DSC [166] (to solve the problem of the explosion of complexity in classical backstepping control) and with direct adaptive fuzzy control [167] to directly approximate the desired control input signals rather than unknown system's nonlinearities.

Generally speaking, this technique attempts to estimate the *norm* of the unknown weight vector (or matrix) rather than estimating its elements [168]. To be more precise, consider again the control problem given in Section 2.1 with the dynamic model as (1). Suppose that $\|W^*\|^2 \leq \varpi$, where ϖ denotes an unknown constant. Defining $\hat{\varpi}$ as the estimation of ϖ, a Lyapunov function can be defined as:

$$V = \frac{1}{2}e^T e + \frac{1}{2\lambda}\tilde{\varpi}^2, \tag{71}$$

where $\tilde{\varpi} = \hat{\varpi} - \varpi$ and λ is a positive constant. Thus, we have:

$$\dot{V} = e^T \left(F(x) + B(x)u - \dot{x}_d + W^{*T}\mu + \varepsilon\right) + \frac{1}{\lambda}\tilde{\varpi}\dot{\hat{\varpi}}. \tag{72}$$

Using Cauchy–Schwarz and Young's inequalities, it is obtained that

$$e^T W^{*T}\mu \leq \frac{a^2\varpi e^T e\mu^T\mu}{2} + \frac{1}{2a^2}, \tag{73}$$

$$e^T\varepsilon \leq \frac{a^2 e^T e}{2} + \frac{\varepsilon_M}{2a^2}, \tag{74}$$

where a represents a positive design constant. Therefore,

$$\begin{aligned}\dot{V} &\leq e^T\left(F(x) + B(x)u - \dot{x}_d\right) + \frac{\varepsilon_M}{2a^2} + \frac{1}{2a^2} + \frac{a^2\varpi e^T e\mu^T\mu}{2} + \frac{a^2 e^T e}{2} + \frac{1}{\lambda}\tilde{\varpi}\dot{\hat{\varpi}} \\ &= e^T\left(F(x) + B(x)u - \dot{x}_d\right) + \frac{\varepsilon_M}{2a^2} + \frac{1}{2a^2} + \frac{a^2\hat{\varpi} e^T e\mu^T\mu}{2} + \frac{a^2 e^T e}{2} \\ &\quad + \tilde{\varpi}\left(\frac{1}{\lambda}\dot{\hat{\varpi}} - \frac{a^2 e^T e\mu^T\mu}{2}\right).\end{aligned} \tag{75}$$

Thus, it is possible to define the control command and the updating rule of $\hat{\varpi}$

as follows:

$$u = B^{-1}\left(-F(x) + \dot{x}_d - \left(\frac{a^2}{2} + k_1\right)e - \frac{a^2 e\hat{\varpi}\mu^T\mu}{2}\right), \tag{76}$$

$$\dot{\hat{\varpi}} = \frac{a^2\lambda e^T e\mu^T\mu}{2} - \sigma\lambda\hat{\varpi}, \tag{77}$$

where k_1 and σ denote positive design parameters, and the second term on the right-hand side of (77) represents the σ-modification term. Substituting (76) and (77) in (75) yields

$$\dot{V} \leq -k_1 e^T e - \sigma\tilde{\varpi}\hat{\varpi} + \frac{\varepsilon_M}{2a^2} + \frac{1}{2a^2}. \tag{78}$$

So, setting $\sigma = \frac{2k_1}{\lambda}$ and knowing that

$$\tilde{\varpi}\hat{\varpi} \geq \frac{\tilde{\varpi}^2}{2} - \frac{\varpi^2}{2}, \tag{79}$$

finally, (78) can be written as follows:

$$\dot{V} \leq -k_1\left(e^T e + \frac{\tilde{\varpi}^2}{\lambda}\right) + \frac{k_1\varpi^2}{\lambda} + \frac{\varepsilon_M}{2a^2} + \frac{1}{2a^2} = -kV + C, \tag{80}$$

where $k = 2k_1$ and $C = \frac{k_1\varpi^2}{\lambda} + \frac{\varepsilon_M}{2a^2} + \frac{1}{2a^2}$. As discussed in Section 2.4, (80) leads to the convergence of both the tracking error and the norm estimation error to a small neighborhood of zero, where the appropriate value of a is determined considering the tradeoff between larger steady-state error and more control effort. Accordingly, we should train only a scalar parameter ($\hat{\varpi}$) rather than a matrix ($\hat{W}$), thereby considerably reducing the computational burden corresponding to the online tuning of the NN parameters. However, by comparing (77) with (7), it can be understood that such an achievement is obtained at the cost of less efficient use of the error vector e in the MLP technique. More precisely, here, we use only the norm of the tracking error (in a scalar updating rule) instead of using all its elements, separately, which results in a conservative design. A similar formulation can also be given for an MLP technique in the case of *direct* adaptive control designs.

This approach can be employed, in a similar way, in the structure of the backstepping control method. The design has been enhanced in [114] by updating only one parameter in the attitude control block that corresponds to

the maximum of the norm of all the three RBFNNs employed in the control system. However, this results in a more conservative design, thereby requiring more control effort. On the other hand, as mentioned earlier, authors in [132] have transformed the longitudinal dynamics of an HFV into the output feedback form in which the new system states were approximated using HGOs. Thus, there is a need for only one neural network in the proposed scheme, where the MLP technique has been employed in the training phase. A similar formulation has been utilized in [99] for a nonaffine model of an HFV. Both the velocity and attitude control blocks have been designed using the pseudocontrol strategy. Also, fuzzy wavelet neural networks have been employed to compensate for model uncertainties, where, owing to the employment of the MLP technique, only the norm of the weight matrix was tuned. Besides, in [115], DSC has been integrated with the MLP technique in the case of an HFV, which is subject to actuator bias fault. Considering unknown control gain functions (g_is), the MLP technique has been utilized to estimate $\lambda_i = {g_{i_{min}}}^{-1}\|W^*\|^2$ to avoid the control singularity problem, while due to the unusual formulation of the employed NNs, the small-gain theorem [169, 166] has been involved to ensure the UUB stability of the system.

Also, regarding the application of the MLP technique in the backstepping control of *VTOL UAVs*, such an approach has been utilized in a backstepping trajectory tracking control scheme for a quadrotor air vehicle in [138] in which the MLP technique has been applied to each of the six RBFNNs employed to estimate model uncertainties. Further, authors in [116] have used the DSC along with the MLP technique in the trajectory tracking control of a multi-rotor UAV considering output constraints. A robust term has also been incorporated to estimate the neural approximation error (Section 3.4). Besides, a neuroadaptive control approach has been proposed in [60] for a quadrotor UAV under model uncertainties and actuator faults. Compared to previously-mentioned studies, a NN has been employed in this paper to estimate *an upper bound for the norm* of model uncertainties (instead of estimating the uncertainty itself), where the MLP technique has been adopted to estimate an upper bound for the norm of

the weight vector of that NN.

The MLP technique has also been employed in discrete-time neural backstepping controllers [170], where the updating rule of the NN weights' norm and the stability analysis can be obtained similarly to the basic discrete-time FEL in Section 2.1. Such an MLP scheme has been used in [135] in a discrete-time neural backstepping controller applied to the longitudinal mode of an HFV. However, the introduced updating rule for ϖ in [170, 135] may result in negative values in some time intervals. Thus, the design has been improved in [137, 42] by assuming that $\|W^*\| \leq \varpi sign(\varpi)$, which allows ϖ to be either positive or negative.

3.3. Systems with unknown control direction

The design of a controller for a dynamic system with unknown control direction is a challenging problem. This is due to the fact that a control command with incorrect direction can simply make the system unstable. In such cases, an interesting idea would be to alternately change the control direction. Accordingly, if the control command is applied in the wrong direction, the systems states get away from the desired trajectory until the control direction changes. Subsequently, the amplitude of the control command should increase by increasing the tracking error to get the system back to the desired trajectory. Such an idea has been first introduced in [171], and a function with the above-mentioned characteristics is known as a *Nussbaum* function. Nussbaum function has been employed in different studies to provide acceptable closed-loop performance in the case of complex systems with unknown control direction [172]. To clarify the control design procedure using the Nussbaum function, consider a SISO dynamic model as $\dot{x} = f(x) + g(t)u$, where $g(t)$ is a time-varying control gain with unknown direction. To ensure the stabilizability of the system, assume that $g(t) \in I = [\underline{g}, \overline{g}]$, where $\underline{g}$ and $\overline{g}$ denote unknown constants and $0 \notin I$. Defining $e = x - x_d$, we have

$$\dot{e} = \dot{x} - \dot{x}_d = f(x) + g(t)u - \dot{x}_d. \tag{81}$$

If $g(t)$ was available, the control command could be computed as $u = g^{-1}(-f(x)+\dot{x}_d - ke)$, where k is a positive constant. However, due to the unknown control direction, such a command is not feasible. Thus, we define a control command as follows:

$$u = N(\zeta)\eta, \tag{82}$$

$$\dot{\zeta} = e\eta, \tag{83}$$

$$\eta = f(x) - \dot{x}_d + ke, \tag{84}$$

where $N(\zeta)$ represents a Nussbaum function like $N(\zeta) = \exp(\zeta^2)\cos(\pi\zeta/2)$. Defining a Lyapunov function as $V = 1/2e^2$, we have

$$\dot{V} = e\dot{e} = e\left(f(x) + gN(\zeta)\eta - \dot{x}_d\right). \tag{85}$$

By adding and subtracting $\dot{\zeta}$ to the right side of the equation, $\dot{V}$ is obtained as

$$\begin{aligned} \dot{V} &= ef(x) + gN(\zeta)e\eta + \dot{\zeta} - \dot{\zeta} - e\dot{x}_d \\ &= gN(\zeta)\dot{\zeta} + \dot{\zeta} - ke^2. \end{aligned} \tag{86}$$

Now, by multiplying both sides of the equation by $\exp(ct)$, where $c = 2k$, the following equation is obtained.

$$\frac{d}{dt}\left(Ve^{ct}\right) = \left(gN(\zeta)\dot{\zeta} + \dot{\zeta}\right)e^{ct}. \tag{87}$$

Thus, we have

$$V = e^{-ct}\int_0^t \left(gN(\zeta) + 1\right)\dot{\zeta}e^{c\tau}d\tau. \tag{88}$$

Consequently, according to Lemma 2 in [173], it is proved that $V(t)$ and $\zeta(t)$ are bounded, thereby guaranteeing the bounded tracking error. In cases where $f(x)$ is also an unknown function, as discussed earlier, we can simply substitute $f(x)$ in (84) by its estimation as $\hat{f}(x) = \hat{W}^T\mu(x)$ and include an additional term $1/2\tilde{W}^T\Gamma_1^{-1}\tilde{W}$ in the Lyapunov function to ensure the closed-loop stability. A similar approach has been employed in [174] in the framework of DSC to control the longitudinal mode of an HFV considering dead-zone input nonlinearity, where a set of NNs have been used to estimate uncertain terms in the control command.

3.4. Neural networks and Disturbance Observers (DOs)

3.4.1. Neural disturbance observer

As discussed earlier, NNs can be effectively employed in the closed-loop control to estimate and compensate for model uncertainties, external disturbances, and also complex parts of the control command. In addition to the above-mentioned control systems, NNs can also be utilized as a powerful DO in an open-loop identification problem. To this end, consider again the nonlinear model (1) where $\Delta(x, u)$ corresponds to the effect of model uncertainties, actuator faults, and external disturbances on the system dynamics [118]. Notice that an external disturbance is generally an explicit function of time (not the system states and inputs). Thus, the identification of Δ as a function of system states (and inputs) requires an implicit assumption that external disturbances can be formulated as a function of the system states (and inputs). Although such an assumption makes sense in the case of some types of external disturbances, in a general case, it is not reasonable. In such circumstances, it may be possible to estimate Δ by a NN with time-dependent weights (or even time-dependent structure). This brings new challenges to the convergence analysis of the NN, which would be an interesting research direction. Another idea would be assuming that external disturbances are smaller than an unknown bounded function of system states, i.e. $|d(t)| \leq W^{*T}\mu(x)$ [165], while it may lead to a conservative design, thereby significantly increasing the control effort in the case of control problems.

Here, assuming that the uncertain terms in the dynamic model can be formulated as a function systems states (and inputs), we can introduce a new state-space model as

$$\dot{\hat{x}} = F(x) + B(x)u + \hat{\Delta} + \kappa(x - \hat{x}), \tag{89}$$

where κ represents a positive constant (which is tuned according to the compromise between the convergence rate of the introduced observer and its sensitivity to measurement noises [175]), and $\hat{\Delta} = \hat{W}^T\mu(x, u)$ denotes the estimation of Δ. Notice that different types of feedforward and recurrent NNs can be formulated

in such a compact form [118]. Now, by defining $e_D = \hat{x} - x$ and $\tilde{W} = \hat{W} - W^*$, a Lyapunov function can be proposed as

$$V = \frac{1}{2} e_D^T e_D + \frac{1}{2} tr\left(\tilde{W}^T \Gamma^{-1} \tilde{W}\right). \tag{90}$$

Thus, we have

$$\dot{V} = e_D^T \left(-\kappa e_D + \tilde{W}^T \mu(x,u) - \varepsilon\right) + tr\left(\tilde{W}^T \Gamma^{-1} \dot{\hat{W}}\right), \tag{91}$$

where ε denotes the bounded estimation error of the NN. As a consequence, if the NN's parameters are updated as $\dot{\hat{W}} = \Gamma \mu e_D^T$, it is obtained that $\dot{V} = -e_D^T(\kappa e_D + \varepsilon)$, which results in $\dot{V} < 0$ for $\|\kappa e_D\| > \|\varepsilon\|$, thereby guaranteeing the bounded estimation error. The employment of one of the modification techniques introduced in Section 2.1 in the updating rule is also recommended to avoid parameter drift. It is notable that, in the proposed DO, there is no need for an affine model, and we can simply substitute $F(x) + B(x)u$ in (89) with $F(x,u)$. A similar method has been used in [176] to estimate uncertain terms in the dynamics model of a UAV including external disturbances induced by different types of atmospheric disturbances, i.e. the wind shear, wind gust, and atmospheric turbulence. Owing to the presence of both v and $\dot{v}$ in the disturbance term (where v represents the wind velocity vector), a dynamic equation is then derived (using the estimated uncertainty) as $\dot{v} = \chi(x,u,v)$ to estimate the total wind velocity. Subsequently, an auto-landing control system has been proposed in [176] for the Sekwa UAV in the presence of external disturbances using a combination of the backstepping control and the dynamic inversion, while the designed scheme attempted to control six independent outputs by only four system inputs, which is not generally feasible. More precisely, the pseudo-inverse operator employed to compute the control command may result in inappropriate commands in the case of inconsistent control objectives.

Moreover, it should be noted that an analogous formulation can be developed to provide a neural *state* observer. A neural observer has been proposed in [162] by incorporating both the kinematic and dynamic equations of the system to estimate the translational and angular velocities of a quadrotor knowing the

position and attitude of the vehicle. Such a DO can also be utilized in the closed-loop control by substituting e_D in the updating rule of the NN by $e_D + e$ with e represents the tracking error. Indeed, this would be a variant of the composite learning method introduced in Section 2.1.

3.4.2. Combination of NN function approximation and DOs

There are a variety of robust control approaches in the literature in which a combination of DOs and NNs has been adopted to simultaneously compensate for external disturbances and model uncertainties, respectively. Using such an identification scheme, it is possible to distinguish between external disturbances, which are explicit functions of time, and internal disturbances, which can be modeled as a function of system states (and inputs). In addition, using a combination of DOs and NN-based estimators, DOs will be capable of compensating for the estimation error of the NN. More specifically, consider a nonlinear dynamic model as

$$\dot{x} = F(x) + B(x)u + \Delta(x, u) + d(t), \tag{92}$$

where $\Delta(x, u)$ and $d(t)$ represent model uncertainties and external disturbances, respectively. Now, considering the following definitions:

$$\dot{\hat{x}} = F(x) + B(x)u + \hat{\Delta}(x, u) + \hat{D} + \kappa(x - \hat{x}), \tag{93}$$

$$\Delta = W^{*^T}\mu(x, u) + \varepsilon, \quad \hat{\Delta} = \hat{W}^T\mu(x, u), \tag{94}$$

$$D(t) = d(t) + \varepsilon, \tag{95}$$

$$e_D = \hat{x} - x, \quad e = x - x_d, \tag{96}$$

and assuming $dim(x) = dim(u) = n$, an appropriate control command can be formulated as follows (the introduced approach can be used in the case of other types of dynamic systems and control methods in a similar manner):

$$u = B(x)^{-1}(x)\left(\dot{x} - F(x) - \hat{\Delta} - \hat{D} - k_1 e\right). \tag{97}$$

Subsequently, a Lyapunov function can be formulated as

$$V = \frac{1}{2}\left(e^T k_2 e + e_D^T k_3 e_D + \tilde{D}^T\tilde{D} + tr(\tilde{W}^T\Gamma^{-1}\tilde{W})\right), \tag{98}$$

where k_i represents positive constants, and $\tilde{D} = \hat{D} - D$. Accordingly, we have

$$\begin{aligned} \dot{V} = k_2 e^T \left(-k_1 e - \tilde{W}^T \mu - \tilde{D}\right) + tr(\tilde{W}^T \Gamma^{-1} \dot{\tilde{W}}) + \\ \tilde{D}^T \left(\dot{\hat{D}} - \dot{D}\right) + k_3 e_D^T \left(-\kappa e_D + \tilde{W}^T \mu + \tilde{D}\right). \end{aligned} \tag{99}$$

Thus, the following updating rules can be defined:

$$\dot{\tilde{W}} = \dot{\hat{W}} = \Gamma \left(\mu \left(k_2 e^T - k_3 e_D^T\right) - \sigma_W \hat{W}\right), \tag{100}$$

$$\dot{\hat{D}} = (k_2 e - k_3 e_D) - k_4 \left[\dot{\hat{x}} - \dot{x} + \kappa e_D\right], \tag{101}$$

where k_4 denotes a positive constant. Concerning the second term on the right-hand side of (101), using (93), we have

$$\dot{\hat{x}} - \dot{x} + \kappa e_D = \tilde{W}^T \mu + \tilde{D}. \tag{102}$$

Consequently, assuming that $\mu(x, u)$ and $\dot{D}$ are bounded, one can simply prove the satisfaction of (58), thereby ensuring the boundedness of all signals in the closed-loop system. Notice that, although the updating rule (101) consists of $\dot{x}$, there is no need for it to compute $\hat{D}$ because the estimated disturbance ($\hat{D}$) is obtained as the integral of (101) (the integral of other terms in the updating rule can be calculated using an auxiliary state variable [177]). Such a combination has been employed in [177] in a backstepping design. The same approach has also been utilized in [129] to provide a decentralized attitude synchronization tracking of multi-UAVs in the presence of actuator faults and wind effects. Similarly, the trajectory tracking control of multiple trailing UAVs has been addressed in [178] using DSC, where an NN+DO has been adopted to compensate for unknown aerodynamic parameters, actuator faults, and wake vortices. A partially analogous scheme has been utilized in [160] to provide a trajectory tracking control for a flapping-wing micro aerial vehicle considering model uncertainties and external disturbances. In a similar manner, a combined NN and DO has been incorporated in [140] in the framework of an FTC to control the attitude of a 3-DOF helicopter. Further, an analogous scheme has been employed in [141] in a backstepping controller designed to control the attitude of an NSV. The same identification approach has also been adopted

in [151], where a DO is utilized to compensate for external disturbances, the estimation error of NNs, and the effect of unknown input dead-zone.

Another effective combination of NNs and DOs with less complexity and no requirement to use the boundedness of $\mu(x,u)$ and $\dot{D}$ in the stability analysis, relies on the estimation of the upper bound of D rather than that of the exact value of it. Such a method, which results in a *conservative* design, can be classified as a *robust adaptive control*. For this purpose, consider again the above-mentioned dynamic model and the following definitions (for simplicity, suppose that $x, u \in \mathbb{R}$, while the introduced approach can be extended to MIMO systems with a similar formulation):

$$\dot{\hat{x}} = F(x) + B(x)u + \hat{\Delta}(x,u) + \upsilon + \kappa(x - \hat{x}), \tag{103}$$

$$\Delta = W^{*^T}\mu(x,u) + \varepsilon, \quad \hat{\Delta} = \hat{W}^T\mu(x,u), \tag{104}$$

$$D(t) = d(t) + \varepsilon, \quad \|D\| \leq D_M, \tag{105}$$

$$e_D = \hat{x} - x, \quad e = x - x_d, \tag{106}$$

$$u = B(x)^{-1}(x)\left(\dot{x}_d - F(x) - \hat{\Delta} - \upsilon - k_1 e\right), \tag{107}$$

where υ should be designed. Now, redefining the Lyapunov function as

$$V = \frac{1}{2}\left(k_2 e^2 + k_3 e_D^2 + k_4 \tilde{D}_M^2 + \tilde{W}^T\Gamma^{-1}\tilde{W}\right), \tag{108}$$

we have

$$\begin{aligned}\dot{V} = k_2 e\left(-k_1 e - \tilde{W}^T\mu + D - \upsilon\right) + \tilde{W}^T\Gamma^{-1}\dot{\tilde{W}} + \\ k_4\tilde{D}_M\dot{\hat{D}}_M + k_3 e_D\left(-\kappa e_D + \tilde{W}^T\mu + \upsilon - D\right).\end{aligned} \tag{109}$$

Thus, using the updating rule (100) and

$$e_a = k_2 e - k_3 e_D, \tag{110}$$

$$\dot{\hat{D}}_M = 1/k_4\left(e_a \tanh(e_a/\epsilon) - \sigma_M\hat{D}_M\right), \tag{111}$$

$$\upsilon = \hat{D}_M \tanh(e_a/\epsilon), \tag{112}$$

where ϵ denotes a positive constant, it is obtained that

$$\begin{aligned}\dot{V} = -k_2 k_1 e^2 - k_3\kappa e_D^2 + \tilde{D}_M\left(e_a\tanh(e_a/\epsilon) - \sigma_M\hat{D}_M\right) \\ - \sigma_W\tilde{W}^T\hat{W} + e_a D - e_a\hat{D}_M\tanh(e_a/\epsilon).\end{aligned} \tag{113}$$

Having the following inequality [179] for any $\epsilon > 0$ and $z \in \mathbb{R}$,

$$0 \leq |z| - z \tanh(z/\epsilon) \leq 0.2785\epsilon, \tag{114}$$

it is easy to show that (58) is satisfied. The utilization of the hyperbolic tangent function instead of the signum function in the presented formulation is an effective way to avoid the chattering phenomenon, while the possible asymptotic stability of the closed-loop system reduces to UUB stability. To be more specific, if we simply employ the signum function in the introduced control scheme and eliminate the σ-modification terms from (100) and (111), one can simply prove the asymptotic convergence of the tracking error to zero, though at the cost of possible parameter drift and previously discussed limitations of discontinuous control systems (Section 2.4.1). Such an approach has been utilized in [168] to control the position and attitude of a helicopter with unknown inertia matrix considering aerodynamic frictions. Accordingly, the unknown aerodynamic forces and moments have been estimated using RBFNNs, where the upper bound of the estimation error corresponding to NNs, as well as external disturbances, has been compensated by the introduced DO.

The introduced identification scheme can be similarly employed in the backstepping control design. It has been employed in [180] in a backstepping trajectory tracking control applied to a model-scaled helicopter in order to deal with the NN's estimation error, where a switching function has been adopted to integrate the NN and the introduced DO. Further, a similar approach has been used in the framework of DSC to control the longitudinal mode of an HFV in the presence of model uncertainties, dead-zone input nonlinearity [174], and actuator faults [115]. Analogously, in [118], recurrent wavelet neural networks have been integrated with such a DO in a DSC to compensate for external disturbances, model uncertainties, and the effect of input constraints in the case of the attitude control of an NSV. An adaptive neural backstepping control has been proposed in [83] for an HFV, where a similar approach has been utilized in each step of the backstepping control to deal with model uncertainties and estimation error of NNs. In [116], DSC has been applied to a multi-rotor UAV

to provide an attitude control system. A similar identification method has been employed to compensate for model uncertainties and external disturbances.

On the other hand, a reverse combination of NN+DO has been introduced in [181], where first, a DO attempts to estimate the entire model uncertainties and external disturbances as a lumped disturbance, and subsequently, a NN has been employed to compensate for estimation error of the DO. The proposed scheme has been utilized to control the roll angle of an air vehicle considering the wing rock phenomenon. However, using such an approach, the estimation error of the NN is not identified and thus remains uncompensated.

3.5. Fault-tolerant control

As mentioned earlier, the introduced (direct and indirect) NN-based adaptive controllers have been applied to faulty systems, as well. They include but not limited to the basic FEL-based control [91, 182, 92], the pseudocontrol strategy [105, 106, 183, 100], the neural backstepping design [70, 184, 124, 125, 185, 140, 147], and hybrid direct-indirect adaptive controllers [107, 38]. In this regard, a typical approach to deal with operational faults is to incorporate the nonlinear terms induced by the actuator faults (or structural damages) into the model uncertainty and estimate (and compensate for) them as a lumped uncertainty by FEL-based NNs [115, 152, 60, 186, 59]. Although the actuator faults (or structural damages) suffer from the same issue as external disturbances (i.e. the explicit dependence on time), by considering a sequence of abrupt faults, the coefficients corresponding to system faults can be deemed as time-independent functions between two sequential faults. Thus, the stability analysis can be performed for a specific time interval between two sequential faults (see the following subsection). Accordingly, the aforementioned combination of NNs and DOs can also be utilized in fault-tolerant flight control systems [140, 129].

To provide more efficient FTC systems with less conservativeness, in addition to the aforementioned generic adaptive neural control methods, there are various NN-based controllers in the literature that have been customized to specifically deal with actuator/sensor faults and structural damages. Some of the more

commonly used schemes in this field are given in the following.

3.5.1. FEL-based fault identification

It is possible to employ the FEL method to directly identify the coefficients corresponding to actuator faults, while simultaneously estimating model uncertainties in the closed-loop control. To clarify the basic idea, consider again the dynamic model (1), and suppose that the actual plant input is determined as

$$u(t) = \xi(t)u_c(t) + \delta(t), \tag{115}$$

where $u_c \in \mathbb{R}^n$ represents the computed control command, and $\xi(t)$ and $\delta(t)$ denote an unknown diagonal matrix and an unknown vector corresponding to multiplicative and additive actuator faults, respectively. Such a formulation can represent different types of actuator faults, such as the stuck type fault and the loss of effectiveness [187]. Considering a sequence of sudden actuator faults, ξ and δ can be considered as piecewise constant functions. Accordingly, defining t_i as the time of the occurrence of the ith actuator fault, one can assume that ξ and δ remain constant for $t \in (t_i, t_{i+1})$. In the following, we will focus on this time interval, while, due to the finite number of such time intervals, the design can be extended to the entire flight time assuming that the occurrence of the fault at t_i does not violate the controllability of the system. Now, if the ideal system input (u^*) is defined as (4) with $\hat{\Delta} = \hat{W}^T\mu$, then a control command can be determined as follows:

$$u_c = k_2 u^* + k_3, \tag{116}$$

where k_2 and k_3 represent unknown constants satisfying

$$\xi k_2 = I, \quad \xi k_3 + \delta = 0, \tag{117}$$

which require ξ to be invertible. Knowing that ξ is a diagonal matrix, the invertibility implies that no actuator should be completely stuck. Owing to the unknown value of k_2 and k_3, their estimations are employed in the control command. Thus, we have:

$$u = \xi(\hat{k}_2 u^* + \hat{k}_3) + \delta = u^* + \xi(\tilde{k}_2 u^* + \tilde{k}_3). \tag{118}$$

Thus, using the following updating rules

$$\dot{\hat{k}}_2 = \Gamma_2 \left(u^* e^T B(x)\right)^T, \tag{119}$$

$$\dot{\hat{k}}_3 = \Gamma_3 \left(e^T B(x)\right)^T, \tag{120}$$

and employing (7), one can define the following Lyapunov function

$$V = \frac{1}{2}\left(e^T e + tr\left(\tilde{W}^T \Gamma \tilde{W} + \tilde{k}_2^T \xi \Gamma_2^{-1} \tilde{k}_2 + \tilde{k}_3^T \xi \Gamma_3^{-1} \tilde{k}_3\right)\right) \tag{121}$$

Here, we have assumed that ξ is a positive definite matrix, which is a reasonable assumption due to the fact that ξ corresponds to the effectiveness ratio of actuators ($0 < \xi_{ii} \leq 1$). Utilizing the above-mentioned updating rules, the time derivative of V is obtained as (8), thereby ensuring the bounded tracking error. As discussed previously, it is recommended to incorporate a modification term in the updating rules (119) and (120) to avoid the parameter drift in the absence of the PE condition.

A similar approach has been used in [188] for a SISO system in the framework of traditional backstepping control. Similarly, in [123], such an approach has been utilized in a command filtered backstepping considering parametric uncertainty in both internal dynamics and the control gain function, while in [83], the prediction error has also be involved in the NN updating rules. All these controllers have been applied to the longitudinal mode of an HFV. An analogous method has been employed in [189], where the designed controller has been applied to an HFV considering flexible dynamics and state constraints. Alternatively, by considering a similar plant in the framework of DSC, authors in [190] attempt to estimate $\frac{1}{\inf|\xi_{ii}|}$, which allows to deal with time-varying actuator faults at the expense of employing a conservative design. DSC has been utilized in [191] to control the skid-to-turn missile in the presence of partial state constraints and actuator faults. Compared to the above-mentioned studies, the additive fault $B(x)\delta$ has been aggregated with the model uncertainty Δ into a single term, where the upper bound of it has been estimated by a NN. Further, instead of estimating k_2, the matrix ξ has been estimated directly, while there is no basic difference between these two design methods.

3.5.2. Using a separate Neural fault detection and identification (FDI) block

Traditionally, NNs were utilized as a separate Fault Detection and Isolation (FDI) scheme in the framework of FTCs. The main idea in it comes from the comparison of the output of the system and pre-trained NNs, where the residuals are interpreted as a fault if they exceed predefined thresholds [192]. However, such an approach cannot ensure closed-loop stability, and it may also lead to false alarms in the presence of severe external disturbances or unexpected damages.

There are other types of indirect fault identification approaches in the literature (which have been designed separately from the control system), as well. The main concern about such a decentralized design is the challenges in analyzing the closed-loop stability considering the estimation error of the fault identification block (which is commonly neglected in the stability analysis). An NN-based fault identification block has been proposed in [143] to estimate multiplicative actuator faults and model uncertainties, distinctly. The introduced method is similar to a FEL-based fault identification scheme, while the tracking error e is substituted by the estimation error of a neural observer (see Section 3.4.1). The estimated model uncertainties and actuator faults have been subsequently employed in the structure of a backstepping attitude controller applied to an NSV. Further, authors in [193], have attempted to identify the combination of fault dynamics and model uncertainties as a lumped uncertainty using a neural state observer. Such an approach has been employed in the paper to tackle sensor and actuator faults in the case of a satellite. The updating rules are obtained using a FEL method considering the estimation error of the neural observer as the learning signal.

In a somewhat similar fashion, a neural observer has been employed in [49, 50] within the framework of *nonlinear geometric approach* for fault detection and identification [194]. The fundamental assumption (which could be a restrictive assumption in flight control problems) in such an approach is the existence of a *coordinate change* in the state space and the output space that provides an

observable subsystem, which is affected by a specific fault but not affected by external disturbances and other faults. By exploiting such subsystems and using the same neural observer as introduced in Section 3.4.1, authors in [49, 50] have attempted to detect and identify different (but not simultaneous) sensor and actuator faults in a satellite, while considering external disturbances. Finally, the proposed scheme in [49] has been employed in an attitude control system based on a typical LQG controller designed for a linear model of the satellite.

Alternatively, an RLS optimization has been adopted in several studies to identify different operational faults of an air vehicle. In [195], multiplicative actuator faults have been identified using a generalized Online Sequential Extreme Learning Machine (OS-ELM) algorithm (for MIMO systems), which is based on the RLS optimization (see Section 4.1). The model uncertainties and external disturbances have been neglected at this stage, while they have been compensated by a robust model predictive control, which is applied to a quadrotor UAV. A neural state observer has been introduced in [196, 197] in which the NN weights have been updated using an Extended Kalman Filter (EKF), which is formulated by a similar formulation to the RLS optimization. The proposed approach has been evaluated in the presence of different faults such as abrupt and intermittent faults. Besides, such a method has been adopted in a dynamic inversion control in [197] to control the attitude of a fixed-wing aircraft considering actuator faults.

According to the obtained results in [196, 195, 38], the use of an RLS optimization-based updating rule results in faster convergence of the NN weights and higher accuracy in comparison with FEL-based approaches (which are developed based on Lyapunov's direct method), particularly in the case of an abrupt actuator fault. In addition, unlike the RLS optimization-based approaches, a fault identification block, which is developed using Lyapunov's direct method (such as in [193]), may result in severe changes in its estimation at the moment of an abrupt actuator fault [196]. This phenomenon, which has not been addressed in typical stability analyses, can be a challenging issue in FEL-based FTCs.

3.5.3. Multimodel approaches

A number of studies attempted to identify the dynamic model of the system using an online identification problem employing recurrent NNs (such as NARX NNs), and then design a controller for the identified model [198]. The challenging issue with such a control system is analogous to that of previously mentioned indirect FDI schemes. More precisely, the identification error is typically neglected in the stability analysis of the closed-loop system.

As a more reliable solution, a multimodel approach has been developed in [199] to identify a 6-DOF model of a fixed-wing aircraft in the presence of different actuator faults. To be more precise, a set of local NARX NNs have been first trained considering different fault conditions, i.e the elevator, aileron, and rudder faults, where each local model corresponds to a specific fault condition. Subsequently, the output of the entire model is computed by a weighted average of the outputs of local NNs, where the relative weight of each local model is determined using an OS-ELM-based optimization. It means that each local model can be considered as a hidden node of an extended ELM, where the output layer of this extended ELM is trained using the OS-ELM algorithm. Accordingly, the entire model can be considered as a deep neural network with two hidden layers, where the first layer (corresponding to local models) is trained offline, and the second layer is trained by the OS-ELM approach. Such an identification algorithm has been adopted in [200] to provide a reliable prediction model for the system. The obtained model has been used in a Model Predictive Control (MPC) to provide a trajectory tracking control for a fixed-wing aircraft. As illustrated in [200], the proposed approach not only can deal with actuator faults that have been considered in the offline training of local NNs, but it can compensate for all the actuator faults and structural damages that can be modeled as a combination of the local models. In this regard, the local NNs can be considered as the basis vectors of a multidimensional space, which are capable of representing all vectors in that space. Also, the prediction error of the model has been tackled by a DO in the proposed model predictive controller.

The stability of the closed-loop system has been analyzed using a terminal constraint in the MPC framework, while the feasibility of such a constrained optimization problem is not trivial [201].

3.6. Consideration of input constraints

Similar to FTC systems, a typical approach to overcome the input constraints is to consider nonlinear terms induced by input constraints (such as dead-zone or saturation function) as an uncertain term, which is estimated and compensated by NNs [136, 202]. Nevertheless, the same issue with neural DOs, i.e. the estimation of an explicit function of time using a NN that is a function of system states, exists here as well. In addition to such a control approach, other types of NN-based control designs have been proposed, which can deal with input constraints. The most commonly used approaches to this goal are given in the following.

3.6.1. Pseudocontrol Hedging (PCH)

A traditional approach to deal with input constraints in the framework of the pseudocontrol strategy called the Pseudo-Control Hedging (PCH), is to prevent the adaptive elements in the control system from *seeing* the effects of input constraints by manipulating the reference trajectory [183, 55]. For this purpose, consider again the dynamic model (25). Considering the desired trajectory x_d, a reference trajectory is defined for the system as $\dot{x}_r = \dot{x}_d + \nu_h$, where ν_h, which represents a residual term induced by input constraints, should be designed. Defining $\bar{e} = x - x_r$, we have:

$$\begin{aligned} \dot{\bar{e}} &= F(x,u) - \dot{x}_r = \hat{F}(x,u) - \dot{x}_r + \Delta(x,u) \\ &= \hat{F}(x,u_c) - \dot{x}_r + \Delta(x,u) + \left(\hat{F}(x,u) - \hat{F}(x,u_c)\right), \end{aligned} \tag{122}$$

where u_c denotes the desired control command and $\Delta(x,u) = F(x,u) - \hat{F}(x,u)$. Thus, if we define

$$\nu_h = \hat{F}(x,u) - \hat{F}(x,u_c), \tag{123}$$

using the control command u_c defined by (26)-(27), and employing the same procedure as given in Section 2.2 (by substituting e by $\bar{e}$), it can be concluded

that both the tracking error $\bar{e}$ and the estimation error of the weight matrix W are bounded. In this regard, due to the substitution of e by $\bar{e}$ in the updating rule of the NN weights, they can be satisfactorily updated even at the time of input saturation owing to the elimination of the effect of input constraints from $\bar{e}$ using the introduced term ν_h. However, concerning the boundedness of the real tracking error $e = x - x_d$, there is a need for a restrictive assumption, i.e.

$$\left\| \int_0^t \nu_h(\tau) d\tau \right\| \leq \nu_M, \tag{124}$$

with ν_M is a positive constant.

This approach has been utilized in [203, 204] to control the attitude of a Reusable Launch Vehicle (RLV) considering actuator faults. As discussed in [204], even in the lack of system controllability, the adaptation mechanism is satisfactory, which results in a rapid recovery once the system controllability is retrieved. Similarly, PCH has been adopted in [71, 205, 206, 207] in a trajectory tracking control problem applied to a helicopter, where the PCH technique has been employed in both the inner and outer control loops. As a result, the interaction between adaptive elements in the outer loop and the characteristics of the inner loop can also be avoided. The same approach has been utilized in [208] to control the trajectory of a ducted-fan VTOL UAV. Further, PCH has been employed in [209] to overcome actuators' nonlinearities in the landing control of a fixed-wing aircraft.

3.6.2. Employment of a modified tracking error

Another effective approach to handle different types of input constraints with less restrictive assumptions is to introduce an auxiliary state variable corresponding to a *filtered* version of the effect of input constraints. More precisely, consider the dynamic model (1). Suppose that the real system input (u) is obtained as $h(u_c)$, where u_c and h represent the computed (desired) control command and a known nonlinear function, respectively. Notice that $h(.)$ can represent different types of input nonlinearities such as the saturation function, dead-zone nonlinearity, etc, or user-defined filters [210] to generate feasible con-

trol commands according to the physical constraints of the system. Now, we define an auxiliary variable γ as follows:

$$\dot{\gamma} = -k\gamma + B(x)\delta u, \tag{125}$$

where $\delta u = u - u_c$. Accordingly, a *modified tracking error* can be defined as

$$z = x - x_d - \gamma = e - \gamma. \tag{126}$$

Notice that, in the absence of input constraints, γ tends to zero, and so the introduced modified tracking error reduces to the real tracking error. Besides, the introduced modified tracking error has a similar formulation to the compensated tracking error used in the command filtered backstepping control (Section 2.3.2). Considering the following definitions,

$$u_c = B(x)^{-1}\left(-F(x) - \hat{\Delta} + \dot{x}_d - ke\right), \tag{127}$$

$$\Delta = W^{*T}\mu(x) + \varepsilon, \quad \hat{\Delta} = \hat{W}^T\mu(x), \tag{128}$$

$$\dot{\hat{W}} = \Gamma\mu z^T, \tag{129}$$

and by defining a Lyapunov function as

$$V = \frac{1}{2}z^T z + \frac{1}{2}tr\left(\tilde{W}^T\Gamma^{-1}\tilde{W}\right), \tag{130}$$

one can prove the boundedness of z. Thus, assuming that $B(x)$ and δu are also bounded (the boundedness of δu is a consequence of the system controllability), it is easy to see that γ is also bounded, thereby resulting in a bounded real tracking error (e). Besides, even if the system controllability is lost at some time intervals, the updating rule of the NN is still stable thanks to the utilization of the bounded term z rather than the real tracking error in (129).

By comparing the above-mentioned approach with the PCH technique, it can be found that both methods attempt to eliminate the effects of input constraints from the tracking error that is involved in the updating rule of the NN parameters, while the employment of the low-pass filter (125) in the current scheme relaxes the necessary assumption on the residual term induced by input constraints.

Such an approach has been employed in [124, 125] in the framework of the command filtered backstepping control. The same technique has been adopted in [117] to control the longitudinal mode of an HFV using a DSC design. Second-order filters have been utilized in these papers ($h(.)$ is defined as a linear second-order transfer function) to deal with the magnitude, rate, and bandwidth limits of the control commands. Notably, as discussed in [124], the constraints on the system states can also be similarly taken into account by filtering the virtual control commands in the backstepping control. Modified tracking errors have also been utilized in [141] and [186], respectively, in a backstepping control and an SMC to deal with input saturation, where the designed controller in [186] has been applied to the longitudinal model of an air-breathing flexible HFV. Similarly, a modified tracking error has been adopted in [211] to tackle input saturation in a backstepping control scheme applied to the longitudinal dynamic model of an HFV considering additive faults, which has been estimated and compensated as a disturbance term using a NN. In addition, an analogous approach has been employed in [127] in a DSC design to control the longitudinal dynamic model of a morphing aircraft in the presence of input saturation. Besides, a somewhat similar scheme, borrowed from [212], has been employed in [191] to deal with partial state constraints in an integrated guidance and control design for skid-to-turn missile using DSC. More precisely, an auxiliary state variable γ has been defined in [191] based on δu, where γ has been involved in the desired virtual control command instead of the tracking error, while the given stability analysis in the paper requires some revision.

3.6.3. Neuro-predictive control

Model predictive control (MPC) is an advanced control method that can satisfactorily deal with input, state, and output constraints. More precisely, an optimization problem is constructed to minimize the tracking error within a prediction horizon, as well as the control effort within a control horizon, while considering the system constraints. The optimization problem is solved at each time step. The first element in the computed control sequence is applied to

the system, and the entire process is repeated in future steps. Despite the numerous advantages of MPC in dealing with nonlinear, MIMO, and constrained system dynamics, there are significant concerns regarding the stability analysis of the system and the high computational burden of MPC. The stability of the closed-loop systems can be ensured using terminal costs and terminal constraints [201]. However, such stabilizing terminal conditions can make the optimization problem infeasible. In this regard, the *recursive feasibility* problem has been extensively addressed by researchers to provide a feasible control design with guaranteed stability [213, 214]. On the other hand, different practical MPC schemes have been introduced in the literature to provide a computationally efficient control system for real applications [215].

Concerning the application of MPC in IFCSs, it should be noted that NNs can be adopted in the framework of MPC in different ways. A straightforward way is the employment of a (typically recurrent) NN to learn the system dynamics as a prediction model and utilizing it in the structure of MPC. A NARX NN, with an RLS optimization-based online adaptation, has been used in [216] as the prediction model of a 6-DOF F-16 fighter aircraft, and afterward, it has been incorporated in an MPC to control the vehicle's attitude considering input constraints. In [217], an adaptive feedforward NN has been employed to estimate the translational acceleration of a fixed-wing aircraft in a moving time window, where the identified model has been adopted in an MPC-based trajectory tracking scheme in the presence of input constraints and model uncertainties. However, the closed-loop stability has not been analyzed in these studies owing to the complicated structure of the proposed nonlinear MPC. Multiple model-based MPC using a set of local NARX NNs as the prediction model of the system has also been introduced in [200], where both the system constraints and actuator faults have been considered in the control design process.

On the other hand, regarding model-based approaches, a linear MPC has been proposed in [218], where the linearization error and unmolded dynamics have been estimated by a feedforward NN in an offline identification problem. The deigned control system has been employed in the altitude control of a

helicopter, while the estimation error of the NN has not been considered in the design process, and the closed-loop stability has not been analyzed. In addition, a fault-tolerant MPC has been introduced in [195], where an OSELM-based algorithm has been adopted to identify actuator faults. Also, the input constraints have been considered in the given design, and the estimation error of the identification block has been compensated by a DO. Further, the closed-loop stability has been proved using terminal constraints, while there are concerns regarding the feasibility problem.

3.6.4. Using Nussbaum function

In [219], Nussbaum functions have been employed in a backstepping control scheme to overcome the input saturation. To this end, the saturation function is approximated by a smooth function $g(v)$, and the approximation error is included in an unknown disturbance term. Subsequently, a Nussbaum function has been utilized to handle $\partial g/\partial v$, which is created in the last step of the backstepping control design. However, there are concerns about the stability analysis of the proposed approach given in the paper. To be more precise, although the Input-to-State Stability (ISS) assumption has been employed in the paper, the boundedness of the introduced Lyapunov function has been proved considering the input saturation without using the ISS condition, while this is an irrational result. Based on such a method, A DSC has been proposed in [118] to control the attitude of an NSV considering the input saturation and external disturbances. Surprisingly, there is no assumption on the stabilizability of the system to ensure closed-loop stability in the presence of input saturation. Apparently, this is due to the employment of the aforementioned theorem in [219]. Similarly, considering a more stringent problem, a backstepping control has been developed in [177] for a SISO model of a helicopter to control the pitch angle of the vehicle in the presence of input and output constraints. Neural networks have been employed to identify model uncertainties, while disturbance observers attempt to compensate for unknown external disturbances. Again, Nussbaum functions have been used to deal with the input saturation, where there is no

assumption on the stabilizability of the air vehicle. In this regard, further investigations should be conducted by researchers to evaluate the application of the Nussbaum function in the control of constrained systems. But, similar to the discussion presented in Section 3.3, Nussbaum functions have been successfully adopted in [174] to deal with dead-zone input nonlinearity as an unknown control gain function.

3.7. Consideration of state/output constraints

As discussed previously, the use of modified tracking errors and the MPC framework can be beneficial in dealing with state constraints, as well. Meanwhile, there are other approaches in the literature to cope with state/output constraints in the structure of IFCSs. The most commonly used method for this purpose is the employment of Barrier Lyapunov Functions (BLFs). A barrier function is defined as a function, $f(z)$, which tends to infinity as its variable, z, tends to a predefined bound [220]. Accordingly, considering a desired bound k_b for the tracking error $e = x - x_d$, a BLF can be defined as follows [221]:

$$V_0 = \frac{1}{2} \ln \left(\frac{k_b^2}{k_b^2 - e^T e} \right), \tag{131}$$

which is a positive definite function for $\|e\| < k_b$ (it is assumed that $\|e(0)\| < k_b$). Thus, considering the dynamic model (1) and using the same NN function approximation as given in Section 2.1, a Lyapunov function can be defined for the system as

$$V = V_0 + \frac{1}{2} tr \left(\tilde{W}^T \Gamma^{-1} \tilde{W} \right). \tag{132}$$

This results in the following equation.

$$\dot{V} = \frac{e^T}{k_b^2 - e^T e} \left(F(x) + B(x)u + \Delta - \dot{x}_d \right) + tr \left(\tilde{W}^T \Gamma^{-1} \dot{\tilde{W}} \right). \tag{133}$$

Consequently, using the control command (4) and by defining the following updating rule,

$$\dot{\hat{W}} = \Gamma \mu \frac{e^T}{k_b^2 - e^T e}, \tag{134}$$

we have:

$$\dot{V} = \frac{e^T}{k_b^2 - e^T e} \left(-k_1 e + \varepsilon \right), \tag{135}$$

which ensures the negative definiteness of $\dot{V}$ for $\|k_1 e\| > \|\varepsilon\|$, thereby guaranteeing the satisfaction of $\|e\| < k_b$ (assuming that $k_b > \|\varepsilon/k_1\|$).

Such a control formulation can also be employed in the backstepping control design to impose both the state and output constraints on the controlled system. In [138], a BLF has been adopted in a backstepping control scheme in the position control loop to keep the trajectory tracking error of a quadrotor UAV in a desired bound. Similarly, a BLF has been utilized in [177] within a backstepping controller to control the pitch angle of a 3-DOF helicopter considering output constraints. The constraint on the angle of attack (AOA) has also been dealt with by a BLF in [83] in a backstepping controller applied to the longitudinal mode of an HFV. In addition, in [127], both the velocity and altitude constraints have been considered in a backstepping design using BLFs, where the designed control system is applied to the longitudinal dynamic model of a morphing aircraft.

As discussed in [116], the satisfaction of output constraint using the BLF is achieved at the expense of excessive control effort in the case of approaching the tracking error to the boundaries of the permissible region. Accordingly, there is a trade-off between choosing a narrow range for the outputs' tracking error and reducing the control effort. More specifically, as a typical BLF imposes a constant upper bound on the system output, it may lead to large control inputs at initial times. An asymmetric BLF with time-varying bounds has been employed in [116] to deal with time-varying output constraints in which the constant parameter k_b in (131) is substituted by a function of time. The introduced scheme has been utilized in a DSC design in case of the attitude control of a multi-rotor UAV.

Another effective approach to tackle output constraints using a time-varying funnel-like bound is known as the *funnel control*. The key point of the funnel control is to construct a time-varying gain to control a dynamic system in such a way that the (norm of the) tracking error falls within a funnel boundary $\frac{1}{\varrho(t)}$, where $\varrho(t)$ is a continuous bounded function [222]. To be more specific, the funnel control is somewhat similar to the BLF-based approach, where the

Lyapunov function (131), in the case of a single-output system, is changed to

$$V_0 = \frac{1}{2}\left(\frac{e}{\Phi(t) - |e|}\right)^2, \tag{136}$$

where $\Phi(t) = \frac{1}{\varrho(t)}$. Such an approach has been employed in [133] in a backstepping design to deal with both the velocity and altitude constraints in the case of the longitudinal mode of an air-breathing HFV with a nonaffine model. Similarly, a Lyapunov function has been introduced in [147] as

$$V_0 = \frac{1}{2}\tan^2\left(\frac{\pi e}{2\xi(t)}\right), \tag{137}$$

where $\xi(t)$ represents a funnel-like function. This method has been utilized in [147] to control an HFV using a typical backstepping control in the presence of external disturbances and actuator faults. Although such approaches suffer from the same issue as the BLF scheme, i.e. the excessive control effort in the vicinity of the permissible output boundaries, the initial large control inputs can be avoided due to the employment of a funnel boundary.

3.8. Self-organizing neural networks

Although due to the universal approximation property, NNs (or FNNs) can approximate almost all nonlinear functions with an acceptable estimation accuracy, the determination of the appropriate number of hidden nodes (or fuzzy rules) in the network is not an easy task. In addition, the development of a variable structure NN rather than a fixed-structure NN (with only variable parameters) provides greater power to deal with time-varying characteristics of dynamics systems. Several self-organizing NNs have been introduced in the literature to deal with such issues. Further, a self-organizing FNN can eliminate the requirement for prior knowledge about the system [223]. Typically, a set of growing and pruning algorithms are defined in a self-organizing network to add or remove hidden nodes to (/from) the NN when necessary. As a result, a set of modifications may be required in the network's parameters (at the time of the change in the network's architecture) to ensure the continuity of the network output.

As a traditional approach in this field, Minimal Resource Allocation Network (MRAN) was introduced in [224], which has been developed based on RBFNNs. The growing phase in MRAN is activated if 1) the incoming data is far away from the center of existing hidden nodes, 2) the estimation error in the current step is larger than a predefined threshold, and 3) the root mean estimation error over a moving window is larger than a predefined threshold. Such an approach can be considered as a *clustering* problem. In this regard, the center of the newly added node is set to the last incoming data, while the output weight of that neuron is equal to the current estimation error of the network. On the other hand, a hidden node is pruned from the network architecture if the normalized output of that neuron becomes less than a predefined threshold in a specific number of consecutive steps. In addition, the network parameters are trained using either a Least Mean Squares (LMS) optimization or an EKF algorithm. An extension to MRAN, called Extended MRAN (EMRAN), has also been introduced in [225] in which only the parameters of the nearest neuron to the current input data are updated at each step. This leads to a significant reduction in the online computational burden of the algorithm. Such a learning strategy has been adopted in several studies. In [91], an MRAN-aided H_∞ control is incorporated in an auto-landing control problem of a conventional aircraft considering external disturbances and actuator faults. The NN, which was trained using the FEL method, augments the control command of the baseline controller. Similarly, an EMRAN-aided controller has been proposed in [92] to control a fighter aircraft in the landing phase considering actuator faults and severe winds, where the NN attempted to learn the inverse dynamics model of the system using a FEL scheme. However, the closed-loop stability has not been analyzed in these two papers. In a similar manner, EMRAN has been adopted in [226] to augment a baseline controller, combined with an SMC to ensure the closed-loop stability, where the designed controller has been applied to an auto-landing problem considering actuator faults and severe winds.

The ambiguity in how to determine the parameters employed in an MRAN is a challenging issue, while there is no explicit relationship between the design

parameters and the estimation error of the NN. As an alternative, the concept of the *significance of a neuron* has been employed in [227] to provide more efficient growing and pruning rules for RBFNNs. The significance of a neuron is defined as the average of its output over all the input samples it has seen. Accordingly, a new neuron is added to the network only if its significance is greater than a chosen learning accuracy, while a neuron is pruned if its significance becomes less than the learning accuracy. The main concern about such an approach is the complexity of computing the significance of a neuron, which has been determined in [227] assuming a uniform distribution for the input data. Such a concept has been extended in [96] to develop the growing and pruning rules within the framework of a Generic Evolving Neuro-Fuzzy Inference System (GENEFIS), which was first introduced in [228]. An ϵ-completeness criterion has also been utilized in the paper to add a new rule when a new incoming sample cannot be covered by any existing rules. According to this criterion, which has been introduced in [229], the firing strength of at least one fuzzy rule corresponding to each data in the operating range, should not be less than ϵ. Subsequently, to update the antecedent parameters of the fuzzy rules, the Generalized Adaptive Resonance Theory+ (GART+), which uses the Bayes decision theory, has been first employed to determine the winning rule corresponding to each newly added data, and then, a *vigilance test* has been conducted to investigate the capability of the winning rule to accommodate the newest data. Further, an SMC-based approach has been adopted to train the consequent parameters of the fuzzy rules. Alternatively, hyperplane-based clusters have been employed in [97], which removes the antecedent parameters in the proposed neuro-fuzzy system. More precisely, the membership function of each fuzzy rule has been defined according to the distance between the current data point and the corresponding hyperplane, where the hyperplane parameters are considered as the consequent parameters of the network. The main idea in the introduced self-organizing network has been borrowed from [223] in which a Parsimonious Learning Machine (PALM) has been developed for data regression. However, different from the basic PALM, which requires various predefined thresholds, the

growing and pruning mechanisms in [97] have been developed using the concept of bias-variance. Accordingly, by defining the expected squared tracking error of the system as the Network Significance (NS), the NS has been derived as a sum of the bias and variance of the plant's expected output. Then, a high variance of the system outputs has been interpreted as the high complexity of the network, which in turn activates the rule pruning mechanism. On the other hand, the rule growing algorithm is activated in the presence of high output bias, which is induced by an oversimplified network. Finally, similarly to [96], an SMC-based training method has been adopted to update the consequent parameters of the network. The proposed schemes in both the above-mentioned research [96, 97] have been utilized in the altitude and attitude control blocks of a hexacopter and a flapping-wing micro aerial vehicle as an aid to a baseline controller. Similarly, a self-constructing FNN has been introduced in [152], where the distance between the incoming data and existing clusters has been considered as a measure to add a new rule, while the distance between the existing clusters has been analyzed to prune insignificant rules. The obtained self-constructing FNN has been utilized to approximate the upper bounds of model uncertainties, while it has been employed in an FTC applied to a longitudinal model of a fixed-wing aircraft. Notice that, despite the development of various effective self-organizing NNs in the literature, significant concerns still remain about the optimality of the network's architecture. As a consequence, the development of a truly *generic* approach to construct an *optimal* network structure depending on different characteristics of the obtained data from a plant is an important research direction, which must be addressed in future studies as a critical step in developing a fully autonomous control scheme.

Finally, concerning multiple-model-based structures, a self-organizing multi-model ensemble has been given in [230], in which a new local NN is added to the proposed multi-model structure if the estimation error of the entire model exceeds a predefined threshold. In addition, a local NN is considered as an insignificant model and pruned from the entire model if the normalized weight of the model in the entire scheme becomes less than a predefined threshold. The

proposed approach has been employed in the paper to identify the time-varying dynamic model of a fixed-wing aircraft at different flight conditions.

3.9. Concerns with air vehicle's characteristics

The position and attitude of an air vehicle can be determined using the kinematic equations based on the translational and angular velocities, respectively. As a result, the position and attitude can be controlled indirectly in a backstepping scheme, where in the first step, the position (or attitude) controller is designed, and the second step deals with the control of the translational (or angular) velocity. Besides, the measurement noises or the simplifications in the kinematic equations, which appear in the first step of the backstepping controller, can be estimated and compensated (as an uncertain term) by NNs [116].

On the other hand, the dynamic equations of an aerial vehicle can be generally derived using either the Newton-Euler or Euler-Lagrange methods. Regarding conventional multi-rotor VTOL UAVs (with no tilt-rotor), the system dynamics are divided into the rotational and translational equations, where, due to the under-actuated dynamics of the vehicle, the translational motion (typically in the horizontal plane) should be indirectly controlled by the vehicle's attitude. Thus, a straightforward control method to deal with such a dynamic model would be a multi-loop control design wherein the desired attitude, which is controlled by the inner loop, is determined using the translational dynamics in the outer control loop [32, 58, 60]. Such a framework may also be expressed within a backstepping control scheme. More precisely, the first step of the backstepping controller would deal with the translational dynamics, while the attitude dynamics have been addressed in the second step [138, 162]. In this regard, the desired attitude is indirectly determined (usually by employing the inverse kinematics method) to provide the desired forces required in the outer loop (or in the first step of the backstepping controller) [231, 95, 151]. Accordingly, owing to the complicated relationship between the vehicles' attitude and the translational dynamics, it may be a requirement for a control law (in the inner loop) that can guarantee the asymptotic stability of the inner loop (rather

than a bounded tracking error). Otherwise, the possible tracking error caused by the inner control loop should be considered in the outer loop, while it can complicate the stability analysis. Further, as discussed earlier, uncertain forces and moments induced by uncertain dynamics or external disturbances in the translational and rotational dynamic models can be estimated by distinct NNs.

Besides, a similar framework can also be employed in the case of a helicopter. In addition to the inverse kinematics method to determine the desired attitude (or the desired angular velocity) [206, 232], it is also possible to define a virtual control input for the attitude dynamics in a backstepping control scheme [180]. Such a virtual control would be computed according to the translational dynamics of the vehicle, which have been addressed in the previous steps of the backstepping design, by taking into account the kinematic equations (corresponding to the rotation matrix or the quaternion) in such a way that the closed-loop stability can be analyzed based on the Lyapunov stability theorem [233]. Again, NNs can be adopted to compensate for model uncertainties, external disturbances, or the inversion model error (in the pseudo-control strategy) [101] in each loop. By employing a backstepping scheme in [110] for the attitude control of a helicopter, the uncertain control gain matrix (g_i) in the dynamic model has been estimated by a distinct NN, while the extra terms due to the minimum estimation error of that NN (ε) has been considered in the last step of the backstepping design corresponding to the actuator dynamics. However, the proposed design suffers from the chattering phenomenon. Further, the issue of unknown inertia matrix has been dealt with in [168] wherein an additional adaptive rule has been defined to estimate it (while taking advantage of the Cholesky decomposition).

A somewhat similar control framework can also be designed in the case of conventional fixed-wing aircraft. A backstepping design has been employed in [125], where the desired trajectory is first transformed to the desired velocity, flight path, and heading angles using the kinematic equations and subsequently, they are converted to the desired thrust force and angular velocity using dynamic equations. Finally, in the last step, the control-surface deflections have been

determined according to the desired angular velocity, where unknown forces and moments have been estimated using the FEL method. In such a framework, the outer loop is typically considered as the guidance loop, while the inner loops are known as the main control system. A similar approach has been utilized in [200], where the desired trajectory is first transformed to the desired Euler angles (and subsequently to the desired angular velocity) in the guidance loop, while in the inner loop, the actuator deflections have been determined based on the desired angular velocity. Another approach is to decouple the control problem of the longitudinal and lateral-directional modes of a fixed-wing air vehicle (using some simplifications) and address them separately [98]. In this regard, several studies have addressed only one of these two subtasks and skipped the other part [152].

In the case of HFVs, almost all of the above-mentioned papers have investigated only the longitudinal model of the vehicle, where the velocity and altitude subsystems are typically decoupled, as well. To this end, the effect of thrust force on the altitude subsystem should be neglected, and the change rate of the velocity is considered much smaller than that of the altitude (known as *time-scale decomposition*) [136]. As a result, the velocity subsystem is obtained as a simple SISO model with a single state, while the altitude subsystem includes four state variables including the altitude (h), the Flight Path Angle (FPA, γ), the pitch angle (θ), and the pitch rate (q). Further, the consideration of flexible modes results in introducing additional states, which are not directly involved in the control design process [117, 114, 174], and they may be considered as a disturbance compensated by a DO. In this regard, the wind effect, which results in an excessive angle of attack, can also be considered as an unknown disturbance [81]. In such a framework, the main system inputs are the throttle setting and the elevator deflection, which directly influence the velocity and altitude subsystems, respectively. Other systems inputs such as the diffuser area ratio and the canard deflection can also be considered in the design, while there are typically assumed to be constant or a linear function of other system inputs [114]. Besides, a coordinate change has been employed in [82] to deal with the non-minimum phase behaviour of the attitude subsystem (due to the coupling

between the lift force and the elevator deflection) that is typically eliminated by the canard deflection as an additional control input in most studies. Knowing that $h = V \sin\gamma$, a typical method to design a controller for the altitude subsystem is incorporating an intermediate PID controller between h and γ to derive the desired FPA [115], and subsequently transforming the remaining system (including $x = [\gamma, \theta, q]^T$) into a strict feedback form, which can be controlled using a backstepping design [135]. In this regard, both the direct and indirect adaptive backstepping designs can be used to control such a strict feedback system [33] On the other hand, the backstepping design can be avoided by deriving a normal output-feedback model (in the case of continuous-time systems) [131, 132] or a prediction model (in the discrete-time domain) [136]. In such a circumstance, it may be possible to use only one NN to compensate for uncertain terms in the control command. Such a normal feedback form has been employed in a pseudocontrol strategy in [99] to deal with nonaffine dynamics of the vehicle. There also few works in the literature, which have addressed the coupled dynamics of the velocity and attitude subsystems. A combination of singular perturbation theory and implicit function theorem has been incorporated in [109] to deal with the longitudinal model of an HFV in a unified manner, while conservative assumptions on the dynamic model are required in the proposed control scheme. In a more effective way, a neural backstepping controller has been proposed in [189] for a MIMO model of an HFV considering the coupling between the velocity and attitude subsystems, where a combined adaptive design and a DO (as discussed in Section 3.4.2) has been adopted.

4. Towards truly model-free control systems

4.1. Neural network-based system identification

As discussed earlier, in the basic indirect FEL-based control, a *nominal* model of the system is derived and subsequently, a set of NNs are employed to identify model uncertainties and external disturbances as a *single term* [209]. This type of dynamic modeling leads to a conservative control design with re-

duced efficiency. To be more precise, most difficulties in modeling a complex dynamic system may originate from the existence of hidden states in the system, not from the model uncertainties caused by a lumped disturbance [234]. In addition, in many of the above-mentioned papers, an initial controller was designed based on a nominal model of the system, and then a control augmentation was proposed considering instantaneous model uncertainties. However, in the case of severe structural damages or significant dynamic changes, such an approach may lead to excessive control effort or even closed-loop instability [182].

In this regard, the development of a valid dynamic model of the system is a crucial task in the control theory. Typically, this is performed by two different approaches: the first-principles modeling and the system identification. Obtaining an acceptable dynamic model of an air vehicle using the first-principles modeling requires detailed information about the aerodynamic and propulsive forces and moments acting on the vehicle, which may not be practical for complicated systems. Alternatively, system identification attempts to fit a mathematical model on the obtained system inputs-outputs. It can be effectively employed to identify the system dynamics, particularly in the case of complex systems. However, the employment of such a black-box model identification suffers from the lack of interpretability of the obtained model [36]. There are various studies in the literature that have demonstrated the superior performance of the integration of two methods compared to that of only one method [235, 236, 237], while such an approach remains an open research topic in the framework of IFCSs [52].

In contrast to the FEL method, there are a variety of intelligent controllers in the literature that include separate identification and control design processes. In this regard, the previously mentioned unique capabilities of NNs make them an ideal candidate to be used in the identification process of such control systems. Different feedforward and Recurrent Neural Networks (RNNs) have been employed for this purpose, where the training process of the network would be performed in the framework of the supervised learning using either offline

or online approaches (or a combination of them). More precisely, the system identification and the control design processes can be fulfilled sequentially or simultaneously in an iterative manner, which is also known as *iterative learning control*. As a result, the iterative learning control can effectively deal with time-varying dynamic systems, and consequently, it can be classified as an intelligent control system, while the employment of a pre-trained NN in the flight control system may not be considered as an IFCS.

Further, the identification problem using a NN can be considered as an *optimization* problem in which the NN's parameters are determined by minimizing the prediction error of the NN with respect to adjustable parameters of the NN. Therefore, different optimization algorithms (from traditional approaches such as the gradient-descent method, the Gauss-Newton method, the Levenberg-Marquardt (LM) method, etc., to heuristic methods such as the genetic algorithm [238]) can be effectively employed to train the NN parameters. Such optimization algorithms have been thoroughly discussed in the literature [40], and thus, in the following, only the Online Sequential Extreme Learning Machine (OS-ELM) method will be briefly introduced as a conventional online training algorithm in the structure of indirect adaptive flight control systems. Besides, concerning the network structure, some of the most commonly used NNs for identifying the system dynamics in the case of an IFCS will be discussed in the following.

4.1.1. Single-hidden-layer neural networks

As a special case of state-space modeling of a dynamic system, input-output representation is a simpler popular approach to model nonlinear systems [198]. Using such a formulation, the system output (in the discrete-time domain) can be represented as follows:

$$y(k) = h\left(y(k-1), \cdots, y(k-P), u(k-1), \cdots, u(k-M), d(k)\right), \tag{138}$$

where u and d denote the system input and the vector of disturbances, respectively. Also, P and M represent the number of past outputs and inputs

employed in the modeling. Here, h is an unknown function, which should be identified. Also, considering (138), the system states are $[y(k), \cdots, y(k-P)]^T$.

In this regard, the assumption on the influence of external disturbances and noises on the system dynamics results in introducing two conventional model structures. More specifically, in the presence of the *state noise* (which is also known as the *equation error*), the dynamic model (138) can be simplified as

$$y(k) = f\left(y(k-1), \cdots, y(k-P), u(k-1), \cdots, u(k-M)\right) + d(k), \tag{139}$$

where f and d represent, respectively, an unknown nonlinear function and an additive disturbance term. Such a model is known as a Nonlinear Autoregressive with exogenous inputs (NARX) model, which can be identified by a NARX NN. According to (139), a NARX NN employs the past *measured system outputs* and system inputs as the network inputs. Consequently, a NARX NN can be considered as a feedforward NN with taped delay lines. The use of the NARX structure in training the network is also known as *series-parallel* identification [239].

On the other hand, in the presence of the *output noise*, the system dynamics model can be formulated as

$$\begin{aligned} x(k) &= f\left(x(k-1), \cdots, x(k-P), u(k-1), \cdots, u(k-M)\right), \\ y(k) &= x(k) + d(k). \end{aligned} \tag{140}$$

Accordingly, the system output at each step is a function of the disturbance that occurs at the same time step only. Nonlinear Output Error (NOE) NNs can be utilized to identify such dynamic models, where the NN employs the past *network's outputs* and system inputs as the network input. Thus, an NOE NN would be a recurrent network, and employing such a scheme in the training process results in a *parallel* identification method [240, 241]. Further, a combination of (139) and (140) can be taken into account to consider both the state and output noises in the dynamic model, simultaneously. Such a method, which results in a Nonlinear Autoregressive with Moving average and exogenous inputs (NARMAX) model, requires both the past measured system outputs and

model outputs in the dynamic model [240]. Besides, a class of identification techniques has been introduced in the literature to combine the advantages of both the parallel and series-parallel approaches, which are typically developed based on the average of the measured outputs and the predicted outputs.

Concerning the difference between these two types of NNs, the use of series-parallel identification prevents several complexities of training a recurrent network, thereby guaranteeing the training convergence. In addition, the series-parallel structure, which is also known as the teacher forcing method, leads to a faster training speed [242, 216]. On the other hand, parallel identification suffers from stability problems and complicated training methods [243, 244]. However, it should be noted that a NARX NN can be used only in the case of the *single-step ahead prediction*, while in multi-step ahead predictions, there is a need for an NOE network. Although one can convert the NARX neural network after the training process to an NOE network, the use of the series-parallel approach in the training phase may lead to inaccurate predictions for long prediction horizons [241].

NARX NNs have been used in [198] to identify the nonlinear model of an F-16 aircraft using a hybrid offline-online training algorithm considering model uncertainties. The NN's parameters have been trained using the LM method. Subsequently, the obtained model has been employed in a fault-tolerant NN-based adaptive PID attitude control system. A similar identification technique has been used in [216] for a similar aircraft model, where the identified model has been utilized in a predictive attitude controller. To train the network parameters, an adaptive updating rule with exponential forgetting has been derived based on a recursive formulation of the Gauss–Newton method, which is similar to the OS-ELM algorithm introduced in the following.

On the other hand, concerning the training algorithm of a NN in an identification problem, in contrast to the FEL method, which has been developed based on the Lyapunov stability theorem, a variety of online training algorithms have been introduced in the framework of an open-loop identification problem, which is typically developed based on the minimization of the mean squared

prediction error [40]. As a simple and popular method, OS-ELM, which has been developed based on the Recursive Least Squares (RLS) optimization [245], can be effectively employed in online identification problems. The use of such an approach to identify the system dynamics (based on the RLS optimization) can result in a significantly better performance (compared to the FEL-based method) in the structure of the trajectory tracking control of a damaged aircraft [38]. In the following, a brief description of the OS-ELM algorithm is given. Extreme learning machine (ELM), which can be considered as a single-hidden-layer feedforward neural network with random constant weights and biases in the hidden layer, has been employed in several studies as a part of the control system. This is due to the simple linear learning method of this type of NNs in which only the output weights of the NN are trained during the identification process [47]. Now, consider an ELM as $f(u) = W^T\mu(u)$ to identify the unknown mapping between system inputs (u) and outputs (y) in the case of a SISO system (a similar formulation can be presented for MIMO systems [195]). Considering a set of system inputs-outputs $\mathcal{D} = \{(u(k), y(k))\,|k = 1, ..., K\}$ with K distinct samples, the introduced ELM can be trained through the data set $\mathcal{D}$. As a result, ideally, we should have $\Phi W = Y$ where,

$$\Phi = [\mu(u(1)) \quad \cdots \quad \mu(u(K))]^T, \tag{141}$$

$$Y = [y(1) \quad \cdots \quad y(K)]^T. \tag{142}$$

Assuming $K > N$, $\Phi_{K\times N}$ becomes a non-square matrix. In such cases, the optimal weights of ELM can be determined using the least-squares optimization as $\hat{W} = \Phi^{\dagger}Y$, where $\Phi^{\dagger} = (\Phi^T\Phi)^{-1}\Phi^T$ denotes the pseudo-inverse of Φ [246]. However, in the case of online training problems, the incoming data are obtained one by one. Thus, $\hat{W}$ can be updated at each iteration using a recursive formulation as follows:

$$\hat{W}(k+1) = \hat{W}(k) + \ell(k)e(k), \tag{143}$$

$$P(k+1) = \left(I - \ell(k)\mu(u(k))^T\right)P(k), \tag{144}$$

where,

$$e(k) = y(k) - \hat{W}(k)^T \mu(u(k)), \tag{145}$$

$$\ell(k) = \frac{P(k)\mu(u(k))}{1 + \mu(u(k))^T P(k)\mu(u(k))}. \tag{146}$$

$$\tag{147}$$

As discussed in [35], there is a need for a persistent exciting regressor $\mu(u(k))$ to ensure the convergence of $\hat{W}$ to its optimal value. Various types of OS-ELM have been proposed in the literature for different identification and control purposes [247, 248]. OS-ELM with constant or variable forgetting mechanism has been widely utilized by researchers to identify time-varying system dynamics [249, 250]. Further, the OS-ELM algorithm can be adopted, in a similar manner, to identify the vector of unknown parameters corresponding to linear-in-parameters model uncertainty in the dynamic model or to relative weights of local models in a multi-model ensemble. Such an approach has been utilized in [195] to identify the unknown coefficients corresponding to actuator faults in the case of a quadrotor UAV. Subsequently, a trajectory tracking method has been proposed using an acceleration-based model predictive control, which ensures the bounded tracking error in the presence of system constraints. Besides, a hybrid offline-online identification scheme has been presented in [199] for a generic transport model in the presence of actuator faults. A set of local NARX NNs has been first trained under specific flight conditions and actuator faults, and subsequently, they have been aggregated as a single model using a set of adaptive weights updated using an OS-ELM-like approach. A similar method has been adopted in [200] to develop a fault-tolerant trajectory tracking control based on a modified model predictive control. The proposed approach leads to acceptable trajectory tracking even in the presence of unexpected actuator faults and flight conditions.

Although, due to the universal approximation property, NNs can estimate almost all continuous dynamic systems using a sufficient number of hidden nodes, increasing the hidden nodes may lead to the overfitting problem [251]. To be

more precise, the generalization capability of NNs in modeling the system dynamics is a crucial issue in utilizing them in a wide range of operating conditions, which are not necessarily covered in the training stage [252]. This, in turn, may lead to different considerations about the training of a NN, such as employing PE input signals, selecting appropriate frequency range for input signals according to dynamic modes of the system, determining optimal network structure, etc. These concerns have been thoroughly addressed in the field of system identification [253], which are beyond the scope of this paper.

4.1.2. Deep neural networks

As an alternative, deep NNs utilize more hidden layers rather than increasing the hidden node in a single hidden layer. In this regard, Convolutional Neural Networks (CNN) can be considered as one of the most important deep NNs. CNN has a cascade connection structure. Each CNN cell has two layers: the convolution layer and the sub-sampling layer. Also, the last layer is fully connected. The output of a CNN can be formulated as $y(k) = V\Phi(x(k))$, where Φ represents the operation of hidden layers and V is the weight vector of the final layer. As discussed in [254], CNN is an extremely powerful tool for the identification of nonlinear systems. This is due to the following facts: the convolution operation in CNN is the same as the input-output relation of the linear time-invariant systems; a CNN employs sparse connectivity and shared weights, thereby reducing the NN parameters and the risk of the over-fitting issue; the multi-level pooling results in a robust identification scheme against the measurement noises. However, despite the above-mentioned characteristics, few researchers have addressed the development of flight control systems using a CNN-based identified model. CNN has been utilized in [232] to identify uncertain terms induced by hidden states, varying inertia, and aerodynamic disturbances in the dynamic model of a helicopter UAV. More precisely, the dynamic model consists of a simple nominal dynamic model and a set of CNNs. A two-step optimization process has been adopted. First, the parameters of a nominal first-principles-based model have been optimized using the least-squares

method, where model uncertainties have been neglected at this stage. Subsequently, the parameters of deep CNNs have been determined in an open-loop optimization using the Stochastic Gradient-Descent (SGD) method. The dynamic model has been trained and validated under different aerobatic maneuvers. Afterward, an adaptive backstepping controller has been designed for the air vehicle which ensures semi-global UUB stability. The use of CNNs in the dynamic model to identify different types of model uncertainties results in a less conservative control system compared to conventional FEL-based controllers, which attempt to compensate for only bounded uncertain terms.

Moreover, concerning the direct employment of CNNs in the control system, CNN is an appropriate choice for high-level control schemes (such as localization and path planning), due to its excellent capability in extracting useful information, particularly from images [255, 256, 257].

4.2. Neuroadaptive optimal control

4.2.1. Optimal control formulation (HJB vs. HJI equations)

The feedback control law may be obtained using an H_2 or H_∞ optimal control problem at each time step. To be more precise, considering a nonlinear affine model as $\dot{x} = F(x) + B(x)u$, we can define a cost-to-go function as follows:

$$V(x) = \int_t^\infty L(x, u) d\tau, \tag{148}$$

where $L(x, u)$ represents a running cost function. The introduced cost function is also known as a *value* function if the running cost $L(x, u)$ is considered as a *reward* function (this is the common notation in the framework of reinforcement learning). Notice that, here, it has been assumed that $x_d = 0$. Thus, in the case of trajectory tracking problems, we should consider the system dynamics as $\dot{e} = F(x) + B(x)u - \dot{x}_d$ and substitute $x(t)$ in (148) by $e(t) = x(t) - x_d(t)$. Now, defining the Hamiltonian as

$$H(x, \lambda, u) = L(x, u) + \lambda^T(t) \left(F(x) + B(x)u \right), \tag{149}$$

with λ denotes the Lagrange multiplier, the optimal control law can be obtained within the framework of *dynamic programming* by the following equation [258],

$$0 = \min_{u} H(x, \frac{\partial V^*}{\partial x}, u), \tag{150}$$

where the superscript * stands for the optimal solution. In the literature, (150) is known as the Hamilton-Jacobi-Bellman (HJB) equation, while, in general, there is no analytic solution for it. It is notable that, in the case of unconstrained linear time-invariant (LTI) systems and using a quadratic running cost L, the HJB equation reduces to the well-known algebraic Riccati equation [259, 260].

Now, using a quadratic running cost as $L(x, u) = x^T Qx + u^T Ru$ with Q and R denote positive definite matrices, one can obtain the optimal control law as

$$u^*(x) = -\frac{1}{2} R^{-1} B^T(x) \frac{\partial V^*}{\partial x}. \tag{151}$$

By substituting (151) in the HJB equation (150), it is obtained that:

$$x^T Qx - \frac{1}{4} \nabla V^{*T} BR^{-1} B^T \nabla V^* + \nabla V^{*T} F(x) = 0, \tag{152}$$

where $\nabla V^* = \frac{\partial V^*}{\partial x}$. Accordingly, the optimal cost function is determined by solving the differential equation (152) considering the boundary conditions, and subsequently, the optimal control law is computed using (151) at each time.

A similar discussion can be provided in the framework of an H_∞ control problem in which the control objective is to achieve closed-loop stability while attenuating external disturbances. More precisely, consider the nonlinear dynamic model $\dot{x} = F(x) + B(x)u + D(x)w$, where w denotes external disturbances. Accordingly, considering a running cost function $L(x, u, w)$, the optimal control problem can be formulated as [261]

$$0 = \min_{u} \max_{w} H(x, \nabla V^*, u, w), \tag{153}$$

where

$$H(x, \lambda, u) = L(x, u) + \lambda^T(t) \left(F(x) + B(x)u + D(x)w \right). \tag{154}$$

Eq. (153), which is known as the Hamilton-Jacobi-Isaacs (HJI) equation, represents a minimax optimization problem. It can be referred to as a two-player

differential game, where the player u attempts to minimize the cost function while the player w tries to maximize it [262]. Again, by defining a quadratic running cost as

$$L(x, u, w) = x^T Qx + u^T Ru - \beta^2 w^T Pw, \tag{155}$$

with β and P represent, respectively, a positive constant and a positive definite matrix, the optimal control law and the worst-case disturbance can be obtained, respectively, as (151) and

$$w^*(x) = \frac{1}{2\beta^2} P^{-1} D^T(x) \nabla V^*. \tag{156}$$

By substituting (151) and (156) in (153), the HJI equation becomes as [263, 262]

$$x^T Qx + \frac{1}{4} \nabla V^{*T} E \nabla V^* + \nabla V^{*T} F(x) = 0, \tag{157}$$

$$E = \frac{1}{\beta^2} DP^{-1} D^T - BR^{-1} B^T. \tag{158}$$

In this regard, V^* should be determined by solving the HJI Partial Differential Equation (PDE) (157) considering the boundary conditions, and the optimal control law is then computed using (151) at each time. Unfortunately, finding the solution of HJ PDEs (152) or (157) is not generally an easy task at all. Another challenging issue in such optimal control problems is that they require the complete system dynamics model, which may not be available in real applications.

4.2.2. Approximate dynamic programming (continuous-time systems)

Different approaches have been introduced in the literature to provide a numerical approximation for these control problems [260]. These approaches are typically addressed within the framework of approximate (or sometimes adaptive) dynamic programming (ADP) [264, 265, 266]. The principal difference between such adaptive controllers and the previously proposed control methods in Section 2 is that here, we attempt to determine the approximate *optimal* control law, adaptively, while the previous control methods do not necessarily satisfy the optimality condition. *Policy iteration* and *Value iteration* are the

well-known methods in the literature to determine the approximate solution of HJ equations [267]. The policy iteration method consists of a policy evaluation and a policy improvement step at each iteration. In the ith iteration, first, the value function $V^{(i)}(x)$ corresponding to the current control law $u^{(i)}(x)$ is computed by solving $H\left(x, \nabla V^{(i)}(x), u^{(i)}(x)\right) = 0$, while in the second step, the control law is updated using (151) (a similar approach can also be taken into account in the case of the HJI equation). Such an iterative method will continue until the convergence of the policy function $u(x)$. As discussed in [268], the policy iteration algorithm will converge to the optimal solution by having an initial stabilizing control law (policy). On the other hand, the value iteration method includes an iterative approach for finding the optimal value function, and once the optimal value function is determined, the optimal policy can be explicitly computed using (151) [269]. Unlike the policy iteration, the value iteration does not require an initial stabilizing control law. In a more general view, however, both methods can be expressed within the framework of the *generalized policy iteration* [269, 270]. The concept of the generalized policy iteration can be defined as a set of interacting approximate policy evaluation and policy improvement steps, in which in the first step, we do not completely evaluate the cost of a given control law, but only update the current cost estimate *towards* that value. Similarly, in the policy improvement step, the control policy is not fully updated to the minimizing policy for the new cost estimate, but we only update the policy *towards* that policy. Nevertheless, the convergence analysis of such an ADP scheme, in a general case, is not trivial.

Owing to the unique capabilities of NNs in learning different nonlinear functions, traditionally, two NNs were employed as the actor and critic network to approximate the optimal policy and value function, respectively [267]. Such an approach can be categorized as a Heuristic Dynamic Programming (HDP) scheme [271, 265]. In the following, we focus on solving the HJB equation, while a similar discussion can be provided in the case of the HJI equation. Accord-

ingly, we have

$$V(x) = W_v^{*T}\mu_v(x) + \varepsilon_v, \hat{V}(x) = \hat{W}_v^T\mu_v(x), \tag{159}$$

$$u(x) = W_u^{*T}\mu_u(x) + \varepsilon_u, \hat{u}(x) = \hat{W}_u^T\mu_u(x), \tag{160}$$

where $V(x)$ represents the corresponding value function of $u(x)$, which satisfies $H\left(x, \nabla V(x), u(x)\right) = 0$. Thus, it is obtained that

$$x^TQx + u^TRu + \left(\nabla V(x)\right)^T\left(F(x) + B(x)u\right) = 0, \tag{161}$$

$$u(x) = -\frac{1}{2}R^{-1}B^T(x)\nabla V. \tag{162}$$

Consequently, knowing that $H\left(x, \nabla V^*, u^*\right) = 0$, we can define

$$\begin{aligned} e_c = H(x, \nabla\hat{V}, u) - H\left(x, \nabla V^*, u^*\right) = x^TQx \\ + u^TRu + \hat{W}_v^T\nabla\mu_v(x)\left(F(x) + B(x)u\right), \end{aligned} \tag{163}$$

where the last equality is obtained using $\nabla\hat{V}(x) = \left(\nabla\mu_v(x)\right)^T\hat{W}_v$. Therefore, an appropriate training rule may be obtained by minimizing $E_c = 1/2e_c^2$. Using a normalized gradient descent algorithm, we have

$$\begin{aligned} \dot{\hat{W}}_v = \dot{\tilde{W}}_v &= -\alpha_c\frac{\partial E_c/\partial\hat{W}_v}{\left(1+\phi^T\phi\right)^2} = -\alpha_c\frac{\phi}{\left(1+\phi^T\phi\right)^2}e_c \\ &= -\alpha_c\frac{\phi\phi^T}{\left(1+\phi^T\phi\right)^2}\tilde{W}_v + \alpha_c\frac{\phi}{\left(1+\phi^T\phi\right)^2}\left(\nabla\varepsilon_v(x)\right)^T\left(F(x) + B(x)u\right), \end{aligned} \tag{164}$$

where $\phi = \nabla\mu_v(x)\left(F(x) + B(x)u\right)$ and $\tilde{W}_v = \hat{W}_v - W_v^*$. However, due to the unknown value of u, it should be substituted by $\hat{u}$. As can be observed in (164), such a training algorithm requires the PE condition to ensure $\lambda_{min}(\phi\phi^T) > 0$ [272]. Further, the updating rule of $\hat{W}_u$ would be obtained according to (162). However, there is a need for a nonstandard modification term in this updating, which consists of the cross-product of the actor and the critic networks' weights to ensure the closed-loop stability. The obtained training rule as well as the proof of the UUB stability of the system (under conservative assumptions) can be found in [270].

Alternatively, event-triggered optimal control schemes have been introduced in the literature, where the control law is updated only at the time instants that a triggering condition is satisfied, while it remains constant in other times. Such a control scheme can significantly reduce the online computational cost of a controller. An event-triggered optimal control has been introduced in [273], where the updating rule of the critic network has been derived similarly to (164). In addition, the actor network's updating rule has been obtained using (162) by defining

$$e_u = \hat{W}_u^T \mu_u(x) + \frac{1}{2} R^{-1} B^T(x) \left(\nabla \mu_v(x)\right)^T \hat{W}_v. \tag{165}$$

Accordingly, by defining $E_u = 1/2 e_u^2$, the updating rule of $\hat{W}_u$ at the jth triggering instant t_j is obtained as

$$\begin{aligned} \hat{W}_u(t_j^+) &= \hat{W}_u(t_j) - \alpha_u \frac{\partial e_u}{\partial \hat{W}_u} e_u^T \\ &= \hat{W}_u(t_j) - \alpha_u \mu_u(x_j) e_u^T(t_j), \end{aligned} \tag{166}$$

where α_u denotes a positive constant. Similar to the above-mentioned design, there is a need for a robustifying term in the control law to ensure the closed-loop stability while requiring several conservative assumptions. Such an approach has been extended in [274] to a trajectory tracking control problem by defining an augmented state, which consists of both the tracking error and the desired trajectory. The designed controller has been subsequently applied to a linear model of the elevation of a Quanser helicopter.

Another alternative training rule for the critic network can be derived based on the method of weighted residuals [275, 276]. More precisely, at each step, $\hat{W}_v$ can be obtained by projecting e_c onto $\partial e_c / \partial \hat{W}_v$ and setting the result to zero, i.e.

$$\left\langle \frac{\partial e_c}{\partial \hat{W}_v}, e_c \right\rangle = 0, \tag{167}$$

where $\langle f, g \rangle = \int f^T g$. Thus, we have

$$\langle \phi, \phi \rangle \hat{W}_v + \langle L(x,u), \phi \rangle = 0, \tag{168}$$

which leads to

$$\hat{W}_v = - \langle \phi, \phi \rangle^{-1} \langle L(x,u), \phi \rangle . \tag{169}$$

Indeed, such an approach leads to the solution of the least-squares optimization. Subsequently, an improved control law can be obtained using $\hat{u}(x) = -\frac{1}{2}R^{-1}B^T(x)\left(\nabla\mu_v(x)\right)^T\hat{W}_v$. This process will continue until convergence. However, as discussed in [277], such an iterative optimization process still requires rich input signals to ensure the existence of $\langle\phi,\phi\rangle^{-1}$. In addition, computing the necessary integrals in (169) may be a complicated task in practice. Thus, they are typically approximated by discretization. This method has been employed in [277] to successively solve the HJI equation, where the designed controller has been applied to a linear model of a fighter aircraft.

It should be noted that the introduced actor-critic scheme can also be implemented using a single NN [262]. This can be performed by employing a critic NN to approximate the value function $V(x)$ and subsequently, computing the approximate optimal control law as

$$\hat{u}(x) = -\frac{1}{2}R^{-1}B^T(x)\left(\nabla\mu_v(x)\right)^T\hat{W}_v. \tag{170}$$

Accordingly, there is a need for a modification term in the updating rule (164) to ensure closed-loop stability. The modification term is obtained by assuming that the optimal control law $u^*(x)$ can stabilize the system such that the following equation holds [278].

$$\dot{J}_s(x) = \left(\nabla J_s\right)^T\left(F(x) + B(x)u^*\right) = -\left(\nabla J_s\right)^T\Lambda\left(\nabla J_s\right), \tag{171}$$

where $J_s(x)$ and $\Lambda(x)$ represent a Lyapunov function of the system (as a polynomial) and a positive definite matrix, respectively [267]. Consequently, the modification term is obtained by preventing the function $J_s(x)$ from increasing as follows:

$$\begin{aligned}\Delta\dot{\hat{W}}_v &= -\alpha_s\frac{\partial\dot{J}_s(x)}{\partial\hat{W}_v}\\ &= -\alpha_s\frac{\partial\left[\left(\nabla J_s\right)^T\left(F(x) + B(x)\hat{u}(x)\right)\right]}{\partial\hat{W}_v},\end{aligned} \tag{172}$$

where α_s represents a positive constant. A similar approach has been employed in [272] in an event-triggered H_∞ control problem to solve an HJI equation,

where the proposed method has been applied to a linear model of an F-16 aircraft. Besides, such a scheme has been adopted in [279] in combination with an NN-based state observer to provide a trajectory tracking controller for a helicopter UAV, where the NNs have been trained online by an *on-policy* learning method. In the on-policy learning, the control law that is applied to the system (called the *behavior policy*) is the same as the control law, which is evaluated and improved (called the estimation or *target policy*). On the other hand, in the *off-policy* learning scheme, the behavior policy and the target policy can be unrelated. The employment of off-policy learning in the control design process provides considerable advantages in comparison with on-policy learning schemes [277]. More specifically, in the on-policy H_∞ control, the external disturbance should be obtained by (156), while specifying the disturbance term is typically impractical in real systems. In addition, the issue of the *exploration* (which is partly related to the PE condition) is of significant importance to guarantee the convergence of the control law to the optimal policy. However, since in the on-policy learning, we should apply the target policy to the system, the exploration would be limited by the UAV trajectory.

Further, a remaining issue with all the above-mentioned designs is that they still depend on the system dynamics (i.e. $F(x)$ and $B(x)$). To tackle such a problem, model-free off-policy learning schemes have been introduced in the literature to provide an acceptable approximate solution for the optimal control problem. To this end, consider again the dynamic model of the system. By adding and subtracting the target policy $u^{(i)}(x)$ (at ith optimization iteration) to the model, it is obtained that

$$\dot{x} = F(x) + B(x)u^{(i)} + B(x)\left(u - u^{(i)}\right). \tag{173}$$

Thus, considering the value function V corresponding to $u^{(i)}(x)$, we can write

$$\dot{V} = (\nabla V)^T \dot{x} = (\nabla V)^T \left(F(x) + B(x)u^{(i)}\right) + (\nabla V)^T B(x)\left(u - u^{(i)}\right). \tag{174}$$

As a result, the policy evaluation equation (161) can be reformulated as follows

[277]:

$$\dot{V} = (\nabla V)^T B(x) \left(u - u^{(i)}\right) - x^T Q x - {u^{(i)}}^T R u^{(i)}, \tag{175}$$

By integrating from both sides of (175) in a specific time interval, a policy evaluation equation is obtained which is independent of the internal dynamics $F(x)$. Thus, we can redefine the design process by employing the policy evaluation equation (175) rather than (161). Such an approach, which is similar to the Integral Reinforcement Learning (IRL) scheme [280, 259], has been utilized in [277] to approximately solve an HJI equation, where the designed controller has been applied to a linear model of the longitudinal dynamics of an F-16 aircraft. Meanwhile, the model-free approach to optimal control may be better expressed within the framework of reinforcement learning, which will be discussed in the following subsection.

Another concern with above mentioned ADP schemes is that the obtained information from the system inputs-outputs data is used to update only a *scalar* function, i.e. the value function. This results in inefficient usage of data which may slow down the convergence. To deal with such an issue, another actor-critic ADP method has been introduced in the literature, which attempts to approximate the optimal value function derivative $\nabla V^*(x)$ (using the critic NN) rather than the value function itself. This method falls into the framework of Dual Heuristic Programming (DHP) [281]. Two sets of NNs have been employed in [282] as the actor and critic networks in a constrained minimum-time optimal control problem, i.e. the control of the flight path angle of a missile given a final Mach number. Indeed, instead of utilizing a single NN, a set of NNs have been trained offline as the actor (and critic) NN, which have been employed sequentially to determine $u^*(x)$ and $\nabla V^*(x)$ during the time. Also, to deal with the free-final time, the dynamic equations of the system have been reformulated considering the flight path angle as the independent variable, thereby providing a fixed-final condition problem. A similar actor-critic method has been utilized in [283], where an offline training stage has been performed using the linearized model of the air vehicle, and an online training phase has been employed to

improve the closed-loop performance. The designed controller has been applied to a fixed-wing aircraft considering model uncertainties, unmodeled dynamics, and actuator faults. However, the closed-loop stability was not analyzed in these papers.

4.3. Direct adaptive control using Reinforcement learning (RL)

The concept of adaptive optimal control, particularly for systems with unknown system dynamics, can be presented within the framework of Reinforcement Learning (RL) as well. Although the notion of RL and the optimal control theory share a somewhat similar idea, i.e. moving towards the optimal solution over time, they possess different mathematical notations due to their different origins [284]. In a conventional RL problem, which is typically formulated in the *discrete-time* domain, the objective is to search for an optimal control law (policy) for a dynamic system (agent) while interacting with an uncertain environment that maximizes the total reward obtained during an episode. Traditionally, the problem is formulated as a Markov Decision Process (MDP) described by a four-tuple (x, u, F, R). Here, x and u represent, respectively, the current system state and inputs. The system inputs are obtained according to a policy π, which in turn, results in receiving a reward $R(x, u)$. Further, F denotes the system dynamics model or (in a probabilistic formulation) a stationary transition distribution $F \sim P\left(x(k+1)|x(k), u(k)\right)$, which satisfies the Markov property [281]

$$P\left(x(k+1)|x(1), u(1), \cdots x(k), u(k)\right) = P\left(x(k+1)|x(k), u(k)\right). \tag{176}$$

Thus, the choice of appropriate states to satisfy the Markov property is of significant importance. However, although most of the theoretical achievements within the framework of RL have been obtained under such a property, many approaches can still work well for different practical problems which do not satisfy the Markov property [285]. Now, similar to the common notation in the

RL framework, consider a discrete-time optimal control problem as follows:

$$\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{k=t}^{\infty} \gamma^{k-t} R(x,u) \right], \tag{177}$$
$$\text{subject to } x(k+1) = F(x(k), u(k), d(k)).$$

Accordingly, the objective is to maximize the obtained accumulative reward, $R(x,u)$, and the expected value is computed considering the random external disturbance $d(k)$. Further, π_k and $\gamma \in (0,1)$ represent the current control command (policy) and the discount factor, respectively. Thus, the control command $u(k)$ is computed at each step using either the stochastic or deterministic policy π (in the case of stochastic policies, π represents the conditional probability distribution of the control command, i.e. $\pi(u|x)$, while concerning deterministic policies, we have $u(k) = \pi(x)$). It is notable that a similar discussion can also be made on the basis of an average reward rather than a discounted reward, which eliminates the requirement for a discount factor (for more details, see [285]).

The focus of this section is on a model-free optimal control approach. Like other adaptive control methods, we can employ either an indirect or a direct control design procedure. To be more precise, it is possible to first derive an estimation of the system dynamics model F and then attempt to (approximately) solve the optimization problem (177), or try directly to develop an optimal control policy. Within the framework of the RL, the former approach is known as the model-based RL, while the latter corresponds to the model-free RL. On the other hand, owing to the unstable behavior of typical aerial vehicles and the inherent trial and error scheme employed in RL, commonly, the learning phase should be performed in a simulation environment on an existing *model* of the system. Thus, even in the model-free RL, there is a requirement for a (simple) dynamic model of the system to be used in the learning phase (in the simulation environment). We will give a short insight into the method of eliminating the requirement for a dynamic model in the RL-based flight control systems at the end of this section. Concerning the model-based RL, however, the model would be obtained by the system identification method as discussed in Section

4.1. Consequently, apart from the model identification phase (required in the model-based RL), the learning process in the flight control design in both the model-based and model-free RL schemes can be discussed in the same fashion.

Now, in a general view, we can solve an adaptive optimal control problem within the RL framework through two different approaches: ADP and direct policy updating [286], which will be addressed in detail in the following.

4.3.1. Approximate dynamic programming (discrete-time systems)

Within the ADP framework, we first attempt to estimate the action-value function $Q(x, u)$, which is defined as follows [269]:

$$Q(x, u) = \mathbb{E}_\pi \left[\sum_{k=t}^{\infty} \gamma^{k-t} R(x, u) \middle| x(t) = x, u(t) = u \right]. \tag{178}$$

Notice that, it is also possible to derive the RL-based control formulation using the value function $V(x)$ rather than the action-value function. Indeed, such an approach would result in the discrete equivalent of the previously discussed ADP scheme for continuous-time systems. However, as will be observed in the following, the employment of the introduced action-value function instead of the value function can help to develop an entirely model-free control system [281]. Now, using the concept of DP, one can obtain the Bellman optimality equation as follows:

$$Q^*(x, u) = R(x, u) + \gamma \mathbb{E}_\pi \left[\max_{u'} Q^* \left(x(k+1), u' \right) \right], \tag{179}$$

where the superscript * denotes the action-value function corresponding to the optimal policy. Such an approach can be utilized in both the on-policy and off-policy iterative learning schemes to estimate the action-value function.

A traditional on-policy learning approach to iteratively estimate the action-value function is known as *Sarsa*. In this regard, by incorporating the *Temporal difference* (TD) error, which is equivalent to the Hamiltonian introduced in the previous subsection for continuous-time systems, an on-policy learning rule can

be derived as

$$Q_{k+1}(x,u) = Q_k(x,u) + \eta\left(R(x,u) + \gamma Q_k\left(x(k+1), u(k+1)\right) - Q_k(x,u)\right), \tag{180}$$

where $\eta \in \mathbb{R}^+$ denotes the learning rate. The second term on the right-hand side of the equation corresponds to the TD error. Subsequently, an improved action is chosen at each step using

$$\pi_{k+1}(x) = \arg\max_u Q_{k+1}(x,u). \tag{181}$$

Such an approach (using the multi-step TD, which is discussed in the following) has been employed in [287] to control a 2-DOF Quanser helicopter. The optimization problem has been presented for a linear model of the system, which leads to the well-known algebraic Riccati equation. Sarsa has also been adopted in [288] for a quite complex control problem, i.e. the control of glider soaring in a turbulent environment (by taking advantage of turbulent fluctuations), while such a problem has been dealt with in [289] by employing an off-policy value-iteration method.

Accordingly, if we estimate the action-value function corresponding to the current control policy using a NN as $\hat{Q}(x,u) = \hat{W}_q^T \mu_q(x,u)$, the network weights $\hat{W}$ can be updated at each step using the gradient descent method as follows [269]:

$$\begin{aligned}\hat{W}_q(k+1) =& \hat{W}_q(k) + \eta\mu_q(x,u)\big(R\left(x,u\right) + \\ & \gamma\hat{Q}\left(x(k+1), u(k+1)\right) - \hat{Q}\left(x(k), u(k)\right)\big),\end{aligned} \tag{182}$$

where η represents a positive learning rate, and x and u correspond to the current value of the system state and input. As seen, the proposed updating rule is similar to the training rule (7) employed in the FEL scheme, where the tracking error has been substituted by the TD error. A notable point, however, is that (182) is obtained using a *semi-gradient* method rather than a true gradient descent scheme. This is due to the employment of $R\left(x,u\right) + \gamma\hat{Q}\left(x(k+1), u(k+1)\right)$ as the target value of the action-value function, which

in turn is a function of $\hat{W}_q$, while the effect of it is not included in the gradient function.

Thereafter, in the policy improvement step, (181) can be solved as

$$\partial \hat{Q}(x,u)/\partial u = 0,$$

which shows well a principal advantage of employing the action-value function instead of the value function, hence we can simply determine the improved policy at each step by maximizing the Q-function with respect to u with no requirement for the system dynamics model.

On the other hand, concerning off-policy learning methods, an off-policy TD-based learning rule called the *Q-learning* can be derived as

$$Q_{k+1}(x,u) = Q_k(x,u) + \eta \left(R(x,u) + \gamma \max_{u'} Q_k\left(x(k+1), u'\right) - Q_k(x,u) \right), \tag{183}$$

while, again, the improved policy would be determined using (181). Such a learning method has been adopted in [290] to learn the optimal servoing gain in an Image-Based Visual Servoing (IBVS) design for the trajectory tracking control of a quadrotor UAV, while the learning rate η has been updated using a fuzzy controller.

In a similar manner to Sarsa, the NN-based estimation can also be adopted in the Q-learning algorithm. The corresponding updating rule is obtained by substituting $\hat{Q}\left(x(k+1), u(k+1)\right)$ in (182) by $\max_{u'} \hat{Q}\left(x(k+1), u'\right)$ or

$$\sum_u \pi\left(u|x(k+1)\right) \hat{Q}\left(x(k+1), u\right)$$

in the case of stochastic target policy π. Such a scheme has been employed in [291] to control an airship in a 3D environment, where the scale of the state space was reduced by a coordinate transformation. Different variants of the Q-learning method have been presented in the literature, which are beyond the scope of this paper (see for example [284, 292]).

In the Q-learning, a common choice for the behavior policy is to choose either the current improved target policy (with a probability of $1-\epsilon$) or a random

action (with a probability of ϵ, where ϵ denotes a small positive constant). Such a behavior policy results in a good exploration, which is critical in the convergence of off-policy algorithms. As an alternative, an evolutionary exploration algorithm has been introduced in [293] in which a set of random trajectories are generated at each step while the mean and the variance of them are updated considering the obtained reward corresponding to each trajectory in such a way that the resultant behavior policy moves toward better trajectories. Such an approach has been adopted in [293] to train a flapping-wing aerial vehicle using the Q-learning. Nevertheless, NN-based off-policy learning algorithms suffer from convergence issues in various problems. The updating rules, which are derived based on the true gradient descent method (such as the gradient-TD method) can address this issue at the expense of excessive computational complexity, while their performance in real applications is still not clear. In this regard, a comprehensive comparison between the performance of semi-gradient methods and TD methods those based on true gradient descent, in the case of intelligent flight control systems, is a necessity in future research.

It is also possible to derive an (on-policy) updating rule by attempting to eliminate the TD error at each time step using a least-squares optimization (or an RLS optimization similar to the OS-ELM approach introduced in Section 4.1) to solve the following equation.

$$\hat{W}_q(k+1)^T \left(\mu_q(x,u) - \gamma\mu_q\left(x(k+1), u(k+1)\right)\right) = R\left(x,u\right). \tag{184}$$

To this end, there is a requirement for the regression vector

$$\left(\mu_q(x,u) - \gamma\mu_q\left(x(k+1), u(k+1)\right)\right)$$

to be persistent exciting [281]. Such a method has been utilized in [294] to generate the desired trajectory for a quadrotor aimed to transport a suspended load.

As discussed, both the Q-learning and Sarsa have been developed based on the TD method, where its iterative updating rule bases in part on the *current estimation* of Q. Thus, they are known as *bootstrapping* methods. Further,

notice that the proposed schemes are developed based on a simple *one-step* TD error. More complex and effective learning rules can be derived by employing *multi-step* TD. Multi-step TD learning is indeed a bridge between the simple one-step TD learning and the Monte Carlo method wherein the updating rule is derived using the entire sequence of rewards obtained from the current state until the end of the episode. A detailed description of multi-step TD can be found in [269]. Compared to the TD method, the Monte Carlo approach could not be used in an online training scheme, because we should wait until the end of the episode to determine the obtained rewards corresponding to the current policy. On the other hand, there are some concerns with the convergence of the TD learning, which is a bootstrapping method, particularly under the usage of neural approximation. The Monte Carlo method has been adopted in [295] to maximize a value function in order to develop a controller for a helicopter in low-speed aerobatic maneuvers, e.g. the inverted flight of the aerial vehicle, where the optimization process has been performed in the simulation environment using an identified stochastic, nonlinear model of the system. Using the Monte Carlo method, a collision-avoidance control system has been proposed in [296]. In this regard, after the training of the action-value function, which was modeled by a CNN, the control command, i.e. the velocity direction of the UAV, could be obtained by maximizing the Q-function at each step. An intelligent trajectory generation approach has been proposed in [297] for a UAV aimed to collect information from the environment considering the constraint on the total energy consumption of the vehicle. A CNN has been utilized to estimate the Q-function using an off-policy modified Deep RL (DRL) method. In contrast to the TD and Monte Carlo methods, in the DRL method, a replay buffer has been utilized, which stores a finite number of tuples of (x_k, u_k, r_k, x_{k+1}) obtained under an exploration (behavior) policy. Subsequently, at each step, a mini-batch of samples is chosen uniformly from the entire buffer allowing for a set of *uncorrelated* samples to be used in the training process. In addition, a copy of the main NN called the target network has been generated, where the target network, which provides the target value for training the main critic

network, is trained with a significantly less learning rate, thereby avoiding the learning divergence. The two above-mentioned high-level control systems can be considered as preliminary intelligent path planning designs, which could be integrated with conventional IFCSs to provide a completely intelligent flight control system. The development of such a combination would be a critical step to develop a truly intelligent UAV, while, due to the complicated and high-dimensional nature of the problem, it has not been thoroughly addressed by researchers yet.

Despite the simplicity of introduced approaches to approximate the optimal action-value function (and subsequently, the optimal policy), they still face fundamental challenges to ensure the convergence to the optimal solution (particularly in the case of off-policy algorithms). More specifically, different impractical assumptions (such as the requirement for visiting all possible state-action pairs for an infinite number of times) have been adopted in the literature to achieve the convergence property [269].

4.3.2. Direct policy updating

Another approach to solve the optimization problem (177) is to directly update the approximate optimal policy rather than employing an estimated action-value function to find the optimal policy. More precisely, here, we attempt to directly find an appropriate updating rule for the approximate optimal policy, which is estimated by a NN as $\pi(x) = \hat{W}_\pi^T \mu_\pi(x)$, or $\pi(u|x) = \hat{W}_\pi^T \mu_\pi(x, u)$ in the case of a stochastic policy (for the ease of notation, we do not use the $\hat{\cdot}$ symbol for the estimated optimal policy in the rest of this section). Such a direct policy parametrization brings a principal advantage into the control design process that we can incorporate the prior knowledge of the optimal policy in the parametrization of the estimated optimal policy.

In this context, the most commonly used approach, called the *policy gradient* method, attempts to update the network weights $\hat{W}_\pi$ by moving in the direction of the gradient of a performance function in order to improve $\pi(x)$. Typically, the value function $V_\pi(x)$ (the subscript π indicates that the value function has

been computed along the trajectory obtained by π) is chosen as the performance function. In the following, we first give a brief introduction to the policy gradient theorem for stochastic policies and then address the corresponding theorem of deterministic policies as a special case. Now, defining the advantage function as

$$A_\pi(x,u) = Q_\pi(x,u) - V_\pi(x), \tag{185}$$

one can derive an equation for the difference between the value functions corresponding to two different policies as follows [298, 299]:

$$V_\pi(x_0) - V_\varpi(x_0) = -\sum_x \rho_\varpi(x) \sum_u \varpi(u|x) A_\pi(x,u), \tag{186}$$

where,

$$\rho_\varpi(x) = \sum_k \gamma^k P\left(x(k) = x | x(0) = x_0\right), \tag{187}$$

denotes the unnormalized discounted visitation frequency, where actions are determined according to ϖ. Let ϖ be a fixed policy (which may be considered as the behavior policy in off-policy methods) and π corresponds to the estimated optimal policy. Thus, using the fact that $\sum_u \pi(u|x) A_\pi(x,u) = 0$ and

$$\sum_u \left(\varpi(u|x) - \pi(u|x)\right) V_\pi(x) = 0,$$

we have [300]:

$$\begin{aligned} V_\pi(x_0) - V_\varpi(x_0) &= \sum_x \rho_\varpi(x) \sum_u \left(\pi(u|x) - \varpi(u|x)\right) Q_\pi(x,u) \\ &= \sum_x \rho_\varpi(x) \sum_u \left(\pi(u|x) - \varpi(u|x)\right) A_\pi(x,u). \end{aligned} \tag{188}$$

By estimating the target policy as $\pi(u|x) = \hat{W}_\pi^T \mu_\pi(x,u)$ and differentiating both sides of the (188) with respect to $\hat{W}_\pi$, one can obtain the gradient of the performance function as follows:

$$\nabla V_\pi(x_0) = \sum_x \rho_\varpi(x) \sum_u \nabla\pi(u|x) Q_\pi(x,u) + \left(\pi(u|x) - \varpi(u|x)\right) \nabla Q_\pi(x,u), \tag{189}$$

where $\nabla = \frac{\partial}{\partial \hat{W}_\pi}$. The obtained result is analogous to the off-policy actor-critic algorithm proposed in [301], while the second term on the right-hand

side of (189) is neglected in [301]. A similar equation can also be derived by substituting $Q_\pi(x,u)$ in (189) by $A_\pi(x,u)$. Now, considering the special case $\pi = \varpi$ in (189), it is obtained that

$$\nabla V_\pi(x_0) = \mathbb{E}_{\rho_\pi,\pi} \frac{\nabla \pi(u|x)}{\pi(u|x)} Q_\pi(x,u), \tag{190}$$

$$\nabla V_\pi(x_0) = \mathbb{E}_{\rho_\pi,\pi} \frac{\nabla \pi(u|x)}{\pi(u|x)} A_\pi(x,u). \tag{191}$$

The first equation is known as the fundamental equation of the policy gradient theorem, while the second one is called the policy gradient with baseline, which in turn reduces the variance of the algorithm, thereby improving the performance. Now, the NN weights can be updated through either an off-policy or on-policy method using each data sample as follows:

$$\hat{W}_\pi(k+1) = \hat{W}_\pi(k) + \eta\rho(x,u) \frac{\nabla \pi(u|x)}{\pi(u|x)} \hat{A}_\pi(x,u), \tag{192}$$

where $\rho(x,u) = \frac{\pi(u|x)}{\varpi(u|x)}$, called the importance sampling ratio, is employed to compensate for the fact that the data samples have been collected under the behavior policy $\varpi(u|x)$ rather than the estimated target policy $\pi(u|x)$ (in the on-policy learning, we have $\rho = 1$). Further,

$$\hat{A}_\pi(x,u) = R(x,u) + \gamma \hat{V}_\pi(x(k+1)) - \hat{V}_\pi(x), \tag{193}$$

denotes an estimation of the advantage function. As seen, it requires the estimation of the value function, which can be obtained by a critic network using the introduced learning schemes in the previous section for the action-value function, while in the case of the off-policy learning of the value function, unlike the learning algorithm of the action-value function, we should again employ the importance sampling ratio in the updating rule [269].

A variety of *conservative* approximated policy gradient approaches have been introduced in the literature to restrict the policy update at each step, thereby improving the performance of the method. This is due to the great effect of the *magnitude* of $\Delta\hat{W}_\pi$ (which can also be controlled by the learning rate) in each iteration of the policy gradient on the performance and the convergence

of the algorithm. Trust Region Policy Optimization (TRPO) [298] and Proximal Policy Optimization (PPO) [302] are two common methods in this field. TRPO employs a constrained optimization problem in which an approximated value function is optimized (through updating the target policy) subject to a constraint on the KL divergence of the old policy and the new policy. The KL divergence represents a measure of the divergence of a distribution from the other one. On the other hand, the PPO introduced a simplified design to keep the ratio of the new policy to the old policy in a permissible range. Such an approach has been utilized in [300] to develop a trajectory tracking control for a quadrotor air vehicle.

Policy gradient theorem can also be extended to deterministic policies, which is called the Determinist Policy Gradient (DPG) [303]. To this end, consider again (188) while substituting the probability distribution $\pi(u|x)$ with the Dirac delta function, i.e. $\pi(u|x) \sim \delta\left(u - \pi(x)\right)$, which is equivalent to a deterministic policy. Subsequently, knowing that

$$\sum_u \left(\delta\left(u - \pi(x)\right)\right) Q_\pi(x, u) = Q_\pi(x, \pi(x)),$$

by estimating $\pi(x)$ as $\hat{W}_\pi^T \mu_\pi(x)$ and differentiating both sides of (188), it is obtained that

$$\nabla V_\pi(x_0) = \sum_x \rho_\varpi(x) \Bigg(\nabla Q_\pi(x, \pi(x)) + \sum_u \left(\delta\left(u - \pi(x)\right) - \varpi(u|x)\right) \nabla Q_\pi(x, u) \Bigg). \tag{194}$$

Thus, knowing that

$$\nabla Q_\pi(x, \pi(x)) = \nabla \pi(x) \nabla_u \, Q_\pi(x, u)|_{u=\pi(x)} \,,$$

one can obtain the on-policy DPG algorithm by setting $\pi = \varpi$ in (194) as follows:

$$\hat{W}_\pi(k+1) = \hat{W}_\pi(k) + \eta \nabla \pi(x) \nabla_u \, \hat{Q}_\pi(x, u)\Big|_{u=\pi(x)} \,, \tag{195}$$

where $\hat{Q}_\pi(x, u)$ is the estimated action-value function, which can be obtained using a critic network trained by the Sarsa algorithm through (182).

Concerning the off-policy DPG, note that there is an additional term in (194), while similar to the stochastic policy gradient theorem, it is neglected in [303]. Thus, the off-policy DPG equation is obtained again as (195), whereas the estimated action-value function computed using the critic network should be trained by the Q-learning method. Similar to the stochastic policy gradient, it is also possible to derive the DPG algorithm using the advantage function rather than the action-value function in (195).

DPG would be more convenient in the control design process since a stochastic policy results in unpredictable behavior, which is not desirable in autonomous vehicles. However, the exploration strategy in DPG is of significant importance to avoid the convergence to local optima. A common choice to provide an acceptable exploration is adding white noise to the current optimized policy at each step to obtain an exploratory behavior policy.

A simplified version of the introduced (on-policy) actor-critic scheme has been employed in [304] and [305] to stabilize and control a nonlinear model of an Apache helicopter, respectively, while in [305], three cascaded NNs have been employed in the action network (equivalently to conventional multi-loop control systems) to improve the training performance. An on-policy DPG employing the Monte-Carlo method (rather than the TD method), which updates the actor and critic networks after the end of each episode, has been used in [10] to control a quadrotor UAV, where a constrained optimization has been utilized in the design to avoid large policy updates at each step, similarly to the TRPO method. In addition, the *natural gradient descent*, which attempts to include the effects of the performance function's curvature induced by higher-order derivatives into the updating rule [306], has been employed in the training rule instead of the conventional gradient descent algorithm. The control scheme has been subsequently applied to a real quadrotor air vehicle, while it suffers from the huge computational cost of the (offline) training phase, which is performed in a simulation environment.

Deep DPG (DDPG) has been introduced in [307] as a combination of the DPG and DRL to employ (deep) NN in a stable manner as the actor and

critic estimators, where, here, *target networks* (which are employed in DRL for the critic network) are defined for both the actor and critic networks. An off-policy DDPG has been adopted in [308] to control a quadrotor UAV considering external disturbances and actuator faults (only in the flight tests). Adopting the concept of DRL results in more efficient training with improved stability, while off-policy learning allows for utilizing an exploratory behavior policy, which is independent of the estimated target policy. However, as discussed in the paper, the combination of the (neural network) function approximation, the bootstrapping scheme (due to the TD learning), and the off-policy learning can lead to significant bias and variance in estimations (while in some cases, it may result in the divergence of the algorithm [269]). To deal with such an issue, an integrator has been placed at the input of the actor network, which significantly reduces the steady-state error. Besides, a hybrid offline-online training method has been employed to improve the target policy during the real flight, while no experimental results have been included in the paper. DDPG has been utilized in [309] to address the autonomous landing of a UAV on a moving platform, while the problem has been dealt with in a 2D environment. As mentioned in the paper, DDPG can be an optimal choice in control problems with low-dimensional continuous states and actions. Further, the *shaping* method has been utilized in the paper to design an appropriate reward function in which the *progress* of the UAV in approaching the desired goal between two successive time steps has been considered as the reward function. It has been claimed that such a technique would results in a faster learning process [310], though at the cost of significant design effort and possible change of the optimal solution. This is similar to the *reward shaping* method introduced in [311], wherein a potential-based function is summed with the basic reward function to speed up the learning process with no effect on the optimal policy. To develop an intelligent UAV navigation system in large-scale complex environments (with no requirement for map reconstruction), authors in [312], involved the concept of *Partially Observable MDPs* (POMDP) within the framework of DRL. In a POMDP, at each step, we can observe only a part of the system state denoted

by o_t, which does not satisfy the Markov property, and so, the current policy requires the entire previous trajectory $\tau_t = (u_0, o_0, \cdots, u_{t-1}, o_{t-1})$ to determine the control command. Such a framework provides the capability of capturing the complex features of the environment by storing the previous trajectory of the system. Accordingly, a determinist policy gradient theorem, called the Fast-recurrent DPG, has been introduced in the paper to deal with POMDPs in which ∇V_π is computed similar to (194) except that the current state x is replaced by τ_t. In a similar manner, a combination of the POMDP and the deep Q-learning concepts has been utilized in [313] to address the obstacle avoidance problem in the case of a UAV with limited environment knowledge, where a recurrent NN has been employed as the estimator of the Q-function to better estimate the current system state using information from an arbitrarily long sequence of observations.

As a notable shortcoming, almost all of the RL-based control strategies require a remarkable time for offline training of NNs to be employed in a real application. A preliminary study has been given in [314] in which a quadrotor can learn to hover by a relatively small amount of training data using the model-based RL. The incoming data are first employed to build a dynamic model of the system followed by a policy updating algorithm, which uses an MPC-like cost function. However, the designed control system results in unstable behavior after about five seconds.

Besides, a well-known issue in utilizing RL in flight control systems arises from the fact that the learning process in RL relies on trial and error, which can simply make the air vehicle unstable. Thus, the learning process (in the current form) should be performed in a simulation environment, and subsequently, the trained policy is employed in a real application. However, the employment of a policy, which is trained in a simulation environment, in a real experiment suffers from the well-known *reality gap* problem. Different approaches have been proposed in the literature to overcome this issue [315]. The generalization of the policy through learning in different simulation environments with different flight conditions has been suggested in [296]. Further, the utilization of *abstracted*

inputs and outputs in the learning process would be an effective approach to tackle the reality gap [316]. In this regard, it may be a need for a mapping (or an intermediate controller) between the abstracted inputs-outputs and real signals in the control system. Besides, one can employ a dynamic model, which involved probabilistic uncertainties in the model, in order to evaluate and bound the worst-case controller performance in real applications [317].

A similar idea can also be beneficial to deal with the issue of the stability analysis in RL-based IFCSs. More specifically, a preliminary idea to analyze the closed-loop stability under the framework of RL could be maximizing the *expected* rewards at the neighborhood of an action sequence rather than that of a specific action sequence [318]. It can be a starting point to develop a *probabilistic stability analysis* framework in contrast to well-known approaches to stability analysis (using the Lyapunov theorem or similar methods) to be employed in the case of dynamic systems controlled by an RL-based scheme. To develop such a framework, we should also provide appropriate answers to principal questions regarding the quality and quantity of data samples required in the learning process.

In addition to the above-mentioned RL scheme based on MDP, there are other types of policy optimization algorithms that directly search for the optimal policy as a black-box optimization without employing the estimated action-value function into the optimization algorithm. Random search [319, 320], guided policy search [321], and evolutionary algorithms [322] are well-known approaches in this category, while due to the lack of a solid mathematical foundation, they are not widely employed in flight control systems yet. A guided policy search based on MPC has been introduced in [323] in which a set of trajectories are first generated at each step using Linear Quadratic Gaussian (LQG) controllers, where their objective is to maximize a quadratic reward by penalizing the deviation from the current policy. Subsequently, a modified MPC was designed in the vicinity of obtained trajectories, where the sampled data from trajectories, which were generated by MPC, are then employed to train the policy network in a supervised learning framework. However, there is a need for an approximate

dynamic model in the proposed design. Such an approach has been utilized in [323] to control a quadrotor trajectory in the presence of obstacles. While the MPC in the training phase requires access to full state observation, the final NN policy employs the data gathered by only the onboard sensors. Since in such a guided policy search, the control commands, in the training phase, are obtained using the MPC rather than a partially trained policy network, it is a beneficial approach to avoid a remarkable drawback of the RL, i.e the occurrence of catastrophic failures during the training. Accordingly, the training phase of RL can be performed safely in a real environment to avoid the reality gap. Another approach to achieve this goal could be the design of a training scenario using *gradually increasing control commands* to learn the optimal policy (in a safe environment) to avoid the systems' instability during the training. This is conceptually similar to teaching a child to walk by his/her parents (with no simulation environment!). Such an idea could be a starting point on the way to a truly *model-free* RL-based IFCS.

Finally, it is notable that the concept of adaptive optimal control can also be incorporated in the framework of the Stochastic Optimal Control (SOC) [324]. Since few studies have addressed the application of such a design in flight control systems, the mathematical details are not given here. Briefly, considering an affine dynamic system and a quadratic cost with respect to system inputs, it can be proved that the HJB equation for a stochastic model can be transformed into a linear PDE by defining a *desirability function* as an exponential value function. The solution of such a linear PDE, called the Cauchy problem, can be represented in a probabilistic manner by applying the Feynman-Kac formula, where the solution can be derived by an expectation over all possible system paths. Accordingly, the Monte-Carlo method involving the importance sampling technique is utilized to approximate it [325]. This problem can also be formulated within the framework of the *information theory* by incorporating the concepts of the *free energy* and the KL divergence, while there is no need for mentioned restrictions (such as an affine model) in such a formulation [326]. To this end, the optimal probability distribution of the control command is first

determined, where the control problem is then converted to the minimization of the KL divergence of the current probability distribution from the optimal distribution. The solution is typically determined iteratively at each step considering a finite prediction horizon in the cost (value) function. Such a method is also known as Model Predictive Path Integral (MPPI). Indeed, MPPI is a variety of MPCs in which a set of trajectories are generated at each step by adding noises to system inputs, and then, future control commands are improved by utilizing a Monte-Carlo sampling and computing the corresponding cost of each trajectory. The first control command in the computed sequence is then applied to the system and the remaining terms are used as the baseline in the next time step [327]. Such a method, which is somewhat similar to the guided policy search, results in a more efficient exploration rather than MPCs based on random trajectories [328]. Consequently, it can be an efficient alternative to conventional RL-based control systems, thereby providing the significant potential to be employed in IFCSs in the future. A vision-based MPPI has been given in [327] in which a deep NN was utilized to learn the optical flow of each pixel in the image, and then an MPC attempted to bring a target pixel to the center of the camera field of view while controlling the UAV path. Note that, as the MPC requires a prediction model, these methods are expressible within the framework of the model-based RL. An iterative learning control has been adopted within the introduced information-theoretic MPPI scheme in [326] and [328] for obstacle-avoidance control of a quadrotor trajectory and to provide a missile guidance law, respectively, where the system dynamics have been modeled by feedforward NNs. In this regard, as a key requirement, there is a need for a large number of samples in sampling-based MPCs, while the mathematical foundation for the analysis of the algorithm convergence and the closed-loop stability in the above-mentioned method should still be strengthened.

Various challenges remain in the way of efficient RL-based model-free control systems yet, and in some cases, a simple PID or LQR controller may behave more effectively than existing RL-based control approaches [286]. However, RL has provided a window into a new look at the control problem of complex systems

in complex environments, and it is expected that such a framework can lead to a generic, fully autonomous, truly model-free, and safe control methodology in the near future such that it can be reliably employed in the case of more complex aerial vehicles (such as nonconventional aircraft) and more complex problems (such as the presence of severe external disturbances and actuator faults).

Five tables are given in the following. Principal characteristics of some key research addressing the NN-based control of VTOL aerial vehicles, HFVs (and NSVs), fixed-wing aircraft, and nonconventional air vehicles are listed in Tables 1-4, respectively. Different specifications of each research, i.e., the control objective, the consideration of system constraints in the design, the use of an MLP technique, the employment of an OFB control scheme, and the type of uncertain dynamics considered in the model, as well as the main features and limitations arising from each control methodology are briefly reported. The provided data would be advantageous to identify considerable capabilities, complexities, and limitations of each control strategy for each type of aerial vehicle, to compare the importance and effectiveness of different control methods, and to figure out the challenging issues remaining unsolved. On the other hand, as a separate category, some novel *high-level* control systems incorporating NNs in their design are given in Table 5 in which the learning method, the control objective, key features, and considerable limitations of each research are listed. The combination of such high-level control strategies with existing low-level intelligent control systems would be a critical research area to develop an intelligent flight management unit.

Table 1: Principal characteristics of some of the introduced intelligent control systems for VTOL aerial vehicles

Ref.	Controller	Characteristics*	Main features	Limitations/ Complexities
[180]	Backstepping	TD	-Utilizing a switching function to integrate the NN and DO	-Neglecting the approximation error of differentiators

[168]	Neuroadaptive	TMD	-Considering aerodynamic frictions in the model -Considering unknown inertia matrix	-Avoiding attitude singularity problem using a BLF rather than employing the well-known quaternion formulation
[95]	Neuroadaptive	TD	-SMC-like-based training algorithm for FNNs	-The system should be decoupled into a set of SISO models -Applicable in second-order systems -Concerns with the stability analysis
[274]	ADP (H_2 control)	T	-Event-triggered control -Using discounted cost	-On-policy learning -The necessity for the PE condition -Several conservative assumptions in the stability analysis -Requires entire dynamic model
[10]	On-policy DPG	WR	-Performing a wide range of maneuvers, stably	-No stability analysis -Huge off-line computational burden
[177]	Backstepping	ADIO	-Adopting combined NN and DO -Using Nussbaum function to deal with input saturation -Using BLF to tackle output constraints	-Concerns with the stability analysis -Consdeiring a SISO model
[116]	Backstepping	AOMD	-DSC for multi-rotor UAV -Adopting combined NN and DO -Using a time-varying BLF	-Large control actions caused by BLF
[96]	Neuroadaptive	AR	-SMC-like-based training algorithm for FNNs -Generality of the control scheme	-The system should be decoupled into a set of SISO models -Chattering phenomenon -The plant must be stabilizable by a PID controller -Concerns with the stability analysis

[308]	Off-policy DDPG	WFR	-Hybrid offline-online learning algorithm -Adopting integrators to eliminate steady-state error	-No stability analysis -Significant off-line computational burden
[156]	Neuroadaptive	TR	-SMC-like-based training algorithm for FNNs	-The system should be decoupled into a set of SISO models -The plant must be stabilizable by a PD controller -Concerns with the stability analysis

* Control objective: A: Attitude control, W: waypoint tracking, T: Trajectory tracking, L: Longitudinal mode/ I: Consideration of input constraints, O: Consideration of output (state) constraints/ M: Minimal-learning parameter/ K: Output feedback control/ D: Disturbance or noise rejection, F: Fault-tolerant control, R: Model-free control

Table 2: Principal characteristics of some of the introduced intelligent control systems for HFVs and NSVs

Ref.	Controller	Characteristics	Main features	Limitations/ Complexities
[118]	Backstepping	ADI	-DSC with WNN-based DO -Using Nussbaum function to deal with input saturation	-Concerns with satiability analysis
[141]	Backstepping	ADI	-Adopting combined NN and DO -Using a modified tracking error to deal with input saturation -Control allocation using a convex optimization solved by an RNN	-Neglecting the control allocation error in the stability analysis
[174]	Backstepping	LI	-DSC with direct neural approximation -Using Nussbaum function to deal with dead-zone input nonlinearity	
[133]	Backstepping	LMO	-Funnel control to guarantee the transient performance -Consideration of flexible states	-Many design parameters -Availability of the third derivative of the tracking error

[127]	Backstepping	LMIO	-FOSMD in the backstepping design -Morphing aircraft (with pure-feedback model) -Using Butterworth filter to avoid algebraic loop in the control design	-Consideration of only the cruise phase -Assuming the bounded filtering error
[41]	Backstepping	L	-Using a discrete-time model -Utlizing a prediciton model	
[130]	Backstepping	L	-Direct neural-backstepping scheme -Using the integral of tracking error to eliminate the steady tracking error	
[132]	Neuroadaptive	LMK	-Defining an OFB model and using HGOs to avoid backstepping	-Large control commands at early times
[115]	Backstepping	LMF	-Avoiding singularity problem using direct DSC	-Considering only the bias actuator fault -Unusual formulation of NNs
[99]	Pseudocontrol	LM	-Avoiding the backstepping design through transforming the model into a normal feedback form -No requirement for the contraction assumption	-Time derivatives of FPA should be measurable
[83]	Backstepping	LFDO	-FOSMD in the backstepping design -Consideration of AOA constraint -Neural fault identification	-Using a SISO model
[147]	Backstepping	LMFDO	-Control of the transient response Asymptotic tracking control	-Parameter drift in the updating rules -Excessive control effort at the vicinity of permissible output bounds

Table 3: Principal characteristics of some of the introduced intelligent control systems for fixed-wing aircraft

Ref.	Controller	Characteristics	Main features	Limitations/ Complexities

[38]	Pseudocontrol	AF	-Hybrid direct-indirect adaptive control -Considering (a specific) structural damage	-No actuator dynamics -Slow convergence of the algorithm
[100]	Pseudocontrol	WF	-Modification of guidance commands to adapt to current flight condition	-No stability analysis
[277]	ADP (H_∞ control)	WD	-Off-policy learning -Partially model-free control -Employing single NN	-No stability analysis
[272]	ADP (H_∞ control)	WD	-Event-triggered control -Employing single NN	-On-policy learning -Requires entire dynamic model -The necessity for the PE condition -Several conservative assumptions in the stability analysis
[197]	Dynamic inversion	AF	-Indirect EKF-based fault identification	-Concerns with the stability analysis
[200]	MPC	TIFR	-Multimodel FTC scheme -Indirect RLS optimization-based fault identification	-Concerns with the feasibility of the proposed control design
[129]	Backstepping	AF	-Fractional-order backstepping control -Decentralized control of multi-UAVs -Adopting combined NN and DO	-Concerns with employing fractional-order control in practice -Conservative assumptions on estimation errors
[178]	Backstepping	TF	-DSC-based distributed formation flight control -Adopting combined NN and DO -Consideration of wake vortices	-Some simplifications in dynamic modeling

Table 4: Principal characteristics of some of the introduced intelligent control systems for nonconventional air vehicles

Ref.	Controller	Characteristics	Main features	Limitations/ Complexities

[160]	Neuroadaptive	TOD	-Flapping wing micro aerial vehicle control -Adopting combined NN and DO	
[293]	Deep Q-learning	TR	-Control of flapping-wing aerial vehicles using RL -Utilizing an evolutionary exploration -Maximizing expected reward near an action sequence to improve the robustness	-No stability analysis

Table 5: Principal characteristics of some of the introduced intelligent high-level control methods

Ref.	Method	Objective*	Main features	Limitations/ Complexities
[255]	Supervised learning	C	-Using simple images for training with no need for determining characteristic features of an object	-Lack of strong mathematical foundation -No stability analysis
[296]	Deep RL	C	-Real flight experiments -Using only monocular images as input	-No stability analysis
[297]	Deep RL	DE	-Training mobile charging stations to autonomously move to the charging point in an optimal manner	-Considering the 2D problem -No stability analysis
[312]	Modified DPG	CW	-Using the POMDP scheme -Navigating in lrage-scale complex environment	-No stability analysis
[309]	DDPG	W	-Auto-landing on a moving platform -Real flight experiments	-Considering 2D problem -No stability analysis
[323]	Guided policy search	CW	-MPC-based guided policy search -Providing a safer training phase using MPC in the training	-Requiring approximate dynamic model -No stability analysis
[326]	MPPI	CW	-Control of nonaffine dynamics -Utilizing information theoretic MPC with a generic cost function	-Requiring the dynamic model -No stability analysis

* Obstacle or Collision avoidance: C/ Data collection: D/ Waypoint tracking: W/ Consideration of total energy constraint: E

5. Concluding remarks and future directions

Intelligent flight control systems have been significantly evolved, particularly during the last two decades. They have been able to satisfactorily deal with different practical issues in a real flight, e.g. atmospheric disturbances, operational faults, model uncertainties, unmodeled dynamics, etc. In addition, concerning model-free control methods, there has been remarkable progress in both indirect adaptive controllers, which employ NNs to provide a valid dynamic model of the system, and direct adaptive control systems using the optimal control or the RL frameworks. Besides, recently, intelligent approaches, particularly those based on RL, have been effectively adopted in high-level control systems to provide intelligent path planning and guidance loops in flight control systems. Such remarkable progress of IFCSs results in introducing aerial robots with an outstandingly high level of autonomy. Despite all these advances, there still is a long way to go to introduce a *generic intelligent flight control system*. In the following, we address a set of crucial bottlenecks along with some suggestions for the direction of future research in developing such an intelligent flight control system.

1. *Design parameters*: The determination of appropriate design parameters in proposed IFCSs is a challenging issue, which is typically carried out by trial and error. Although the training of the controller's parameters (using an additional learning loop [329, 47] or evolutionary algorithms [207]) or the reduction of the design parameters (by incorporating self-organizing [228] or new data analysis approaches [223]) can deal with such a problem to a certain extent, the development of *generic* intelligent control systems with no (or at least very low) design parameters is still an open problem in the field of intelligent control. Thinking about more flexible control

structures organized by the incoming system data using machine learning approaches can be a gateway to efficient solutions.

2. *High-level control*: An intelligent guidance loop to ensure that a feasible trajectory is commanded to the aircraft is critical in developing a reliable flight control system in the presence of internal and external disturbances. However, this loop is typically remained unchanged after the fault occurrence in classical IFCSs [100]. Adaptive estimation of the flight envelope in the presence of operational faults would be the first step to provide a feasible FTC system [330]. In a more general view, the problem of the intelligent trajectory generation (for different purposes such as collision avoidance) is a challenging problem, which has received less attention from academia. Such a problem considering different design criteria, such as obstacle/collision avoidance, optimal resource allocation, etc., have been addressed in [296, 297, 312] using RL, while there are concerns about the definition of an appropriate reward function. In this respect, there is a significant need for the improvement and unification of such high-level control systems with conventional low-level IFCSs, consistently, to develop a fully autonomous flight management system. Further, by developing novel machine learning algorithms along with the development of the computing power, there will be an opportunity to redefine an entire flight control problem (which, in the existing framework, includes various control loops) as a new framework with a more integrated and concise structure that can map high-level commands to low-level inputs with less human intervention using novel machine learning methods.
3. *Evolutionary algorithms*: Although evolutionary algorithms are not currently a mainstream topic in aerial robotics, they may be an appropriate candidate in the near future to be adopted in NN-based flight control systems to enhance the effectiveness and efficiency of training algorithms, while reducing the computational complexity, and to learn the networks' architecture and learning hyperparameters [14, 316]. To this end, there would be a requirement for a new mathematical framework (maybe in a

probabilistic representation) to analyze the convergence of such learning approaches in order to develop a reliable control design procedure.

4. *Controllability region*: The provided stability analysis in almost all of the existing literature introduces a region of stability, where the boundaries of that region are determined by upper limits of a set of parameters, which do not have a physical meaning or are not measurable [60]. This is a serious problem in utilizing such adaptive controllers in a real application because we can not determine the controllability region of the system based on physical parameters. This problem becomes more challenging in model-free control systems. In this regard, there is a need for introducing a set of *tangible criteria* for analyzing closed-loop stability. More specifically, the borrowing of concepts from the information theory to analyze the controllability of the system according to different characteristics of incoming data would be an attractive idea to provide a beneficial stability analysis framework even for model-free control systems [326].
5. *Input-output constraints*: Simultaneous consideration of input and output constraints in the control design process is a challenging problem. This is due to the fact that the satisfaction of input constraints may lead to larger tracking errors, and conversely, the consideration of output constraints may necessitate impractical control commands. Integration of the funnel control with input saturation constraint has been addressed in [331] and [332] for a linear minimum phase and a SISO nonlinear system without considering model uncertainty in the control problem, while the problem becomes more challenging in the presence of uncertain dynamics. Reinforcement learning would be an effective candidate in such control problems. More precisely, using the RL, it is possible to learn an optimal policy as a mapping from permissible system inputs to desired outputs. Further, concerning *fault-tolerant* flight control systems, an iterative learning control scheme integrated by RL methods (such as MPPI [328]) would be an appropriate solution in future studies.
6. *Structural constraints*: Elastic modes of an air vehicle can result in a va-

riety of undesirable phenomena such as flutter and control reversal, which may affect the closed-loop performance. This can become more challenging in the case of damaged aircraft due to the uncertainty in structural margins and elastic modes shifting (caused by changes in the structural stiffness and mass of the airframe) [38]. The consideration of such structural constraints in the design process of IFCSs in future studies is a vital issue.

7. *RL computational complexity*: The considerable computational cost of RL is still a challenging issue, which is more problematic in the case of high dimensional problems [10]. In this regard, the development of integrated multi-loop control systems in which RL is employed in the outer control loop design, while existing classical intelligent controllers (discussed in Section 2) are utilized in the inner loop, would be an effective solution to reduce the dimension of the action space, thereby significantly reducing the learning complexity.
8. *Reward function in RL*: The definition of an appropriate reward function in RL-based control systems is of great importance, where there is still no well-known intelligent approach to define that. Although *reward shaping* is a well-known approach to speed up the learning process, there are significant concerns about the optimality of the computed policy and the convergence of the algorithm [311]. Inverse RL could be another solution to identify appropriate reward function using the *learning from demonstration* (i.e. the task of learning from an expert) [333]. Further, the concept of learning from demonstration (also known as *apprenticeship learning*) would be a useful approach to train an intelligent trajectory generation scheme [334].
9. *NNs' adaptation speed*: There are still considerable concerns about the adaptation speed of NNs in both the model-based and model-free control systems, particularly in the case of time-varying systems and environments with rapid changes [200, 195]. To deal with such an issue, there would be a requirement for more effective learning schemes with a faster convergence

rate, while not violating the robustness of the closed-loop system. On the other hand, the computation of the *best* adaptation rate in different flight conditions is another complicated problem with no clear answer [335]. Concerning the adaptive control of dynamic systems with parametric uncertainty, the above-mentioned problems have been addressed in the past two decades within the framework of L_1-adaptive control, which attempts to decouple the estimation loop from the control loop in order to decouple the adaptation from the robustness [336]. However, several issues have been reported in the literature regarding the claims made about the capabilities of L_1-adaptive control [37]. In this regard, there is a serious need to develop novel learning frameworks to be used in the case of dynamics systems subject to rapid changes in the dynamic model and the environment (considering both parametric and nonparametric uncertainties).

10. *NNs' structure in RL*: Typically, feedforward NNs are used as the actor and critic networks within the actor-critic framework (particularly, in RL [308]). RNNs can be an efficient alternative to feedforward NNs within such a framework to improve the closed-loop stability while reducing the bias and variance of the learning process.
11. *Time-dependent NNs*: There are more complex NNs in the literature, which can be adopted in the control design procedure to enhance the modeling and training performance. For instance, continuous-time RNNs [337] can be used to incorporate the time-varying sampling times [316]. Further, in spiking NNs, which are more close to biological NNs, each neuron has a *membrane potential* affected by incoming signals, where the neuron emits a spike once its membrane potential exceeds a specific threshold, and subsequently, the membrane potential is reset to a rest value. Although, due to the complex training process of such NNs, currently, they are mostly employed in relatively simple control problems [338, 339], spiking NNs can be a beneficial choice to imitate complex behavior of intelligent systems, thereby providing a significant potential to be used in complex intelligent controllers. Further, as these NNs encompass the concept of time in

their models, they could be appropriate solutions to deal with explicitly time-dependent model uncertainties and external disturbances.

12. *Complex NNs*: Different types of deep NNs, wavelet NNs, and CNNs have demonstrated their superior performance in the identification of complex nonlinear systems [254, 340]. Although several studies have addressed the development of direct (and rarely indirect) adaptive flight control systems consisting of such complicated NNs [341, 342, 99, 232], the effective employment of them in both the identification and control design steps, which may also require the development of more efficient training algorithms rather than existing ones, can impressively reduce the NNs' estimation error, which in turn, results in reducing the conservativeness of designed controllers, significantly.
13. *Aggressive maneuvers*: Most of the introduced trajectory tracking control schemes have been designed and validated under simple trajectories with no aggressive maneuvers [293]. In recent years, some researchers have addressed the development of autonomous aircraft aerobatics employing the concept of learning from demonstrations [234]. Such an approach can be effectively employed in the framework of IFCSs, particularly those based on RL to provide the ability to perform a wide range of maneuvers, and in the near future, IFCSs are expected to be able to fulfill more complex tasks, such as take-off, landing, and different aggressive maneuvers, with guaranteed performance.

Acknowledgements

This work was supported by Iran National Science Foundation (INSF) and Iran's National Elites Foundation (INEF) grant 98027065.

References

[1] Bruce T. Clough and Wright Patterson. Metrics, Schmetrics! How The Heck Do You Determine A UAV's Autonomy Anyway? In *Proceed-*

ings of the 2002 Performance Metrics for Intelligent Systems Workshop, Gaithersburg, MD, 2002.

[2] Linda S. Gottfredson. Mainstream Science on Intelligence: An Editorial With 52 Signatories, History, and Bibliography. *INTELLIGENCE*, 24(1):13–23, 1997.

[3] Lyle Long and Troy Kelley. The Requirements and Possibilities of Creating Conscious Systems. In *AIAA Infotech@ Aerospace Conference*, Seattle, Washington, 2009.

[4] Thaddeus Eze, Richard Anthony, Alan Soper, and Chris Walshaw. A Generic Approach towards Measuring Level of Autonomicity in Adaptive Systems. *International Journal on Advances in Intelligent Systems*, 5(3-4), 2012.

[5] Marialena Vagia, Aksel A. Transeth, and Sigurd A. Fjerdingen. A literature review on the levels of automation during the years. What are the different taxonomies that have been proposed? *Applied ergonomics*, 53 Pt A:190–202, 2016.

[6] Marco Protti and Riccardo Barzan. UAV Autonomy – Which level is desirable? – Which level is acceptable? Alenia Aeronautica Viewpoint. In *Platform Innovations and System Integration for Unmanned Air, Land and Sea Vehicles*, Neuilly-sur-Seine, France, 2007.

[7] Andriy Sarabakha, Changhong Fu, Erdal Kayacan, and Tufan Kumbasar. Type-2 Fuzzy Logic Controllers Made Even Simpler: From Design to Deployment for UAVs. *IEEE Transactions on Industrial Electronics*, 65(6):5069–5077, 2018.

[8] Kirk Y. W. Scheper, Sjoerd Tijmons, Cornelis C. de Visser, and Guido C. H. E. de Croon. Behavior Trees for Evolutionary Robotics. *Artificial life*, 22(1):23–48, 2016.

[9] G. C. H. E. de Croon, M. Perçin, B. D. W. Remes, C. de Wagter, and R. Ruijsink. *The DelFly: Design, aerodynamics, and artificial intelligence of a flapping wing robot / G.C.H.E. de Croon, M. Perçin, B.D.W. Remes, R. Ruijsink, C. de Wagter*. Springer, Berlin, 2016.

[10] Jemin Hwangbo, Inkyu Sa, Roland Siegwart, and Marco Hutter. Control of a Quadrotor With Reinforcement Learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103, 2017.

[11] Lu Cheng, ChangSheng Jiang, and Ming Pu. Online-SVR-compensated nonlinear generalized predictive control for hypersonic vehicles. *Science China Information Sciences*, 54(3):551–562, 2011.

[12] Eunjung Ju, Jungdam Won, Jehee Lee, Byungkuk Choi, Junyong Noh, and Min Gyu Choi. Data-driven control of flapping flight. *ACM Transactions on Graphics*, 32(5):1–12, 2013.

[13] Dario Floreano, Jean-Christophe Zufferey, and Jean-Daniel Nicoud. From wheels to wings with evolutionary spiking circuits. *Artificial life*, 11(1-2):121–138, 2005.

[14] Fernando Silva, Miguel Duarte, Luís Correia, Sancho Moura Oliveira, and Anders Lyhne Christensen. Open Issues in Evolutionary Robotics. *Evolutionary Computation*, 24(2):205–236, 2016.

[15] Fendy Santoso, Matthew A. Garratt, and Sreenatha G. Anavatti. State-of-the-Art Intelligent Flight Control Systems in Unmanned Aerial Vehicles. *IEEE Transactions on Automation Science and Engineering*, 15(2):613–627, 2018.

[16] C. C. Lee. Fuzzy logic in control systems: Fuzzy logic controller. I. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2):404–418, 1990.

[17] Giampiero Campa, Mario L. Fravolini, Brad Seanor, Marcello R. Napolitano, Diego Del Gobbo, Gu Yu, and Srikanth Gururajan. On-line learning neural networks for sensor validation for the flight control system of a

B777 research scale model. *International Journal of Robust and Nonlinear Control*, 12(11):987–1007, 2002.

[18] Janusz Kacprzyk, Johann Schumann, and Yan Liu, editors. *Applications of Neural Networks in High Assurance Systems*. Studies in Computational Intelligence. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[19] James E. Tomayko. *The Story of Self-Repairing Flight Control Systems*. NASA Dryden Flight Reasearch Center, 2003.

[20] M. Steinberg. Historical Overview of Research in Reconfigurable Flight Control. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 219(4):263–275, 2005.

[21] Johann Schumann, Pramod Gupta, and Yan Liu. Application of Neural Networks in High Assurance Systems: A Survey. In Janusz Kacprzyk, Johann Schumann, and Yan Liu, editors, *Applications of Neural Networks in High Assurance Systems*, volume 268 of *Studies in Computational Intelligence*, pages 1–19. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[22] Markus Kaminski. *A Rapid Robust Fault Detection Algorithm for Flight Control Reconfiguration*. Master Thesis, Germany, 2017.

[23] Peggy Williams. Selected Flight Test Results for Online Learning Neural Network-Based Flight Control System. In *AIAA 1st Intelligent Systems Technical Conference*, Chicago, Illinois, 2004. AIAA.

[24] Jacob Hageman, Mark Smith, and Susan Stachowiak. Integration of On-line Parameter Identification and Neural Network for In-Flight Adaptive Control. *NASA Dryden Flight Research Center*, 2003.

[25] Tim Smith, Jim Barhorst, and James M. Urnes. Design and Flight Test of an Intelligent Flight Control System. In Janusz Kacprzyk, Johann Schumann, and Yan Liu, editors, *Applications of Neural Networks in High Assurance Systems*, volume 268 of *Studies in Computational Intelligence*, pages 57–76. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[26] John T. Bosworth and Peggy S. Williams-Hayes. Flight Test Results from the NF-15B Intelligent Flight Control System (IFCS) Project with Adaptation to a Simulated Stabilator Failure. In *AIAA Conference and exhibit*, AIAA infotech@Aerospace Conference and Exhibit, Rohnert Park, California, 2007. AIAA.

[27] John Burken, Curt Hanson, Jim Lee, and John Kaneshige. Flight Test Comparison of Different Adaptive Augmentations of Fault Tolerant Control Laws for a Modified F-15 Aircraft. In *AIAA Infotech@Aerospace Conference*, Seattle, Washington, 2009.

[28] Curt Hanson, Jacob Schaefer, Marcus Johnson, and Nhan Nguyen. Design of Low Complexity Model Reference Adaptive Controllers. *NASA Dryden Flight Research Center*, 2012.

[29] Curt Hanson, Jacob Schaefer, John J. Burken, David Larson, and Marcus Johnson. Complexity and Pilot Workload Metrics for the Evaluation of Adaptive Flight Controls on a Full Scale Piloted Aircraft. *NASA Dryden Flight Research Center*, 2014.

[30] Konstantinos Dalamagkidis, Kimon P. Valavanis, and Les A. Piegl. Nonlinear Model Predictive Control With Neural Network Optimization for Autonomous Autorotation of Small Unmanned Helicopters. *IEEE Transactions on Control Systems Technology*, 19(4):818–831, 2011.

[31] Zhijun Li, Jun Deng, Renquan Lu, Yong Xu, Jianjun Bai, and Chun-Yi Su. Trajectory-Tracking Control of Mobile Robot Systems Incorporating Neural-Dynamic Optimized Model Predictive Approach. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(6):740–749, 2016.

[32] Mehmet Önder Efe. Neural Network Assisted Computationally Simple PID Control of a Quadrotor UAV. *IEEE Transactions on Industrial Informatics*, 7(2):354–361, 2011.

[33] Bin Xu, Chenguang Yang, and Yongping Pan. Global neural dynamic surface tracking control of strict-feedback systems with application to hypersonic flight vehicle. *IEEE Transactions on Neural Networks and Learning Systems*, 26(10):2563–2575, 2015.

[34] Bin Xu, Daipeng Yang, Zhongke Shi, Yongping Pan, Badong Chen, and Fuchun Sun. Online Recorded Data-Based Composite Neural Control of Strict-Feedback Systems With Application to Hypersonic Flight Dynamics. *IEEE Transactions on Neural Networks and Learning Systems*, 29(8):3839–3849, 2018.

[35] Karl J. Åström and Björn Wittenmark. *Adaptive control*. Dover books on engineering. Dover Publications, Mineola N.Y., 2nd ed., dover ed. edition, 2008.

[36] Weibin Gu, Kimon P. Valavanis, Matthew J. Rutherford, and Alessandro Rizzo. A Survey of Artificial Neural Networks with Model-based Control Techniques for Flight Control of Unmanned Aerial Vehicles. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 362–371, Atlanta, GA, USA, 2019. IEEE.

[37] P. A. Ioannou, A. M. Annaswamy, K. S. Narendra, S. Jafari, L. Rudd, R. Ortega, and J. Boskovic. L1-Adaptive Control: Stability, Robustness, and Interpretations. *IEEE Transactions on Automatic Control*, 59(11):3075–3080, 2014.

[38] Nhan Nguyen, Kalmanje Krishnakumar, John Kaneshige, and Pascal Nespeca. Flight dynamics and hybrid adaptive control of damaged aircraft. *Journal of Guidance, Control, and Dynamics*, 31(3):751–764, 2008.

[39] A. J. Calise. Neural networks in nonlinear aircraft flight control. *IEEE Aerospace and Electronic Systems Magazine*, 11(7):5–10, 1996.

[40] Martin T. Hagan, Howard B. Demuth, Mark Hudson Beale, and Orlando de Jesús. *Neural network design*. 2nd ed. edition, 2016.

[41] Bin Xu and Yu Zhang. Neural discrete back-stepping control of hypersonic flight vehicle with equivalent prediction model. *Neurocomputing*, 154:337–346, 2015.

[42] Bin Xu, Yongping Pan, Danwei Wang, and Fuchun Sun. Discrete-time hypersonic flight control based on extreme learning machine. *Neurocomputing*, 128:232–241, 2014.

[43] F. Rinaldi, S. Chiesa, and F. Quagliotti. Linear Quadratic Control for Quadrotors UAVs Dynamics and Formation Flight. *Journal of Intelligent & Robotic Systems*, 70(1-4):203–220, 2013.

[44] Valeria Artale, Mario Collotta, Cristina Milazzo, Giovanni Pau, and Angela Ricciardello. An Integrated System for UAV Control Using a Neural Network Implemented in a Prototyping Board. *Journal of Intelligent and Robotic Systems*, 2016.

[45] Jih-Gau Juang and Kai-Chung Cheng. Application of Neural Networks to Disturbances Encountered Landing Control. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):582–588, 2006.

[46] Sefer Kurnaz, Omer Cetin, and Okyay Kaynak. Adaptive neuro-fuzzy inference system based autonomous flight control of unmanned air vehicles. *Expert Systems with Applications*, 37(2):1229–1234, 2010.

[47] Seyyed Ali Emami and Alireza Roudbari. Multimodel ELM-Based Identification of an Aircraft Dynamics in the Entire Flight Envelope. *IEEE Transactions on Aerospace and Electronic Systems*, 55(5):2181–2194, 2019.

[48] Erdal Kayacan. *Fuzzy neural networks for real time control applications: Concepts, modeling and algorithms for fast learning.* Elsevier and Butterworth-Heinemann, Amsterdam and Oxford UK, 2016.

[49] P. Baldi, P. Castaldi, N. Mimmo, and S. Simani. Satellite attitude active FTC based on Geometric Approach and RBF Neural Network. In *2013*

Conference on Control and Fault-Tolerant Systems (SysTol), pages 667–673, Nice, France, 2013. IEEE.

[50] P. Baldi, M. Blanke, P. Castaldi, N. Mimmo, and S. Simani. Combined Geometric and Neural Network Approach to Generic Fault Diagnosis in Satellite Actuators and Sensors. *IFAC-PapersOnLine*, 49(17):432–437, 2016.

[51] M. Verhaegen, S. Kanev, R. Hallouzi, C. Jones, J. Maciejowski, and H. Smail. Fault Tolerant Flight Control - A Survey. In Christopher Edwards, Thomas Lombaerts, and Hafid Smaili, editors, *Fault tolerant flight control*, Lecture notes in control and information sciences, 0170-8643. Springer, Berlin, 2010.

[52] Weibin Gu, Kimon P. Valavanis, Matthew J. Rutherford, and Alessandro Rizzo. UAV Model-based Flight Control with Artificial Neural Networks: A Survey. *Journal of Intelligent and Robotic Systems*, 100(3-4):1469–1491, 2020.

[53] Mohd Ariffanan Mohd Basri, Abdul Rashid Husain, and Kumeresan A. Danapalasingam. Intelligent adaptive backstepping control for MIMO uncertain non-linear quadrotor helicopter systems. *Transactions of the Institute of Measurement and Control*, 37(3):345–361, 2015.

[54] Anthony J. Calise, Naira Hovakimyan, and Moshe Idan. Adaptive output feedback control of nonlinear systems using neural networks. *Automatica*, 37(8):1201–1211, 2001.

[55] Nakwan Kim. *Improved methods in neural network based adaptive output feedback control, with applications to flight control.* PhD Thesis, School of Aerospace Engineering, 2003.

[56] Girish Chowdhary, Maximilian Mühlegg, and Eric Johnson. Exponential parameter and tracking error convergence guarantees for adaptive con-

trollers without persistency of excitation. *International Journal of Control*, 87(8):1583–1603, 2014.

[57] Taeyoung Lee and Youdan Kim. Nonlinear Adaptive Flight Control Using Backstepping and Neural Networks Controller. *Journal of Guidance, Control, and Dynamics*, 24(4):675–682, 2001.

[58] Shushuai Li, Yaonan Wang, Jianhao Tan, and Yan Zheng. Adaptive RBFNNs/integral sliding mode control for a quadrotor aircraft. *Neurocomputing*, 216:126–134, 2016.

[59] Samir Zeghlache, Hemza Mekki, Abderrahmen Bouguerra, and Ali Djerioui. Actuator fault tolerant control using adaptive RBFNN fuzzy sliding mode controller for coaxial octorotor UAV. *ISA Transactions*, 80:267–278, 2018.

[60] Yongduan Song, Liu He, Dong Zhang, Jiye Qian, and Jin Fu. Neuroadaptive fault-tolerant control of quadrotor UAVs: a more affordable solution. *IEEE Transactions on Neural Networks and Learning Systems*, 30(7):1975–1983, 2019.

[61] Dong-Ho Shin and Youdan Kim. Nonlinear discrete-time reconfigurable flight control law using neural networks. *IEEE Transactions on Control Systems Technology*, 14(3):408–422, 2006.

[62] T. Zhang, S. S. Ge, and C. C. Hang. Design and performance analysis of a direct adaptive controller for nonlinear systems. *Automatica*, 35(11):1809–1817, 1999.

[63] S. S. Ge and Cong Wang. Direct adaptive NN control of a class of nonlinear systems. *IEEE Transactions on Neural Networks*, 13(1):214–221, 2002.

[64] M. Vijaya Kumar, S. Suresh, S. N. Omkar, Ranjan Ganguli, and Prasad Sampath. A direct adaptive neural command controller design for an unstable helicopter. *Engineering Applications of Artificial Intelligence*, 22(2):181–191, 2009.

[65] E. Tzirkel-Hancock and F. Fallside. Stable control of nonlinear systems using neural networks. *International Journal of Robust and Nonlinear Control*, 2(1):63–86, 1992.

[66] S. Fabri and V. Kadirkamanathan. Dynamic structure neural networks for stable adaptive control of nonlinear systems. *IEEE Transactions on Neural Networks*, 7(5):1151–1167, 1996.

[67] Jovan D. Boskovic, Lingji Chen, and Raman K. Mehra. Adaptive Control Design for Nonaffine Models Arising in Flight Control. *Journal of Guidance, Control, and Dynamics*, 27(2):209–217, 2004.

[68] Petros A. Ioannou and Petar V. Kokotovic, editors. *Adaptive Systems with Reduced Models*, volume 47 of *Lecture Notes in Control and Information Sciences*. Springer, Berlin and Heidelberg, 1983.

[69] P. A. Ioannou and Jing Sun. *Robust adaptive control*. Dover Publications Inc, Mineola New York, 2012.

[70] D.-H. Shin and Y. Kim. Reconfigurable Flight Control System Design Using Adaptive Neural Networks. *IEEE Transactions on Control Systems Technology*, 12(1):87–100, 2004.

[71] Naira Hovakimyan, Nakwan Kim, Anthony Calise, and J.V.R. Prasad. Adaptive Output Feedback for High-Bandwidth Control of an Unmanned Helicopter. In *AIAA Guidance, Navigation, and Control Conference*, Montreal, Canada, 2001. American Institute of Aeronautics and Astronautics.

[72] N. Hovakimyan, F. Nardi, A. Calise, and Nakwan Kim. Adaptive output feedback control of uncertain nonlinear systems using single-hidden-layer neural networks. *IEEE Transactions on Neural Networks*, 13(6):1420–1431, 2002.

[73] S. S. Ge, B. Ren, Keng Peng Tee, and T. H. Lee. Approximation-based control of uncertain helicopter dynamics. *IET Control Theory & Applications*, 3(7):941–956, 2009.

[74] Kumpati S. Narendra and Anuradha M. Annaswamy. A New Adaptive Law for Robust Adaptation without Persistent Excitation. *IEEE Transactions on Automatic Control*, pages 1067–1072, 1987.

[75] R. Rysdyk and A. J. Calise. Robust nonlinear adaptive flight control for consistent handling qualities. *IEEE Transactions on Control Systems Technology*, 13(6):896–910, 2005.

[76] R. Rysdyk and A. Calise. Fault tolerant flight control via adaptive neural network augmentation. In *Guidance, Navigation, and Control Conference and Exhibit*, Guidance, Navigation, and Control Conference and Exhibit, Boston, USA, 1998.

[77] C.J.B. Macnab. Robust Associative-Memory Adaptive Control in the Presence of Persistent Oscillations. *Neural Information Processing*, 10(12):277–287, 2006.

[78] C. Nicol, C.J.B. Macnab, and A. Ramirez-Serranob. Robust neural network control of a quadrotor helicopter. In *Canadian Conference on Electrical and Computer Engineering*, Canadian Conference on Electrical and Computer Engineering, Niagara Falls, Canada, 2008.

[79] C. Coza, C. Nicol, C.J.B. Macnab, and A. Ramirez-Serrano. Adaptive fuzzy control for a quadrotor helicopter robust to wind buffeting. *Journal of Intelligent & Fuzzy Systems*, 22(5,6):267–283, 2011.

[80] Bin Xu, Zhongke Shi, Chenguang Yang, and Fuchun Sun. Composite Neural Dynamic Surface Control of a Class of Uncertain Nonlinear Systems in Strict-Feedback Form. *IEEE Transactions on Cybernetics*, 44(12):2626–2634, 2014.

[81] Bin Xu, Danwei Wang, Youmin Zhang, and Zhongke Shi. DOB-Based Neural Control of Flexible Hypersonic Flight Vehicle Considering Wind Effects. *IEEE Transactions on Industrial Electronics*, 64(11):8676–8685, 2017.

[82] Bin Xu, Xia Wang, and Zhongke Shi. Robust Adaptive Neural Control of Nonminimum Phase Hypersonic Vehicle Model. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(2):1107–1115, 2021.

[83] Bin Xu, Zhongke Shi, Fuchun Sun, and Wei He. Barrier Lyapunov Function Based Learning Control of Hypersonic Flight Vehicle With AOA Constraint and Actuator Faults. *IEEE Transactions on Cybernetics*, 49(3):1047–1057, 2019.

[84] Girish V. Chowdhary and Eric N. Johnson. Theory and Flight-Test Validation of a Concurrent-Learning Adaptive Controller. *Journal of Guidance, Control, and Dynamics*, 34(2):592–607, 2011.

[85] Chuan-Kai Lin. Robust adaptive critic control of nonlinear systems using fuzzy basis function networks: An LMI approach. *Information Sciences*, 177(22):4934–4946, 2007.

[86] Chuan-Kai Lin. H∞ reinforcement learning control of robot manipulators using fuzzy wavelet networks. *Fuzzy Sets and Systems*, 160(12):1765–1786, 2009.

[87] Xiangwei Bu, Yu Xiao, and Humin Lei. An Adaptive Critic Design-Based Fuzzy Neural Controller for Hypersonic Vehicles: Predefined Behavioral Nonaffine Control. *IEEE/ASME Transactions on Mechatronics*, 24(4):1871–1881, 2019.

[88] Yanhong Luo, Qiuye Sun, Huaguang Zhang, and Lili Cui. Adaptive critic design-based robust neural network control for nonlinear distributed parameter systems with unknown dynamics. *Neurocomputing*, 148(2):200–208, 2015.

[89] Haojian Xu, Maj Mirmirani, and Petros A. Ioannou. Robust Neural Adaptive Control of a Hypersonic Aircraft. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Austin, Texas, 2003.

[90] Hiroaki Gomi and Mitsuo Kawato. Neural network control for a closed-loop System using Feedback-error-learning. *Neural Networks*, 6(7):933–946, 1993.

[91] Yan Li, N. Sundararajan, P. Saratchandran, and Zhifeng Wang. Robust neuro-H∞ controller design for aircraft auto-landing. *IEEE Transactions on Aerospace and Electronic Systems*, 40(1):158–167, 2004.

[92] A. A. Pashilkar, N. Sundararajan, and P. Saratchandran. A fault-tolerant neural aided controller for aircraft auto-landing. *Aerospace Science and Technology*, 10(1):49–61, 2006.

[93] Y. Li, N. Sundararajan, and P. Saratchandran. Neuro-controller design for nonlinear fighter aircraft maneuver using fully tuned RBF networks. *Automatica*, 37(8):1293–1301, 2001.

[94] Mojtaba Ahmadieh Khanesar, Erdal Kayacan, and Okyay Kaynak. Optimal sliding mode type-2 TSK fuzzy control of a 2-DOF helicopter. In *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6, Istanbul, Turkey, 2015. IEEE.

[95] Erdal Kayacan and Reinaldo Maslim. Type-2 Fuzzy Logic Trajectory Tracking Control of Quadrotor VTOL Aircraft With Elliptic Membership Functions. *IEEE/ASME Transactions on Mechatronics*, 22(1):339–348, 2017.

[96] Md Meftahul Ferdaus, Mahardhika Pratama, Sreenatha G. Anavatti, Matthew A. Garratt, and Yongping Pan. Generic Evolving Self-Organizing Neuro-Fuzzy Control of Bio-Inspired Unmanned Aerial Vehicles. *IEEE Transactions on Fuzzy Systems*, 28(8):1542–1556, 2020.

[97] Md Meftahul Ferdaus, Mahardhika Pratama, Sreenatha G. Anavatti, Matthew A. Garratt, and Edwin Lughofer. PAC: A novel self-adaptive neuro-fuzzy controller for micro aerial vehicles. *Information Sciences*, 512(4):481–505, 2020.

[98] Byoung S. Kim and Anthony J. Calise. Nonlinear Flight Control Using Neural Networks. *Journal of Guidance, Control, and Dynamics*, 20(1):26–33, 1997.

[99] Xiangwei Bu and Humin Lei. A fuzzy wavelet neural network-based approach to hypersonic flight vehicle direct nonaffine hybrid control. *Nonlinear Dynamics*, 94(3):1657–1668, 2018.

[100] Girish Chowdhary, Eric N. Johnson, Rajeev Chandramohan, M. Scott Kimbrell, and Anthony Calise. Guidance and Control of Airplanes Under Actuator Failures and Severe Structural Damage. *Journal of Guidance, Control, and Dynamics*, 36(4):1093–1104, 2013.

[101] S. Lee, C. Ha, and B. S. Kim. Adaptive nonlinear control system design for helicopter robust command augmentation. *Aerospace Science and Technology*, 9(3):241–251, 2005.

[102] A. Rahideh, A. H. Bajodah, and M. H. Shaheed. Real time adaptive nonlinear model inversion control of a twin rotor MIMO system using neural networks. *Engineering Applications of Artificial Intelligence*, 25(6):1289–1297, 2012.

[103] Anthony Calise. Development of a reconfigurable flight control law for the X-36 tailless fighter aircraft. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Dever,CO,U.S.A, 2000. American Institute of Aeronautics and Astronautics.

[104] Joseph Brinker and Kevin Wise. Flight testing of a reconfigurable flight control law on the X-36 tailless fighter aircraft. In *AIAA Guidance,*

Navigation, and Control Conference and Exhibit, Dever,CO,U.S.A, 2000. American Institute of Aeronautics and Astronautics.

[105] Anthony J. Calise, Seungjae Lee, and Manu Sharma. Development of a Reconfigurable Flight Control Law for Tailless Aircraft. *Journal of Guidance, Control, and Dynamics*, 24(5):896–902, 2001.

[106] Joseph S. Brinker and Kevin A. Wise. Flight Testing of Reconfigurable Control Law on the X-36 Tailless Aircraft. *Journal of Guidance, Control, and Dynamics*, 24(5):903–909, 2001.

[107] Nhan Nguyen, Kalmanje Krishnakumar, John Kaneshige, and Pascal Nespeca. Dynamics and adaptive control for stability recovery of damaged asymmetric aircraft. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, Colorado, 2006. American Institute of Aeronautics and Astronautics.

[108] Keng Peng Tee, Shuzhi Sam Ge, and F.E.H. Tay. Adaptive Neural Network Control for Helicopters in Vertical Flight. *IEEE Transactions on Control Systems Technology*, 16(4):753–762, 2008.

[109] DaoXiang Gao, ShiXing Wang, and Houjiang Zhang. A Singularly Perturbed System Approach to Adaptive Neural Back-stepping Control Design of Hypersonic Vehicles. *Journal of Intelligent & Robotic Systems*, 73(1-4):249–259, 2014.

[110] S. S. Ge, B. Ren, and M. Chen. Robust attitude control of helicopters with actuator dynamics using neural networks. *IET Control Theory & Applications*, 4(12):2837–2854, 2010.

[111] Xiaolong Zheng and Xuebo Yang. Improved adaptive NN backstepping control design for a perturbed PVTOL aircraft. *Neurocomputing*, 410(7):51–60, 2020.

[112] D. Swaroop, J. K. Hedrick, P. P. Yip, and J. C. Gerdes. Dynamic surface control for a class of nonlinear systems. *IEEE Transactions on Automatic Control*, 45(10):1893–1899, 2000.

[113] Waseem Aslam Butt, Lin Yan, and Amezquita S. Kendrick. Adaptive dynamic surface control of a hypersonic flight vehicle with improved tracking. *Asian Journal of Control*, 15(2):594–605, 2013.

[114] Qun Zong, Fang Wang, Bailing Tian, and Rui Su. Robust adaptive dynamic surface control design for a flexible air-breathing hypersonic vehicle with input constraints and uncertainty. *Nonlinear Dynamics*, 78(1):289–315, 2014.

[115] Bin Xu, Qi Zhang, and Yongping Pan. Neural network based dynamic surface control of hypersonic flight dynamics using small-gain theorem. *Neurocomputing*, 173:690–699, 2016.

[116] Chunyang Fu, Wei Hong, Huiqiu Lu, Lei Zhang, Xiaojun Guo, and Yantao Tian. Adaptive robust backstepping attitude control for a multi-rotor unmanned aerial vehicle with time-varying output constraints. *Aerospace Science and Technology*, 78:593–603, 2018.

[117] Waseem Aslam Butt, Lin Yan, and Kendrick Amezquita S. Adaptive integral dynamic surface control of a hypersonic flight vehicle. *International Journal of Systems Science*, 46(10):1717–1728, 2013.

[118] Mou Chen, Yanlong Zhou, and William W. Guo. Robust tracking control for uncertain MIMO nonlinear systems with input saturation using RWNNDO. *Neurocomputing*, 144(20):436–447, 2014.

[119] Li Zhou and Liping Yin. Dynamic surface control based on neural network for an air-breathing hypersonic vehicle. *Optimal Control Applications and Methods*, 36(6):774–793, 2015.

[120] Bin Xu, Xia Wang, Weisheng Chen, and Peng Shi. Robust Intelligent Control of SISO Nonlinear Systems Using Switching Mechanism. *IEEE Transactions on Cybernetics*, 51(8):3975–3987, 2021.

[121] J. A. Farrell, M. Polycarpou, M. Sharma, and Wenjie Dong. Command Filtered Backstepping. *IEEE Transactions on Automatic Control*, 54(6):1391–1395, 2009.

[122] Wenjie Dong, J. A. Farrell, M. M. Polycarpou, V. Djapic, and M. Sharma. Command Filtered Adaptive Backstepping. *IEEE Transactions on Control Systems Technology*, 20(3):566–580, 2012.

[123] Bin Xu, Yuyan Guo, Yuan Yuan, Yonghua Fan, and Danwei Wang. Fault-tolerant control using command-filtered adaptive back-stepping technique: Application to hypersonic longitudinal flight dynamics. *International Journal of Adaptive Control and Signal Processing*, 30(4):553–577, 2016.

[124] Lars Sonneveldt, Q. P. Chu, and J. A. Mulder. Nonlinear Flight Control Design Using Constrained Adaptive Backstepping. *Journal of Guidance, Control, and Dynamics*, 30(2):322–336, 2007.

[125] L. Sonneveldt, E. R. van Oort, Q. P. Chu, and J. A. Mulder. Nonlinear adaptive trajectory control applied to an F-16 model. *Journal of Guidance, Control, and Dynamics*, 32(1):25–39, 2009.

[126] Arie Levant. Robust exact differentiation via sliding mode technique. *Automatica*, 34(3):379–384, 1998.

[127] Zhonghua Wu, Jingchao Lu, Qing Zhou, and Jingping Shi. Modified adaptive neural dynamic surface control for morphing aircraft with input and output constraints. *Nonlinear Dynamics*, 87(4):2367–2383, 2017.

[128] Arie Levant. Higher-order sliding modes, differentiation and output-feedback control. *International Journal of Control*, 76(9-10):924–941, 2003.

[129] Ziquan Yu, Youmin Zhang, Bin Jiang, Chun-Yi Su, Jun Fu, Ying Jin, and Tianyou Chai. Decentralized fractional-order backstepping fault-tolerant control of multi-UAVs against actuator faults and wind effects. *Aerospace Science and Technology*, 104(6):105939, 2020.

[130] Xiangwei Bu, Xiaoyan Wu, Zhen Ma, and Rui Zhang. Nonsingular direct neural control of air-breathing hypersonic vehicle via back-stepping. *Neurocomputing*, 153:164–173, 2015.

[131] Bin Xu, DaoXiang Gao, and ShiXing Wang. Adaptive neural control based on HGO for hypersonic flight vehicles. *Science China Information Sciences*, 54(3):511–520, 2011.

[132] Bin Xu, Yonghua Fan, and Shangmin Zhang. Minimal-learning-parameter technique based adaptive neural control of hypersonic flight dynamics without back-stepping. *Neurocomputing*, 164:201–209, 2015.

[133] Xiangwei Bu. Air-Breathing Hypersonic Vehicles Funnel Control Using Neural Approximation of Non-affine Dynamics. *IEEE/ASME Transactions on Mechatronics*, 23(5):2099–2108, 2018.

[134] Bin Xu, Fuchun Sun, Chenguang Yang, DaoXiang Gao, and Jianxin Ren. Adaptive discrete-time controller design with neural network for hypersonic flight vehicle via back-stepping. *International Journal of Control*, 84(9):1543–1552, 2011.

[135] Bin Xu, Danwei Wang, Fuchun Sun, and Zhongke Shi. Direct neural discrete control of hypersonic flight vehicle. *Nonlinear Dynamics*, 70(1):269–278, 2012.

[136] Bin Xu, Zhongke Shi, Chenguang Yang, and ShiXing Wang. Neural control of hypersonic flight vehicle model via time-scale decomposition with throttle setting constraint. *Nonlinear Dynamics*, 73(3):1849–1861, 2013.

[137] Bin Xu, Danwei Wang, Fuchun Sun, and Zhongke Shi. Direct neural control of hypersonic flight vehicles with prediction model in discrete time. *Neurocomputing*, 115:39–48, 2013.

[138] Zongyu Zuo and Chenliang Wang. Adaptive trajectory tracking control of output constrained multi-rotors systems. *IET Control Theory & Applications*, 8(13):1163–1174, 2014.

[139] Bin Xian, Chen Diao, Bo Zhao, and Yao Zhang. Nonlinear robust output feedback tracking control of a quadrotor UAV using quaternion representation. *Nonlinear Dynamics*, 79(4):2735–2752, 2015.

[140] Mou Chen, Peng Shi, and Cheng-Chew Lim. Adaptive neural fault-tolerant control of a 3-DOF model helicopter system. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(2):260–270, 2016.

[141] Mou Chen. Constrained Control Allocation for Overactuated Aircraft Using a Neurodynamic Model. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2015.

[142] M. Krstić, I. Kanellakopoulos, and P. V. Kokotović. Adaptive nonlinear control without overparametrization. *Systems & Control Letters*, 19(3):177–185, 1992.

[143] Peng Shi, Cheng Chew Lim, Bin Jiang, and Dezhi Xu. Adaptive neural observer-based backstepping fault tolerant control for near space vehicle under control effector damage. *IET Control Theory & Applications*, 8(9):658–666, 2014.

[144] Parag M. Patre, William MacKunis, Kent Kaiser, and Warren E. Dixon. Asymptotic Tracking for Uncertain Dynamic Systems Via a Multilayer Neural Network Feedforward and RISE Feedback Control Structure. *IEEE Transactions on Automatic Control*, 53(9):2180–2185, 2008.

[145] Hadi Razmi and Sima Afshinfar. Neural network-based adaptive sliding mode control design for position and attitude control of a quadrotor UAV. *Aerospace Science and Technology*, 91(3):12–27, 2019.

[146] Jongho Shin, H. J. Kim, Youdan Kim, and Warren E. Dixon. Autonomous Flight of the Rotorcraft-Based UAV Using RISE Feedback and NN Feedforward Terms. 20(5):1392–1399, 2012.

[147] Yajun Li, Mingshan Hou, Shuai Liang, and Gang Jiao. Predefined-time adaptive fault-tolerant control of hypersonic flight vehicles without overparameterization. *Aerospace Science and Technology*, 104:105987, 2020.

[148] Hongwei Mo and Ghulam Farid. Nonlinear and Adaptive Intelligent Control Techniques for Quadrotor UAV – A Survey. *Asian Journal of Control*, 21(2):989–1008, 2018.

[149] Zhijun Zhang, Lunan Zheng, and Qi Guo. A Varying-Parameter Convergent Neural Dynamic Controller of Multirotor UAVs for Tracking Time-Varying Tasks. *IEEE Transactions on Vehicular Technology*, 67(6):4793–4805, 2018.

[150] Zheng Zhu, Yuanqing Xia, and Mengyin Fu. Attitude stabilization of rigid spacecraft with finite-time convergence. *International Journal of Robust and Nonlinear Control*, 21(6):686–702, 2011.

[151] Bin Xu. Composite Learning Finite-Time Control With Application to Quadrotors. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(10):1806–1815, 2018.

[152] Xiang Yu, Yu Fu, Peng Li, and Youmin Zhang. Fault-Tolerant Aircraft Control Based on Self-Constructing Fuzzy Neural Networks and Multivariable SMC Under Actuator Faults. *IEEE Transactions on Fuzzy Systems*, 26(4):2324–2335, 2018.

[153] Dandan Wang, Qun Zong, Bailing Tian, Shikai Shao, Xiuyun Zhang, and Xinyi Zhao. Neural network disturbance observer-based distributed finite-

time formation tracking control for multiple unmanned helicopters. *ISA Transactions*, 73:208–226, 2018.

[154] Dandan Wang, Qun Zong, Bailing Tian, Hanchen Lu, and Jie Wang. Adaptive finite-time reconfiguration control of unmanned aerial vehicles with a moving leader. *Nonlinear Dynamics*, 95(2):1099–1116, 2019.

[155] E. Kayacan, O. Cigdem, and O. Kaynak. Sliding Mode Control Approach for Online Learning as Applied to Type-2 Fuzzy Neural Networks and Its Experimental Evaluation. *IEEE Transactions on Industrial Electronics*, 59(9):3510–3520, 2012.

[156] Efe Camci, Devesh Raju Kripalani, Linlu Ma, Erdal Kayacan, and Mojtaba Ahmadieh Khanesar. An aerial robot for rice farm quality inspection with type-2 fuzzy neural networks tuned by particle swarm optimization-sliding mode control hybrid algorithm. *Swarm and Evolutionary Computation*, 41:1–8, 2018.

[157] S. Seshagiri and H. K. Khalil. Output feedback control of nonlinear systems using RBF neural networks. *IEEE Transactions on Neural Networks*, 11(1):69–79, 2000.

[158] Hassan K. Khalil. *Nonlinear systems*. Prentice Hall and London : Pearson Education, Upper Saddle River, N.J., 3rd ed. edition, 2002.

[159] Jongho Shin, H. Jin Kim, and Youdan Kim. Adaptive support vector regression for UAV flight control. *Neural networks : the official journal of the International Neural Network Society*, 24(1):109–120, 2011.

[160] Wei He, Zichen Yan, Changyin Sun, and Yunan Chen. Adaptive Neural Network Control of a Flapping Wing Micro Aerial Vehicle With Disturbance Observer. *IEEE Transactions on Cybernetics*, 47(10):3452–3465, 2017.

[161] SAMAN BEHTASH. Robust output tracking for non-linear systems. *International Journal of Control*, 51(6):1381–1407, 2007.

[162] Travis Dierks and Sarangapani Jagannathan. Output feedback control of a quadrotor UAV using neural networks. *IEEE Transactions on Neural Networks*, 21(1):50–66, 2010.

[163] Yansheng Yang, Changjiu Zhou, and Jusheng Ren. Model reference adaptive robust fuzzy control for ship steering autopilot with uncertain nonlinear systems. *Applied Soft Computing*, 3(4):305–316, 2003.

[164] Y. Yang, G. Feng, and J. Ren. A Combined Backstepping and Small-Gain Approach to Robust Adaptive Fuzzy Control for Strict-Feedback Nonlinear Systems. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 34(3):406–420, 2004.

[165] Yansheng Yang, Tieshan Li, and Xiaofeng Wang. Robust Adaptive Neural Network Control for Strict-Feedback Nonlinear Systems Via Small-Gain Approaches. In David Hutchison, Takeo Kanade, Josef Kittler, Bao-Liang Lu, and Yin Hujun, editors, *Advances in Neural Networks - ISNN 2006*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin/Heidelberg, 2006.

[166] Tie-Shan Li, Dan Wang, Gang Feng, and Shao-Cheng Tong. A DSC approach to robust adaptive NN tracking control for strict-feedback nonlinear systems. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 40(3):915–927, 2010.

[167] Bing Chen, Xiaoping Liu, Kefu Liu, and Chong Lin. Direct adaptive fuzzy control of nonlinear strict-feedback systems. *Automatica*, 45(6):1530–1535, 2009.

[168] Guanyu Lai, Zhi Liu, Yun Zhang, and C. L. Philip Chen. Adaptive Position/Attitude Tracking Control of Aerial Robot With Unknown Inertial Matrix Based on a New Robust Neural Identifier. *IEEE Transactions on Neural Networks and Learning Systems*, 27(1):18–31, 2016.

[169] Z.-P. Jiang, A. R. Teel, and L. Praly. Small-gain theorem for ISS systems and applications. *Mathematics of Control, Signals, and Systems*, 7(2):95–120, 1994.

[170] Yan-Jun Liu, Guo-Xing Wen, and Shao-Cheng Tong. Direct adaptive NN control for a class of discrete-time nonlinear strict-feedback systems. *Neurocomputing*, 73(13-15):2498–2505, 2010.

[171] Roger D. Nussbaum. Some remarks on a conjecture in parameter adaptive control. *Systems & Control Letters*, 3(5):243–246, 1983.

[172] Zhiyong Chen. Nussbaum functions in adaptive control with time-varying unknown control coefficients. *Automatica*, 102:72–79, 2019.

[173] Shuzhi Sam Ge and J. Wang. Robust adaptive tracking for time-varying uncertain nonlinear systems with unknown control coefficients. *IEEE Transactions on Automatic Control*, 48(8):1463–1469, 2003.

[174] Bin Xu. Robust adaptive neural control of flexible hypersonic flight vehicle with dead-zone input nonlinearity. *Nonlinear Dynamics*, 80(3):1509–1520, 2015.

[175] P. Castaldi, N. Mimmo, R. Naldi, and L. Marconi. Robust Trajectory Tracking for Underactuated VTOL Aerial Vehicles: Extended for Adaptive Disturbance Compensation. In *Proceedings of the19th IFAC world congress*, Cape Town, South Africa,, 2014. IFAC.

[176] Mihai Lungu. Auto-landing of UAVs with variable centre of mass using the backstepping and dynamic inversion control. *Aerospace Science and Technology*, 103(2):105912, 2020.

[177] Rong Li, Mou Chen, and Qingxian Wu. Adaptive neural tracking control for uncertain nonlinear systems with input and output constraints using disturbance observer. *Neurocomputing*, 235:27–37, 2017.

[178] Ziquan Yu, Youmin Zhang, Bin Jiang, Xiang Yu, Jun Fu, Ying Jin, and Tianyou Chai. Distributed adaptive fault-tolerant close formation flight control of multiple trailing fixed-wing UAVs. *ISA Transactions*, 106:181–199, 2020.

[179] M. M. Polycarpou. Stable adaptive neural control scheme for nonlinear systems. *IEEE Transactions on Automatic Control*, 41(3):447–451, 1996.

[180] Yao Zou and Zewei Zheng. A Robust Adaptive RBFNN Augmenting Backstepping Control Approach for a Model-Scaled Helicopter. *IEEE Transactions on Control Systems Technology*, 23(6):2344–2352, 2015.

[181] Zeng Lian Liu and J. Svoboda. A new control scheme for nonlinear systems with disturbances. *IEEE Transactions on Control Systems Technology*, 14(1):176–181, 2006.

[182] S. Suresh, S. N. Omkar, V. Mani, and N. Sundararajan. Nonlinear Adaptive Neural Controller for Unstable Aircraft. *Journal of Guidance, Control, and Dynamics*, 28(6):1103–1111, 2005.

[183] Eric N. Johnson and Anthony Calise. Neural Network Adaptive Control of Systems with Input Saturation. In *Proceedings of the 2001 American Control Conference*, pages 3527–3532, Crystal Gateway Marriot, Arlington, VA, USA, 2000. American Automatic Control Council.

[184] A. A. Pashilkar, N. Sundararajan, and P. Saratchandran. Adaptive backstepping neural controller for reconfigurable flight control systems. *IEEE Transactions on Control Systems Technology*, 14(3):553–561, 2006.

[185] Wenxing Fu, Yuji Wang, Supeng Zhu, and Yingzhou Xia. Neural adaptive control of hypersonic aircraft with actuator fault using randomly assigned nodes. *Neurocomputing*, 174:1070–1076, 2016.

[186] Zian Cheng, Fuyang Chen, and Jingxiu Gong. Self-repairing control of air-breathing hypersonic vehicle with actuator fault and backlash. *Aerospace Science and Technology*, 97(10):105608, 2020.

[187] Heng Liu, Hongxing Wang, Jinde Cao, Ahmed Alsaedi, and Tasawar Hayat. Composite learning adaptive sliding mode control of fractional-order nonlinear systems with actuator faults. *Journal of the Franklin Institute*, 356(16):9580–9599, 2019.

[188] Xidong Tang, Gang Tao, and Suresh M. Joshi. Adaptive actuator failure compensation for parametric strict feedback systems and an aircraft application. *Automatica*, 39(11):1975–1982, 2003.

[189] Zhiyu Peng, Ruiyun Qi, and Bin Jiang. Adaptive fault tolerant control for hypersonic flight vehicle system with state constraints. *Journal of the Franklin Institute*, 357(14):9351–9377, 2020.

[190] Yuan Yuan, Zheng Wang, Lei Guo, and Huaping Liu. Barrier Lyapunov Functions-Based Adaptive Fault Tolerant Control for Flexible Hypersonic Flight Vehicles With Full State Constraints. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(9):3391–3400, 2020.

[191] Wangkui Liu, Yiyin Wei, Mingzhe Hou, and Guangren Duan. Integrated guidance and control with partial state constraints and actuator faults. *Journal of the Franklin Institute*, 356(9):4785–4810, 2019.

[192] Marcello R. Napolitano, Younghwan An, and Brad A. Seanor. A fault tolerant flight control system for sensor and actuator failures using neural networks. *Aircraft Design*, 3(2):103–128, 2000.

[193] H. A. Talebi, K. Khorasani, and S. Tafazoli. A recurrent neural-network-based sensor and actuator fault detection and isolation for nonlinear systems with application to the satellite's attitude control subsystem. *IEEE Transactions on Neural Networks*, 20(1):45–60, 2009.

[194] C. de Persis and A. Isidori. A geometric approach to nonlinear fault detection and isolation. *IEEE Transactions on Automatic Control*, 45(6):853–865, 2001.

[195] Seyyed Ali Emami and Afshin Banazadeh. Fault–tolerant predictive trajectory tracking of an air vehicle based on acceleration control. *IET Control Theory & Applications*, 14(5):750–762, 2020.

[196] Alireza Abbaspour, Payam Aboutalebi, Kang K. Yen, and Arman Sargolzaei. Neural adaptive observer-based sensor and actuator fault detection in nonlinear systems: Application in UAV. *ISA Transactions*, 67:317–329, 2017.

[197] Alireza Abbaspour, Kang K. Yen, Parisa Forouzannezhad, and Arman Sargolzaei. A Neural Adaptive Approach for Active Fault-Tolerant Control Design in UAV. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–11, 2018.

[198] Aydogan Savran, Ramazan Tasaltin, and Yasar Becerikli. Intelligent adaptive nonlinear flight control for a high performance aircraft with neural networks. *ISA Transactions*, 45(2):225–247, 2006.

[199] S. A. Emami and A. Banazadeh. Online Identification of Aircraft Dynamics in the Presence of Actuator Faults. *Journal of Intelligent and Robotic Systems*, 96(3-4):541–553, 2019.

[200] Seyyed Ali Emami and Afshin Banazadeh. Intelligent trajectory tracking of an aircraft in the presence of internal and external disturbances. *International Journal of Robust and Nonlinear Control*, 29(16):5820–5844, 2019.

[201] David Mayne. An apologia for stabilising terminal conditions in model predictive control. *International Journal of Control*, 86(11):2090–2095, 2013.

[202] Shuyi Shao, Mou Chen, and Youmin Zhang. Adaptive Discrete-Time Flight Control Using Disturbance Observer and Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 30(12):3708–3721, 2019.

[203] Eric Johnson, Anthony Calise, Hesham El-Shirbiny, and Rolf Eysdyk. Feedback linearization with Neural Network augmentation applied to X-33 attitude control. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Denver, CO, 2000.

[204] Eric N. Johnson and Anthony J. Calise. Limited Authority Adaptive Flight Control for Reusable Launch Vehicles. *Journal of Guidance, Control, and Dynamics*, 26(6):906–913, 2003.

[205] Eric N. Johnson and Suresh K. Kannan. Adaptive Flight Control for an Autonomous Unmanned Helicopter. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Monterey, California, 2002.

[206] Eric N. Johnson and Suresh K. Kannan. Adaptive Trajectory Control for Autonomous Helicopters. *Journal of Guidance, Control, and Dynamics*, 28(3):524–538, 2005.

[207] Alireza Abaspour, Seyed Hossein Sadati, and Mohammad Sadeghi. Nonlinear optimized adaptive trajectory control of helicopter. *Control Theory and Technology*, 13(4):297–310, 2015.

[208] Eric N. Johnson and Michael A. Turbe. Modeling, Control, and Flight Testing of a Small-Ducted Fan Aircraft. *Journal of Guidance, Control, and Dynamics*, 29(4):769–779, 2006.

[209] Mihai Lungu and Romulus Lungu. Landing Auto-Pilots for Aircraft Motion in Longitudinal Plane using Adaptive Control Laws Based on Neural Networks and Dynamic Inversion. *Asian Journal of Control*, 2016.

[210] J. Farrell, M. Polycarpou, and M. Sharma. Adaptive backstepping with magnitude, rate, and bandwidth constraints: Aircraft longitude control. In *Proceedings of the 2003 American Control Conference*, pages 3898–3904, Colorado, USA, 2003. IEEE.

[211] ShiXing Wang, Yu Zhang, YuQiang Jin, and YongQuan Zhang. Neural control of hypersonic flight dynamics with actuator fault and constraint. *Science China Information Sciences*, 58(7):1–10, 2015.

[212] Mou Chen, Shuzhi Sam Ge, and Beibei Ren. Adaptive tracking control of uncertain MIMO nonlinear systems with input constraints. *Automatica*, 47(3):452–465, 2011.

[213] Thomas Besselmann, Johan Lofberg, and Manfred Morari. Explicit MPC for LPV Systems: Stability and Optimality. *IEEE Transactions on Automatic Control*, 57(9):2322–2332, 2012.

[214] De-Feng He, Hua Huang, and Qiu-Xia Chen. Quasi-min–max MPC for constrained nonlinear systems with guaranteed input-to-state stability. *Journal of the Franklin Institute*, 351(6):3405–3423, 2014.

[215] Maciej Ławryńczuk. *Computationally efficient model predictive control algorithms*, volume 3. Springer International Publishing, Cham, 2014.

[216] Vincent A. Akpan and George D. Hassapis. Nonlinear model identification and adaptive model predictive control using neural networks. *ISA Transactions*, 50(2):177–194, 2011.

[217] Gonzalo Andres Garcia, Shawn Shahriar Keshmiri, and Thomas Stastny. Robust and Adaptive Nonlinear Model Predictive Controller for Unsteady and Highly Nonlinear Unmanned Aircraft. *IEEE Transactions on Control Systems Technology*, 23(4):1620–1627, 2015.

[218] Zheng Yan and Jun Wang. Model Predictive Control of Nonlinear Systems With Unmodeled Dynamics Based on Feedforward and Recurrent Neural Networks. *IEEE Transactions on Industrial Informatics*, 8(4):746–756, 2012.

[219] Changyun Wen, Jing Zhou, Zhitao Liu, and Hongye Su. Robust Adaptive Control of Uncertain Nonlinear Systems in the Presence of Input Satura-

tion and External Disturbance. *IEEE Transactions on Automatic Control*, 56(7):1672–1678, 2011.

[220] Khoi B. Ngo, Robert Mahony, and Jiang Zhong-Ping. Integrator Backstepping using Barrier Functions for Systems with Multiple State Constraints. In *44th IEEE Conference on Decision and Control, and European Control Conference*, pages 8306–8312, Seville, Spain, 2005. IEEE Operations Center.

[221] Keng Peng Tee, Shuzhi Sam Ge, and Eng Hock Tay. Barrier Lyapunov Functions for the control of output-constrained nonlinear systems. *Automatica*, 45(4):918–927, 2009.

[222] Achim Ilchmann, Eugene P. Ryan, and Philip Townsend. Tracking with Prescribed Transient Behavior for Nonlinear Systems of Known Relative Degree. *SIAM Journal on Control and Optimization*, 46(1):210–230, 2007.

[223] Md Meftahul Ferdaus, Mahardhika Pratama, Sreenatha G. Anavatti, and Matthew A. Garratt. PALM: An Incremental Construction of Hyperplanes for Data Stream Regression. *IEEE Transactions on Fuzzy Systems*, 27(11):2115–2129, 2019.

[224] Y. Lu, N. Sundararajan, and P. Saratchandran. Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm. *IEEE Transactions on Neural Networks*, 9(2):308–318, 1998.

[225] P. Saratchandran, N. Sundararajan, and Y. Li. Analysis of minimal radial basis function network algorithm for real-time identification of nonlinear dynamic systems. *IEE Proceedings - Control Theory and Applications*, 147(4):476–484, 2000.

[226] Shaik Ismail, Abhay A. Pashilkar, Ramakalyan Ayyagari, and Narasimhan Sundararajan. Improved neural-aided sliding mode controller for autolanding under actuator failures and severe winds. *Aerospace Science and Technology*, 33(1):55–64, 2014.

[227] Guang-Bin Huang, P. Saratchandran, and Narasimhan Sundararajan. An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 34(6):2284–2292, 2004.

[228] Mahardhika Pratama, Sreenatha G. Anavatti, and Edwin Lughofer. GENEFIS: Toward an Effective Localist Network. *IEEE Transactions on Fuzzy Systems*, 22(3):547–562, 2014.

[229] Shiqian Wu, Meng Joo Er, and Yang Gao. A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks. *IEEE Transactions on Fuzzy Systems*, 9(4):578–594, 2001.

[230] Seyyed Ali Emami and Kasra K. A. Ahmadi. A self-organizing multi-model ensemble for identification of nonlinear time-varying dynamics of aerial vehicles. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 235(7):1164–1178, 2021.

[231] Abhijit Das, Frank Lewis, and Kamesh Subbarao. Backstepping Approach for Controlling a Quadrotor Using Lagrange Form Dynamics. *Journal of Intelligent and Robotic Systems*, 56(1-2):127–151, 2009.

[232] Yu Kang, Shaofeng Chen, Xuefeng Wang, and Yang Cao. Deep Convolutional Identifier for Dynamic Modeling and Adaptive Control of Unmanned Helicopter. *IEEE Transactions on Neural Networks and Learning Systems*, 30(2):524–538, 2019.

[233] Chia-Wei Kuo, Ching-Chih Tsai, and Chi-Tai Lee. Intelligent Leader-Following Consensus Formation Control Using Recurrent Neural Networks for Small-Size Unmanned Helicopters. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(2):1288–1301, 2021.

[234] Pieter Abbeel, Adam Coates, and Andrew Y. Ng. Autonomous Helicopter Aerobatics through Apprenticeship Learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[235] Qianying Li. *Masters Thesis: Grey-Box System Identification of a Quadrotor Unmanned Aerial Vehicle*. Master Thesis, Faculty of Mechanical, Maritime and Materials Engineering, 2014.

[236] Shuai Tang, Zhiqiang Zheng, Shaoke Qian, and Xinye Zhao. Nonlinear system identification of a small-scale unmanned helicopter. *Control Engineering Practice*, 25:1–15, 2014.

[237] Clément Hamel, Ruxandra Botez, and Margaux Ruby. Cessna Citation X Airplane Grey-Box Model Identification without Preliminary Data. In *SAE 2014 Aerospace Systems and Technology Conference*, SAE Technical Paper Series. SAE International400 Commonwealth Drive, Warrendale, PA, United States, 2014.

[238] F. F. Leung, H. K. Lam, S. H. Ling, and P. S. Tam. Tuning of the structure and parameters of a neural network using an improved genetic algorithm. *IEEE Transactions on Neural Networks*, 14(1):79–88, 2003.

[239] Francisco da Costa Lopes, Edson H. Watanabe, and Luis Guilherme B. Rolim. A control-oriented model of a PEM fuel cell stack based on NARX and NOE neural networks. *IEEE Transactions on Industrial Electronics*, 62(8):5155–5163, 2015.

[240] Gérard Dreyfus. *Neural networks: Methodology and applications*, volume 24. Springer, Berlin, 2005.

[241] Qinghua Zhang. Nonlinear system identification with output error model through stabilized simulation. *IFAC Proceedings Volumes*, 37(13):501–506, 2004.

[242] Heidar A. Talebi, Farzaneh Abdollahi, Rajni V. Patel, and Khashayar Khorasani. *Neural Network-Based State Estimation of Nonlinear Systems*, volume 395. Springer New York, New York, NY, 2010.

[243] Krzysztof Patan and Józef Korbicz. Nonlinear model predictive control of a boiler unit: A fault tolerant control study. *International Journal of Applied Mathematics and Computer Science*, 22(1):225–237, 2012.

[244] Krzysztof Patan. Neural network-based model predictive control: fault tolerance and stability. *IEEE Transactions on Control Systems Technology*, 23(3):1147–1155, 2015.

[245] Nan-Ying Liang, Guang-Bin Huang, P. Saratchandran, and N. Sundararajan. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on Neural Networks*, 17(6):1411–1423, 2006.

[246] Zheng Yan and Jun Wang. Robust model predictive control of nonlinear systems with unmodeled dynamics and bounded uncertainties based on neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 25(3):457–469, 2014.

[247] Chao Jia, Xiaoli Li, Kang Wang, and Dawei Ding. Adaptive control of nonlinear system using online error minimum neural networks. *ISA Transactions*, 2016.

[248] Ning Wang, Jing-Chao Sun, Meng Joo Er, and Yan-Cheng Liu. A Novel Extreme Learning Control Framework of Unmanned Surface Vehicles. *IEEE Transactions on Cybernetics*, 46(5):1106–1117, 2016.

[249] Jianwei Zhao, Zhihui Wang, and Dong Sun Park. Online sequential extreme learning machine with forgetting mechanism. *Neurocomputing*, 87:79–89, 2012.

[250] Symone G. Soares and Rui Araújo. An adaptive ensemble of on-line extreme learning machines with variable forgetting factor for dynamic system prediction. *Neurocomputing*, 171:693–707, 2016.

[251] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[252] Somil Bansal, K. Anayo Akametalu, Frank J. Jiang, Forrest Laine, and Claire J. Tomlin. Learning Quadrotor Dynamics Using Neural Network for Flight Control. In *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016.

[253] Mark B. Tischler and Robert K. Remple. *Aircraft & Rotorcraft system identification: Engineering Methods with Flight-Test Examples*. AIAA, Blacksburg, Virginia, 2006.

[254] Wen Yu and Mario Pacheco. Impact of random weights on nonlinear system identification using convolutional neural networks. *Information Sciences*, 477:1–14, 2019.

[255] Alessandro Giusti, Jerome Guzzi, Dan C. Ciresan, Fang-Lin He, Juan P. Rodriguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jurgen Schmidhuber, Gianni Di Caro, Davide Scaramuzza, and Luca M. Gambardella. A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2016.

[256] Dong Ki Kim and Tsuhan Chen. Deep Neural Network for Real-Time Autonomous Indoor Navigation.

[257] Adrian Carrio, Carlos Sampedro, Alejandro Rodriguez-Ramos, and Pascual Campoy. A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles. *Journal of Sensors*, 2017(2):1–13, 2017.

[258] Arthur E. Bryson and Yu-Chi Ho. *Applied optimal control*. Taylor & Francis, 1975.

[259] Lemei M. Zhu, Hamidreza Modares, Gan Oon Peen, Frank L. Lewis, and Baozeng Yue. Adaptive Suboptimal Output-Feedback Control for Linear Systems Using Integral Reinforcement Learning. *IEEE Transactions on Control Systems Technology*, 23(1):264–273, 2015.

[260] Dante Kalise, Sudeep Kundu, and Karl Kunisch. Robust Feedback Control of Nonlinear PDEs by Numerical Approximation of High-Dimensional Hamilton–Jacobi–Isaacs Equations. *SIAM Journal on Applied Dynamical Systems*, 19(2):1496–1524, 2020.

[261] Murad Abu-Khalaf, Jie Huang, and Frank L. Lewis. *Nonlinear H_2/H_infinity Constrained Feedback Control: A practical design approach using neural networks*. Advances in industrial control, 1430-9491. Springer, London, 2006.

[262] Travis Dierks and Sarangapani Jagannathan. Optimal Control of Affine Nonlinear Continuous-time Systems Using an Online Hamilton-Jacobi-Isaacs Formulation. In *49th IEEE Conference on Decision and Control (CDC)*, pages 3048–3053, Atlanta, GA, USA, 2010. IEEE.

[263] A. J. van der Schaft. L/sub 2/-gain analysis of nonlinear systems and nonlinear state-feedback H/sub infinity / control. *IEEE Transactions on Automatic Control*, 37(6):770–784, 1992.

[264] Randal W. Bea. Successive Galerkin approximation algorithms for nonlinear optimal and robust control. *International Journal of Control*, 71(5):717–743, 1998.

[265] Asma Al-Tamimi, Frank L. Lewis, and Murad Abu-Khalaf. Discrete-time nonlinear HJB solution using approximate dynamic programming: Convergence proof. *IEEE transactions on systems, man, and cybernetics. Part*

B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, 38(4):943–949, 2008.

[266] Jennie Si. *Handbook of learning and approximate dynamic programming*, volume 2 of *IEEE press series on computational intelligence*. IEEE Press, Hoboken, New Jersey, 2004.

[267] Derong Liu, Ding Wang, Fei-Yue Wang, Hongliang Li, and Xiong Yang. Neural-network-based online HJB solution for optimal robust guaranteed cost control of continuous-time uncertain nonlinear systems. *IEEE Transactions on Cybernetics*, 44(12):2834–2847, 2014.

[268] Randal W. Beard, George N. Saridis, and John T. Wen. Galerkin approximations of the generalized Hamilton-Jacobi-Bellman equation. *Automatica*, 33(12):2159–2177, 1997.

[269] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass. and London, 1998.

[270] Kyriakos G. Vamvoudakis and Frank L. Lewis. Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica*, 46(5):878–888, 2010.

[271] Paul J. Werbos. Approximate dynamic programming for real-time control and neural modeling. In David A. White and Donald A. Sorge, editors, *Handbook of intelligent control*. Van Nostrand Reinhold, New York, 1992.

[272] Ding Wang, Haibo He, and Derong Liu. Improving the Critic Learning for Event-Based Nonlinear H_infinity Control Design. *IEEE Transactions on Cybernetics*, 47(10):3417–3428, 2017.

[273] Kyriakos G. Vamvoudakis. Event-triggered optimal adaptive control algorithm for continuous-time nonlinear systems. *IEEE/CAA Journal of Automatica sinica*, 1(3):282–293, 2014.

[274] Kyriakos G. Vamvoudakis, Arman Mojoodi, and Henrique Ferraz. Event-triggered optimal tracking control of nonlinear systems. *International Journal of Robust and Nonlinear Control*, 27(4):598–619, 2017.

[275] Murad Abu-Khalaf and Frank L. Lewis. Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach. *Automatica*, 41(5):779–791, 2005.

[276] Draguna Vrabie and Frank Lewis. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural networks : the official journal of the International Neural Network Society*, 22(3):237–246, 2009.

[277] Biao Luo, Huai-Ning Wu, and Tingwen Huang. Off-policy reinforcement learning for H∞ control design. *IEEE Transactions on Cybernetics*, 45(1):65–76, 2015.

[278] Shan Xue, Biao Luo, and Derong Liu. Event-Triggered Adaptive Dynamic Programming for Unmatched Uncertain Nonlinear Continuous-Time Systems. *IEEE Transactions on Neural Networks and Learning Systems*, 32(7):2939–2951, 2021.

[279] David Nodland, Hassan Zargarzadeh, and Sarangapani Jagannathan. Neural network-based optimal adaptive output feedback control of a helicopter UAV. *IEEE Transactions on Neural Networks and Learning Systems*, 24(7):1061–1073, 2013.

[280] Draguna Vrabie and Frank Lewis. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural networks : the official journal of the International Neural Network Society*, 22(3):237–246, 2009.

[281] Frank L. Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, 9(3):32–50, 2009.

[282] Dongchen Han and S. N. Balakrishnan. State-constrained agile missile control with adaptive-critic-based neural networks. *IEEE Transactions on Control Systems Technology*, 10(4):481–489, 2002.

[283] Silvia Ferrari and Robert F. Stengel. Online Adaptive Critic Flight Control. *Journal of Guidance, Control, and Dynamics*, 27(5):777–786, 2004.

[284] Said G. Khan, Guido Herrmann, Frank L. Lewis, Tony Pipe, and Chris Melhuish. Reinforcement learning and optimal adaptive control: An overview and implementation examples. *Annual Reviews in Control*, 36(1):42–59, 2012.

[285] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[286] Benjamin Recht. A Tour of Reinforcement Learning: The View from Continuous Control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):253–279, 2019.

[287] Biao Luo, Huai-Ning Wu, and Tingwen Huang. Optimal Output Regulation for Model-Free Quanser Helicopter With Multistep Q-Learning. *IEEE Transactions on Industrial Electronics*, 65(6):4953–4961, 2018.

[288] Gautam Reddy, Antonio Celani, Terrence J. Sejnowski, and Massimo Vergassola. Learning to soar in turbulent environments. *Proceedings of the National Academy of Sciences of the United States of America*, 113(33):E4877–84, 2016.

[289] Gautam Reddy, Jerome Wong-Ng, Antonio Celani, Terrence J. Sejnowski, and Massimo Vergassola. Glider soaring via reinforcement learning in the field. *Nature*, 562(7726):236–239, 2018.

[290] Haobin Shi, Xuesi Li, Kao-Shing Hwang, Wei Pan, and Genjiu Xu. Decoupled Visual Servoing With Fuzzy Q -Learning. *IEEE Transactions on Industrial Informatics*, 14(1):241–252, 2018.

[291] Chunyu Nie, Zewei Zheng, and Ming Zhu. Three-Dimensional Path-Following Control of a Robotic Airship with Reinforcement Learning. *International Journal of Aerospace Engineering*, 2019:1–12, 2019.

[292] Beakcheol Jang, Myeonghwi Kim, Gaspard Harerimana, and Jong Wook Kim. Q-Learning Algorithms: A Comprehensive Classification and Applications. *IEEE Access*, 7:133653–133667, 2019.

[293] Jungdam Won, Jongho Park, Kwanyu Kim, and Jehee Lee. How to train your dragon. *ACM Transactions on Graphics*, 36(6):1–13, 2017.

[294] Ivana Palunko, Aleksandra Faust, Patricio Cruz, Lydia Tapia, and Rafael Fierro. A Reinforcement Learning Approach Towards Autonomous Suspended Load Manipulation Using Aerial Robots. In *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013. IEEE.

[295] Andrew Y. Ng, Adam Coates, Mark Diel, Varun Ganapathi, Jamie Schulte, Ben Tse, Eric Berger, and Eric Liang. Autonomous Inverted Helicopter Flight via Reinforcement Learning. In M. H. Ang, Marcelo H. Ang, and Oussama Khatib, editors, *Experimental Robotics IX*, volume 21 of *Springer Tracts in Advanced Robotics*, pages 363–372. Springer, Secaucus, 2006.

[296] Fereshteh Sadeghi and Sergey Levine. CAD2RL: Real Single-Image Flight without a Single Real Image. In *Robotics: Science and Systems Conference (RSS)*, Cambridge MA, USA, 2017.

[297] Bo Zhang, Chi Harold Liu, Jian Tang, Zhiyuan Xu, Jian Ma, and Wendong Wang. Learning-Based Energy-Efficient Data Collection by Unmanned Vehicles in Smart Cities. *IEEE Transactions on Industrial Informatics*, 14(4):1666–1676, 2018.

[298] John Schulman, Sergey Levine, Philipp Moritz, Michael Jordan, and

Pieter Abbeel. Trust Region Policy Optimization. In *31 st International Conference on Machine Learning*, Lille, France, 2015.

[299] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *19th International Conference on Machine Learning*, pages 267–274, 2002.

[300] Chen-Huan Pi, Kai-Chun Hu, Stone Cheng, and I-Chen Wu. Low-level autonomous control and tracking of quadrotor using reinforcement learning. *Control Engineering Practice*, 95(4):104222, 2020.

[301] Thomas Degris, Martha White, and Richard S. Sutton. Off-Policy Actor-Critic. In *International Conference on Machine Learning*, Scotland, UK, 2012.

[302] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms.

[303] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic Policy Gradient Algorithms. In *31st International Conference on Machine Learning*, volume 32, Beijing, China, 2014.

[304] Russell Enns and Jennie Si. Apache Helicopter Stabilization Using Neural Dynamic Programming. *Journal of Guidance, Control, and Dynamics*, 25(1):19–25, 2002.

[305] R. Enns and Jennie Si. Helicopter trimming and tracking control using direct neural dynamic programming. *IEEE Transactions on Neural Networks*, 14(4):929–939, 2003.

[306] Shun-ichi Amari. Natural Gradient Works Efficiently in Learning. *Neural computation*, 10(2):251–276, 1998.

[307] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous

control with deep reinforcement learning. In *International Conference on Learning Representations*, San Juan, Puerto Rico, 2016.

[308] Yuanda Wang, Jia Sun, Haibo He, and Changyin Sun. Deterministic Policy Gradient With Integral Compensator for Robust Quadrotor Control. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–13, 2019.

[309] Alejandro Rodriguez-Ramos, Carlos Sampedro, Hriday Bavle, Paloma de La Puente, and Pascual Campoy. A Deep Reinforcement Learning Strategy for UAV Autonomous Landing on a Moving Platform. *Journal of Intelligent & Robotic Systems*, 93(1-2):351–366, 2019.

[310] Carlos Sampedro, Hriday Bavle, Alejandro Rodriguez-Ramos, Paloma de La Puente, and Pascual Campoy. Laser-Based Reactive Navigation for Multirotor Aerial Robots using Deep Reinforcement Learning. In *International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 2018. IEEE.

[311] Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In Ivan Bratko and Saso. Ed Dzeroski, editors, *Proceedings of the Sixteenth International Conference on Machine Learning*, Bled, Slovenia, 1999.

[312] Chao Wang, Jian Wang, Yuan Shen, and Xudong Zhang. Autonomous Navigation of UAVs in Large-Scale Complex Environments: A Deep Reinforcement Learning Approach. *IEEE Transactions on Vehicular Technology*, 68(3):2124–2136, 2019.

[313] Abhik Singla, Sindhu Padakandla, and Shalabh Bhatnagar. Memory-Based Deep Reinforcement Learning for Obstacle Avoidance in UAV With Limited Environment Knowledge. *IEEE Transactions on Intelligent Transportation Systems*, 22(1):107–118, 2021.

[314] Nathan O. Lambert, Daniel S. Drew, Joseph Yaconelli, Sergey Levine, Roberto Calandra, and Kristofer S. J. Pister. Low-Level Control of a Quadrotor With Deep Model-Based Reinforcement Learning. *IEEE Robotics and Automation Letters*, 4(4):4224–4230, 2019.

[315] Sylvain Koos, J-B Mouret, and S. Doncieux. The Transferability Approach: Crossing the Reality Gap in Evolutionary Robotics. *IEEE Transactions on Evolutionary Computation*, 17(1):122–145, 2013.

[316] Kirk Y.W. Scheper and Guido C.H.E. de Croon. Evolution of robust high speed optical-flow-based landing for autonomous MAVs. *Robotics and Autonomous Systems*, 124(46):103380, 2020.

[317] J. Andrew Bagnell and Jeff G. Schneider. Autonomous Helicopter Control Using Reinforcement Learning Policy Search Methods. In *International Conference on Robotics and Automation (ICRA)*, Seoul, Korea, 2001. IEEE.

[318] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Optimizing walking controllers for uncertain inputs and environments. *ACM Transactions on Graphics*, 29(4):1–8, 2010.

[319] Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search provides a competitive approach to reinforcement learning.

[320] S. L. Waslander, G. M. Hoffmann, Jung Soon Jang, and C. J. Tomlin. Multi-agent quadrotor testbed control design: Integral sliding mode vs. reinforcement learning. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3712–3717, Edmonton, AB, Canada, 2005. IEEE.

[321] Sergey Levine and Vladlen Koltun. Guided Policy Search. In *30th International Conference on Machine Learning*, Atlanta, Georgia, USA, 2013.

[322] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution Strategies as a Scalable Alternative to Reinforcement Learning.

[323] Tianhao Zhang, Gregory Kahn, Sergey Levine, and Pieter Abbeel. Learning Deep Control Policies for Autonomous Aerial Vehicles with MPC-Guided Policy Search. In Allison Okamura and Arianna Menciassi, editors, *2016 IEEE International Conference on Robotics and Automation*, pages 528–535. IEEE, 2016.

[324] H. J. Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(11):P11011, 2005.

[325] Jung-Su Ha, Soon-Seo Park, and Han-Lim Choi. Topology-guided path integral approach for stochastic optimal control in cluttered environment. *Robotics and Autonomous Systems*, 113(28):81–93, 2019.

[326] Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M. Rehg, Byron Boots, and Evangelos A. Theodorou. Information Theoretic MPC for Model-Based Reinforcement Learning. In *IEEE International Conference on Robotics and Automation (ICRA))*, IEEE International Conference on Robotics and Automation, Singapore, 2017. IEEE.

[327] Keuntaek Lee, Jason Gibson, and Evangelos A. Theodorou. Aggressive Perception-Aware Navigation Using Deep Optical Flow Dynamics and PixelMPC. *IEEE Robotics and Automation Letters*, 5(2):1207–1214, 2020.

[328] Chen Liang, Weihong Wang, Zhenghua Liu, Chao Lai, and Benchun Zhou. Learning to Guide: Guidance Law Based on Deep Meta-Learning and Model Predictive Path Integral Control. *IEEE Access*, 7:47353–47365, 2019.

[329] Atilim Gunes Baydin, Robert Cornish, David Martinez Rubio, Mark Schmidt, and Frank Wood. Online Learning Rate Adaptation with Hypergradient Descent. *ArXiv e-prints*, pages 1–10, 2017.

[330] Liang Tang, Michael Roemer, Jianhua Ge, Agamemnon Crassidis, J.V.R. Prasad, and Christine Belcastro. Methodologies for Adaptive Flight En-

velope Estimation and Protection. In *AIAA Guidance, Navigation, and Control Conference*, Chicago, Illinois, 2009. AIAA.

[331] N. Hopfe, A. Ilchmann, and E. P. Ryan. Funnel Control With Saturation: Linear MIMO Systems. *IEEE Transactions on Automatic Control*, 55(2):532–538, 2010.

[332] Norman Hopfe, Achim Ilchmann, and Eugene P. Ryan. Funnel Control With Saturation: Nonlinear SISO Systems. *IEEE Transactions on Automatic Control*, 55(9):2177–2182, 2010.

[333] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of 21st International conference on Machine learning*, Banff, Canada, 2004.

[334] Adam Coates, Pieter Abbeel, and Andrew Y. Ng. Apprenticeship learning for helicopter control. *Communications of the ACM*, 52(7):97–105, 2009.

[335] Naira Hovakimyan, Chengyu Cao, Evgeny Kharisov, Enric Xargay, and Irene M. Gregory. L 1 adaptive control for safety-critical systems. *IEEE Control Systems*, 31(5):54–104, 2011.

[336] Naira Hovakimyan and Chengyu Cao. *L1 adaptive control theory: Guaranteed robustness with fast adaptation*, volume 21 of *Advances in design and control*. Society for Industrial and Applied Mathematics, Philadelphia, 2010.

[337] Randall D. Beer. On the Dynamics of Small Continuous-Time Recurrent Neural Networks. *Adaptive Behavior*, 3(4):469–509, 1995.

[338] Taylor Clawson, Silvia Ferrari, Sawyer Fuller, and Robert Wood. Spiking Neural Network (SNN) Control of a Flapping Insect-Scale Robot. In *55th Conference on Decision and Control (CDC)*, pages 3381–3388, Las Vegas, USA, 2016. IEEE.

[339] Jesse J. Hagenaars, Federico Paredes-Valles, Sander M. Bohte, and Guido C. H. E. de Croon. Evolved Neuromorphic Control for High Speed Divergence-Based Landings of MAVs. *IEEE Robotics and Automation Letters*, 5(4):6239–6246, 2020.

[340] Antonios K. Alexandridis and Achilleas D. Zapranis. Wavelet neural networks: A practical guide. *Neural networks : the official journal of the International Neural Network Society*, 42:1–27, 2013.

[341] Chih-Min Lin, Ching-Fu Tai, and Chang-Chih Chung. Intelligent control system design for UAV using a recurrent wavelet neural network. *Neural Computing and Applications*, 24(2):487–496, 2014.

[342] Chih-Min Lin and Enkh-Amgalan Boldbaatar. Autolanding Control Using Recurrent Wavelet Elman Neural Network. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(9):1281–1291, 2015.