A Survey of Artificial Neural Networks with Model-based Control Techniques for Flight Control of Unmanned Aerial Vehicles

A Survey of Artificial Neural Networks with Model-based Control Techniques for Flight Control of Unmanned Aerial Vehicles

Weibin Gu[1], Kimon P. Valavanis[1], Matthew J. Rutherford[2], Alessandro Rizzo[3]

***Abstract*— Model-based control (MBC) techniques have been successfully developed for flight control applications of unmanned aerial vehicles (UAVs) in recent years. However, their heavy reliance on the fidelity of the plant model coupled with high computational complexity make the design and implementation process challenging. To overcome such challenges, attention has been focused on the use of artificial neural networks (ANNs) to study complex systems since they show promise in system identification and controller design, to say the least. This survey aims to provide a literature review on combining MBC techniques with ANNs for UAV flight control, with the goal of laying the foundation for efficient controller designs with performance guarantees. A brief discussion on frequently-used ANNs is presented along with an analysis of their time complexity. Classification/comparison of existing dynamic modeling approaches and control techniques is provided. Challenging research questions and an envisaged control architecture are also posed for future development.**

I. INTRODUCTION

Model-based control (MBC) techniques have been successfully developed for flight control applications of unmanned aerial vehicles (UAVs) in recent years such as feedback linearization (FL), a.k.a. nonlinear dynamic inversion (NDI) [1], adaptive control [2], model predictive control (MPC) [3], sliding mode control (SMC) [4], backstepping control [5] and $\mathcal{H}_\infty$ robust control [6]. However, one major problem with MBC systems is their dependency on the accuracy of the mathematical model of the real plant [7]. Undoubtedly, there is no perfect model owing to the fact that the fidelity of the model is always affected by (i) parametric uncertainties; (ii) unmodeled dynamics; and (iii) disturbances and noise. For UAVs, unmodeled dynamics include advanced aerodynamics effects such as blade flapping and effect of airflow, which are generally neglected in modeling for simplicity reasons (and it is indeed hard to capture their dynamics as well) but do have a great impact on the real flight [8]. Moreover, disturbances like wind gusts and sensor noise are ubiquitous in the real world, keeping in mind that assumptions on noise are not always realistic as noise is mostly non-Gaussian.

To cope with these issues, modern control techniques are available and have been applied to flight control, despite some limitations and drawbacks. A widely-used technique is gain scheduling (GS) [9] which demonstrates capabilities of handling parameter variations and nonlinearities. However, frequent and rapid changes of controller gains are likely to drive the system unstable [10]. Moreover, high cost of design and implementation increases with the number of operating points [11]. As an alternative, robust control is only good for presumed bounded parametric uncertainties and not able to handle either unbounded ones or unmodeled dynamics [6], [12]. Lastly, adaptive control is specialized in dealing with parametric uncertainties, and there barely exists widely accepted solution to robust adaptive control problem so far [13].

Apart from the heavy reliance on the accuracy of the plant model, another problem with MBC systems is called "explosion of complexity" [14] which could arise from (i) mathematical inverse model required in FL; (ii) repeated differentiation of virtual controller in backstepping control; and (iii) prediction over a future horizon of plant behavior as well as online optimization process in MPC, just to list a few. Notably for those highly nonlinear complex systems such as UAVs, the complexity of the controller grows substantially as the order of the system increases.

To overcome the aforementioned challenges, attention has been focused on the use of artificial neural networks (ANNs) to study complex control systems since they show promise in system identification and controller design, to say the least [15]–[17], which greatly thanks to their merits as follows:

- Capability of identifying nonlinear and multi-variable systems [18], [19];
- Ability to learn and adapt in real time;
- Relatively easy processing procedure and hardware implementation.

However, there are also downsides for ANNs being used as listed below:

- Requirement for a large number of training data;
- Easy to learn spurious relationships which may lead to poor generalization capabilities [20];
- Lack of interpretability due to black-box characteristics;
- No systematic approach for ANN architecture design: For instance, decisions on the number of layers and neurons, activation functions, weight adjustment mechanisms and so forth are generally arbitrarily made, not in a systematic fashion.

In spite of these drawbacks, it is still quite straightforward to observe a complementary relation between MBC techniques and ANNs. Specifically, the explicit system knowledge that

[1]Weibin Gu and Kimon P. Valavanis are with the Department of Electrical and Computer Engineering, University of Denver, Colorado, USA {Weibin.Gu, Kimon.Valavanis}@du.edu

[2]Matthew J. Rutherford is with the Department of Computer Science, University of Denver, Colorado, USA Matthew.Rutherford@du.edu

[3]Alessandro Rizzo is with the Department of Electronics and Telecommunications, Politecnico di Torino, Torino, Italy alessandro.rizzo@polito.it

MBC relies heavily on could be leveraged to facilitate the training process and to improve the performance and the interpretability of ANNs. In return, ANNs can offer a powerful means to control system design, particularly in complex dynamics modeling as well as real-time adaptation and implemention. Therefore, a combination between MBC techniques and ANNs is naturally expected by intuition and has indeed yielded a large amount of literature.

The main objective of this paper is to provide a literature review on MBC techniques combined with ANNs for UAV flight control (a.k.a. low-level control) with the aim to lay the foundation for efficient controller designs with performance guarantees. Classification/comparison of existing dynamic modeling and control techniques is provided to show their disadvantages and limitations, giving an insight into the possible future development. To the best knowledge of the authors, this survey is believed to be the first work on such topic. A relevant and recent research refers to [16] where deep learning techniques and their applications for UAV-based solutions are reviewed, focusing on high-level control (i.e., path planning, situation awareness, etc).

The remainder of this paper is organized as follows. Preliminaries of ANNs are concisely introduced in Section II where several frequently-used ANNs are presented along with an analysis of their time complexity. Section III provides the literature review on UAV dynamic modeling with the use of ANNs, followed by Section IV where MBC techniques combined with ANNs for controller design is surveyed. Classification/comparison is elaborated in Section V to remark the limitations of existing dynamic modeling approaches and control techniques. Some challenging research questions are also posed for future development together with an envisaged control architecture. Finally, this survey is concluded with Section VI.

II. ARTIFICIAL NEURAL NETWORKS

A. Preliminaries

The smallest element in ANNs is called *neuron* by analogy with neurophysiology which was firstly studied in [21] and behaves as a function in principle. It generally consists of a scalar-valued activation function $f : \mathbb{R} \to \mathbb{R}$ and two training parameters, namely input weight matrix $\mathbf{W} \in \mathbb{R}^{1\times R}$ and bias weight $b \in \mathbb{R}^{1\times 1}$, where R is the number of elements in the input vector $\mathbf{p} \in \mathbb{R}^{R\times 1}$. Given an input $\mathbf{p}$, a neuron generates output $a \in \mathbb{R}^{1\times 1}$ by $a = f(\mathbf{W}\mathbf{p}+b)$ where activation function $f(\cdot)$ can be chosen as hard-limit, linear, log-sigmoid or tan-sigmoid function and so forth based on the requirements for network performance [22].

An ANN is a network usually made up of a single input and output layer and several hidden layers. Each of the layer (except input layer) comprises a number of neurons, possibly governed by different activation functions. Throughout this survey, we follow the convention that the total number of layers of an ANN is the sum of hidden layers plus one (accounting for the single output layer), regardless of input layer.

Depending on the connections between each neuron, the architecture of ANNs can be categorized into *feed-forward* and *recurrent*. In feed-forward networks, a neuron in one layer can receive inputs only from those in the previous layer. On the contrary, feedback connections are allowed in recurrent networks which put on more dynamics at the cost of increasing complexity. On the other hand, ANNs can be also classified into *offline* and *online* networks based on whether the network parameters (i.e., weight matrices and biases) are fixed a priori or adjustable in real time. Offline ANNs are appealing mostly because the time for offline training is not critical while online ANNs show promise in learning changes in the dynamics. Table I summarizes the frequently-used representatives of feed-forward and recurrent networks in the literature of UAV flight control. The distinguished differences among them mainly lie in the choices of activation functions and connection architecture, which may further result in different training techniques for network parameters. Thorough explanation of each type of ANNs is omitted here due to space limitation. For those interested in details, Table I provides some straightforward references.

TABLE I
CLASSIFICATION OF ANNS

Type of Connection	**ANNs**
Feed-forward	Multi-layer perceptron (MLP) [22] Radial basis function (RBF) network [22] Rectified-linear unit (ReLU) network [23] Wavelet neural network (WNN) [24]
Recurrent	Elman network [22] Hopfield network [22] Echo state network (ESN) [24]–[26] Long-short-term-memory (LSTM) network [27]

B. Time Complexity

Since it is crucial for controller design to be implemented in real time, analysis on time complexity (TC) should be given a high priority. Unfortunately, not much literature on combination of MBC techniques and ANNs for UAV applications treats this issue meticulously, particularly say an analytical analysis that offers a clear mind on the required computational resources beforehand. In the sequel, we attempt to generalize the concept of TC of ANNs and provide analytical formulation for one single contribution.

We arbitrarily attribute TC of an ANN to three processes: (i) offline training process; (ii) online training process; and (iii) output generation given certain inputs. For offline ANNs, TC can be merely regarded as the time required for ANNs to generate outputs because offline training time is not critical in many applications and therefore can be neglected. Hence, TC for offline ANNs is simply associated to the network architecture since it can be expressed as a function of number of input nodes, output nodes and hidden neurons as will be shown later. For online ANNs, apart from the time for output generation, online training time should be also taken into account because the online training process

for parameter adaptation does have a significant impact on the real-time performance of ANNs. In the following, we formulate the time required for output generation of several ANNs appearing in Table I. We denote the number of input nodes, output nodes and hidden neurons by $N, K,$ and $L \in \mathbb{Z}^+$, respectively.

It is commonly known that TC refers to the time required for running algorithms, which is evidently influenced by the computer being used. However, since it can be generally expressed as the number of elementary operations included in the algorithms which are normally assumed to take constant execution time on a given computer, using a different computer to run algorithms can be therefore regarded as changing the execution time by a constant factor (which is determined by the computer configurations). Motivated by [28], Table II shows TC for different operations that might be encountered in the ANNs in the later formulation. Note that the choice of multiplication algorithm affects the complexity of division, exponentiation, exponential function and square root. We choose 3-way Toom-Cook algorithm for multiplication in our case, though there are other variations of algorithm available such as Schoolbook long multiplication and Karatsuba algorithm [29]. The scalar constants $K_i \in \mathbb{R}^+$ with $i = A, E, M, \phi$ represent the ratios between the time cost for different operations calculated with algorithms specified in Table II and the unit time cost (which is defined as the time cost for addition/subtraction operation), hence K_i is always equal or greater than unity [28]. Since the multiplication algorithm determines the complexity of other operations and the constants K_i for multiplication, division and square root are identical, these constants are equivalently denoted by K_M with a slight abuse of notation. Besides, subscript A denotes addition/subtraction operation, E denotes exponentiation, and ϕ denotes exponential function. Similarly, $N_i \in \mathbb{R}$ with $i = A, E, M, \phi$ represent the number of each operation involved in a network. To make the results more numerical, we assume that all the computations are performed on a 32-bit microcontroller in spite of the fact that industrial embedded controllers vary in number of bits, hence n and k are all equal to 32 in Table II, where n denotes the input size in units of bits needed to represent the input and k is the number of digits of the exponent.

1) Time Complexity of a Feed-forward Network: Given a fully-connected feed-forward network with a single hidden layer whose activation functions are log-sigmoid functions and output functions are linear functions, the time complexity (TC) for output generation can be formulated as

$$\begin{aligned} \text{TC} &= N_A \cdot K_A + N_M \cdot K_M + N_\phi \cdot K_\phi \\ &= L(N+K+1) + L(N+K+1)K_M + LK_\phi. \end{aligned} \tag{1}$$

Remark. For a fully-connected feed-forward network with a single hidden layer, the outputs of hidden layer with log-sigmoid activation functions can be expressed as

$$z_i = \phi(\sum_{j=1}^{N} w_{ji}x_j) + b_i, \quad i = 1, \cdots, L \tag{2}$$

where x_j denotes j^{th} input, w_{ji} denotes the weight of i^{th} hidden neuron associated with j^{th} input, b_i denotes the bias of i^{th} hidden neuron, and $\phi(\cdot)$ denotes log-sigmoid activation function $\phi(n) = \frac{1}{1+e^{-n}}$. The output function at one output node can be expressed as

$$y_k = \sum_{i=1}^{L} w_{ik}z_i + \mu_k, \quad k = 1, \cdots, K \tag{3}$$

where w_{ik} denotes the weight of k^{th} output associated with i^{th} hidden neuron, and μ_k denotes the bias of k^{th} output. Now, let us consider the number of elementary operations included in (2) and (3) to compute TC. For a single hidden neuron, the number of addition/subtraction, multiplication/division, and exponential function are $N_A = N+1, N_M = N+1, N_\phi = 1$, respectively. The number of operations associated to a single output node are $N_A = L, N_M = L$. Taken into account there are L hidden neurons and K output nodes in total, the number of operations for the full network are $N_A = (N+K+1)L, N_M = (N+K+1)L, N_\phi = L$, which eventually result in the total time cost in (1) by multiplying constants $K_i, i = A, M, \phi$. □

2) Time Complexity of a RBF Network: Given a radial basis function (RBF) network with a single hidden layer whose activation functions are radial basis functions and output functions are linear functions, the time complexity (TC) for output generation can be formulated as

$$\begin{aligned} \text{TC} &= N_A \cdot K_A + N_M \cdot K_M + N_\phi \cdot K_\phi \\ &= L(2N+K-1) + L(N+K+3)K_M + LK_\phi. \end{aligned} \tag{4}$$

3) Time Complexity of a ReLU Network: Given a rectified-linear unit (ReLU) network with rectified-linear units in the single hidden layer and linear functions in the output layer, the time complexity (TC) for output generation can be formulated as

$$\begin{aligned} \text{TC} &= N_A \cdot K_A + N_M \cdot K_M \\ &= L(N+K+1) + L(N+K)K_M. \end{aligned} \tag{5}$$

Remark. Computation of TC for (4) and (5) is similar to that for (1), and hence is omitted here for brevity. □

4) Time Complexity of an ESN: Given a fully-connected echo state network (ESN) with hyperbolic tangent functions as reservoir activation functions in the hidden layer and linear functions as output functions in the output layer, the time complexity (TC) for output generation can be formulated as

$$\begin{aligned} \text{TC} &= N_A \cdot K_A + N_M \cdot K_M + N_\phi \cdot K_\phi \\ &= ((N+K+L)L + (N+L-1)K) \\ &\quad + ((N+K+L+3)L + (N+L)K)K_M + 2LK_\phi. \end{aligned} \tag{6}$$

Remark. Here we discuss a universal architecture of ESN as shown in Fig. 1. w^I, w^R, w^B are the weights of inputs, reservoir activation states and output feedbacks, respectively. w^{OI}, w^{OR} are the weights of readouts from input and reservoir layer, respectively, which need to be trained. Note that only a few of them are indicated in the figure for clarity. $\phi(\cdot)$ denotes hyperbolic tangent function $\phi(n) = \tanh(n) =$

Operation	Algorithm	Complexity	Constant K_i $(n=32, k=32)$
Addition/Subtraction	Basic	$O(n)$	$K_A = 1$
Multiplication	3-way Toom-Cook multiplication	$O(n^{1.465})$	$K_M = n^{0.465} = 5.01$
Division	Newton-Raphson division	$O(n^{1.465})$	$K_M = n^{0.465} = 5.01$
Exponentiation	Exponentiation by squaring	$O(k \cdot n^{1.465})$	$K_E = k \cdot n^{0.465} = 160.34$
Exponential function	Taylor series	$O(n^{0.5} \cdot n^{1.465})$	$K_\phi = n^{0.5} \cdot n^{0.465} = 28.34$
Square root	Newton's method	$O(n^{1.465})$	$K_M = n^{0.465} = 5.01$

TABLE II

TIME COMPLEXITY OF DIFFERENT OPERATIONS

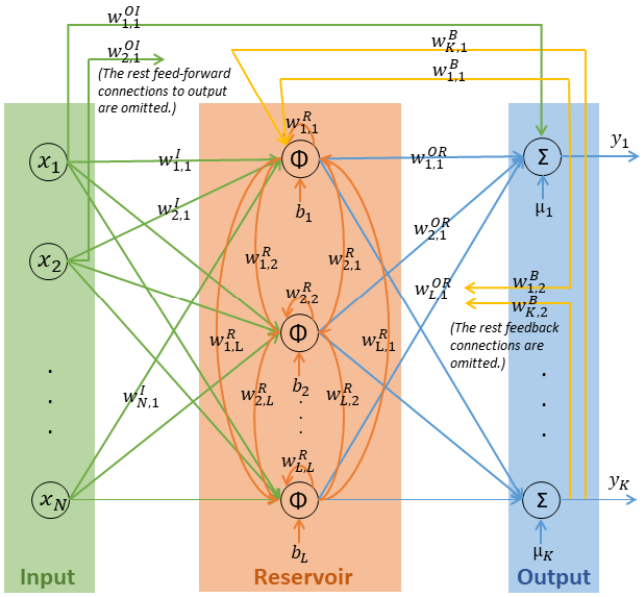

Fig. 1. A universal ESN architecture, which consists of an input layer, a reservoir (or hidden) layer and an output (or readout) layer. The colors — green, orange, blue and yellow, indicate connections from input layer to reservoir and output layer, connections within the reservoir, connections from reservoir to output layer, and output feedback connections from output layer to reservoir, respectively.

$\frac{e^{2n}-1}{e^{2n}+1}$ and $\sum$ denotes sum operation. The reservoir neuron activation of ESN given in Fig. 1 can be expressed as

$$z_i(n+1) = (1-\alpha)z_i(n) + \alpha\phi\Big(\sum_{j=1}^{N} w_{ji}^{I} x_j + \sum_{t=1}^{L} w_{ti}^{R} z_t(n) + \sum_{k=1}^{K} w_{ki}^{B} y_k(n) + b_i\Big), \quad i = 1, \cdots, L \tag{7}$$

where $z(n)$ denotes reservoir activation state at time instant $n \in \mathbb{Z}$, $\alpha \in (0, 1]$ denotes leaking rate [26] which is assumed to be equal to unity for the given architecture, and b_i denotes bias for each hidden node. It should be noted that (7) is not the unique expression of reservoir activation since certain terms are optional such as the leaky integration $\big($i.e., $(1-\alpha)z_i(n)\big)$ and the output feedback term $\big($i.e., $\sum_{k=1}^{K} w_{ki}^{B} y_k(n)\big)$, hence one can choose to use different variations in applications. The output function at one output node can be expressed as

$$y_k(n) = \sum_{i=1}^{L} w_{ik}^{OR} z_i(n) + \sum_{j=1}^{N} w_{jk}^{OI} x_j + \mu_k, \quad k = 1, \cdots, K \tag{8}$$

where μ_k denotes bias for each output node. To compute TC, the number of operations for a single reservoir neuron are $N_A = N+K+L, N_M = N+K+L+3, N_\phi = 2$ and those for a single output node are $N_A = N+L-1, N_M = N+L$. Therefore, the total number of operations taken into account L reservoir neurons and K output nodes are $N_A = (N+K+L)L+(N+L-1)K, N_M = (N+K+L+3)L+(N+L)K, N_\phi = 2L$, which yield (6) by multiplying constants $K_i, i = A, M, \phi$. □

III. DYNAMIC MODELING

Dynamic modeling is of great importance to MBC techniques since it lays the foundation and paves the road for control synthesis. Mathematical modeling has been one of the most ubiquitous methods for years provided that the system to be controlled is well studied and easy to model. Unfortunately, this is not true for complex systems such as UAVs because of inertial coupling, advanced aerodynamics effects, not to mention a variety of disturbances from the environment. To cope with these issues, ANN-based models have been developed and used thanks to their function approximation and learning ability, also named as *data-driven* or *black-box approach*. By collecting sufficient informative data and relying on data science techniques, ANN-based models can be trained without too much effort bearing almost no physical knowledge, which greatly eases the modeling process and reduces the computational cost for mathematical formulation of complex dynamics. Besides data-driven approach, there also exists another branch of modeling methods called *hybrid approach* which combines mathematical modeling with data-driven approach. Such method aims to (i) improve the performance of ANN-based model with the aid of physical model; or (ii) enhance ANN's interpretability by incorporating physical knowledge into modeling. Consequently, generalization capability of hybrid model may be improved, meanwhile complete ignorance about model behavior could be avoided. In this section, the existing literature focusing on dynamic modeling for UAVs is reviewed, categorized into data-driven and hybrid approach. Note that papers working on both modeling and control aspects will be surveyed in Section IV.

A. Modeling with Data-driven Approach

Data-driven approach, as introduced previously, refers to ANN which is purely constructed from data without physical knowledge incorporated. Depending on the architecture, the ANN-based models appeared in the literature can be classified into feed-forward, recurrent, and *mixed*. Mixed models refer to those combine both feed-forward and recurrent networks (explained in Section II) in one single model. Note that these models are sometimes called "hybrid" in some literature, however, we find it more appropriate to name them "mixed" to distinguish from hybrid model we defined ahead. In the sequel, the review is elaborated divided into offline and online modeling.

1) Offline Modeling: In [30], [31], mixed models of supervised neural networks are proposed for modeling of a mini-helicopter. Motivated by [32], a series of context-neurons for storing previous states are connected with a two-layer feed-forward neural network, forming a feedback loop to add recurrent feature to the mixed ANN-based model. Besides, two types of training architectures, namely daisy chain (or cascade) and decoupled (or parallel), are discussed and compared since the dynamic system of helicopter presents two distinct subsystems (radio-signal-to-attitude and attitude-to-position). Simulation results show daisy chain architecture has better performance due to the fact that the error propagated from the attitude network (model of subsystem 1) is taken into account and corrected in the position network (model of subsystem 2). A comparative study is also carried out to examine the performance of MLP and RBF used for feed-forward network. The results reveal that MLP outperforms RBF network for global approximation but falls behind for local approximation. However, the number of context-neurons are picked empirically in the paper which has a direct influence on the training time. For instance, their choice leads to 102 inputs for feed-forward network, which eventually requires about 18 hours for offline training given 60 hidden neurons used in MLP for both attitude and position networks.

In [33], black-box system identification using ESN with real flight data is performed for a quadrotor. Thanks to reservoir computing (RC) [25], the training process of ESN is facilitated since the traditional training methods for RNN such as back-propagation through time (BPTT) and real time recurrent learning (RTRL) are not required [34]–[36]. A single ESN model is trained offline with initially 100 echo states and spectral radius of 0.1 chosen empirically. Then, an evolutionary algorithm named covariance matrix adaptation evolution strategy (CMA-ES) is used to optimize some of the parameters of ESN in order to reduce the mean square error (MSE) produced at the testing stage, which finally leads to 127 echo states and spectral radius of 0.9191. From simulation, the optimized ESN shows imperceptible results compared with real flight data.

In [37], a novel ReLU network model is used for dynamic modeling of an aerobatic helicopter presented with details about parameter initialization and optimization methods. Three baseline models are described to make comparisons with the proposed model given real data, namely linear acceleration model, linear lag model and quadratic lag model. The proposed ReLU model is a combination of quadratic lag model and a two-layer simple ReLU network. Since the proposed model is non-convex unlike the three baseline models, it is trained by two steps: First, least-squares regression; Second, stochastic gradient descent (SGD). Simulation results show the proposed ReLU model improves 58% overall in root mean square (RMS) acceleration prediction over state-of-the-art methods. The training time takes less than 1 hour on a 6-core Intel i7 server with 32GB RAM for 2500 hidden units and 10 past samples of (lag) horizon.

In [38], the problem is investigated that whether an ANN-based dynamic model can be employed for control synthesis for trajectories different than those used for training. Similar to [37], two simple two-layer ReLU networks are used respectively to learn linear and angular acceleration components of a quadrotor offline, however, without augmenting quadratic lag model. The two network models have 100 hidden neurons for each and are trained by resilient backpropagation learning algorithm. Coupled with linear quadratic regulator (LQR) and proportional-derivative (PD) control for trajectory tracking, experiment shows even simple ANN architecture is capable of generalizing the dynamics beyond the training data to a satisfactory accuracy, hence can be further used for control purposes.

In [39], a modular deep recurrent neural network (MODERNN) framework [40] is used for MIMO modeling of a quadrotor in the presence of noise and ground effect. The MODERNN framework is proposed intentionally to address the problem of vanishing/exploding gradient in deep neural networks [41]. To reduce the computational complexity, three separate MODERNNs are trained with Levenberg-Marquardt (LM) method, each of dimension 3×10, 3×8, and 4×20, where the first number is the number of layers and the second is the number of neurons inside each layer. It takes approximately 24 hours for each training on an i7 core machine with 16GB of RAM. Simulation results show good learning capability of a complex quadrotor model.

2) Online Modeling: In [42], a comparison of an offline and online neural network architecture for the identification of a fixed-wing UAV is drawn. In particular, two decoupled networks representing lateral and longitudinal dynamics are employed with compared to a single network model. To facilitate online training, small fixed batches of input and output data are used, empirically ranging from 5 to 10. Both decoupled networks have two feed-forward layers with 4 hidden neurons and a maximum of 3 past inputs used for the prediction of the present output. From simulation results, it can be seen that online model is more adaptive to changes in the inputs to the system while offline model has a smoother approximation. However, for a complex nonlinear system such as UAV, it is expected to use both online and offline model to achieve better performance. For instance, a well trained offline model can be used as a reference model and the online model can be used to aid the autopilot.

B. Modeling with Hybrid Approach

Hybrid approach is usually referred to as modeling using ANNs in conjunction with mathematical model, particularly first principle model (FPM). In a broader sense, however, it refers to any ANN-based models that directly incorporate (explicit) physical knowledge into training process, which unfortunately, has not been fully developed for UAV applications yet (will be further discussed in Section V). The existing literature on hybrid approach presented subsequently focuses only on offline modeling.

1) Offline Modeling: In [43], a hybrid model for quadrotor in a multi-step prediction scenario is proposed. Different from the above modeling focusing on single-step prediction, multi-step prediction problem seeks an accurate estimate of the system output over the same time horizon given an input sequence. Long-short-term-memory (LSTM) networks in conjunction with a mathematical motion model of the quadrotor are proposed, in which LSTMs are initialized following process given in [44] for accurate system states prediction. Two configurations, namely hybrid-series and hybrid-parallel, and the use of tapped delay lines (TDLs) are discussed. Simulation results show the proposed approach, especially hybrid-parallel model, outperforms first principle model as well as full black-box model. However, the network specifications are not presented in detail which poses difficulties to understand computational complexity. Besides, according to the paper, for full black-box model training, velocity and body rates are decoupled as two MIMO subsystems to be learned by two separate LSTM networks to avoid divergence. Nonetheless, the velocity network is trained on the actual body rates rather than predicted body rates from body rate network. This can be actually viewed as the decoupled architecture in [31] where it has been already pointed out that daisy chain architecture outperforms decoupled architecture as the error propagated from the first network is very likely to be corrected in the second network. Hence, it would be more convincing if the authors can compare hybrid architecture with state-of-the-art daisy chain architecture.

In [28], different identification techniques are compared for control-oriented modeling for quadrotor, including parametric techniques (ARX, ARMAX, and OE), ANNs (RBF networks) and neuro-fuzzy inference systems. In particular, they show the hybridization of them (i.e., parametric technique with ANN or neuro-fuzzy in cascade connection) provides models with the better balance between accuracy and complexity than if they are individually applied. Moreover, they also discuss the influence of the training dataset partition on the final model, which proves that ANN with online learning strategies has almost always smaller error than that of the offline model and hence is less sensitive to dataset partition. Unfortunately, it is not shown in this paper the performance of recurrent networks and that of online learning in hybridization.

IV. CONTROL TECHNIQUES

In this section, the focus of literature review is on the use of ANN for controller design, different from Section III where attention is particularly paid to dynamic modeling. It is noted that most of the literature applies adaptive ANN-based elements to deal with uncertainties, which can be regarded as adaptive control technique. However, for the sake of classification, we categorize the existing work based on the main control technique adopted for flight control.

A. Feedback Linearization

In [45], an adaptive output feedback control design is proposed for command augmentation system (CAS) for an aircraft. Two RBF networks with sigma-pi units are used for both offline model inversion and online adaptation to cancel out model inversion error, with 21 RBF units and 40 sigma-pi units and the total number of weights of 840 for each. Since the weights of offline network appear linearly, a standard linear least square approximation method is used as the training method. A stable weights adjustment rule for the online neural network is also derived, which ensures that all of the signals in the loop are uniformly bounded and that the weights of the online network tend to constant values. Simulation results show that online adaptive network is able to cancel out model inversion error without full knowledge of uncertainties under a high-g, fixed throttle turn maneuver. A similar extended work refers to [46] in which σ-modification is used and more simulation results are given to show the capabilities of dealing with parametric uncertainties and unmodeled dynamics, however, for a Van der Pol oscillator.

In [47], a high-performance tracking controller is designed for an autonomous helicopter using feedback linearization with an ANN-based adaptive element accounting for inversion error. Pseudo control hedging is particularly used to protect adaptive element from actuator saturation nonlinearities and from inner-outer-loop interaction. A feedforward neural network with single hidden layer is used with 5 hidden neurons. Network parameters are updated online using a stable adaptive law derived from Lyapunov stability analysis, which guarantees signal boundedness of all associated variables through analytical proof. Both simulation and flight tests show satisfactory tracking performance for various maneuvers. However, the potential of ANN-based adaptive element is not investigated, for instance, the flight in which flight dynamics change moderately significant.

In [48], a hybrid adaptive control method is proposed for stability recovery of damaged aircraft operating in off-nominal flight conditions under single damage. A direct adaptive control augmentation to cancel out model inversion error is realized by a single-hidden-layer sigma-pi neural network. The hybrid (implicit-explicit) adaptive control incorporates an explicit parameter identification based on an adaptive law derived from the Lyapunov stability method or a recursive least-square method to estimate the true plant dynamics, which is then used for dynamic inversion control. By doing this, the dynamic inversion error that direct

adaptive ANN has to compensate for is reduced, which avoids saturating control authority and exciting unmodel dynamics due to aggressive learning (high learning rate) of direct adaptive ANN. Simulation results show good ability of damage effect aerodynamic modeling and control under single damage.

In [49], a concurrent-learning adaptive controller with feedback linearization is proposed for a helicopter to improve weight convergence properties and tracking performance by alleviating the rank-1 condition on weight updates in adaptive control. Inclusion of concurrent learning is the main contribution of this paper, with the use of a similar baseline control as presented in [47], that is a single-hidden layer neural network with 8 hidden neurons used as online adaptive element to reduce model inversion error. Such modification makes the controller able to adapt using current as well as stored data without affecting the responsiveness of the adaptive law to current data. Since the choice of baseline adaptive law and the projection matrix does not affect the stability properties of the concurrent-learning adaptive law, the proposed methodology can be further generalized to other adaptive schemes. From simulation and flight test results, improved tracking performance is observed for both repeated forward-step maneuvers and aggressive trajectory tracking maneuver.

In [50], two MLPs are trained for a feedback linearization controller design of a fixed-wing UAV of which the offline MLP replaces the inverse model required in feedback linearization while the online MLP is to account for inversion error. Both of them update the network parameters through back-propagation. From software-in-the-loop (SIL) simulation, where simulated data are used for online training at the moment, satisfactory tracking results are achieved for a step command in roll rate under some sensor noise. However, the details of MLP configurations are not elaborated such as number of neurons and layers. Moreover, the trained MLP and online MLP is the same object as described in this paper. Such architecture can be actually regarded as an online MLP with pre-determined weights and biases, which might require more theoretical proof or realistic tests to validate its stability and performance.

In [51], a nonlinear 7dof of control reconfigurable model for F-16 aircraft is developed, which is then used to develop an explicit model following direct adaptive controller with adaptive neural networks and using nonlinear dynamic inversion and control allocation (CA). The adaptive neural network is designed to cope with inversion error online by Lyapunov stability analysis, with a two layer design having 50 hidden neurons. Simulation results show notable performance under mass properties changes and loss of an aileron. The main advantage of the proposed approach is incorporation of changes or deviations in the plant on/off-flight while ensuring stability, manoeuvrability, and performance. Besides, it relaxes the need to develop extensive and precise fault detection and isolation (FDI) algorithms.

In [52], ESN is used for both offline and online training in the control design of a fixed-wing UAV. The offline training realizes model inversion required for feedback linearization while online training is performed to reduce the inversion errors. Since ESN does not require back-propagation for training, Weiner-Hopf method for linear regression is used which significantly reduces training time. Simulation results show improved tracking performance of bank angle command. However, no theoretical proof for system stability is provided. No comparative studies are carried out to show the promise of ESN over other feed-forward or recurrent networks. Besides, system performance under uncertainties is not considered for the proposed control architecture.

B. Sliding Mode Control

In [53], a double-loop integral sliding mode control (IntSMC) along with RBF networks are developed for position and attitude tracking of a quadrotor subject to disturbances and parametric uncertainties. RBF networks are trained online using an adaptive law based on Lyapunov approach to estimate and cancel out uncertainties. The proposed controller shows faster convergence of the state variables to their desired values under disturbances and uncertainties. Comparisons with other control techniques are also given from simulation. However, unmodeled dynamics are not considered.

C. Backstepping Control

In [54], backstepping technique along with ANNs are used for flight control of a helicopter. Two-layer online MLPs with 15 hidden neurons are used to estimate some unknown nonlinearities as well as physical parameters. Analytical analysis of stability is provided using the Lyapunov theory for online adaptation of the network parameters. Simulation results show good performance of the proposed controller even with a sudden mass change perturbation.

V. Discussion: Challenges and Opportunities

A. Classification and Comparison

Table III shows a general classification of reviewed literature on dynamic modeling for UAVs. As surveyed in Section III, the previous studies are categorized into data-driven and hybrid approach (as shown in the row) and offline and online modeling (as shown in the column). It is straightforward to see that most of them rely on offline data-driven approach. This is mostly because the training time for offline modeling is not critical, which is quite appealing for practical applications to train ANNs with a large number of data for accuracy purposes. Nonetheless, whenever system dynamics change, the performance of offline model degrades since nothing can be done to adjust the parameters to capture those changes in the dynamics in real time. From this perspective, an online model (or a combination of online and offline model) is highly preferable, especially for nonlinear complex systems.

Besides, data-driven approach is known as black-box approach, which has been unceasingly criticized for the lack of interpretability. Though from Table III we see that hybrid models have been proposed which attempt to improve

the model accuracy by exploiting data information from a mathematical model in series, only little literature works in this direction and no online solution has been developed so far. More importantly, the issue on interpretability is still not intentionally dealt with as no explicit knowledge is directly incorporated. Motivated by this, it is worth mentioning a recent study [55] in which physics-guided neural network (PGNN) framework is proposed for modeling aiming to make ANNs more interpretable by means of incorporating an explicit physical law in the cost function as well as using simulated data from a mathematical model for network training. Unlike in the majority of previous studies, where simplicity and accuracy are the only two criteria for ANN's performance, the modified cost function included with a physical law proposed in this paper is optimized for simplicity, accuracy and *physical consistency*, which yields a model that has a good balance between computational complexity, model accuracy and generalization capability. Though the PGNN-based model is proposed for offline SISO modeling of lake temperature with feed-forward architecture, it shows satisfactory preliminary results and deserves more investigation for UAV applications. To conclude, we believe that online hybrid modeling will outperform all the existing dynamic modeling due to the aforementioned reasons. Hence, it is one of our future research topics.

	Data-driven			**Hybrid**
	Feed-forward	**Recurrent**	**Mixed**	
Offline	[37], [38], [42]	[33], [39], [40]	[30], [31]	[28], [43]
Online	[42]	N/A	N/A	N/A

TABLE III
LITERATURE CLASSIFICATION OF UAV DYNAMIC MODELING

Table IV illustrates the classification of reviewed literature on UAV flight control, which is categorized based on control objectives. Several observations can be made from the existing studies. First, most of the ANN-based adaptive elements have update mechanisms of weight matrices and biases through adaptive laws, which are intended to provide theoretical proof for closed-loop stability and weight convergence. As for ANN-based adaptive element updated using data science approach, there is a lack of theoretical development. Second, using ANNs in control to cope with uncertainties is one of the major reasons for the employment of ANNs, however, very few studies push the simulation results towards experimental ones to validate the effectiveness in the real-world situation. Third, there is a preference for selection of feed-forward networks in the design, most probably due to their simple architecture that facilitates the theoretical analysis. However, it is widely acknowledged that recurrent networks are more powerful since feedback connections give them a temporal learning ability which feed-forward networks do not possess. Hence, a comparative study or analysis might be appreciated to support the choice of ANN architecture in each design and the future development of recurrent networks is also anticipated. Fourth, as pointed out in [48], significant modeling error will lead to aggressive learning due to the high learning rate, which might further result in saturation of control authority and excitement of unmodeled dynamics. This stresses the importance of architecture design for control systems where the capability of proposed solution should be clarified and validated, for example, what type of uncertainties can be dealt with and what is the safe and reliable operational range of the controller. Lastly, it is also expected to enhance the interpretability of ANN-based solution in control as that in dynamic modeling.

To improve tracking performance	[47], [49], [53]
To handle uncertainties	[45], [53], [54]
To provide fault tolerance	[48], [51]
To verify feasibility of the methods	[50], [52]

TABLE IV
LITERATURE CLASSIFICATION OF UAV FLIGHT CONTROL

B. Future Work

Based on the analyses of the literature surveyed in this paper, we envisage a novel control architecture for UAV flight control using MBC techniques combined with ANNs for future studies. The control scheme is briefly illustrated in Fig. 2, which in essence applies adaptive ANNs with feedback linearization technique motivated by [45]. Differently, an online inverse model is used for model inversion which forms a quasi-linear plant in real time to avoid significant inversion error and aggressive learning, motivated by [48]. The functionality of each block of the controller is explained as follows: (i) Reference model is used to specify desired handling qualities; (ii) Linear dynamic compensator is included to shape system response; (iii) Adaptive control augmentation is aimed to reduce inversion error and deal with parametric uncertainties, unmodeled dynamics and disturbances and noise; and (iv) PGNN-based inverse model is designed to cancel out inherent nonlinearities and yield a quasi-linear plant.

Compared with [45], [48], the main anticipated advancement of the proposed architecture lies in the use of PGNN framework for online inverse modeling (i.e., yielding a hybrid online model), which is intended to improve the generalization capability as well as the model accuracy. On the development of the PGNN-based inverse model, there are several research questions that are of particular interest to the authors. First, it is worth investigating if the use of a recurrent architecture can empower the PGNN framework since a feed-forward architecture is arbitrarily chosen in [55]. Second, the use of PGNN framework for MIMO modeling needs to be explored. Third, computational complexity should be studied for the PGNN-based model to verify the feasibility of real-time implementation. In Section II, we only derive the time complexity of output generation

for some ANNs, while issues on online training time as well as space complexity are still missing. Fourth, according to [55], the PGNN-based model is trained through data science approach, hence theoretical development on system stability and weight convergence is needed when the model is embedded in the control architecture. Last but not least, it is also essential to verify if the PGNN-based inverse model can be employed for control synthesis, leading to a robust solution.

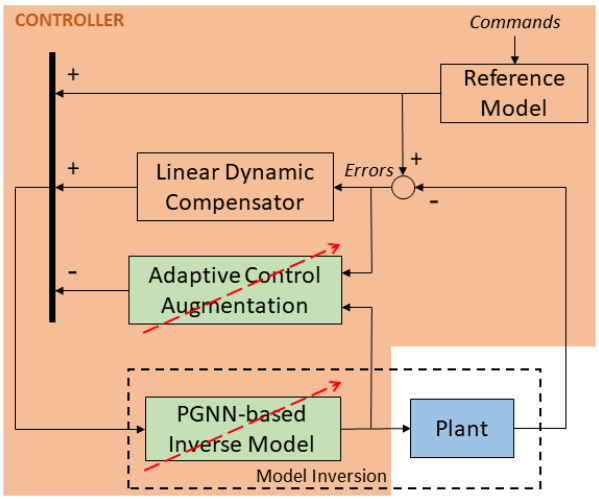

Fig. 2. Proposed control architecture for UAV flight control. Green blocks with red arrows represent online adaptive ANN-based solutions.

VI. CONCLUSIONS

In this survey, we thoroughly analyze the reasons why ANNs are now playing an increasingly crucial role in MBC systems. To implement a real-time control system, the time complexity of certain ANNs is also addressed. A literature review on MBC techniques combined with ANNs for flight control applications of UAVs is presented in detail. Despite the promising results achieved in the existing work, there are still opportunities for further enhancements in dynamic modeling and control techniques. Consequently, we envisage a novel control architecture which attempts to obtain an optimal control solution by meticulously combining ANNs and MBC techniques. Several challenging research questions are also posed for future development.

ACKNOWLEDGMENT

This work is partially supported by an NSF Grant, CMMI-DCSD-1728454, TIM JOL "Crab", the Siebel Energy Institute, and Compagnia di San Paolo.

REFERENCES

[1] H. Voos, "Nonlinear control of a quadrotor micro-uav using feedback-linearization," *2009 IEEE International Conference on Mechatronics*, pp. 1–6, 2009.

[2] C. Nicol, C. Macnab, and A. Ramirez-Serrano, "Robust adaptive control of a quadrotor helicopter," *Mechatronics*, vol. 21, no. 6, pp. 927 – 938, 2011.

[3] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Experimental model predictive attitude tracking control of a quadrotor helicopter subject to wind-gusts," *18th Mediterranean Conference on Control and Automation, MED'10*, pp. 1461–1466, 2010.

[4] R. Xu and U. Ozguner, "Sliding mode control of a quadrotor helicopter," *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 4957–4962, 2006.

[5] A. Das, F. L. Lewis, and K. Subbarao, "Backstepping approach for controlling a quadrotor using lagrange form dynamics," *Journal of Intelligent and Robotic Systems*, vol. 56, pp. 127–151, 2009.

[6] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "An underactuated h infinity control strategy for a quadrotor helicopter," *2009 European Control Conference (ECC)*, pp. 3845–3850, 2009.

[7] F. Santoso, M. A. Garratt, and S. G. Anavatti, "State-of-the-art intelligent flight control systems in unmanned aerial vehicles," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 613–627, 2018.

[8] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, p. 6461, 2007.

[9] J. S. Shamma and J. R. Cloutier, "Gain-scheduled missile autopilot design using linear parameter varying transformations," *Journal of guidance, Control, and dynamics*, vol. 16, no. 2, pp. 256–263, 1993.

[10] K. S. Tsakalis and P. A. Ioannou, *Linear time-varying systems: control and adaptation*. Prentice-Hall, Inc., 1993.

[11] P. A. Ioannou and J. Sun, *Robust adaptive control*, vol. 1. PTR Prentice-Hall Upper Saddle River, NJ, 1996.

[12] M. Sadraey and R. Colgren, "Robust nonlinear controller design for a complete uav mission," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, p. 6687, 2006.

[13] S. Fekri, M. Athans, and A. Pascoal, "Issues , progress and new results in robust adaptive control," 2006.

[14] P. P. Yip and J. K. Hedrick, "Adaptive dynamic surface control: a simplified algorithm for adaptive backstepping control of nonlinear systems," *International Journal of Control*, vol. 71, no. 5, pp. 959–979, 1998.

[15] K. J. Hunt, D. Sbarbaro, R. Żbikowski, and P. J. Gawthrop, "Neural networks for control systemsa survey," *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992.

[16] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, and P. Campoy, "A review of deep learning methods and applications for unmanned aerial vehicles," *Journal of Sensors*, vol. 2017, 2017.

[17] Y. Jiang, C. Yang, J. Na, G. Li, Y. Li, and J. Zhong, "A Brief Review of Neural Networks based Learning and Control and their Applications for Robots," vol. 2017, pp. 1–30, 2017.

[18] K. Hornik, M. Stinchcomb, and H. White, "Multilayer feedforward networks are universal approximator," *IEEE Transactions on Neural Networks*, vol. 2, 01 1989.

[19] G. Cybenko, "Approximations by superpositions of a sigmoidal function," *Mathematics Control Signals Systems*, vol. 2, 1989.

[20] D. Lazer, R. Kennedy, G. King, and A. Vespignani, "The parable of google flu: Traps in big data analysis," *Science (New York, N.Y.)*, vol. 343, pp. 1203–5, 03 2014.

[21] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity. 1943.," *Bulletin of mathematical biology*, vol. 52 1-2, pp. 99–115; discussion 73–97, 1990.

[22] M. H. Beale, M. T. Hagan, and H. B. Demuth, "Neural network toolbox user's guide," *The Mathworks Inc*, 1992.

[23] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *AISTATS*, 2011.

[24] H. Khodabandehlou and M. S. Fadali, "Echo state versus wavelet neural networks: Comparison and application to nonlinear system identification," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 2800 – 2805, 2017. 20th IFAC World Congress.

[25] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," in *NIPS*, 2002.

[26] M. Lukoševičius, "A practical guide to applying echo state networks," in *Neural networks: Tricks of the trade*, pp. 659–686, Springer, 2012.

[27] K. Greff, R. K. Srivastava, J. Koutnx00EDk, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, pp. 2222–2232, 2017.

[28] J. E. Sierra and M. S. Peñas, "Modelling engineering systems using analytical and neural techniques: Hybridization," *Neurocomputing*, vol. 271, pp. 70–83, 2018.
[29] D. J. Struik, *A source book in mathematics, 1200-1800*, vol. 11. Harvard University Press, 1969.
[30] R. S. Martin, A. Barrientos, P. Gutiérrez, and J. del Cerro, "Unmanned aerial vehicle (uav) modelling based on supervised neural networks," *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2497–2502, 2006.
[31] R. S. Martín, A. Barrientos, P. Gutiérrez, and J. D. Cerro, "Neural networks training architecture for uav modelling," *2006 World Automation Congress*, pp. 1–6, 2006.
[32] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.
[33] A. Vargas, M. Ireland, and D. Anderson, "System identification of multi-rotor uavs using echo state networks," 2015.
[34] B. A. Pearlmutter, "Learning state space trajectories in recurrent neural networks," *International 1989 Joint Conference on Neural Networks*, pp. 365–372 vol.2, 1988.
[35] F. J. Pineda, "Generalization of back propagation to recurrent and higher order neural networks," in *NIPS*, 1987.
[36] P. J. Werbos, "Backpropagation through time: What it does and how to do it," 1990.
[37] A. Punjani and P. Abbeel, "Deep learning helicopter dynamics models," *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3223–3230, 2015.
[38] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine, and C. J. Tomlin, "Learning quadrotor dynamics using neural network for flight control," *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 4653–4660, 2016.
[39] N. Mohajerin and S. L. Waslander, "Modelling a quadrotor vehicle using a modular deep recurrent neural network," *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 376–381, 2015.
[40] N. Mohajerin and S. L. Waslander, "Modular deep recurrent neural network: Application to quadrotors," *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1374–1379, 2014.
[41] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.
[42] V. R. Puttige and S. G. Anavatti, "Comparison of real-time online and offline neural network models for a uav," *2007 International Joint Conference on Neural Networks*, pp. 412–417, 2007.
[43] N. Mohajerin, M. Mozifian, and S. L. Waslander, "Deep learning a quadrotor dynamic model for multi-step prediction," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2454–2459, 2018.
[44] N. Mohajerin and S. L. Waslander, "State initialization for recurrent neural network modeling of time-series data," *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2330–2337, 2017.
[45] B. S. Kim and A. J. Calise, "Nonlinear flight control using neural networks," *Journal of Guidance, Control, and Dynamics*, vol. 20, no. 1, pp. 26–33, 1997.
[46] A. J. Calise, N. Hovakimyan, and M. Idan, "Adaptive output feedback control of nonlinear systems using neural networks," *Automatica*, vol. 37, no. 8, pp. 1201–1211, 2001.
[47] S. K. Kannan and E. N. Johnson, "Adaptive trajectory based control for autonomous helicopters," 2002.
[48] N. T. Nguyen, K. S. Krishnakumar, J. Kaneshige, and P. Nespeca, "Dynamics and adaptive control for stability recovery of damaged asymmetric aircraft," 2006.
[49] G. V. Chowdhary and E. N. Johnson, "Theory and flight-test validation of a concurrent-learning adaptive controller," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 2, pp. 592–607, 2011.
[50] S. Bhandari, A. Raheja, D. Tang, K. Ortega, O. Dadian, and A. Bettadapura, "Nonlinear control of uavs using multi-layer perceptrons with off-line and on-line learning," *2014 American Control Conference*, pp. 2875–2880, 2014.
[51] D. T. Ocaña, H.-S. Shin, and A. Tsourdos, "Development of a nonlinear reconfigurable f-16 model and flight control systems using multilayer adaptive neural networks," *IFAC-PapersOnLine*, vol. 48, no. 9, pp. 138–143, 2015.
[52] O. Dadian, S. Bhandari, and A. Raheja, "A recurrent neural network for nonlinear control of a fixed-wing uav," in *American Control Conference (ACC), 2016*, pp. 1341–1346, IEEE, 2016.
[53] S. Li, Y. Wang, J. Tan, and Y. Zheng, "Adaptive rbfnns/integral sliding mode control for a quadrotor aircraft," *Neurocomputing*, vol. 216, pp. 126–134, 2016.
[54] T. Madani and A. Benallegue, "Adaptive control via backstepping technique and neural networks of a quadrotor helicopter," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 6513–6518, 2008.
[55] A. Karpatne, W. Watkins, J. Read, and V. Kumar, "Physics-guided neural networks (pgnn): An application in lake temperature modeling," *arXiv preprint arXiv:1710.11431*, 2017.