JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 34, No. 2, March–April 2011

Theory and Flight-Test Validation of a Concurrent-Learning Adaptive Controller

Girish V. Chowdhary* and Eric N. Johnson†
Georgia Institute of Technology, Atlanta, Georgia 30332

DOI: 10.2514/1.46866

Theory and results of flight-test validation are presented for a novel adaptive law that concurrently uses current as well as recorded data for improving the performance of model reference adaptive control architectures. This novel adaptive law is termed *concurrent learning*. This adaptive law restricts the weight updates based on stored data to the null-space of the weight updates based on current data for ensuring that learning on stored data does not affect responsiveness to current data. This adaptive law alleviates the rank-1 condition on weight updates in adaptive control, thereby improving weight convergence properties and improving tracking performance. Lyapunov-like analysis is used to show that the new adaptive law guarantees uniform ultimate boundedness of all system signals in the framework of model reference adaptive control. Flight-test results confirm expected improvements in performance.

Nomenclature

A	=	state matrix for a linear system
a	=	sigmoidal function activation potential
$\bar{a}$	=	maximum sigmoidal function activation potential
B	=	input matrix for a linear system
b_W, b_V	=	neural network biases
e	=	tracking error
f	=	function representing the vehicle dynamics
$\hat{f}$	=	function representing approximate inversion model
K_p, K_d	=	proportional and derivative linear gains
m, n	=	dimension of the input and the state
n_1, n_2, n_3	=	number of input-layer, hidden-layer, and output-layer neurons
P	=	positive-definite solution to the Lyapunov equation
p	=	number of stored data points
p, q, r	=	aircraft roll rate, pitch rate, and yaw rate in rad/s
Q	=	positive-definite matrix appearing in the Lyapunov equation
r	=	neural network training signal formed by a linear transformation of tracking error
t	=	time
u, v, w	=	aircraft velocity along the body X, Y, and Z axes, ft/s
V, W	=	matrices containing neural network hidden- and output-layer weights
V^*, W^*	=	matrices containing ideal neural network weights
x	=	state vector
$\bar{x}$	=	input to neural network
Z	=	matrix formed by stacking the columns of V and W
z	=	input to sigmoidal activation function
α	=	variable used to bound the neural network input
Γ_V, Γ_W	=	adaptive law learning rates for W and V matrices
Δ	=	unknown model error function
δ	=	control input
ϵ	=	neural network approximation bound
$\bar{\epsilon}$	=	largest neural network approximation bound
κ	=	e-modification scaling term
ν	=	pseudo control
ν_{ad}	=	output of the adaptive element
ν_{pd}	=	output of the proportional–derivative compensator
ν_{rm}	=	output of the reference model
σ	=	neural network sigmoidal activation function

Subscript

c	=	commanded variable

Superscripts

i, j, k	=	numerical variables

I. Introduction

ADAPTIVE control has been extensively studied for aerospace applications. Many active research directions exist: for example, Lewis [1], Kim and Lewis [2], and Patiño et al. [3] have developed methods in adaptive control for the control of robotic arms. Calise et al. [4], Johnson et al. [5–8], Kannan [9], and others have developed model reference adaptive control (MRAC) methodology for control of unmanned aerial systems (UASs). Hovakimyan et al. [10] studied output feedback adaptive control and Cao and Hovakimyan [11] have developed the paradigm of $\mathcal{L}_1$ adaptive control. Lavretsky and Wise [12], Nguyen et al. [13], and others have extended direct adaptive control methods to fault-tolerant control and developed techniques in composite/hybrid adaptive control. The increasing interest in adaptive control stems from its ability to handle changes in system dynamics and to mitigate uncertainty. These abilities allow a relatively direct extension of adaptive control to fault-tolerant control.

Many of these approaches use a parameterized model, e.g., a neural network (NN), to capture the plant uncertainty. The parameters of the model are tuned online using an adaptive law and are often referred to as adaptive weights. The existence of an ideal (albeit unknown) set of weights that guarantee a sufficiently accurate parameterization of the plant uncertainty is an inherent assumption in approaches using such parametric models. The convergence of adaptive weights to these ideal values is a problem of considerable interest in adaptive control. If the online adaptive control laws are indeed able to drive the adaptive weights to the ideal values, then the tracking performance of the adaptive controller should improve significantly. Furthermore, in the framework of MRAC, such accurate parameterization will allow the linear tracking-error dynamics to

Received 25 August 2009; revision received 1 October 2010; accepted for publication 3 October 2010.

*Research Assistant, Ph.D. Candidate, School of Aerospace Engineering; Girish.Chowdhary@gatech.edu. Member AIAA.

†Associate Professor, School of Aerospace Engineering. Member AIAA.

dominate, allowing the use of powerful linear control metrics for making inferences about the robustness of the tracking-error dynamics. However, most adaptive laws in MRAC suffer from local adaptation; that is, the weights take on values such that the adaptive element dominates the uncertainty pointwise in time, rather than approximating it uniformly over the entire domain. The three major contributors that may cause local adaptation are as follows:

1) Tracking error is used for adaptation rather than modeling error. Modeling error in the system (that is, the difference between the output of the parametric model of the uncertainty and the true plant uncertainty) cannot be directly used for training, since the true plant uncertainty is not known. Hence, most adaptive laws are designed to update the weights in order to minimize an instantaneous tracking-error cost (e.g., $V = e^T e$) by using a gradient-descent-type method (e.g., [1,14]) rather than to approximate the true uncertainty.

2) Weight updates in traditional adaptive laws occur based only on the instantaneous data. This leaves the adaptive law with access only to local information for forming a global approximation of the true plant uncertainty.

3) Due to the rank-1 condition [5], the rank of gradient-descent-based adaptive laws that use only instantaneous data for adaptation is always, at most, one. Conceptually, the rank-1 condition indicates that the adaptive law is constrained to searching adaptive weights only along one direction in the underlying vector space at that instant. It is well known that in order to guarantee parameter convergence when using rank-1 laws the system states are required to be persistently excited [14,15]. Particularly, Boyd and Sastry [16] have shown that in order to guarantee weight convergence for a class of MRAC adaptive laws, the exogenous reference input must have as many spectral lines as the number of unknown parameters. This condition relates to the persistency of excitation in the reference input. For flight vehicle control, however, persistently exciting inputs cannot always be used, as they may result in waste of fuel and may cause undue stress on the aircraft. Hence, adaptive flight controllers with the rank-1 condition are unlikely to exhibit weight convergence in practice.

Ioannou and Sun [14] suggested the use of a σ modification term that adds damping to the adaptive law to counter weight drift when persistently exciting inputs cannot be used, and Narendra and Annaswamy [17] suggested the e modification term to scale the added damping by using the norm of the tracking error. These methods are primarily designed to ensure boundedness of adaptive weights and do little to ensure weight convergence to ideal values. One way to overcome the problem of local adaptation is to incorporate memory in the adaptive law by using stored information. The most commonly used approach for incorporating memory is the use of a momentum term [18–20]. The momentum term scales the most recent weight update in the direction of the last weight update. This speeds up the convergence of weights when in the vicinity of local minima and slows the divergence. This modification is heuristic and results only in a modest improvement. Furthermore, it does not address the issue of susceptibility to local training due to high reliance on instantaneous data. Another common approach is the use of a forgetting factor that can be tuned to indicate the degree of reliance on past data [21]. This approach suffers from the drawbacks that the forgetting factor is difficult to tune, and an improper value can adversely affect the performance of the adaptive controller. Particularly, a smaller value of the forgetting factor indicates higher reliance on recent data, which could lead to a local parameterization, while a larger value of the forgetting factor indicates higher reliance on past data, which could lead to sluggish adaptation performance. Patiño et al. [3] suggested the use of a bank of NNs trained around different operating conditions as a basis for the space of all operating conditions. The required model error was then calculated by using a linear combination of the outputs of these different NNs. To overcome the shortcomings of online training algorithms, Patiño et al. also suggested that the bank of NNs be adapted offline using recorded data. The reliance on offline training makes this approach inappropriate for adaptive flight applications. To improve weight convergence, Volyanskyy et al. [22] proposed Q modification to baseline MRAC-based NN adaptation law. Q modification adds a modification term to the baseline adaptive law that uses the integral of the tracking error over a finite time window. This method is computationally intensive; furthermore, its performance depends on the size of the window over which the integral is performed, which is normally selected to be a short time interval in the instantaneous past. It is noted that to the best of the authors' knowledge, no current method exists that can incorporate data points stored anytime across the flight envelope in the adaptive law.

This work presents a novel adaptive control method that is aimed at improving the convergence of adaptive weights in MRAC. This novel method is termed as concurrent learning. Concurrent learning uses current as well as stored data concurrently for adaptation and has an associated stability proof. This ability allows the adaptive law to incorporate memory consisting of specifically selected and stored data points across the flight envelope. Weight updates based on stored data are restricted to the null-space of the weight updates based on current data in concurrent learning. This key feature ensures that concurrent-learning adaptive laws alleviate the rank-1 condition and ensures that adaptation on stored data does not affect the responsiveness of the adaptive element to sudden changes. Furthermore, concurrent learning allows the adaptive element to train on multiple specifically stored input-output data pairs and enables further processing of stored information to allow the use of direct training information. In this paper the stability of concurrent-learning adaptive laws is established in the sense of uniform ultimate boundedness (UUB) using Lyapunov-like analysis. Along with the theoretical results, this paper also presents flight-test results of the presented concurrent-learning adaptive controller onboard the Georgia Institute of Technology (Georgia Tech) GTMax rotorcraft UAS. The results show improved tracking-error performance and the removal of the rank-1 condition.

This paper begins by presenting a brief overview of approximate model-inversion NN-based adaptive control in Sec. II. In Sec. III the structure of the current learning law is discussed. In Sec. IV the concurrent-learning adaptive law is presented, along with the associated proof of its UUB. Sections V, VI, and VII are dedicated to the discussion of simulation and flight-test results of a concurrent-learning adaptive law.

II. Model Reference Adaptive Control Using Neural Networks

This section introduces the approximate dynamic inversion-based adaptive control architecture used in this paper.

A. Approximate Model-Inversion-Based Adaptive Control

Let $D_x \in \Re^n$ be a compact set, let $x, \dot{x} \in D_x$ be the known state and its time derivative, let $\delta \in \Re^m$ be the control input, and consider the following system describing the dynamics of an aircraft:

$$\ddot{x} = f(x, \dot{x}, \delta) \tag{1}$$

The function f is assumed to be continuously differentiable on D_x, and the control input is assumed to be bounded and piecewise-continuous. The conditions for the existence and uniqueness of the solution to Eq. (1) are assumed to be met. Let $\nu \in \Re^{n_3}$ denote the pseudo control input, which represents a desired $\ddot{x}$ that is expected to be achieved by the controller. Since the exact model in Eq. (1) is usually not available or not invertible, an approximate inversion model is introduced:

$$\nu = \hat{f}(x, \dot{x}, \delta) \tag{2}$$

It is assumed that the approximate inversion model is chosen such that it can be inverted to obtain δ for all pseudo control inputs ν. Hence,

$$\delta = \hat{f}^{-1}(x, \dot{x}, \nu) \tag{3}$$

This results in a model error of the form

$$\ddot{x} = \nu + \Delta(x, \dot{x}, \delta) \tag{4}$$

where the modeling error Δ is given by

$$\Delta(x, \dot{x}, \delta) = f(x, \dot{x}, \delta) - \hat{f}(x, \dot{x}, \delta) \tag{5}$$

A reference model f_{rm} can be chosen to characterize the desired response of the system:

$$\ddot{x}_{\text{rm}} = f_{\text{rm}}(x_{\text{rm}}, \dot{x}_{\text{rm}}, x_c) \tag{6}$$

where x_{rm} denotes the state of the reference model and x_c denotes the desired reference command. It is assumed that all the requirements for guaranteeing a unique and bounded solution to Eq. (6) are satisfied and that the reference command is piecewise-continuous.

Figure 1 depicts the control architecture for MRAC control discussed in this section. In the above discussion the effects of actuator nonlinearities is not considered, and it is assumed that the actual servo deflection is equal to the commanded input δ.

B. Model Tracking-Error Dynamics

A tracking control law consisting of the output of a linear feedback compensator ν_{pd}, a feedforward part $\nu_{\text{rm}} = \ddot{x}_{\text{rm}}$, and the output of the adaptive element ν_{ad} is proposed to have the following form:

$$\nu = \nu_{\text{rm}} + \nu_{\text{pd}} - \nu_{\text{ad}} \in \Re^{n_3} \tag{7}$$

For a second-order system, the proportional–derivative compensation term ν_{pd} in Eq. (7) is expressed by

$$\nu_{\text{pd}} = [\,K_p \quad K_d\,]e \tag{8}$$

where the reference model tracking error is defined as

$$e = \begin{bmatrix} x_{\text{rm}} - x \\ \dot{x}_{\text{rm}} - \dot{x} \end{bmatrix} \tag{9}$$

The model tracking-error dynamics are found by differentiating e and using Eqs. (4) and (7) [5,9]:

$$\dot{e} = Ae + B[\nu_{\text{ad}}(x, \dot{x}, \delta) - f(x, \dot{x}, \delta) + \hat{f}(x, \dot{x}, \delta)] \tag{10}$$

where

$$A = \begin{bmatrix} 0 & I \\ -K_p & -K_d \end{bmatrix}, \qquad B = \begin{bmatrix} 0 \\ I \end{bmatrix} \tag{11}$$

and both K_p and K_d are chosen such that A is Hurwitz. Let Δ represent the model error given by Eq. (5), then the error dynamics in Eq. (10) can be rewritten as

$$\dot{e} = Ae + B[\nu_{\text{ad}}(x, \dot{x}, \delta) - \Delta(x, \dot{x}, \delta)] \tag{12}$$

Based on the above equation, note that if $\nu_{\text{ad}}(x, \dot{x}, \delta) - \Delta(x, \dot{x}, \delta) = 0$ for all x, $\dot{x}$, and δ, then the tracking-error dynamics are reduced to an exponentially stable linear system. This would be the case if the model error is cancelled uniformly by ν_{ad}.

Since the matrix A is designed to be Hurwitz, for any positive-definite matrix $Q \in \Re^{2n \times 2n}$ a positive-definite solution to the Lyapunov equation $P \in \Re^{2n \times 2n}$ exists:

$$A^T P + PA + Q = 0 \tag{13}$$

C. Neural-Network-Based Adaptation

In this paper a single hidden-layer (SHL) NN is used as the adaptive element. Let $\bar{x}$ denote the input vector, let V denote the matrix containing the input-layer to hidden-layer synaptic weights, let W denote the matrix containing hidden-layer to output-layer synaptic weights, and let $\sigma(z)$ denote the output of the sigmoidal activation function vector with $z = V^T \bar{x} \in \Re^{n_2}$, then the SHL NN output ν_{ad} can be expressed in a compact form as

$$\nu_{\text{ad}}(W, V, \bar{x}) = W^T \sigma(V^T \bar{x}) \in \Re^{n_3} \tag{14}$$

Let n_1 denote the number of input-layer neurons, let n_2 denote the number of hidden-layer neurons, let n_3 denote the number of output-layer neurons, let $x_{in} \in \Re^{n_1}$ denote the system signals used for adaptation, and let $b_v \geq 0$ and $b_w \geq 0$ denote input biases that allow the thresholds θ_V and θ_W to be included in the weight matrices V and W, then the terms in Eq. (14) can be expressed as

$$\bar{x} = \begin{bmatrix} b_v \\ x_{in} \end{bmatrix} \in \Re^{(n_1+1)} \tag{15}$$

$$V = \begin{bmatrix} \theta_{v,1} & \cdots & \theta_{v,n2} \\ v_{1,1} & \cdots & v_{1,n2} \\ \vdots & \ddots & \vdots \\ v_{n1,1} & \cdots & v_{n1,n2} \end{bmatrix} \in \Re^{(n_1+1) \times n_2} \tag{16}$$

$$W = \begin{bmatrix} \theta_{w,1} & \cdots & \theta_{w,n_3} \\ w_{1,1} & \cdots & w_{1,n_3} \\ \vdots & \ddots & \vdots \\ w_{n_{2+1},1} & \cdots & w_{n_{2+1},n_3} \end{bmatrix} \in \Re^{(n_2+1) \times n_3} \tag{17}$$

The function $\sigma(z)$: $\Re^{n_2} \rightarrow \Re^{n_2+1}$ maps the elements z_j of the vector signal z in the following manner:

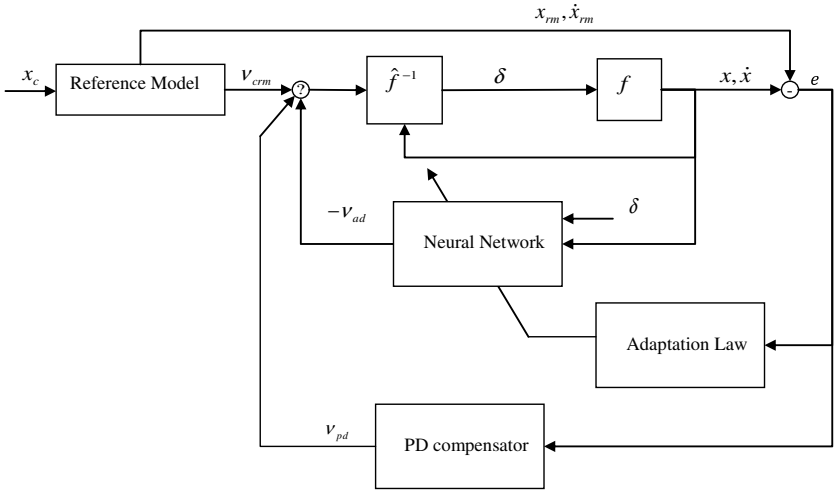

Fig. 1 Neural network adaptive control using approximate inversion model.

$$\sigma(z) = \begin{bmatrix} b_w \\ \sigma_1(z_1) \\ \vdots \\ \sigma_{n_2}(z_{n_2}) \end{bmatrix} \in \Re^{n_2+1} \tag{18}$$

The elements of σ consist of sigmoidal activation functions given by

$$\sigma_j(z_j) = \frac{1}{1+e^{-a_j z_j}} \in \Re \tag{19}$$

Further details on NN theory and MRAC can be found in [1,2,7,18–20].

III. Online-Learning NN Adaptive Control and Rank-1 Limitation

SHL NNs are considered to be universal function approximators [23]. That is, given an $\epsilon > 0$ (for all $\bar{x} \in D$, where D is a compact set), there exists a number of hidden-layer neurons n_2 and an ideal set of weights (W^*, V^*) that brings the NN output to within an ϵ neighborhood of the function approximation error Δ, with the largest such ϵ given by

$$\bar{\epsilon} = \sup_{\bar{x}\in D} \| W^{*T}\sigma(V^{*T}\bar{x}) - \Delta(x, \dot{x}, \delta) \| \tag{20}$$

Recall that the model tracking-error dynamics represented by Eq. (12) have a term that is linear in e. Hence, if A is chosen to be Hurwitz and if the adaptive law is such that $W(t) \to W^*$ and $V(t) \to V^*$, then by choosing a sufficient number of hidden-layer neurons, $\bar{\epsilon}$ can be made sufficiently small to allow the exponentially stable linear tracking-error dynamics to dominate. Therefore, convergence of weights to their ideal values not only ensures attractive stability properties of the tracking error, but also makes well established metrics in linear stability analysis available for ascertaining the robustness of the tracking controller.

Traditional adaptive laws in MRAC, including those in [1,2,5,12], are designed to update the adaptive weights such that $V = e^T e$ is minimized. The following theorem provides one such law in the framework of MRAC. In the following theorem Γ_W and Γ_V denote the positive-definite matrices containing the NN W and V matrix learning rates; κ denotes the e-modification scaling term; r, V, W, and $\bar{x}$ are as defined previously; and $\acute{\sigma}$ is the derivative of σ with respect to $\bar{x}$.

Theorem 1: Consider the system in Eq. (1) and the inverting controller of Eq. (3), the following online adaptive law guarantees that the tracking error e and the adaptive weights W and V stay uniformly ultimately bounded [2]

$$\dot{W} = -\Gamma_W(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})V^T\bar{x})e^T PB - k\|e\|\Gamma_W W \tag{21}$$

$$\dot{V} = -\Gamma_V \bar{x} e^T PBW^T\acute{\sigma}(V^T\bar{x}) - k\|e\|\Gamma_V V \tag{22}$$

Proof: This theorem is a special case of Theorem 2 (Sec. IV) when using only current data (i.e., $p = 0$). □

Consider now the following fact from linear algebra [24].

Fact 1: Let $u \in \Re^n$ and $v \in \Re^m$ be nonzero vectors, then the matrix $A = uv^T$, where $A \in \Re^{n\times m}$, is of rank 1.

Using the above fact, it can be seen that even though the NN weight adaptation laws of Eqs. (21) and (22) have a matrix form, they are always, at most, rank 1. Particularly, $\dot{W}$ is always, at most, a rank-1 matrix, since $\sigma \in \Re^{n_2+1}$ and $e^T PB \in \Re^{1\times n_3}$, and $\dot{V}$ is always, at most, rank 1, because $\Gamma_V \bar{x} \in \Re^{n_1+1}$ and $e^T PBW^T\acute{\sigma}(V^T\bar{x}) \in \Re^{1\times n_2}$. For these adaptive laws it is well known that convergence of adaptive weights to their ideal values is guaranteed only if the system states are made persistently exciting by choosing persistently exciting reference commands [14–16]. However, the use of persistently exciting reference commands is not always desirable in flight control, especially since such inputs may cause nuisance, waste of fuel, and undue stress on the aircraft. Therefore, rank-1 adaptive laws are unlikely to enjoy parameter convergence in practice.

IV. Overcoming the Rank-1 Limitation with Concurrent Learning on Past and Current Data

In this work a novel adaptive law is presented that overcomes the rank-1 condition by using current data with specifically selected and stored data concurrently for adaptation. The presented law is termed as concurrent learning. Let $\dot{W}_t$ and $\dot{V}_t$ denote any traditional adaptive laws evaluated using current data, and let $\dot{W}_b$ and $\dot{V}_b$ denote adaptive laws evaluated using stored data, then the concurrent-learning law takes the following form:

$$\dot{W}(t) = \dot{W}_t(t) + W_c(t)\dot{W}_b(t) \tag{23}$$

$$\dot{V}(t) = \dot{V}_t(t) + V_c(t)\dot{V}_b(t) \tag{24}$$

where $W_c(t)$ and $V_c(t)$ are orthogonal projection operators that project $\dot{W}_b$ into the null-space of $\dot{W}_t$ and $\dot{V}_b$ into the null-space of $\dot{V}_t$. Hence, the operators $W_c(t)$ and $V_c(t)$ restrict the weight updates based on stored data to the null-space of the weight updates based on current data, thereby ensuring that the magnitude of weight update in the direction of $\dot{W}_t(t)$ and $\dot{V}_t(t)$ remains unaffected. This enforced separation of training spaces ensures that the adaptation on past data does not affect the responsiveness of the adaptive law to current data, which may reflect sudden changes in the environment. Hence, concurrent-learning adaptive laws are not susceptible to sluggish adaptation performance due to overreliance on memory, a phenomenon that may be experienced when using forgetting-factor-based adaptation laws [21]. Furthermore, this enforced separation allows concurrent learning to use data stored anytime in the past. Hence, the stored data can be chosen to represent the full flight envelope of the vehicle. Finally, since the weight update based on the instantaneous data $(\dot{W}_t, \dot{V}_t)$ is always, at most, rank 1, learning on stored data can occur in an $n-1$ dimensional subspace, where n is the maximum possible rank of the appropriate weight matrix. The combined learning on stored and current data thus allows the concurrent-learning law to overcome the rank-1 condition.

A. Selection of Data Points for Concurrent Learning

The choice of the selected data points affects the rank and performance of the adaptive law. For a given $\epsilon_{\bar{x}} > 0$, a simple way to choose points that are sufficiently different from the last point stored is to enforce the following criterion:

$$\frac{\|\bar{x} - \bar{x}_p\|^2}{\|\bar{x}\|} \geq \epsilon_{\bar{x}} \tag{25}$$

where the subscript p denotes the index of the last data point stored. The above method ascertains that only those data points that are sufficiently different from the last data point are selected for storage. Let the subscript i denote the ith stored data point. Once the data points x_i are selected, the model error $\Delta(x, \dot{x}, \delta)$ of Eq. (5) can be directly estimated by noting that

$$\Delta(x, \dot{x}, \delta) = f(x_i, \dot{x}_i, \delta_i) - \hat{f}(x_i, \dot{x}_i, \delta_i) = \ddot{x}_i - \nu_i \tag{26}$$

Since ν_i is known, the above equation reduces the problem of estimating the model error Δ to that of estimating $\ddot{x}$. If an explicit measurement of $\ddot{x}$ is not available, $\ddot{x}$ can be estimated using an implementation of an optimal fixed-point smoother [25]. Optimal fixed-point smoothing uses all available measurements up to the current time T to arrive at an estimate of the system state at time t, where $0 \leq t \leq T$. Let $\ddot{\hat{x}}_i$ denote the smoothed estimate of $\ddot{x}_i$, then an estimate of the model error can be formed as follows:

$$\hat{\Delta}(x_i, \dot{x}_i) = \ddot{\hat{x}}_i - \nu_i \tag{27}$$

The Appendix describes the details of the fixed-point smoothing procedure for estimating $\ddot{\bar{x}}_i$. The residual signal that is used for training on stored data then becomes

$$r_{b_i} = W^T\sigma(V^T\bar{x}_i) - \hat{\Delta}(x_i, \dot{x}_i, \delta_i) \tag{28}$$

The residual signal in this form is the difference between the current estimate of the model error and a stored estimate of the model error and allows the use of direct training information in the update law. It is important to note here that fixed-point smoothing [25] uses a forward and a backward Kalman filter for improving the estimate of $\ddot{x}$. Hence, using fixed-point smoothing for estimating the model error will entail a finite time delay before r_{b_i} can be calculated for that data point. However, the enforced separation of training spaces in the concurrent learning ensures that this delay does not affect the instantaneous tracking performance of the controller.

B. Combined Instantaneous and Concurrent-Learning NN, Proof of Boundedness

In this section Lyapunov-like analysis is used to guarantee stability of a particular concurrent-learning adaptive law (Theorem 2) in the sense of UUB [26,27]. The adaptive law is presented in the framework of approximate model-inversion-based adaptive control described in Sec. II.

Let $\dot{W}_t$ and $\dot{V}_t$ be given by Eqs. (21) and (22). Let $\tilde{W} \triangleq W - W^*$ and $\tilde{V} \triangleq V - V^*$, where (W^*, V^*) are the ideal NN weights. Let $i \in \aleph$ denote the index of a stored data point $\bar{x}_i$. Furthermore, let $\ddot{\hat{x}}_i$ denote the fixed-point smoother estimate of $\ddot{x}_i$. Define the estimate of the model error for the ith data point as $\hat{\Delta}(x_i, \dot{x}_i, \delta_i) = \ddot{\hat{x}}_i - \nu_i$. Let $r_{b_i} = W^T\sigma(V^T\bar{x}_i) - \hat{\Delta}(x_i, \dot{x}_i, \delta_i)$, and consider the following assumptions:

Assumption 1: The norm of the ideal weights (W^*, V^*) is bounded by a known positive value:

$$0 < \|Z^*\|_F \le \bar{Z} \tag{29}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, and

$$Z \triangleq \begin{bmatrix} V & 0 \\ 0 & W \end{bmatrix}$$

.

Assumption 2: The external reference commands to the system x_c remain bounded.

Assumption 3: The reference model is chosen such that the states of the reference model remain bounded for a bounded reference command x_c.

Assumption 4: The NN approximation $\Delta(x, \dot{x}, \delta) = W^{*T}\sigma(V^{*T}\bar{x}) + \bar{\epsilon}$, holds on a compact domain D, which is sufficiently large and includes $\bar{x}(0)$.

Assumption 5: The model error of the system can be observed accurately for points sufficiently far in the past using optimal fixed-point smoothing. This assumption can be relaxed by accounting for inaccuracies in model error observability, which leads to an increase in the size of bounds on $\tilde{Z}$. In essence, this assumption enforces that $\ddot{\hat{x}}_i = \ddot{x}_i$ for all data points.

Lemma 1: Let

$$W_c = \left(I - \frac{(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})V^T\bar{x})(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})V^T\bar{x})^T}{(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})V^T\bar{x})^T(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})V^T\bar{x})}\right) \tag{30}$$

$$V_c = \left(I - \frac{\Gamma_V\bar{x}\bar{x}^T\Gamma_V}{\bar{x}^T\Gamma_V^T\Gamma_V\bar{x}}\right) \tag{31}$$

where I is an identity matrix of the appropriate dimension; σ, $\bar{x}$, and Γ_V are as defined in Eqs. (15), (21), and (22); then W_c and V_c are orthogonal projection operators projecting into the null-space of $\dot{W}_t$ and $\dot{V}_t$, respectively.

Proof: Note that W_c and V_c are idempotent and symmetric; hence, they are orthogonal projection operators [28]. The proof for showing that W_c, and V_c are projection operators into the null-space of $\dot{W}_t$ and $\dot{V}_t$ follows by noting that $W_c\dot{W}_t = 0$ and $V_c\dot{V}_t = 0$. □

Note that both the projection matrices are rank-deficient in only one direction, which is the direction of the weight update based on current data $[\dot{W}_t(t), \dot{V}_t(t)]$.

Theorem 2: Consider the system in Eq. (1), the inverting controller of Eq. (3), Assumptions 1–5, the NN training signal based on stored data given by $r_{b_i} = W^T\sigma(V^T\bar{x}_i) - \Delta(x_i, \dot{x}_i, \delta_i)$, SHL NN output ν_{ad} given by Eq. (14), and the structure of the adaptive law characterized by Eqs. (23) and (24), with W_c and V_c as in Eqs. (30) and (31). The following concurrent-learning adaptive law (32) and (33) guarantees that the tracking error e and the NN weight errors $\tilde{W}$ and $\tilde{V}$ are uniformly ultimately bounded:

$$\dot{W} = -\Gamma_W(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})V^T\bar{x})e^TPB - k\|e\|\Gamma_W W - W_c\Gamma_W\sum_{i=1}^{P}(\sigma(V^T\bar{x}_i) - \acute{\sigma}(V^T\bar{x}_i)V^T\bar{x}_i)r_{b_i}^T \tag{32}$$

$$\dot{V} = -\Gamma_V\bar{x}e^TPBW^T\acute{\sigma}(V^T\bar{x}) - k\|e\|\Gamma_V V - V_c\sum_{i=1}^{P}(\Gamma_V\bar{x}_i r_{b_i}^T W^T\acute{\sigma}(V^T\bar{x}_i)) \tag{33}$$

Proof: Begin by noting that for all V and $\bar{x}$ the sigmoidal activation function (19) and its derivatives are bounded, due to its exponential form, as follows [5,7]:

$$\|\sigma(V^T\bar{x})\| \le b_w + n_2 \tag{34}$$

$$\|\acute{\sigma}(V^T\bar{x})\| \le \bar{a}(b_w + n_2)(1 + b_w + n_2) = \bar{a}k_1k_2 \tag{35}$$

where $\bar{a}$ is the maximum activation potential in the sigmoidal activation functions, and $k_1 = b_w + n_2$ and $k_2 = 1 + b_w + n_2$ are constants defined for convenience.

The Taylor series expansion of the sigmoidal activation function about the ideal weights is given by

$$\sigma(V^{*^T}\bar{x}) = \sigma(V^T\bar{x}) + \left.\frac{\partial\sigma(s)}{\partial s}\right|_{s=V^T\bar{x}}(V^{*^T}\bar{x} - V^T\bar{x}) + \text{H.O.T.} \tag{36}$$

where H.O.T. represents the higher-order terms. A bound on H.O.T. can be found by rearranging Eq. (36) and letting $\tilde{Z} = Z - Z^*$, where Z is defined by Assumption 1:

$$\|\text{H.O.T.}\| \le \|\sigma(V^{*T}\bar{x})\| + \|\sigma(V^T\bar{x})\| + \|\acute{\sigma}(V^T\bar{x})\|\|\tilde{V}\|\|\bar{x}\| \le 2k_2 + \bar{a}k_1k_2\|\bar{x}\|\|\tilde{Z}\|_F \tag{37}$$

The error in the NN parameterization for a given state $[x, \dot{x}]$ can be written as

$$\nu_{\text{ad}}(x, \dot{x}, \delta) - \Delta(x, \dot{x}, \delta) = W^T\sigma(V^T\bar{x}) - W^{*^T}\sigma(V^{*T}\bar{x}) + \epsilon \tag{38}$$

Using the Taylor series expansion for the sigmoidal activation function from Eq. (36), this can be further expanded to

$$\nu_{\text{ad}}(x, \dot{x}, \delta) - \Delta(x, \dot{x}, \delta) = W^T\sigma(V^T\bar{x}) - W^{*^T}(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})\tilde{V}^T\bar{x} + \text{H.O.T.}) + \epsilon \tag{39}$$

$$\nu_{\text{ad}}(x, \dot{x}, \delta) - \Delta(x, \dot{x}, \delta) = \tilde{W}^T(\sigma(V^T\bar{x}) - \acute{\sigma}(V^T\bar{x})V^T\bar{x}) + W^T\acute{\sigma}(V^T\bar{x})\tilde{V}^T\bar{x} + w \tag{40}$$

where w is given by

$$w = \tilde{W}^T \acute{\sigma}(V^T \bar{x}) V^{*^T} \bar{x} - W^{*^T}(\text{H.O.T.}) + \epsilon \tag{41}$$

Bounds on w can now be obtained:

$$\|w\| \leq \|\tilde{W}^T\| \|\acute{\sigma}(V^T \bar{x})\| \|V^*\| \|\bar{x}\| + \|W^*\| \|\text{H.O.T.}\| + \bar{\epsilon} \tag{42}$$

which leads to

$$\|w\| \leq \bar{a} k_1 k_2 \bar{Z} \|\tilde{Z}\|_F \|\bar{x}\| + \bar{Z}(2k_1 + \bar{a} k_1 k_2 \|\bar{x}\| \|\tilde{Z}\|_F) + \bar{\epsilon} \tag{43}$$

Letting

$$c_0 = \bar{\epsilon} + 2\bar{Z} k_1, \qquad c_1 = \bar{a} k_1 k_2 \bar{Z} + \bar{Z}\, \bar{a}\, k_1 k_2 \tag{44}$$

we have that

$$\|w\| \leq c_0 + c_1 \|\bar{x}\| \|\tilde{Z}\| \tag{45}$$

To show the boundedness of the reference model errors and the NN weights a Lyapunov-like analysis is used [26]. A radially unbounded and positive-definite Lyapunov-like candidate [26,27] is

$$L(e, \tilde{W}, \tilde{V}) = \tfrac{1}{2} e^T P e + \tfrac{1}{2} \text{tr}\{\tilde{W}^T \Gamma_W^{-1} \tilde{W}\} + \tfrac{1}{2} \text{tr}\{\tilde{V}^T \Gamma_V^{-1} \tilde{V}\} \tag{46}$$

where tr{.} denotes the trace operator. Differentiating the Lyapunov candidate along the trajectory of system described by Eq. (10) results in

$$\begin{aligned} \dot{L}(e, \tilde{W}, \tilde{V}) = &-\tfrac{1}{2} e^T Q e + e^T P B (\nu_{\text{ad}} - \Delta) \\ &+ \text{tr}\{(\tilde{W}^T \Gamma_w^{-1} \dot{W})\} + \text{tr}\{(\tilde{V}^T \Gamma_v^{-1} \dot{V})\} \end{aligned} \tag{47}$$

Expanding the NN model parameterization error using Eq. (39) and adding and subtracting

$$\sum_{i=1}^{P} (\nu_{\text{ad}}(x_i, \dot{x}_i) - \Delta(x_i, \dot{x}_i))^T (\nu_{\text{ad}}(x_i, \dot{x}_i) - \Delta(x_i, \dot{x}_i)),$$
$$\text{tr}\{k\|e\| W \tilde{W}^T\}, \qquad \text{tr}\{k\|e\| V \tilde{V}^T\}$$

results in

$$\begin{aligned} \dot{L}(e, \tilde{W}, \tilde{V}) = &-\tfrac{1}{2} e^T Q e + e^T P B (\tilde{W}^T (\sigma(V^T \bar{x}) \\ &- \acute{\sigma}(V^T \bar{x}) V^T \bar{x}) + W^T \acute{\sigma}(V^T \bar{x}) \tilde{V}^T \bar{x} + w) \\ &+ \text{tr}\{\tilde{W}^T \Gamma_W^{-1} (\dot{W}_t + W_c \dot{W}_b)\} + \text{tr}\{\tilde{V}^T \Gamma_V^{-1} (\dot{V}_t + V_c \dot{V}_b)\} \\ &+ \sum_{i=1}^{P} (\nu_{\text{ad}}(x_i, \dot{x}_i) - \Delta(x_i, \dot{x}_i))^T (\nu_{\text{ad}}(x_i, \dot{x}_i) - \Delta(x_i, \dot{x}_i)) \\ &- \sum_{i=1}^{P} (\nu_{\text{ad}}(x_i, \dot{x}_i) - \Delta(x_i, \dot{x}_i))^T (\nu_{\text{ad}}(x_i, \dot{x}_i) - \Delta(x_i, \dot{x}_i)) \\ &+ \text{tr}\{k\|e\| W \tilde{W}^T\} - \text{tr}\{k\|e\| W \tilde{W}^T\} + \text{tr}\{k\|e\| V \tilde{V}^T\} \\ &- \text{tr}\{k\|e\| V \tilde{V}^T\} \end{aligned} \tag{48}$$

Using Eq. (40) to expand $\nu_{\text{ad}}(x_i, \dot{x}_i) - \Delta(x_i, \dot{x}_i)$ and collecting terms, the following terms can be set to zero:

$$\text{tr}\{\tilde{W}^T ((\sigma(V^T \bar{x}) - \acute{\sigma}(V^T \bar{x}) V^T \bar{x}) e^T P B + k\|e\| W + \Gamma_W^{-1} \dot{W}_t)\} = 0 \tag{49}$$

$$\text{tr}\{\tilde{V}^T (\bar{x} e^T P B W \acute{\sigma}(V^T \bar{x}) + k\|e\| V + \Gamma_V^{-1} \dot{V}_t)\} = 0 \tag{50}$$

$$\text{tr}\left\{\tilde{W}^T \left(\sum_{i=1}^{P} (\sigma(V^T \bar{x}_i) - \acute{\sigma}(V^T \bar{x}_i) V^T \bar{x}_i) r_{b_i}^T + \Gamma_W^{-1} W_c \dot{W}_b\right)\right\} = 0 \tag{51}$$

$$\text{tr}\left\{\tilde{V}^T \left(\sum_{i=1}^{P} \bar{x}_i r_{b_i}^T W^T \acute{\sigma}(V^T \bar{x}_i) + \Gamma_V^{-1} V_c \dot{V}_b\right)\right\} = 0 \tag{52}$$

This leads to

$$\dot{W}_t = -\Gamma_W (\sigma(V^T \bar{x}) - \acute{\sigma}(V^T \bar{x}) V^T \bar{x}) e^T P B - k\|e\| \Gamma_W W \tag{53}$$

$$\dot{V}_t = -\Gamma_V \bar{x} e^T P B W^T \acute{\sigma}(V^T \bar{x}) - k\|e\| \Gamma_V V \tag{54}$$

$$W_c \dot{W}_b = -\sum_{i=1}^{P} (\Gamma_W (\sigma(V^T \bar{x}_i) - \acute{\sigma}(V^T \bar{x}_i) V^T \bar{x}_i) r_{b_i}^T) \tag{55}$$

$$V_c \dot{V}_b = -\sum_{i=1}^{P} (\Gamma_V \bar{x}_i r_{b_i}^T W^T \acute{\sigma}(V^T \bar{x}_i)) \tag{56}$$

Noting that orthogonal projection operators are idempotent and multiplying both sides of Eqs. (55) and (56) by W_c and V_c, respectively, results in

$$W_c \dot{W}_b = -W_c \sum_{i=1}^{P} (\Gamma_W (\sigma(V^T \bar{x}_i) - \acute{\sigma}(V^T \bar{x}_i) V^T \bar{x}_i) r_{b_i}^T) \tag{57}$$

$$V_c \dot{V}_b = -V_c \sum_{i=1}^{P} (\Gamma_V \bar{x}_i r_{b_i}^T W^T \acute{\sigma}(V^T \bar{x}_i)) \tag{58}$$

Summing Eq. (53) with Eq. (57), and Eq. (54) with Eq. (58), we arrive at the required training law of Theorem 2. The derivative of the Lyapunov candidate is now reduced to

$$\begin{aligned} \dot{L}(e, \tilde{W}, \tilde{V}) \leq &-\frac{1}{2} \lambda_{\min} Q \|e\|^2 + \|e^T P B\| \|w\| \\ &- \sum_{i=1}^{P} \|r_{b_i}\|^2 + \sum_{i=1}^{P} \|r_{b_i}\| \|w_i\| - k\|e\| \|\tilde{Z}\|_F^2 + k\|e\| \|\tilde{Z}\|_F \bar{Z} \end{aligned} \tag{59}$$

Using previously computed bounds,

$$\begin{aligned} \dot{L}(e, \tilde{W}, \tilde{V}) \leq &-\frac{1}{2} \lambda_{\min}(Q) \|e\|^2 + \|e\| \|PB\| (c_0 + c_1 \|\tilde{Z}\|_F \|\bar{x}\|) \\ &- \sum_{i=1}^{P} \|r_{b_i}\|^2 + \sum_{i=1}^{P} \|r_{b_i}\| (c_0 + c_1 \|\tilde{Z}\|_F \|\bar{x}\|) - k\|e\| \|\tilde{Z}\|_F^2 \\ &+ k\|e\| \|\tilde{Z}\|_F \bar{Z} \end{aligned} \tag{60}$$

Hence, when $\lambda_{\min}(Q)$ and k are sufficiently large, $\dot{L}(e, \tilde{W}, \tilde{V}) \leq 0$ everywhere outside of a compact set. Therefore, the system states are bounded; hence, letting α denote some positive constant, the input to the NN can be bounded as follows:

$$\|\bar{x}\| \leq b_v + \alpha \tag{61}$$

With this definition, let $\hat{c}_1 = \bar{a} k_1 k_2 \bar{Z} + \bar{Z}\, \bar{a}\, k_1 k_2 (b_v + \alpha)$; hence, $\|w\| \leq c_0 + \hat{c}_1 \|\tilde{Z}\|$.

To see that the set is indeed compact, consider $\dot{L}(e, \tilde{W}\, \tilde{V}) \leq 0$ when

$$\|e\| \geq \frac{-a_0 + \sqrt{a_0^2 + 2\lambda_{\min}(Q) \left(-\sum_{i=1}^{P} \|r_{b_i}\|^2 + \sum_{i=1}^{P} \|r_{b_i}\| (c_0 + \hat{c}_1 \|\tilde{Z}\|_F)\right)}}{\lambda_{\min}(Q)} = b_e \left(\|\tilde{Z}\|, \lambda_{\min}(Q), \sum_{i=1}^{P} \|r_{b_i}\|\right) \tag{62}$$

where

$$a_0 = \|PB\|(c_0 + c_1\|\tilde{Z}\|_F) - k\|\tilde{Z}\|_F^2 + k\|\tilde{Z}\|_F\bar{Z}$$

or $\|e\| = 0$, $\|w_i\| = 0$, or $\|e\| \neq 0$,

$$\sum_{i=1}^{P} \|r_{b_i}\| \neq 0$$

and

$$\|\tilde{Z}\| \geq \frac{-b_0 + \sqrt{b_0^2 + 4k\|e\|\left(-\frac{1}{2}\lambda_{\min}(Q)\|e\|^2 + \|PB\|\|e\|c_0 - \sum_{i=1}^{P}\|r_{b_i}\|^2 + \sum_{i=1}^{P}\|r_{b_i}\|c_0\right)}}{2k\|e\|} = b_Z\left(\|e\|, \sum_{i=1}^{P}\|r_{b_i}\|\right) \tag{63}$$

where

$$b_0 = \|e\|\|PB\|\hat{c}_1 + \sum_{i=1}^{p}\|r_{b_i}\|\hat{c}_1 + k\|e\|\bar{Z}$$

or $\|e\| \neq 0$, $\|\tilde{Z}\| \neq 0$ and

$$\sum_{i=1}^{P}\|r_{b_i}\| \geq \frac{-(c_0 + \hat{c}_1\|\tilde{Z}\|_F) + \sqrt{(c_0 + c_1\|\tilde{Z}\|_F)^2 + 4d_0}}{2} = b_{r_b}(\|e\|, \|\tilde{Z}\|) \tag{64}$$

where

$$d_0 = -\tfrac{1}{2}\lambda_{\min}(Q)\|e\|^2 - k\|e\|\|\tilde{Z}\|_F^2 + k\|e\|\|\tilde{Z}\|_F\bar{Z} + \|e\|\|PB\|(c_0 + \hat{c}_1\|\tilde{Z}\|_F)$$

.

The curves represented by

$$b_e\left(\|\tilde{Z}\|, \lambda_{\min}(Q), \sum_{i=1}^{P}\|r_{b_i}\|\right), \qquad b_Z\left(\|e\|, \sum_{i=1}^{P}\|r_{b_i}\|\right)$$

$$b_{r_b}(\|e\|, \|\tilde{Z}\|)$$

are guaranteed to intersect. □

Remark 1: When a data point is added or removed, the discrete change in the Lyapunov function is zero. Hence, the Lyapunov candidate holds for any number of recorded data points. Furthermore, arbitrary addition or removal of data points does not affect the uniform ultimate boundedness properties.

Remark 2: The ultimate bound on the tracking error can be reduced by increasing $\lambda_{\min}(Q)$.

Remark 3: Note that if no concurrent points are stored, then the NN weight adaptation law reduces to that of Theorem 1. This indicates that the purely online NN weight adaptation method can be considered as a special case of the more general online and concurrent weight adaptation method of Theorem 2.

Remark 4: A key point to note is that the proof of Theorem 2 does not depend on specific forms of W_c and V_c as long as they are orthogonal projection operators mapping into the null-space of $\dot{W}_t$ and $\dot{V}_t$, respectively. Hence, similar results as those in Theorem 2 can be formed for other stable baseline laws and modifications, including sigma modification, adaptive loop recovery modification, and projection-operator-based modifications.

Corollary 1: If the concurrent-learning law of Theorem 2 is used in the control law of Eq. (3), then all plant states x of the system of Eq. (1) are uniformly ultimately bounded.

Proof: Considering the ultimate boundedness of e, $\tilde{W}$, and $\tilde{V}$ from Theorem 2, along with the boundedness of the reference model states (Assumption 3), and the definition of reference model tracking error in Eq. (9), the ultimate boundedness of the plant states x is immediate.

V. Demonstration of Concept for the Adaptive Control of an Inverted Pendulum

In this section a simple example is presented to illustrate the concept of concurrent-learning augmented adaptive control. Consider an inverted pendulum system described by the following equation:

$$\ddot{x} = \delta + \sin(x) - |\dot{x}|\dot{x} + 0.5e^{x\dot{x}} \tag{65}$$

where δ is the control input and x describes the angular position of the pendulum. The system in Eq. (65) is unstable, and the controller structure described in Sec. II is used to stabilize the system. The last three terms are regarded as unknown and result in a significant model error when the approximate inversion model is chosen to have the simple form: $\nu = \delta$.

It is assumed that a measurement for $\ddot{x}$ is not available and that the system outputs are corrupted by Gaussian white noise. Consequently, an optimal fixed-lag smoother is used to estimate the model error (26) for points sufficiently far in the past. The reference model desired dynamics are that of a second-order system. A cyclic history stack of five data points is used and the oldest data point is replaced by the most recently selected data point. Concurrent-learning point selection is based on Eq. (27) with $\epsilon_{\bar{x}} = 0.3$. An SHL NN with eight hidden nodes is used as the adaptive element.

Figure 2 shows the performance of the NN-based adaptive controller (without concurrent learning) for the plant in Eq. (65). The external reference command x_c consists of square waves commanded at regular intervals. No noticeable improvement in tracking-error performance is noted, even though the system tracks a repeated command. The forgetting nature of the adaptive law is further reflected in the periodic nature of the adaptive weights shown in Fig. 3.

Figure 4 shows the performance of the adaptive controller with concurrent-learning adaptive law of Theorem 2. It can be clearly seen that the tracking performance of the controller improves through repeated commands. This is easily ascertained by considering Fig. 5,

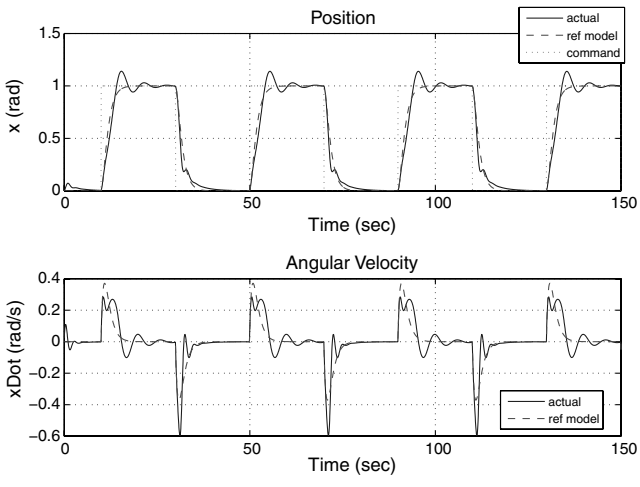

Fig. 2 Comparison of states without concurrent learning.

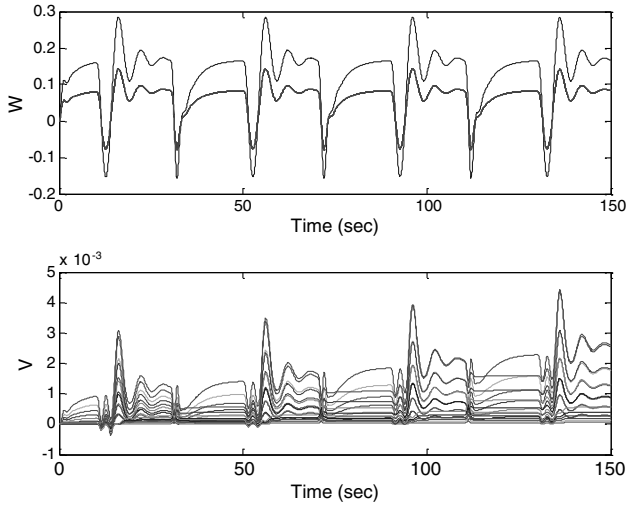

Fig. 3 Evolution of NN weights and W and V matrix without concurrent learning.

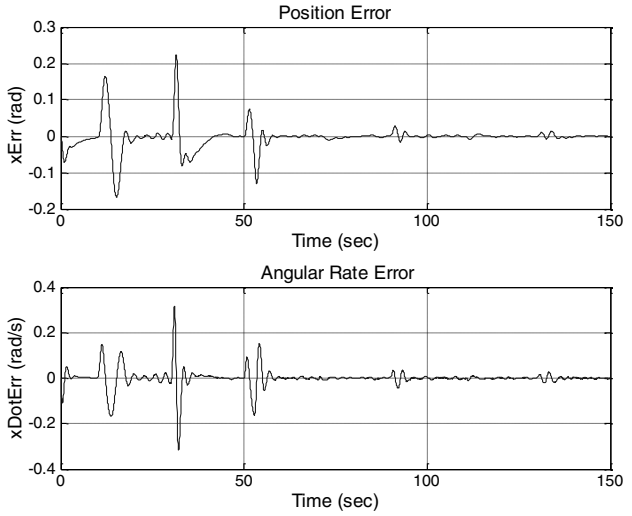

Fig. 5 Position and angular rate errors with concurrent-learning adaptive controller.

which explicitly shows the evolution of both position and angular velocity tracking error. To further characterize the impact of concurrent learning, the following is noted:

1) When the concurrent-learning controller of Theorem 2 is used, comparatively quick convergence of NN weights to almost constant values is observed, as depicted in Fig. 6. This behavior is in clear contrast with the periodic behavior of the weights when no concurrent learning is present. Furthermore, the weights attained a larger value when concurrent learning was used; however, the NN output was of comparable magnitude in both cases, except during transient peaks, where the concurrent-learning NN was better able to approximate the uncertainty.

2) Figure 7 plots the difference between the stored estimate of model error and the current estimate of model error (28) for each stored data point when using the concurrent-learning adaptive controller. The five lines in the figure correspond to the output of Eq. (28) for data points in the five slots of the cyclic history stack. The convergence of these terms indicates that the NN is concurrently adapting to the model error over various data points, indicating semiglobal model error parameterization.

3) The concurrent-learning adaptive law had a rank greater than unity. Figure 6 shows the resulting separation of NN weights when using concurrent-learning adaptive law. Furthermore, the weights are seen to approach constant values in that figure. On the other hand, when concurrent learning is not used, the weights do not approach

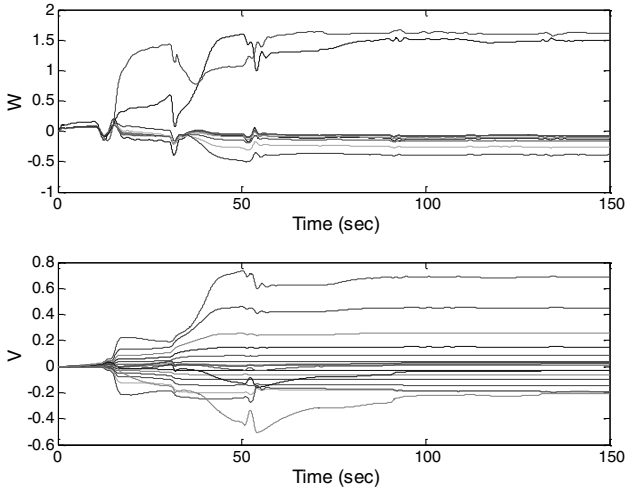

Fig. 6 Evolution of NN weights and W and V matrix with concurrent learning.

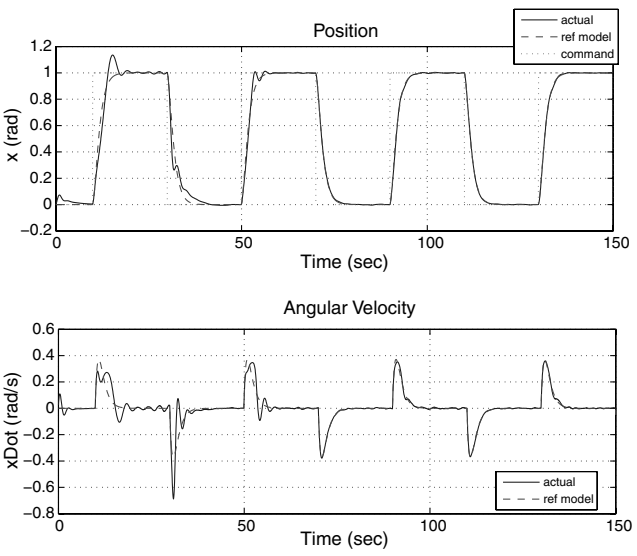

Fig. 4 Comparison of states with concurrent learning.

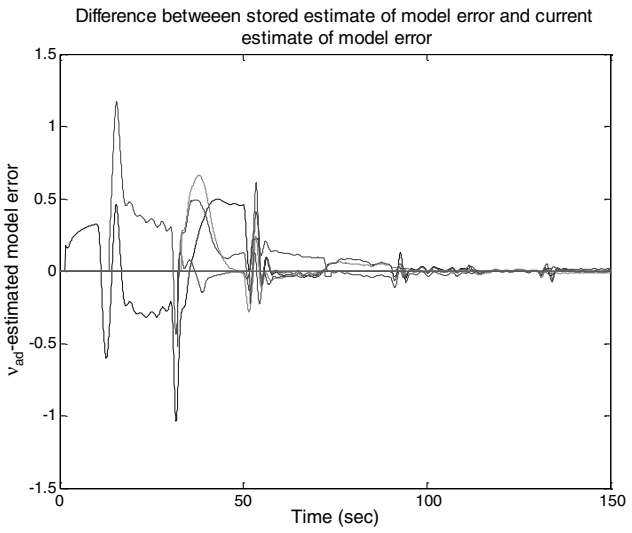

Fig. 7 Difference between stored estimate of model error and current estimate of model error for different data points with concurrent learning.

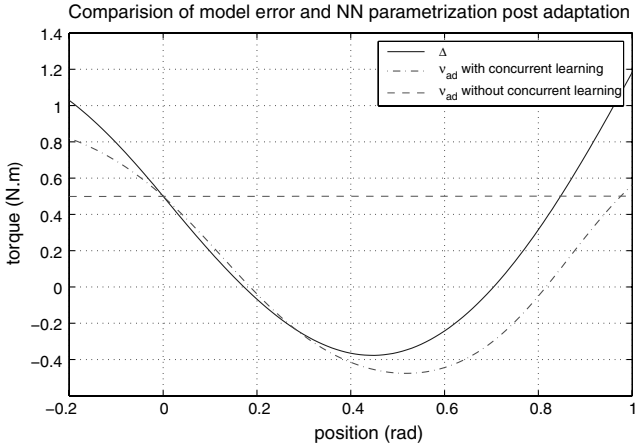

Fig. 8 NN postadaptation approximation of the unknown model error Δ as a function of state x.

Fig. 9 Georgia Tech GTMax, in autonomous landing approach.

constant values, as shown in Fig. 3. This suggests that alleviation of the rank-condition by concurrent-learning adaptive laws results in faster weight convergence and that concurrent-learning adaptive laws need less excitation in the system states for weight convergence.

4) Figure 8 compares the NN output ν_{ad} with the model error Δ as a function of the state x with NN weights set to the final weight values obtained at the end of the simulation. This plot indicates that the concurrent-learning adaptive controller of Theorem 2 is able to find a set of synaptic weights to form an approximation of the model error

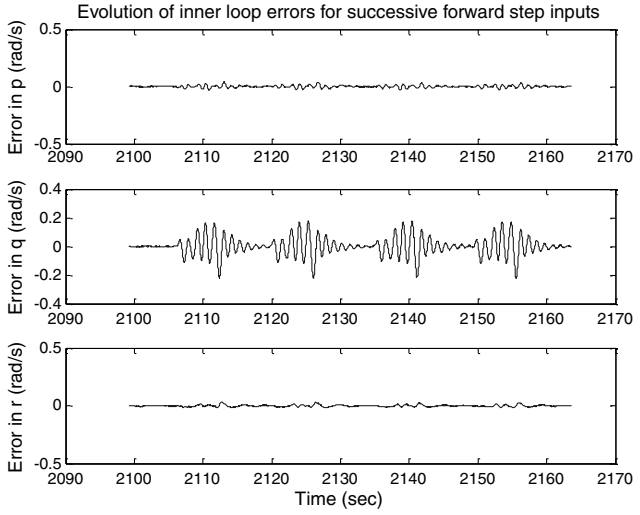

Fig. 10 Evolution of inner-loop errors for successive forward-step maneuvers without concurrent learning.

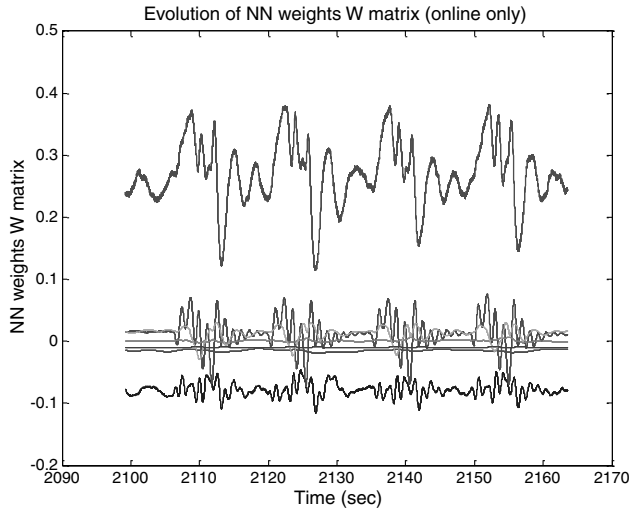

Fig. 11 Evolution of NN weights, W matrix, without concurrent learning.

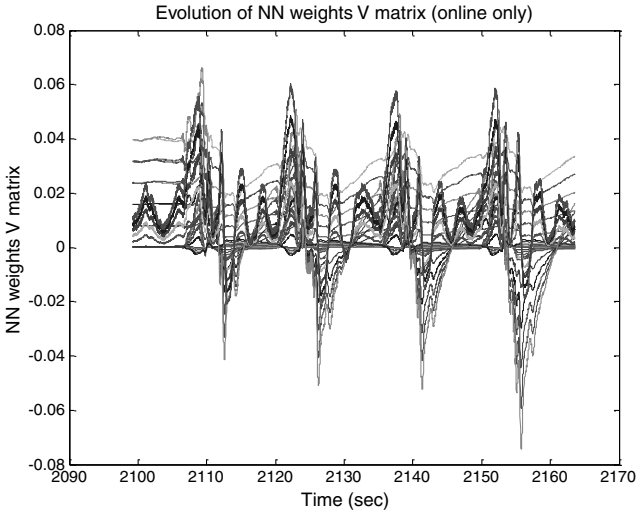

Fig. 12 Evolution of NN weights, V matrix, without concurrent learning.

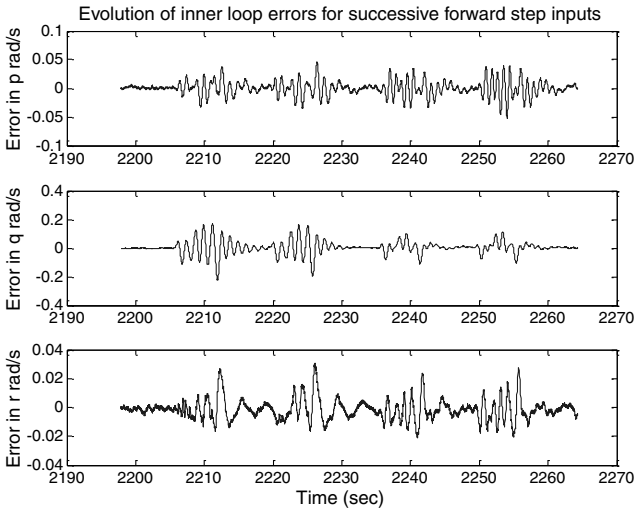

Fig. 13 Evolution of inner-loop error for successive forward-step maneuvers with concurrent learning.

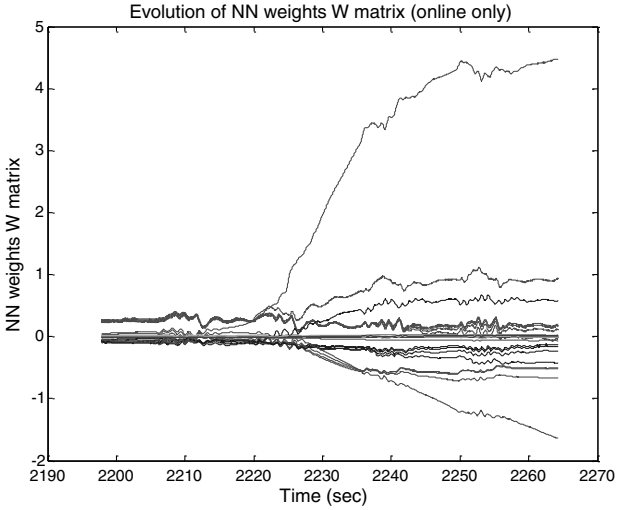

Fig. 14 Evolution of NN weights, *W* matrix, with concurrent learning.

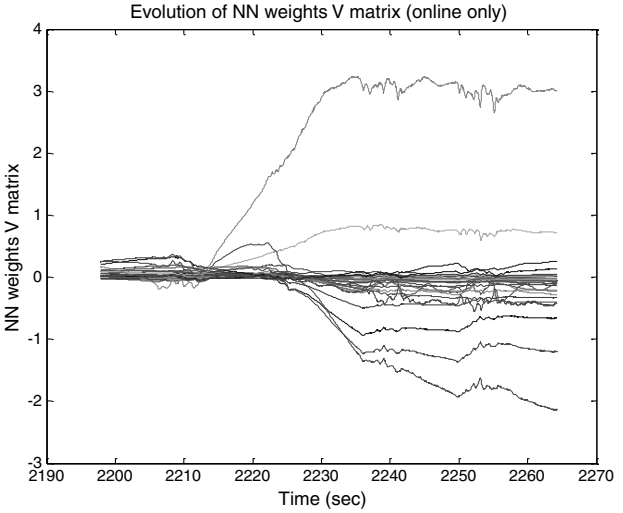

Fig. 15 Evolution of NN weights, *V* matrix, with concurrent learning.

over the range of the presented data (semiglobal model error parameterization). Figure 8 also shows output of the NN with the NN weights set to the final weight values obtained at the end of the simulation without concurrent learning. It can be seen that without concurrent learning the postadaptation NN approximation of the uncertainty is a straight line. This is expected, as the baseline adaptive law of Theorem 1 is designed to minimize the tracking error rather than approximate the uncertainty; hence, it results in dominating the uncertainty pointwise in time.

These effects in combination highlight the performance gain when using concurrent-learning adaptive controllers. Note that in order to make a fair comparison, the same controller parameters (including learning rates Γ_W and Γ_V, reference model parameters, e-modification gain κ, and linear gains K_p and K_d) were used, regardless of whether concurrent learning was used.

It is interesting to note that the inclusion of concurrent learning does not increase the computational requirements significantly. The increased computational load due to the inclusion of concurrent learning consists of 1) screening of current data to determine which points to store in the history stack, 2) fixed-point smoothing algorithm to estimate the model error of the stored data point using Eq. (26), and 3) evaluating the adaptive law of Theorem 2 over points stored in the history stack. The most significant of these is the fixed-point smoothing; however, this process does not necessarily need to update as often as the control output itself.

VI. Implementation on a High-Fidelity Flight Simulator

In this section simulation results of the presented controller on a high-fidelity simulation of the Georgia Tech GTMax UAS are presented. The GTMax is based on the Yamaha RMAX helicopter (Fig. 9). The software simulation used relies on a high-fidelity dynamical model of the GTMax, including a detailed emulation of sensors and actuators.

The GTMax uses an approximate model-inversion adaptive controller equivalent to that described in Sec. II and uses an SHL NN with eight hidden-layer nodes as the adaptive element. The baseline

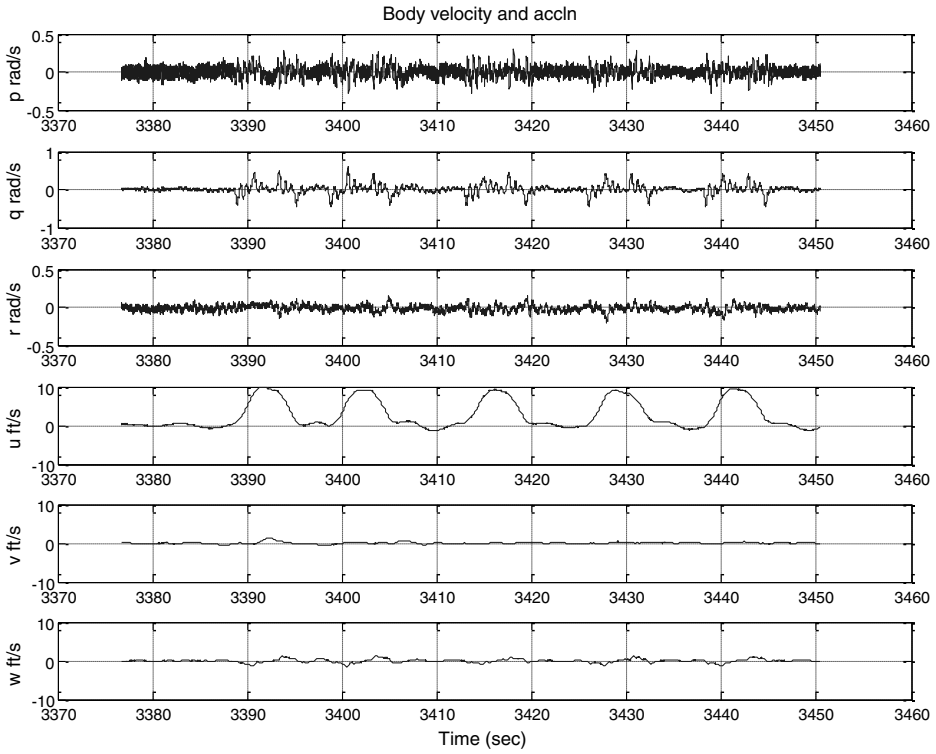

Fig. 16 Body-frame states: repeated forward steps.

adaptive control structure for the GTMax has been described in great detail in [7,9]. Inclusion of concurrent learning is the main modification made to the baseline control structure for the purpose of this paper.

Four successive position steps in the body x axis after an arbitrary interval between any two successive steps are commanded. This control input is used to mimic simple control tasks with repeated commands. Through these maneuvers, the UAS is expected to transition between the forward flight and the hover domain repeatedly. The performance of the inner-loop controller is characterized by the errors in the three body angular rates (roll rate p, pitch rate q, and yaw rate r), with the dominating variable being the pitch rate q as the rotorcraft accelerates and decelerates longitudinally. Figure 10 shows the performance of the inner-loop controller without concurrent learning. No considerable improvement in the pitch-rate tracking error is seen, even when the controller tracks a repeated command. The forgetting nature of the controller is further characterized by the evolution of NN W and V weights matrices. Figures 11 and 12 clearly show that the NN weights do not converge to a constant value; in fact, as the rotorcraft tracks the successive steps, the NN weights oscillate accordingly, characterizing the instantaneous (forgetting) nature of the adaptation.

On the other hand, when concurrent-learning adaptive controller of Theorem 2 is used, an improvement in performance is observed, as characterized by the reduction in pitch-rate tracking error after the first two steps. Figure 13 shows the tracking performance of the concurrent-learning adaptive controller. Figures 14 and 15 show that when concurrent learning is used the NN weights do not exhibit periodic behavior and tend to separate and approach constant values. This behavior indicates the alleviation of the rank-1 condition. Concurrent learning is initiated at about 2200 s into the simulation. Note that it takes a finite amount of time for the recorded data to be used in the concurrent-learning law, as smoothing must be performed to estimate $\ddot{x}$. This results in the effect concurrent learning being clearly visible only after about 20 s after being initiated.

VII. Implementation of Concurrent-Learning Adaptive Controller on a Vertical-Takeoff-and-Landing UAS

In this section flight-test results that characterize the benefits of using concurrent-learning adaptive control are presented. The flight tests presented here were executed on the Georgia Tech GTMax rotorcraft UAS (Fig. 9). As in the previous section, the baseline control structure for the GTMax is similar to that described in [7,9], with the main modification being the inclusion of concurrent learning. Note that the same baseline law learning rates, reference model parameters, approximate inversion model, and other parameters were used with or without concurrent learning. The discussion begins by presenting flight-test results for a series of forward steps. The results from an aggressive trajectory tracking maneuver, in

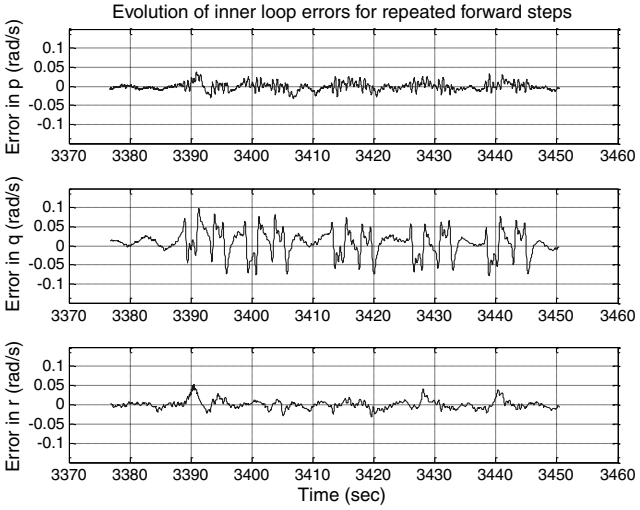

Fig. 17 Evolution of inner-loop errors: repeated forward steps.

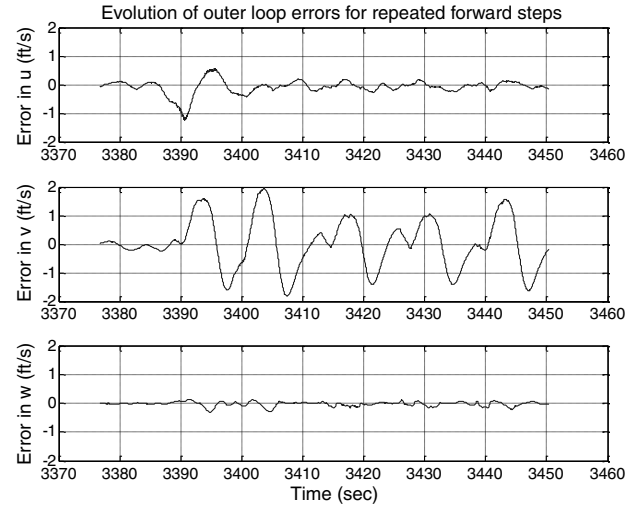

Fig. 18 Evolution of outer-loop errors: repeated forward steps.

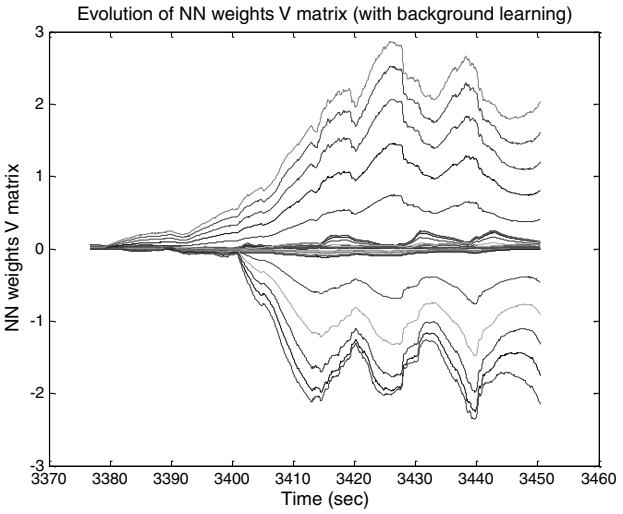

Fig. 19 Evolution NN weights V matrix with concurrent learning.

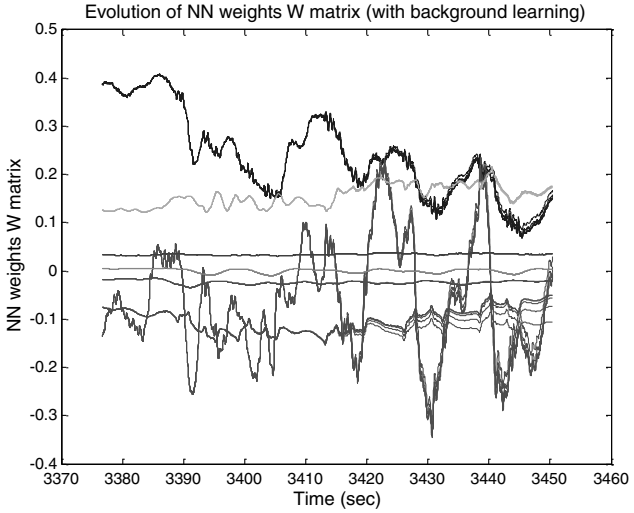

Fig. 20 Evolution NN weights W matrix with concurrent learning.

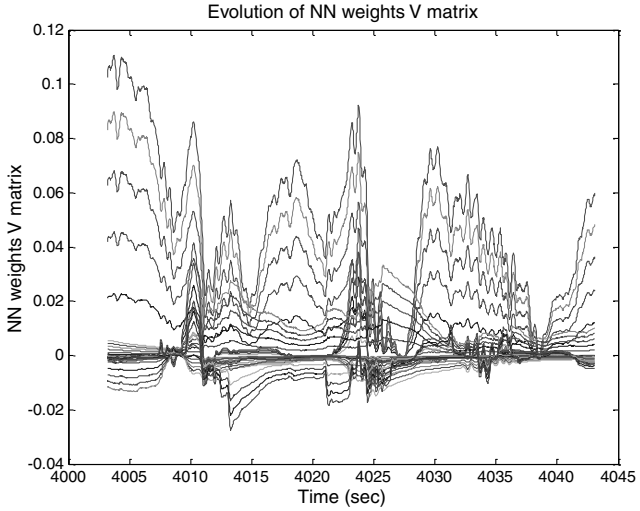

Fig. 21 Evolution NN weights *V* matrix without concurrent learning.

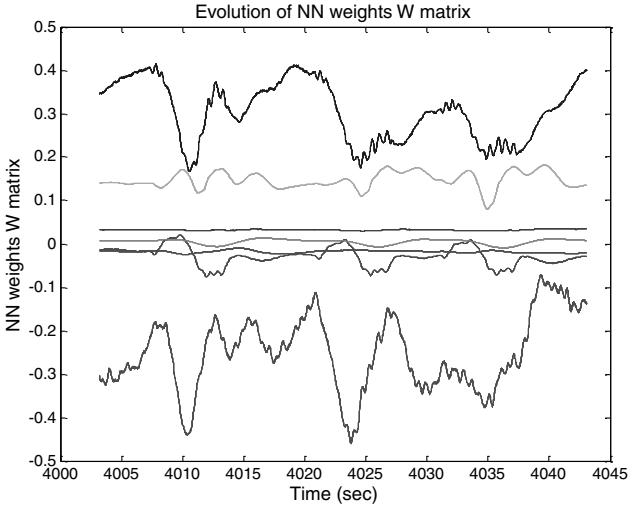

Fig. 22 Evolution of NN weights *W* matrix without concurrent learning.

which the UAS tracks an elliptical trajectory with aggressive velocity, and acceleration profiles are then presented.

A. Repeated Forward-Step Maneuvers

The repeated forward-step maneuvers are chosen to demonstrate a situation in which the controller performs a simple repeated task. Figure 16 shows the body-frame states from recorded flight data for a chain of forward-step commands. Figures 17 and 18 show the evolution of inner- and outer-loop errors, respectively, for the repeated forward-step maneuvers. These results assert the stability (in the sense of UUB) of the concurrent-learning controller of Theorem 2.

Figures 19 and 20 show the evolution of NN weights as the rotorcraft tracks repeated steps using the concurrent-learning controller of Theorem 2. Figures 21 and 22 show the evolution of NN weights as the rotorcraft tracks another sequence of repeated steps without concurrent learning. The NN *V* weights (Fig. 19) appear to approach constant values when concurrent learning is used. This behavior can be contrasted with Fig. 21, which shows that the NN *V* weights without concurrent learning do not appear to approach constant values. NN *W* weights for both cases remain bounded, with the main difference being the increased separation in *W* weights when concurrent learning is used, indicating the alleviation of the rank-1 condition. Furthermore, it is noted that the weights take on much larger values when using concurrent learning. This indicates that concurrent-learning adaptive law is searching for ideal weights in areas of the weight space that are never reached by the baseline adaptive without concurrent learning. The flight-test results also indicate an improvement in the error profile. In Fig. 18 it is seen that the lateral velocity tracking error reduces over repeated commands. These effects in combination indicate improved performance of the concurrent-learning adaptive controller. These results are of particular interest, since the maneuvers performed were conservative and the baseline adaptive controller had already been extensively tuned.

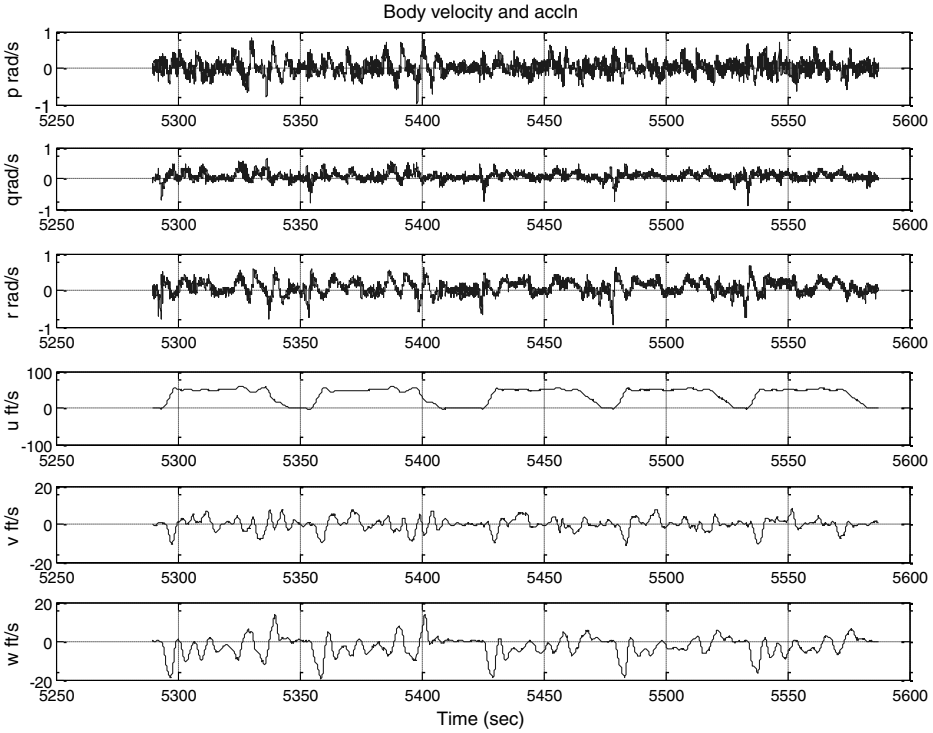

Fig. 23 Recorded inner- and outer-loop states when repeatedly tracking aggressive trajectories.

B. Aggressive Trajectory Tracking Maneuver

The results presented in this subsection are from a trajectory tracking maneuver in which the UAS repeatedly tracks an elliptical trajectory with aggressive velocity (50 ft/s) and acceleration ($\sim$20 ft/s^2) profile. This maneuver requires the rotorcraft to track commands in multiple states; hence, it is more complicated than the simple step commands used in the previous section. Since this maneuver involves state commands in multiple states, it is harder to visually observe an improvement in performance. Hence, the Euclidian norm of the error signal calculated at each time step is used as a rudimentary performance metric.

Figure 23 shows the recorded inner- and outer-loop states as the rotorcraft repeatedly tracks the elliptical trajectory pattern. In this flight, the first two ellipses (until $t = 5415$ s) are tracked with a commanded acceleration of 30 ft/s^2, while the rest of the ellipses are tracked at 20 ft/s^2. In the following these parts of the flight test are treated separately.

1. *Part 1: Aggressive Trajectory Tracking with Saturation in the Collective Channel*

Because of the aggressive acceleration profile of 30 ft/s^2, the rotorcraft collective channel was observed to saturate while performing high-velocity turns. This leads to an interesting challenge for the adaptive controller. Figures 24 and 25 show the evolution of the inner- and outer-loop tracking errors. It can be seen that the tracking error in the body u channel reduces in the second pass through the ellipse, indicating improved tracking performance of the concurrent-learning adaptive controller. This result is further characterized by the noticeable reduction in the norm of the tracking error calculated at every time step, as shown in Fig. 26. Note that the baseline controller on the GTMax uses pseudo control hedging to hide possible actuator saturation from the NN [5,7].

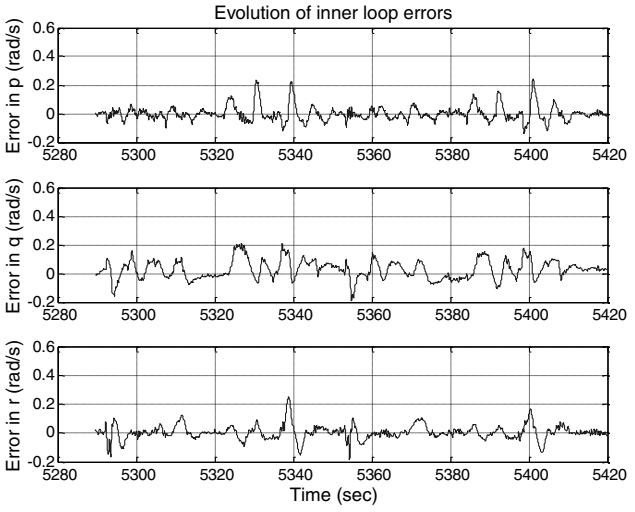

Fig. 24 Evolution of inner-loop errors: aggressive trajectory tracking maneuvers with collective saturation.

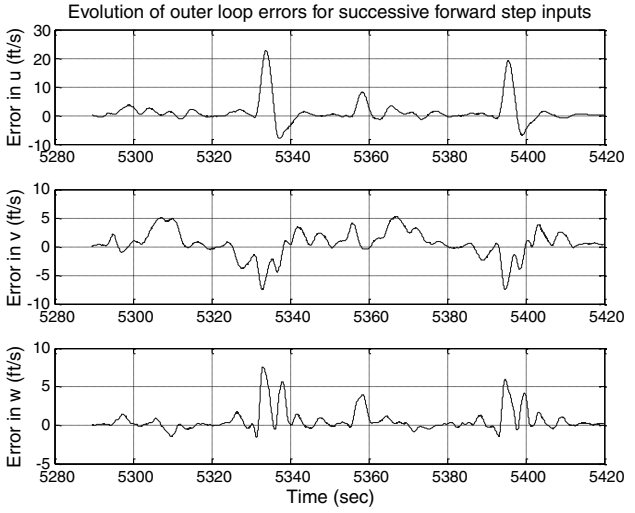

Fig. 25 Evolution of outer-loop errors: aggressive trajectory tracking maneuvers with collective saturation.

2. *Part 2: Aggressive Trajectory Tracking Maneuver*

In this part of the maneuver the acceleration profile was reduced to 20 ft/s^2. At this acceleration profile, no saturation in the collective channel was observed. Figures 27 and 28 show the evolution of the inner- and outer-loop tracking errors for this case.

3. *Aggressive Trajectory Maneuvers with Only Online-Learning NN*

To illustrate the benefit of the concurrent-learning adaptive controller, flight-test results are presented as the rotorcraft tracks the same trajectory command as in part 2, but without concurrent-learning adaptive control.

It is instructive to compare Figs. 29 and 30, which show the evolution of the NN weights with only online learning, with Figs. 31 and 32, which show evolution of the NN weights with concurrent learning.

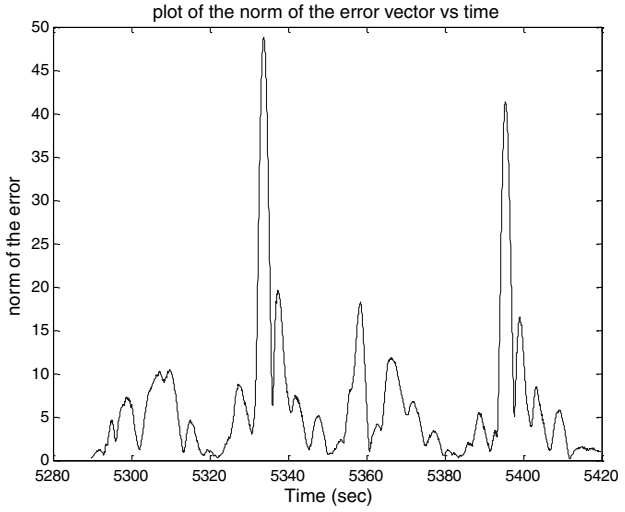

Fig. 26 Plot of the norm of the error at each time step when repeatedly tracking aggressive trajectories with collective saturation.

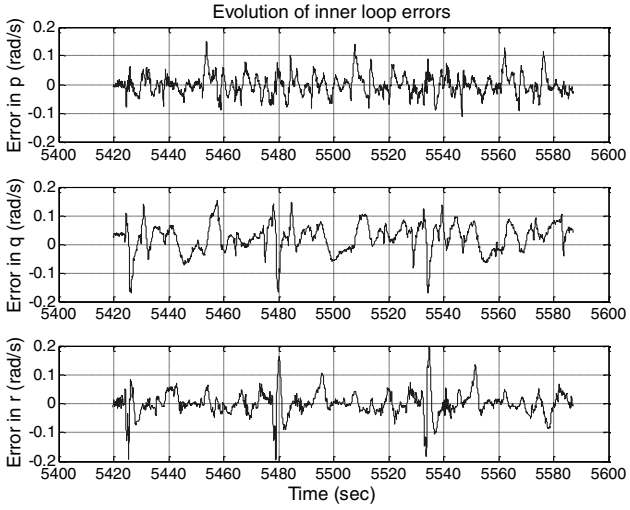

Fig. 27 Evolution of inner-loop errors: aggressive trajectory tracking maneuvers without collective saturation.

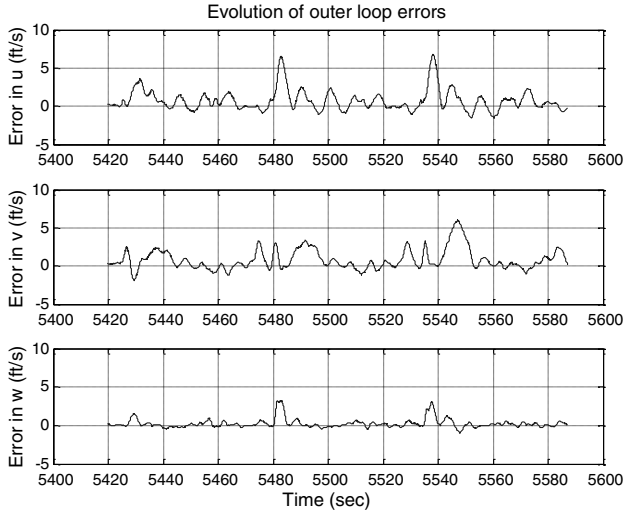

Fig. 28 Evolution of outer-loop errors: aggressive trajectory tracking maneuvers without collective saturation.

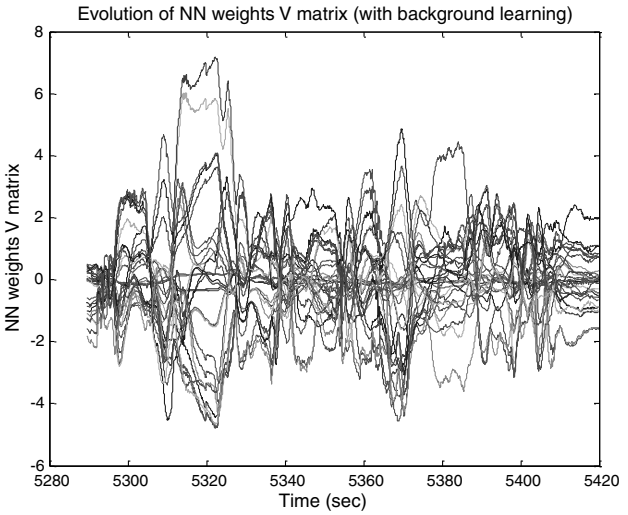

Fig. 31 Evolution of NN *V* matrix weights as the UAS tracks aggressive trajectory with concurrent-learning adaptation.

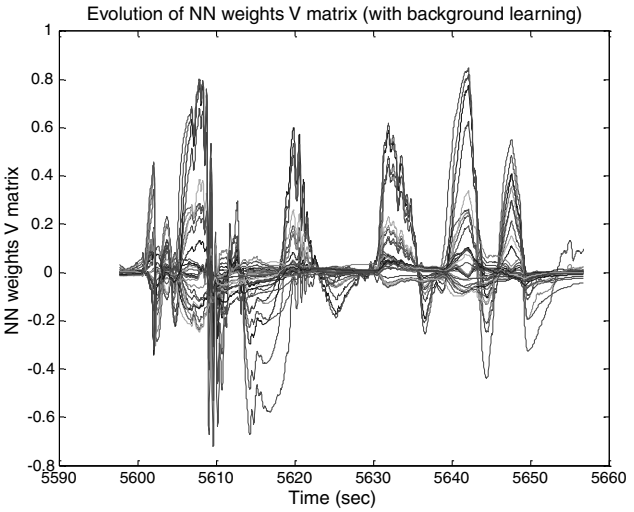

Fig. 29 Evolution of NN *V* matrix weights without concurrent-learning adaptation.

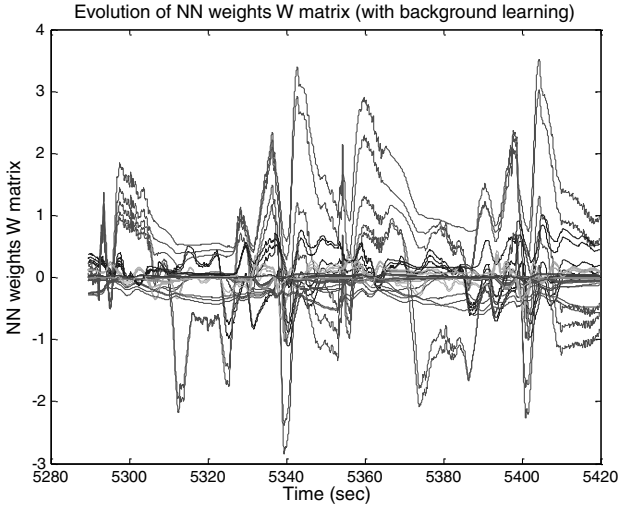

Fig. 32 Evolution of NN *W* matrix weights as the UAS tracks an aggressive trajectory with concurrent-learning adaptation.

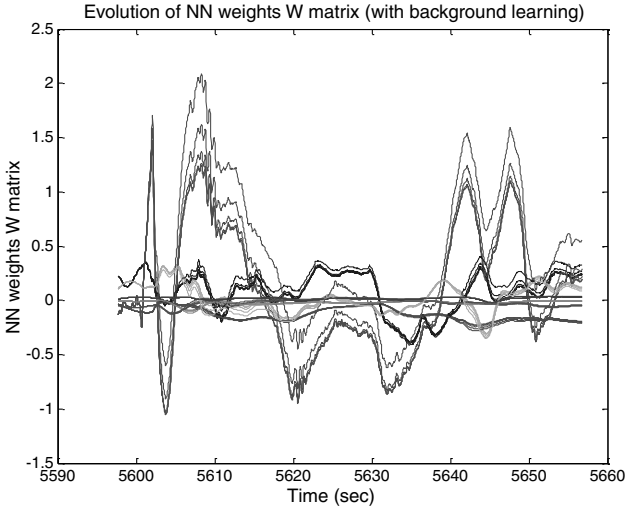

Fig. 30 Evolution of NN *W* matrix weights without concurrent-learning adaptation.

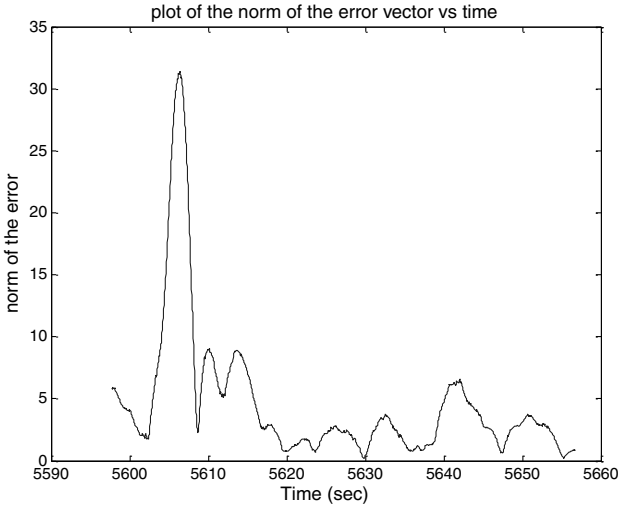

Fig. 33 Evolution of the norm of the tracking-error vector without concurrent learning when tracking aggressive trajectories.

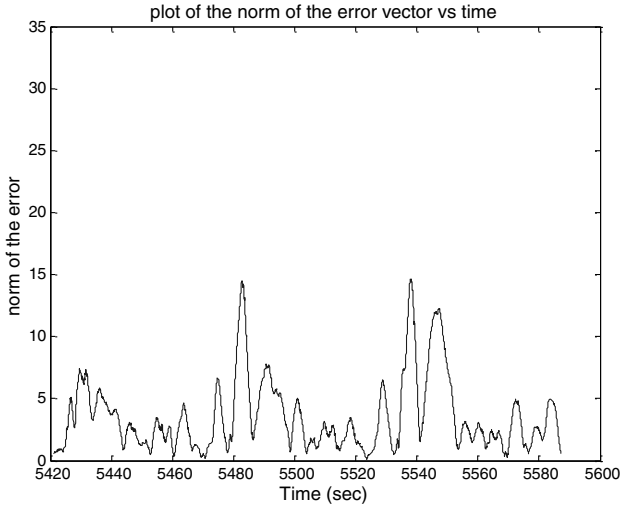

Fig. 34 Evolution of the norm of the tracking-error vector with concurrent learning when tracking aggressive trajectories.

Although absolute convergence of weights is not seen, it is interesting to note that when using concurrent learning the weights tend to be less oscillatory. Also, when using concurrent learning, the adaptation is retained (that is, the weights do not approach zero) when the UAS is hovering between two successive maneuvers. Furthermore, the weights take on different values when using concurrent learning. Figure 33 shows the plot of the tracking-error norm without concurrent learning evaluated at each time step, and Fig. 34 shows the plot of the tracking-error norm with current learning. The peaks in the tracking-error norms in these figures represent the beginning of a maneuver. Note that Fig. 33 contains only one peak, as data for only one maneuver without concurrent learning were recorded, whereas Fig. 34 contains two peaks, as data for two consecutive maneuvers were recorded. Contrasting the peak of the tracking-error norm in Fig. 33 with Fig. 34, it can be seen that the norm of the error vector is much higher without concurrent learning, indicating that the combined online- and concurrent-learning adaptive controller has improved trajectory tracking performance.

VIII. Conclusions

In this paper theory and flight-test results for a concurrent-learning adaptive control algorithm were presented. The ultimate boundedness of all system signals using Lyapunov-like stability analysis for the presented adaptive control law in the framework of approximate model reference adaptive control was shown. Concurrent learning simulates memory by recording interesting data points and storing them in a history stack. The novel aspect of this learning law is its ability to adapt using current as well as stored data without affecting the responsiveness of the adaptive law to current data. Concurrent learning achieves this by restricting the training on recorded data to the null-space of the training based on current data. Concurrent learning also allows the use of direct training information, which can be used to improve the performance of the adaptive controller. It was also shown that concurrent learning alleviates the rank-1 condition of traditional backpropagation-based NN training laws that use only current data for adaptation. Furthermore, since the choice of the baseline adaptive law and the projection matrix does not affect the stability properties of the concurrent-learning adaptive law, it is expected that the concurrent-learning methodology can be further generalized to other adaptive schemes.

Another contribution of this paper was the presentation of flight-test results with concurrent-learning adaptive controllers on a rotorcraft unmanned aerial vehicle. The flight-test results ascertain that all states remain uniformly ultimately bounded when using concurrent-learning adaptation. Expected improvement in tracking performance was noted. Furthermore, the evolutions of the neural network W and V matrix weights were observed to have different characteristics when concurrent learning was employed, including weight separation, a tendency toward weight convergence in some cases, and different numerical values of the adaptive weights. This difference in neural network weight behavior demonstrates the effect of overcoming the rank-1 condition. Because of the fact that other training algorithms are enabled by the stability proof, and since many other ways for selecting data points can be developed, this can be achieved even more consistently by improved implementation.

Appendix: Optimal Fixed-Point Smoothing

Numerical differentiation for estimation of state derivates suffers from high sensitivity to noise. An alternate method is to use a Kalman-filter-based approach. Let x be the state of the system, let $\dot{x}$ be its first derivative, and consider the following system:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} \tag{A1}$$

Suppose that x and $\dot{x}$ are available as sensor measurements, then using the above system, optimal fixed-point smoothing can be used to estimate $\ddot{x}$ from available noisy measurements [25].

Optimal fixed-point smoothing is a non-real-time method for arriving at a state estimate at some time t, where $0 \leq t \leq T$, by using all available data up to time T. Optimal smoothing combines a forward filter that operates on all data before time t and a backward filter that operates on all data after time t to arrive at an estimate of the state that uses all the available information. This Appendix presents brief information on implementation of optimal fixed-point smoothing; the interested reader is referred to Gelb [25] for further details. For ease of implementation on modern avionics, the relevant equations are presented in the discrete form. Let $\hat{x}_{k|N}$ denote the estimate of the state $x = [x \quad \dot{x} \quad \ddot{x}]^T$, let Z_k denote the measurements, let $(-)$ denote predicted values, and let $(+)$ denote corrected values. Then the forward Kalman filter equations can be given as follows:

$$\Phi_k = e^{\left(\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \mathrm{d}t \right)} \tag{A2}$$

$$z_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} \tag{A3}$$

$$\hat{x}_k(-) = \Phi_k \hat{x}_{k-1} \tag{A4}$$

$$P_k(-) = \Phi_k P_{k-1} \Phi_k^T + Q_k \tag{A5}$$

$$K_k = P_k(-) H_k^T [H_k P_k(-) H_k^T + R_k]^{-1} \tag{A6}$$

$$\hat{x}_k(+) = \hat{x}_k(-) + K_k [Z_k - H_k \hat{x}_k(-)] \tag{A7}$$

$$P_k(+) = [I - K_k H_k] P_k(-) \tag{A8}$$

The smoothed state estimate can be given as

$$\hat{x}_{k|N} = \hat{x}_{k|N-1} + B_N [\hat{x}_N(+) - \hat{x}_N(-)] \tag{A9}$$

where $\hat{x}_{k|k} = \hat{x}_k$, $N = k+1, k+2, \ldots$, and

$$B_N = \Pi_{i=k}^{N-1} P_i(+) \Phi_i^T P_{i+1}^{-1}(-) \tag{A10}$$

The smoothed error covariance matrix is propagated as

$$P_{k|N} = P_{k|N-1} + B_N[P_k(+) - P_k(-)]B_N^T \tag{A11}$$

where $P_{k|k} = P_k(+)$.

Acknowledgments

This work was supported in part by National Science Foundation grant ECS-0238993 and NASA Research Announcement NNX 08AD06A. The authors also thank the members of the Georgia Institute of Technology's unmanned aerial vehicle research facility for their support in flight-testing activities. Particularly, the authors wish to thank Suresh Kannan, Jeong Hur, R. Wayne Pickell, Henrik Christophersen, Nimord Rooz, Allen Wu, H. Claus Christmann, and M. Scott Kimbrell.

References

[1] Lewis, F. L., "Nonlinear Network Structures for Feedback Control," *Asian Journal of Control*, Vol. 1, Dec. 1999, pp. 205–228. doi:10.1111/j.1934-6093.1999.tb00021.x

[2] Kim, Y. H., and Lewis, F. L., *High Level Feedback Control with Neural Networks*, World Scientific Series in Robotics and Intelligent Systems, Vol. 21, World Scientific, Singapore, 1998.

[3] Patiño, D. H., Carelli, R., and Kuchen, B. R., "Neural Networks for Advanced Control of Robot Manipulators," *IEEE Transactions on Neural Networks*, Vol. 13, No. 2, March 2002, pp. 343–354. doi:10.1109/72.991420

[4] Calise, A., Hovakimyan, N., and Idan, M., "Adaptive Output Feedback Control of Nonlinear Systems Using Neural Networks," *Automatica*, Vol. 37, No. 8, Aug. 2001, pp. 1201–1211. doi:10.1016/S0005-1098(01)00070-X

[5] Johnson, E. N., "Limited Authority Adaptive Flight Control," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2000.

[6] Johnson, E. N., and Calise, A., "Limited Authority Adaptive Flight Control for Reusable Launch Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 6, pp. 906–913. doi:10.2514/2.6934, 2003

[7] Johnson, E. N., and Kannan, S. K., "Adaptive Trajectory Control for Autonomous Helicopters," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 3, May–June 2005, pp. 524–538. doi:10.2514/1.6271

[8] Johnson, E., Turbe, M., and Wu, A., edited by S. Kannan, "Flight Results of Autonomous Fixed-Wing UAS Transitions to and from Stationary Hover," AIAA Guidance, Navigation, and Control Conference, AIAA Paper 2006-6775, Aug. 2006.

[9] Kannan, S. K., "Adaptive Control of Systems in Cascade with Saturation," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2005.

[10] Hovakimyan, N., Yang, B. J., and Calise, A., "Adaptive Output Feedback Control Methodology Applicable to Non-Minimum Phase Nonlinear Systems," *Automatica*, Vol. 42, No. 4, 2006, pp. 513–522. doi:10.1016/j.automatica.2005.11.001

[11] Cao, C., and Hovakimyan, N., "Design and Analysis of a Novel $\mathcal{L}_1$ Adaptive Control Architecture with Guaranteed Transient Performance," *IEEE Transactions on Automatic Control*, Vol. 53, No. 2, 2008, pp. 586–590. doi:10.1109/TAC.2007.914282

[12] Lavretsky, E., and Wise, K., "Adaptive Flight Control of Manned/Unmanned Military Aircraft," *Proceedings of the American Control Conference*, June 2005.

[13] Nguyen, N., Krishnakumar, K., Kaneshige, J., and Nespeca, P., "Dynamics and Adaptive Control for Stability Recovery of Damaged Asymmetric Aircraft," AIAA Guidance, Navigation, and Control Conference, AIAA Paper 2006-6049, Keystone, CO, Aug. 2006.

[14] Ioannou, P., and Sun, J., *Robust Adaptive Control*, Prentice–Hall, Upper Saddle River, NJ, 1996.

[15] Gang, T., *Adaptive Control Design and Analysis*, Wiley, New York, 2003.

[16] Boyd, S., and Sastry, S., "Necessary and Sufficient Conditions for Parameter Convergence in Adaptive Control," *Automatica*, Vol. 22, No. 6, 1986, pp. 629–639. doi:10.1016/0005-1098(86)90002-6

[17] Narendra, K., and Annaswamy, A., "A New Adaptive Law for Robust Adaptation without Persistent Excitation," *IEEE Transactions on Automatic Control*, Vol. 32, No. 2, Feb. 1987, pp. 134–145. doi:10.1109/TAC.1987.1104543

[18] Haykin, S., *Neural Networks a Comprehensive Foundation*, 2nd ed., Prentice–Hall, Upper Saddle River, NJ, 1998.

[19] Ponzyak, A. S., Sanchez, E. N., and Yu, W., *Differential Neural Networks for Robust Nonlinear Control, Identification, State Estimation, and Trajectory Tracking*, World Scientific, Singapore, 2001.

[20] Suykens, J. A. K., Vandewalle, J. P. L., and De Moor, B. L. R., *Artificial Neural Networks for Modeling and Control of Non-Linear Systems*, Kluwer Academic, Dordrecht, The Netherlands, 1996.

[21] Ioannou, P. A., and Kokotovic, P. V., *Adaptive Systems with Reduced Models*, Springer–Verlag, Berlin, 1983.

[22] Volyanskyy, K., Calise, A. J., Yang, B. J., and Lavretsky, E., "An Error Minimization Method in Adaptive Control," AIAA Guidance Navigation and Control Conference, AIAA Paper 2006-6346, Keystone, CO, 2006.

[23] Hornik, K., Stinchcombe, M., and White, H., "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, Vol. 2, 1989, pp. 359–366. doi:10.1016/0893-6080(89)90020-8

[24] Strang, G., *Linear Algebra and its Applications*, 3rd ed., Thomson Learning, San Diego, CA, 1988.

[25] Gelb, A., *Applied Optimal Estimation*, MIT Press, Cambridge, MA, 1974.

[26] Hadad, W., and Chellebaonia, V., *Nonlinear Dynamical Systems and Control: A Lyapunov Based Approach*, Princeton Univ. Press, Princeton, NJ, 2006.

[27] Khalil, H., *Nonlinear Systems*, 3rd ed., Prentice–Hall, Upper Saddle River, NJ, 2001.

[28] Berberian, S. K., *Introduction to Hilbert Spaces*, AMS Chelsea, New York, 1976.