JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 34, No. 3, May–June 2011

Self-Organizing Radial Basis Function Networks for Adaptive Flight Control

Praveen Shankar*
Arizona State University, Tempe, Arizona 85287
Rama K. Yedavalli†
Ohio State University, Columbus, Ohio 43210
and
John J. Burken‡
NASA Dryden Flight Research Center, Edwards Air Force Base, California

DOI: 10.2514/1.51135

The performance of nonlinear flight-control algorithms, such as feedback linearization and dynamic inversion, is heavily dependent on the fidelity of the dynamic model being inverted. Incomplete or incorrect knowledge of the aircraft dynamics results in reduced performance and may lead to instability. A self-organizing parametrization structure is developed to augment the baseline dynamic inversion controller for a high-performance military aircraft. This algorithm is proven to be stable and can guarantee arbitrary tracking error performance. The training algorithm to grow the network and adapt the parameters is derived from Lyapunov theory. In addition to growing the network of basis functions, a pruning strategy is incorporated to keep the size of the network as small as possible. The controller is simulated for different situations, including control surface failures, modeling errors, and external disturbances. A performance measure of maximum tracking error is specified for the controllers a priori. Excellent tracking error minimization to a prespecified level using the adaptive component is achieved. The performance of the self-organizing radial-basis-function network-based controller is also compared with a fixed radial-basis-function network-based adaptive controller. While the fixed radial-basis-function network-based controller, which is tuned to compensate for control surface failures, fails to achieve the same performance under modeling uncertainty and disturbances, the self-organizing radial-basis-function network is able to achieve good tracking convergence under all specified error conditions.

I. Introduction

FLIGHT vehicles such as highly maneuverable military aircraft and hypersonic vehicles pose a significant challenge to control design due to the complex nonlinearities associated with the dynamics. These vehicles require control laws that are adaptable to not only the various operating conditions but also, as in the case of hypersonic vehicles, to those that account for changing geometry of the vehicle during reentry into the atmosphere and other uncertainties in the environment. Subsystem interactions also play a major role in defining the guidance and control laws for these flight vehicles. The development of adaptive control, which can permit controlled flight without a detailed aerodynamic model or with deviations away from the model (unplanned departures from aerodynamic geometry, accidents, or failures), is therefore very important. Linearized equations of motion that are used for designing flight controllers fall short of achieving the desired performance when the dynamics vary widely with operating conditions. Nonlinear control methods such as adaptive control have found great use in aircraft flight control. One of the methods of adaptive control is feedback linearization or dynamic inversion. The main feature of feedback linearization is that the nonlinear dynamics of the aircraft are transformed to an equivalent linear system, which has a standard form. The linear model is then used to design a controller to achieve a desired response to pilot commands, based on handling qualities criteria and maneuvering objectives. The difficulty, however, with the use of dynamic inversion for aircraft flight control is that a detailed knowledge of the nonlinear plant dynamics is required. While extensive aerodynamic derivative databases can be used online to reconstruct the inverse dynamics of the aircraft, perfect inversion may not be possible, resulting in tracking errors. Other factors that may cause dynamic inversion errors include external disturbances or failures in the control surfaces. Augmenting the baseline controller with approximators that use a parametrization structure that is adapted online reduces the effect of these inversion errors [1]. One of the parameterizations that is used is the neural network. Neural networks can be trained offline or online to compensate for the errors. Offline neural networks are trained using extensive input–output pairs of aerodynamic data. Even though a high-fidelity inverse model can generally be reconstructed, inversion errors are magnified when a failure/unknown error occurs and the pretrained neural network cannot accurately reconstruct the inverse dynamics in such a scenario. Such situations require a neural network to be implemented that has the ability to learn online. Over the past several years, online neural networks have been successfully used for adaptive control in various applications [2–4]. More recent papers use neural networks in combination with other technologies to improve performance of the controller [5,6]. Neural-networks-based adaptive control has also found wide application for various aircraft systems, including highly maneuverable military aircraft, hypersonic vehicles, and unmanned aerial vehicles [7–15]. The implementation of neural-network-based controllers in these papers requires the designer to decide the structure of the network, including the number of nodes and the training algorithms. A significant change to neural-network learning algorithms was made by Platt [16] through the development of a resource allocation network (RAN), in which hidden nodes are added sequentially based on the novelty of the new data. Platt showed the radial-basis-function (RBF) networks were amenable to such a resource allocation scheme. Sanner and Slotine [17] presented a structurally dynamic wavelet network for adaptive control of a

Received 13 July 2010; revision received 17 December 2010; accepted for publication 21 December 2010.

*Lecturer, School for Engineering of Matter Transport and Energy; praveen.shankar@asu.edu. Senior Member AIAA.
†Professor, Department of Mechanical and Aerospace Engineering; yedavalli.1@osu.edu. Associate Fellow AIAA.
‡Aerospace Engineer; john.burken@nasa.gov. Member AIAA.

robotic system and developed the stability analysis without any assumptions on the process used for adding and deleting the basis functions. A similar algorithm was developed by Fabri and Kadirkamanathan [18], where the neural networks use RBFs that are equally spaced on a given region of state space. The structural adaptation mechanism is directed toward selectively choosing from this mesh of nodes as the state evolves within the region. This work is extended in [19–21], where the RBFs are replaced by piecewise linear approximators that are locally weighted. Yan et al. [22] and Huang et al. [23,24] present a growing and pruning algorithm for RBF networks with application to nonlinear system identification. Yan et al. also provide an exhaustive compilation of RBF neural networks for adaptive flight control in [25]. In the development of the growing and pruning RBF-based controllers, the authors also allow for the adaptation of the centers and variances of the Gaussian basis functions but do not provide parameter convergence proofs. Additionally, the stability proofs focus on the weight update of the network without any consideration provided for the structural adaptation, and the conditions for adding a node to the network depend on the novelty of the input to the network rather than a stability requirement. The disadvantages of such a mechanism for structural adaptation is that there is no guarantee that addition of a node will destabilize the system and the possibility of unnecessary addition of nodes, which may increase the computational burden. In the paper by Farrell and Zhao [26], however, the application of locally weighted piecewise linear functions as basis functions for self-organizing approximators is discussed with a stability proof presented for not only the adaptation of parameters but also for the structural adaptation. The authors also present a condition for adding a new basis function that finds its origin from Lyapunov analysis. In this paper, the application of an adaptive controller to the flight dynamics of a high-performance military aircraft is discussed. A baseline dynamic inversion controller that is designed to close the loop around the rotational dynamics of the aircraft is augmented with the self-organizing RBF network (SORBFN) to compensate for errors that may occur due to modeling uncertainties, failures, and disturbances. The SORBFN is trained using an algorithm that finds its origin in Lyapunov theory for not only the weight update but also for the structural adaptation. The rest of the paper is organized as follows. The problem statement is developed, followed by the presentation of the design of the SORBFN-based adaptive controller. Next, the stability analysis of the controller is addressed, followed by application of the controller to the rotational dynamics of a modified F-15.

II. Problem Statement

Consider the scalar nonlinear system of the form

$$\dot{x} = f(x) + g(x)u \tag{1}$$

where $x, u \in \Re$. It is assumed that the dynamics are affine in control u. Now, consider a control law given by

$$u = \hat{g}(x)^{-1}[\dot{x}_d - \hat{f}(x) - k\tilde{x}] \tag{2}$$

where $\hat{f}(x)$ and $\hat{g}(x)$ are approximations of $f(x)$ and $g(x)$, and k is a positive scalar. These approximated functions are in the form of a parametrization chosen by the designer. The control law is implemented such that the parameters associated with this parametrization are adapted continuously to achieve the performance requirement. This is generally referred to as adaptive approximation-based control [1]. Now. the closed loop system is of the form

$$\dot{x} = f(x) + g(x)\hat{g}(x)^{-1}[\dot{x}_d - \hat{f}(x) - k\tilde{x}] \tag{3}$$

It is evident from the preceding equation that stability is heavily dependent on the approximating functions. In the following analysis, it is assumed that $g(x)$ is known exactly and is invertible. Let us choose an approximation structure

$$\hat{f}(x) = \hat{\theta}_f^T \phi_f(x) \tag{4}$$

The continuous function $f(x)$ can be parameterized as follows:

$$f(x) = \theta_f^{*T} \phi_f(x) + \varepsilon_f(x) \tag{5}$$

$$\theta_f^{*T} = \arg\min_{\theta_f}(\sup_{x\in\Re}|f(x) - \theta_f^T \phi(x)|) \tag{6}$$

where $\varepsilon_f(x)$ is the minimum function approximation error (MFAE) for $f(x)$ and $\theta_f^*, \phi_f(.) \in \Re^N$. Now, the closed loop is

$$\dot{\tilde{x}} = -k\tilde{x} + \tilde{\theta}_f^T \phi_f(x) + \varepsilon_f(x) \tag{7}$$

where

$$\tilde{\theta}_f = \theta_f^* - \hat{\theta}_f \tag{8}$$

$$\tilde{x} = x - x_d \tag{9}$$

and k is a positive scalar chosen by the designer. To analyze the stability of the closed loop, consider the Lyapunov function

$$V(t) = \tfrac{1}{2}\tilde{x}^2 + \tfrac{1}{2}\tilde{\theta}_f^T \Gamma \tilde{\theta}_f \tag{10}$$

where Γ is a diagonal matrix with positive elements. Evaluating the time derivative of $V(t)$,

$$\dot{V}(t) = -k\tilde{x}^2 + \tilde{\theta}_f^T[\phi_f(x)\tilde{x} + \Gamma\dot{\tilde{\theta}}_f] + \tilde{x}\varepsilon_f(x) \tag{11}$$

From the preceding equation, if the parameter adaptation is chosen as

$$\dot{\hat{\theta}}_f = \Gamma^{-1}\phi_f(x)\tilde{x} \tag{12}$$

then

$$\dot{\tilde{\theta}}_f = -\Gamma^{-1}\phi_f(x)\tilde{x} \tag{13}$$

We now have

$$\dot{V}(t) = -k\tilde{x}^2 + \tilde{x}\varepsilon_f(x) \tag{14}$$

Consider the trivial case when the parametrization chosen can exactly approximate the functions $f(x)$; i.e., $\varepsilon_f(x) = 0$. Then,

$$\dot{\bar{V}}(t) = -k\tilde{x}^2 \tag{15}$$

The derivative of the Lyapunov function is negative semidefinite, implying that the signal $\tilde{x}$ is bounded. Moreover, $\tilde{x} \to 0$ as $t \to \infty$. The parameters $\tilde{\theta}_f$ are not guaranteed to converge to zero but are bounded. Now, consider the case where the MFAE is not zero. The derivative of the Lyapunov function is now given by Eq. (14). Let us now assume that the MFAE is bounded by a constant; that is,

$$|\varepsilon_f(x)| \le \bar{\varepsilon}_f \quad \forall\ x$$

Now, Eq. (14) can be rewritten as

$$\dot{V}(t) \le -k|\tilde{x}|^2 + |\tilde{x}||\varepsilon_f(x)| \qquad \dot{V}(t) \le -|\tilde{x}|(k|\tilde{x}| - \bar{\varepsilon}_f)$$

Within the dead zone, from the preceding analysis, $\dot{V}(t)$ will be guaranteed to be negative semidefinite if the following condition is satisfied:

$$|\tilde{x}| \ge \frac{\bar{\varepsilon}_f}{|k|}$$

If the error signal were to decrease below the bound, the adaptation parameters could start to diverge and cause the system to go unstable. To prevent this, a dead zone is implemented so that parameter adaptation does not occur once the tracking error reaches a particular value. One of the characteristics of this method is that an arbitrarily small tracking error as a performance requirement is not guaranteed.

Additionally, it is impossible to determine with confidence the bound $\bar{\varepsilon}_f$ since the function being approximated is not known. This causes the designer to choose conservative estimates leading to a larger tracking error. Lastly, there is no guarantee of accurate function approximation unless persistence of excitation conditions are met [27]. With this background, the problem statement can be defined as follows.

Given the nonlinear system of the form equation (1) with a dynamic inversion control law, determine a parametrization structure and training algorithm to approximate the unknown system dynamics $f(x)$ such that an arbitrarily small tracking error as a performance criterion can be guaranteed for the closed-loop system.

This performance criterion is achieved by incorporating a parametrization structure (network of RBFs) that is self-organizing. Unlike the previously reviewed papers on growing and pruning RBF networks, the training algorithm to both grow the network and adapt the parameters is derived from Lyapunov theory. In addition to growing the network of basis functions, a pruning strategy is incorporated to keep the size of the network as small as possible.

III. Self-Organizing Radial-Basis-Function Network-Based Controller

Let us consider the affine nonlinear system from Eq. (1). The variable $f(x)$ is the unknown design model that must be approximated online using state and tracking error information.

Please consider the following assumptions:

1) The desired output $x_d(t)$ and its derivative $\dot{x}_d(t)$ are available for any $t \geq 0$ where $x_d(t) \in D$.

2) $D \in \Re$ is a compact region where the system is desired to operate.

3) The variable $f(x)$ is smooth and bounded on D.

4) Perfect knowledge of the control effectiveness function $g(x) \in \Re$ is available, and it is invertible.

Consider a baseline dynamic inversion law given by Eq. (2). The approximation algorithm to stabilize the system and achieve prespecified tracking error ϵ criterion is implemented as follows. We assume that, initially, there are zero nodes in the approximation network. This implies that $\hat{f}(x) = 0$. Now, consider a Lyapunov function of the form

$$V_0(t) = \tfrac{1}{2}\tilde{x}^T\tilde{x} \tag{16}$$

The derivative of the Lyapunov function is then given by

$$\dot{V}_0(t) = \tilde{x}[-k\tilde{x} + f(x)] \tag{17}$$

$$\dot{V}_0(t) \leq -|\tilde{x}|(|k||\tilde{x}| - |f(x)|) \tag{18}$$

where k is a positive scalar. From the preceding equation, if

$$k|\tilde{x}| \geq |f(x)| \tag{19}$$

the Lyapunov derivative is negative definite. Therefore, if the unknown function $f(x)$ is bounded by $k\epsilon$, then the lower bound on the tracking error will be ϵ. Since it is assumed that $f(x)$ is bounded on D, a large gain k of the system should, in theory, be able to stabilize the system for any arbitrarily small value of ϵ. However, in most applications, there exist actuator constraints, and it is desired to limit the bandwidth of the controller. Therefore, choosing a large value for gain k is impractical.

A. Weight Update

Let us now assume that j RBFs have been added to the approximating network, and $\hat{f}(x) \neq 0$. Then, the closed loop system is given by

$$\dot{x} = f(x) + \dot{x}_d - \sum_{i=1}^{j} \hat{\theta}_{f_i}\phi_{f_i}(x) - k\tilde{x} \tag{20}$$

$$\dot{\tilde{x}} = -k\tilde{x} + \sum_{i=1}^{j} \tilde{\theta}_{f_i}\phi_{f_i}(x) + \varepsilon_{f_j}(x) \tag{21}$$

where each element in $f(x)$ is assumed to be ideally parameterized with the j RBFs as

$$f(x) = \sum_{i=1}^{j} \theta^*_{f_i}\phi_{f_i}(x) + \varepsilon_{f_j}(x) \tag{22}$$

where $\varepsilon_{f_j}(x)$ represents the MFAE for a network of j RBFs, and this MFAE is bounded; that is,

$$|\varepsilon_{f_j}(x)| \leq \bar{\varepsilon}_{f_j} \quad \forall\ x \tag{23}$$

Now, consider a Lyapunov function of the form

$$V_j(t) = V_0(t) + V_\theta(t) = \frac{1}{2}\tilde{x}^2 + \frac{1}{2}\sum_{i=1}^{j} \tilde{\theta}_{f_i}\gamma\tilde{\theta}_{f_i} \tag{24}$$

Choosing the adaptive law such that

$$\dot{\hat{\theta}}_{f_i} = \gamma^{-1}\phi_{f_i}(x)\tilde{x} \tag{25}$$

we have

$$\dot{V}_j(t) = -k\tilde{x}^2 + \tilde{x}\varepsilon_{f_j}(x) \tag{26}$$

$$\dot{V}_j(t) \leq -|\tilde{x}|^T(|k||\tilde{x}| - \bar{\varepsilon}_{f_j}) \tag{27}$$

Similar to earlier analysis, if the error signal is such that

$$|\tilde{x}| \geq \frac{\bar{\varepsilon}_{f_j}}{k} \tag{28}$$

$\dot{V}_j(t)$ will be negative semidefinite; hence, the system is asymptotically stable.

B. Addition of Nodes

From Eq. (28), choosing a fixed number of nodes for the network results in a minimum tracking error that is bounded by a function of the MFAE of the network, and the designer cannot specify a desired tracking error. It may be possible through multiple simulations to determine the number of nodes that can result in a desired tracking error, but the continued performance of the system under changes to the dynamics cannot be guaranteed. Let us consider the case when the number of nodes may be such that $\bar{\varepsilon}_{f_j}/|k|$ is greater than the desired tracking error. In such a case, continued adaptation of the weights $\hat{\theta}_i$ will occur within the dead zone, and this may cause the weights to diverge. This may result in the function approximation $\hat{f}(x)$ to grow and diverge away from $f(x)$. With the closed-loop system,

$$\dot{\tilde{x}} = -k\tilde{x} + f(x) - \hat{f}(x) \tag{29}$$

and the Lyapunov function $V_0(t)$,

$$\dot{V}_0(t) = \tilde{x}[-k\tilde{x} + f(x) - \hat{f}(x)] \tag{30}$$

$$\dot{V}_0(t) \leq -|\tilde{x}|(|k||\tilde{x}| - |f(x) - \hat{f}(x)|) \tag{31}$$

If the adaptation of $\hat{\theta}$ is continued in the dead zone, the parameters will start to diverge. This will cause $\hat{f}(x)$ to grow, eventually resulting in a situation such that

$$|f(x) - \hat{f}(x)| > |k||\tilde{x}| \tag{32}$$

causing the Lyapunov function $V_0(t)$ to increase.

Alternatively, the inherent dynamics of the system may change so substantially that the current number of nodes may not even guarantee the original least-squares approximation bounds. Consider the closed-loop system in Eq. (29). At some time $T > t_0$, let a sudden failure or disturbance affect the system such that

$$\dot{\tilde{x}} = -k\tilde{x} + f(x) - \hat{f}(x) + \chi(x, \bar{u}) \tag{33}$$

where $\bar{u}$ is an external disturbance, and $\chi(.)$ represents the effect of the disturbance on the system dynamics. Now, the current parametrization is required to approximate the new function

$$\bar{f}(x) = f(x) + \chi(x, \bar{u}) \tag{34}$$

Let us assume that the new function $\bar{f}(x)$ can be parameterized as

$$\bar{f}(x) = \sum_{i=1}^{j} \theta^*_{\text{new}_{f_i}} \phi_{f_i}(x) + \varepsilon_{\text{new}_{f_j}}(x) \tag{35}$$

The adaptation of the weights $\hat{\theta}_{f_i}$ is originally oriented toward $\theta^*_{f_i}$, and $\theta^*_{\text{new}_{f_i}}$ may be substantially different from it. This may cause the function approximation error $\bar{f}(x) - \hat{f}(x)$ to be large. Even if $\theta^*_{\text{new}_{f_i}}$ is not substantially different from $\theta^*_{f_i}$, the new MFAE $\varepsilon_{\text{new}_{f_j}}(x)$ may have a bound greater than $k|\tilde{x}|$. Since the objective of the training algorithm is to achieve the desired tracking error, adaptation may continue within the dead zone, causing the approximating function to diverge and, as in the previous case, the derivative of $V_0(t)$ may become positive, destabilizing the system.

Several conditions, as discussed, may make the derivative of the Lyapunov function $V_0(t)$ positive. Since we have assumed that the state x is measurable, and we can assume that the gain k is specified a priori, it is possible to monitor the Lyapunov function $V_0(t)$. If, at any time, $V_0(t)$ increases, a node is added to the network. However, in order to control the growth of the network, another condition is checked before adding the node, which may be explained as follows. The contribution of each node is directly dependent on the distance of the current state from the center of the respective node. If the state is evolving such that it is continuously moving away from all the allocated nodes, the output of each node is going to decrease monotonically. Therefore, before adding a node, it is verified whether the output of every node is continuously decreasing for a prespecified amount of time. Therefore, a new node is added only when 1) $V_0(t)$ increases and 2) the output of each node $\phi_i(x)$ continuously decreases for N_a time steps.

The new node added to the network is a RBF centered at the current state of the system. The variance σ is chosen to be the same value for all the nodes. The initial weight of the node is equal to the tracking error. This initial weight is required to prove the boundedness of the Lyapunov function when a node is added, as detailed in the next section.

C. Deletion of Nodes

The adaptation of the weights and growth of the network is derived from Lyapunov theory. It is, however, possible that, based on the choice of learning rate γ and node addition threshold N_{t_a}, the network may grow substantially large to cause a computational burden. In such an event, the nodes that have been nonperforming for a large amount of time are deleted. The condition for deletion can be explained similarly to the growing criterion. If the state evolves in such a way so as to move so far away from a particular node that the contribution of that node is zero, the nodes may be deleted. Therefore, the output of each node is continuously monitored, and if this output is continuously decreasing or becomes zero over N_d time steps, the node is deleted. N_d is chosen to be greater than N_a, so that nodes are deleted at slower rate than they are added. This reduces the possibility of deleting all the nodes in the network at any time.

IV. Stability Analysis for Self-Organization

The stability analysis of the SORBFN controller consists of proving that the following conditions are met:

1) The number of nodes in the network $N(t)$ is bounded.

2) The variable $\tilde{x}$ is ultimately bounded by the desired tracking error ϵ.

We will follow the stability analysis provided for local approximators developed by Farrell and Zhao in [26]. Let the time interval of operation be specified as $[T_0, T_f]$, where T_f can be infinite. Denote the time at which the jth basis function is added at T_j; that is, $N(T_j) = j$ and

$$\lim_{\kappa \longrightarrow 0} N(T_j - \kappa) = j - 1$$

Therefore, the value of $N(t)$ is constant between the times T_j and T_{j+1}. It is assumed that, for some number of nodes N_f, the RBF network has sufficient nodes to approximate the unknown function, with an MFAE having an upper bound $\bar{\varepsilon}_{f_{N_f}} \leq |k|\epsilon$. In such a case, the node addition would stop and $T_{N_f+1} = \infty$.

A. Boundedness of Number of Nodes $N(t)$

The boundedness of $N(t)$ can be derived from the assumption that the region of operation D is compact and $f(x)$ is bounded on D. It is known that an infinite number of RBFs can exactly approximate a continuous function [28]. Therefore, it can be assumed that a finite number of RBFs are sufficient to approximate the unknown bounded continuous function on a compact set to within a MFAE of $|k|\epsilon$. Thus, $N_f < \infty$.

B. Ultimate Bound on $\tilde{x}$

The adaptive algorithm consists of a discrete update to the number of nodes in addition to the weight update for existing nodes. It is assumed that when a node is added, no weight update occurs, and vice versa. To prove that the error $\tilde{x}$ is bounded by the desired tracking error ϵ, it should be shown that the Lyapunov function $V(t)$ is continuously decreasing during both the discrete update to the number of nodes and weight update of the existing nodes. The Lyapunov function $V_j(t)$ was shown to be decreasing when the weight of the nodes was updated using the update law in Eq. (25). For a fixed number of nodes (RBFs), the tracking error is bounded by a function of the MFAE, since adaptation is halted at the boundary of this dead zone. In the self-organizing algorithm for RBFs, the adaptation is not stopped beyond the dead zone, but an additional node is added if the Lyapunov function $V_0(t)$ increases. The additional node causes the MFAE of the current network with $N(T_{j+1})$ nodes to be lower than the network with $N(T_j)$ nodes [28]. The stability analysis of this process now proceeds in the same way as before (for j nodes); however, it needs to be shown that the Lyapunov function is bounded when the number of nodes increases from j to $j + 1$. The Lyapunov function at time $t = T_j$ is given by $V_j(t)$. Note that the subscript j refers to the number of nodes in the network. The term T_j represents the time at which node j is added:

$$V_j(T_j) = \frac{1}{2}[\tilde{x}^2(T_j) + \sum_{i=1}^{j} \tilde{\theta}_{f_i}(T_j)\gamma\tilde{\theta}_{f_i}(T_j)] \tag{36}$$

Since $\tilde{x}$ is continuous, $\tilde{x}(T_j) = \tilde{x}(T_j^-)$. Also, a node is added only when there are no contributions from the remaining nodes. This implies that the weights of the remaining $j - 1$ nodes do not change from T_j^- to T_j. Therefore,

$$V_j(T_j) = \frac{1}{2}\left[\tilde{x}^2(T_j^-) + \sum_{i=1}^{j-1} \tilde{\theta}_{f_i}(T_j^-)\gamma\tilde{\theta}_{f_i}(T_j^-)\right] + \frac{1}{2}[\tilde{\theta}_{f_j}(T_j)\gamma\tilde{\theta}_{f_j}(T_j)] \tag{37}$$

$$V_j(T_j) = V_j(T_j^-) + \tfrac{1}{2}\tilde{\theta}_{f_j}(T_j)\gamma\tilde{\theta}_{f_j}(T_j) \tag{38}$$

Since the adaptation process is proved to be stabilizing with a fixed number of nodes and that the Lyapunov function is continuously decreasing, it can be assumed that, for any $t \in [T_{j-1}, T_j^-]$, $V_{j-1}(t) \leq V_{j-1}(T_{j-1})$. Now,

$$V_j(T_j) \leq V_{j-1}(T_{j-1}) + \tfrac{1}{2}\tilde{\theta}_{f_j}(T_j)\gamma\tilde{\theta}_{f_j}(T_j) \tag{39}$$

due to the addition of the jth node. Let us now consider the scenario when the first node is added:

$$V_1(T_1) \leq V_0(T_0) + \tfrac{1}{2}\tilde{\theta}_{f_1}(T_1)\gamma\tilde{\theta}_{f_1}(T_1) \tag{40}$$

$$V_0(T_0) = \tfrac{1}{2}\tilde{x}(T_0)^2 \tag{41}$$

where $\tilde{x}(T_0)$ is the initial error, which has a finite value. Also, $\tilde{\theta}_{f_1}(T_1) = \theta^*_{f_1} - \theta_{f_1}(T_1)$ has a finite value, because the initial parameter value $\theta_{f_1}(T_1)$ is fixed at the value of the tracking error $\tilde{x}(T_0)$. If the learning rate γ is very small, as compared with the order of the tracking error, then

$$V_0(T_0) + \tfrac{1}{2}\tilde{\theta}_{f_1}(T_1)\gamma\tilde{\theta}_{f_1}(T_1) \simeq V_0(T_0) \tag{42}$$

Hence, $V_1(T_1)$ has a finite value, and

$$V_1(T_1) \leq V_0(T_0) \tag{43}$$

The same analysis can be carried out for $V_j(T_j)$, such that $j = 1, 2, \ldots, N_f$ and

$$V_j(T_j) \leq V_{j-1}(T_{j-1}) \tag{44}$$

We have thus shown that the Lyapunov function $V(t)$ is continuously decreasing when either the weights or number of nodes of the SORBFN is updated. Overall, it is concluded that the error $\tilde{x}$ is also continuously decreasing until it reaches the prespecified design error ϵ. At this time, the adaptation is stopped, and the tracking error remains bounded below ϵ.

V. Application to Modified F-15 Rotational Dynamics

In this section, the application of the SORBFN-based adaptive controller to the rotational dynamics of a highly maneuverable aircraft is discussed. The simulation model used in this research is a high-fidelity six-degree-of-freedom linear model of a preproduction F-15 aircraft, which has been modified to support various test programs at NASA Dryden Flight Research Center [29]. The most visible modification to the aircraft is the addition of a set of canards near the pilot station. The canards are not used for direct longitudinal control, but it is scheduled through angle of attack and Mach number. Roll control is provided by differential ailerons and differential stabilators. Directional control is provided by rudders and limited differential canards. The simulation model, while being linear, incorporates the complete sensor and actuator dynamics. It matches extremely well with the nonlinear simulation used on the NASA flight simulator, which itself matches the actual flight very well. The block diagram of the adaptive control architecture is shown in Fig. 1.

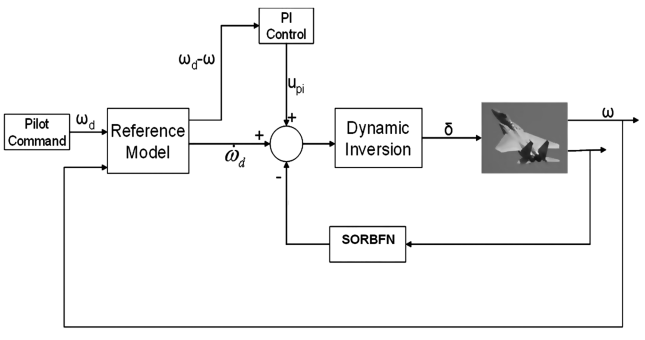


Fig. 1 Adaptive flight-control architecture.

The rotational dynamics of the aircraft can be written as

$$\dot{\omega} = -I^{-1}(\omega \times I\omega) + I^{-1}M \tag{45}$$

where $\omega = (\,p \quad q \quad r\,)$, and M is the generalized moment given by

$$M = M_{\text{aero}} + M_\delta \tag{46}$$

where I is the moment of inertia matrix. M_{aero} is the moment vector generated due to the base aerodynamics of the aircraft:

$$M_{\text{aero}} = \begin{pmatrix} L_{\text{BAE}} \\ M_{\text{BAE}} \\ N_{\text{BAE}} \end{pmatrix} \tag{47}$$

where L_{BAE}, M_{BAE}, and N_{BAE} are the base aerodynamic rolling, pitching, and yawing moments, respectively. Since deflections of the canards are scheduled according to the flight condition and angle of attack, its effect on the pitching moment is included in the base aerodynamics itself. $M_\delta = G(P)\delta$ is the moment generated due to control surface deflections, where $G(.)$ is the control effectiveness function that is dependent on parameters P such as Mach number, altitude, and angle of attack. The dynamic inversion control law for the rotational dynamics can now be written as

$$\delta = [I^{-1}G(P)]^{-1}\{\dot{\omega}_d - [-I^{-1}(\omega \times I\omega)] - (I^{-1}M_{\text{aero}})\} \tag{48}$$

where $\dot{\omega}_d = (\,\dot{p}_d \quad \dot{q}_d \quad \dot{r}_d\,)$ is the desired angular acceleration vector.

A. Reference Model

The pilot commands to the aircraft are in the form of roll and pitch stick combined with a yaw pedal command. These roll, pitch, and yaw command signals (p_d, q_d, r_d) along with their derivatives are generated using reference models for each of the three channels. The longitudinal reference model is designed to characterize the second-order short-period mode, while the lateral reference model represents the roll subsidence mode of the aircraft, and the directional reference model is implemented to characterize the second-order dutch roll mode of the aircraft.

B. Proportional–Integral Controller

The closed-loop system, when the dynamic inversion controller given by Eq. (48), is implemented is given by

$$\dot{\omega} = \dot{\omega}_d \tag{49}$$

If the approximation of the base aerodynamic moments is accurate, then the closed-loop dynamics follow the desired dynamics exactly. However, to meet performance specifications and pilot handling qualities, a proportional-plus-integral controller is implemented for each of the channels of roll, pitch, and yaw. The gains are chosen so as to satisfy pilot handling qualities. An integral antiwindup algorithm [30] is also included to back off the actuator when it is saturated. The inner loop dynamic control law and outer loop proportional integral controller can be assumed to decouple the closed-loop dynamics, which are then given by

$$\dot{p} = \dot{p}_d - K_{pp}(p - p_d) - K_{ip}\int p - p_d \tag{50}$$

$$\dot{q} = \dot{q}_d - K_{pq}(q - q_d) - K_{iq}\int q - q_d \tag{51}$$

$$\dot{r} = \dot{r}_d - K_{pr}(r - r_d) - K_{ir}\int r - r_d \tag{52}$$

For the roll rate, defining an error

$$e_p = \begin{bmatrix} \int (p - p_d) & p - p_d \end{bmatrix}^T$$

the closed-loop system is

$$\dot{e}_p = \begin{bmatrix} 0 & 1 \\ -K_{ip} & -K_{pp} \end{bmatrix} e_p \tag{53}$$

Similar equations can be written for pitch and yaw velocity components.

C. Control Allocator

A control allocator is implemented to distribute the required aerodynamic moments among the actuators. Since the canards are scheduled according to the flight condition (Mach number and altitude) and the angle of attack, the control allocator allocates the moments among the aileron, rudder, and stabilator. Ideally, the deflections are calculated using the equation

$$\delta = [I^{-1}G(P)]^{-1}\left\{\dot{\omega}_d - [-I^{-1}(\omega \times I\omega)] - (I^{-1}M_{\text{aero}}) - K_p(\omega - \omega_d) - K_i\int(\omega - \omega_d)\right\} \tag{54}$$

where $G(P)^{-1}$ is the generalized pseudoinverse. Since actuator position constraints must to be taken into consideration, the allocation of the moments is achieved through a constrained optimization problem given by

$$\min_{\delta} J_1 = \|I^{-1}G(P)\delta - M_{\text{alloc}}\|, \quad \text{subject to } \delta_{\min} \le \delta \le \delta_{\max}$$

where $\delta_{\min}$ and $\delta_{\max}$ represent the most restrictive of the instantaneous rate or position limits of the actuator for a given flight-control system update point, and

$$M_{\text{alloc}} = \dot{\omega}_d - [-I^{-1}(\omega \times I\omega)] - (I^{-1}M_{\text{aero}}) - K_p(\omega - \omega_d) - K_i\int(\omega - \omega_d)$$

D. Control Hedging

Input saturation and input rate saturation are significant issues in adaptive control, since adaptation during saturation may result in instability of the system. Several methods, including reducing the learning rate and bounding the feedback control [31], have been suggested to alleviate this problem. In this paper, pseudocontrol hedging based on [32,33] is implemented to avoid adaptation during actuator saturation. The principle behind this method is to modify the reference model based on the error between the control moment desired by the adaptive controller and the actual control moment available due to the saturated input. A control hedging signal v_h is generated as

$$v_h = M_{\text{alloc}} - I^{-1}G(P)\delta$$

A new reference or desired angular acceleration $\dot{\omega}_{d_{\text{new}}}$ is calculated using

$$\dot{\omega}_{d_{\text{new}}} = \dot{\omega}_d - v_h$$

If the actuator is not saturated, this signal is zero, and the reference angular acceleration is not modified. If, however, the actuator is saturated, then a nonzero control hedging signal is generated that modifies the desired dynamics.

E. Dynamic Inversion Errors

The dynamic inversion controller ideally decouples the closed loop into roll, pitch, and yaw channels, as discussed earlier. The stability of these decoupled dynamics is dependent on the proportional and integral gains. The analysis in the previous section, however, made an assumption that knowledge of the moments due to the base aerodynamics M_{aero}, and hence the inversion model, is accurate. However, uncertainties in the stability derivatives cause errors in the dynamic inversion. Additionally, other factors that may cause errors in the inversion are external disturbances and control surface failures. These errors can be generalized as an unknown nonlinear function of the rotational velocities $\Delta(\omega)$. Under this assumption, the closed-loop system in each channel is

$$\dot{e}_i = A_i e_i + B_i\Delta_i(\omega) \qquad i = p, q, r \tag{55}$$

where $\Delta_i(.)$ is a scalar function of the rotational velocities.

F. Adaptive Component

To minimize the effect of the error on the closed loop, the baseline controller is augmented with adaptive approximators. The adaptive approximator implemented is the self-organizing RBF network discussed earlier. Since it is assumed that the dynamic inversion control law decouples the three rotational dynamics of motion, the implementation of the SORBFN on only the roll channel is discussed. The same implementation and analysis can be extended to the pitch and yaw channels. Consider the closed-loop equation of the roll channel:

$$\dot{e}_p = A_p e_p + B_p\Delta_p(\omega) \tag{56}$$

where

$$A_p = \begin{pmatrix} 0 & 1 \\ -K_{ip} & -K_{pp} \end{pmatrix} \qquad B_p = (0 \quad 1)^T$$

and $\Delta_p(\omega)$ is the dynamic inversion error in the roll channel. Consider an adaptive approximator that is introduced into the dynamic inversion controller such that

$$\delta = [I^{-1}G(P)]^{-1}\left[\dot{\omega}_d - f(\omega) - I^{-1}M_{\text{aero}} - K_p(\omega - \omega_d) - K_i\int(\omega - \omega_d) - \hat{\Delta}(\omega)\right] \tag{57}$$

where

$$\hat{\Delta}(\omega) = [\,\hat{\Delta}_p(\omega) \quad \hat{\Delta}_q(\omega) \quad \hat{\Delta}_r(\omega)\,]$$

The closed loop for the roll channel with the adaptive approximator is given by

$$\dot{e}_p = A_p e_p + B_p\Delta_p(\omega) - B_p\hat{\Delta}_p(\omega) \tag{58}$$

Let us consider a Lyapunov function given by

$$V_j(t) = V_0(t) + V_\theta(t) = e_p^T P e_p + \sum_{i=1}^{j}\gamma\tilde{\theta}_i^2 \tag{59}$$

where $\gamma > 0$, and P is a positive definite matrix such that

$$A_p^T P + PA_p = -Q, \qquad Q > 0$$

since A_p is Hurwitz. Consider the situation when there are no errors in the dynamic inversion controller. Then, $\Delta_p(\omega) = 0$. With no nodes in the network ($j = 0$), the Lyapunov function is reduced to $V_0(t)$. Evaluating its derivative

$$\dot{V}_0(t) = e_p^T P\dot{e}_p + \dot{e}_p^T P e_p \qquad \dot{V}_0(t) = -e_p^T Q e_p$$

Since Q is a positive definite matrix, the Lyapunov derivative is negative definite, which implies that the error signal e_p asymptotically converges to zero. Now, consider the situation in which the error in dynamic inversion $\Delta_p(\omega) \neq 0$. In this case, the Lyapunov derivative is given by

$$\dot{V}_0(t) = e_p^T P\dot{e}_p + \dot{e}_p^T P e_p \qquad \dot{V}_0(t) = -e_p^T Q e_p + 2e_p^T PB_p\Delta_p(\omega)$$

$$\dot{V}_0(t) \le -\|e_p\|^T[\lambda_{\max}(Q)\|e_p\| - 2\|PB_p\Delta_p(\omega)\|]$$

Therefore, the derivative of the Lyapunov function is dependent on the magnitude of the dynamic inversion error. If

$$\|e_p\| > \frac{2\|PB_p\Delta_p(\omega)\|}{\lambda_{\max}(Q)}$$

then there is no necessity for an approximating the function to cancel the effect of the dynamic inversion error. However, if the preceding condition is not satisfied, the effect of the dynamic inversion error is to increase the Lyapunov function. This is one of the conditions used to add a node to the network. Consider the situation when j nodes have been added to the network. Then, the derivative of the Lyapunov function $V_j(t)$ is evaluated to analyze the stability of the closed-loop system. Assume that the unknown dynamic inversion error $\Delta_p(\omega)$ can be parameterized with j basis functions as

$$\Delta_p(\omega) = \sum_{i=1}^{j} \theta_i^* \phi_i(\omega) + \varepsilon(\omega)$$

Let

$$\hat{\Delta}_p(\omega) = \sum_{i=1}^{j} \hat{\theta}_i \phi_i(\omega) \qquad \tilde{\theta}_i = \theta_i^* - \hat{\theta}_i \qquad \|\varepsilon(\omega)\| \le \bar{\varepsilon}$$

Evaluating the derivative of the Lyapunov function $V_j(t)$,

$$\dot{V}_j(t) = e_p^T P \dot{e}_p + \dot{e}_p^T P e_p + \sum_{i=1}^{j} 2\gamma \tilde{\theta}_i \dot{\tilde{\theta}}_i$$

$$\dot{V}_j(t) = -e_p^T Q e_p + 2e_p^T P B_p \sum_{i=1}^{j} \tilde{\theta}_i \phi_i(\omega) + 2\sum_{i=1}^{j} \gamma \tilde{\theta}_i \dot{\tilde{\theta}}_i + 2e_p^T P B_p \varepsilon(\omega)$$

$$\dot{V}_j(t) = -e_p^T Q e_p + 2\sum_{i=1}^{j} \tilde{\theta}_i [\phi_i(\omega) B_p^T P e_p + \gamma \dot{\tilde{\theta}}_i] + 2e_p^T P B_p \varepsilon(\omega)$$

Choosing the adaptation law as

$$\dot{\tilde{\theta}}_i = -\gamma^{-1} \phi_i(\omega) B_p^T P e_p$$

we have the Lyapunov derivative as

$$\dot{V}_j(t) = -e_p^T Q e_p + 2e_p^T P B_p \varepsilon_j(\omega)$$

$$\dot{V}_j(t) \le -\lambda_{\max}(Q)\|e_p\|^2 + 2\|\varepsilon_j(\omega)\|\|B_p^T P\|\|e_p\|$$

$$\dot{V}_j(t) \le -\|e_p\|(\lambda_{\max}(Q)\|e_p\| - \bar{\varepsilon}_j\|B_p^T P\|)$$

The negative (semi) definiteness of the Lyapunov derivative is dependent on the condition that

$$\|e_p\| \ge \frac{\bar{\varepsilon}\|B_p^T P\|}{\lambda_{\max}(Q)}$$

If the tracking error e_p is not smaller than the prespecified tracking error, an additional node is added, as discussed in Sec. III.

G. Implementation

The SORBFN is implemented as the adaptive approximator in each of the angular velocity channels. The training algorithm for the SORBFN includes strategies for the addition and deletion of nodes along with updating its weights. Nodes are added to the individual networks (roll, pitch, or yaw) based on the Lyapunov function $V_0(t)$ evaluated in that channel. For example, in the roll channel, $V_0(t)$ is given by

$$V_0(t) = e_p^T P e_p \tag{60}$$

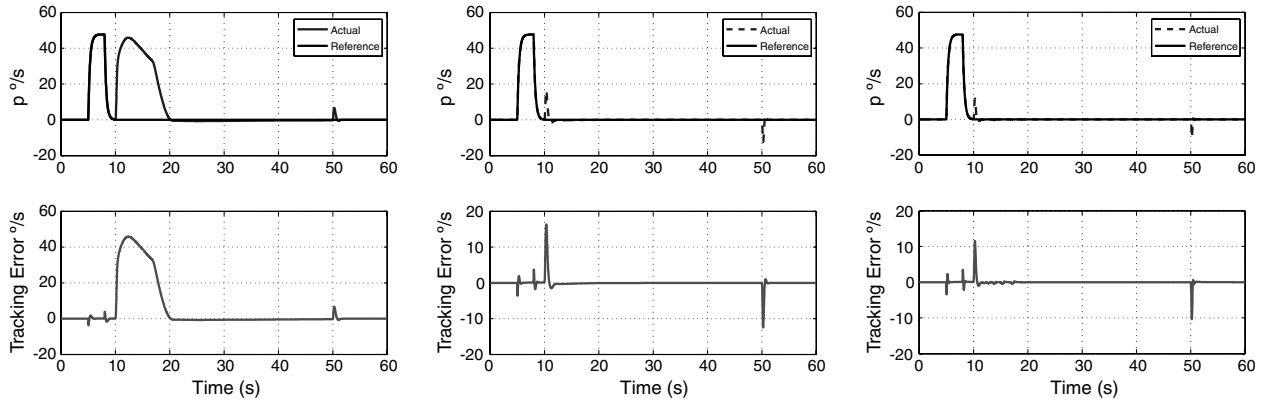
The Lyapunov matrix P is calculated by considering Q to be an identity matrix. The neural network in the roll, pitch, and yaw axes are trained independently. The inputs to the three networks include all the angular velocities (p, q, r), while the update of weights is based on the error in that channel. Thus, the network is trained with information from the entire state space but adapted only to error in the particular channel. In the simulation results presented, the free parameters (i.e., learning rate, node addition threshold N_a, pruning threshold N_d, and variances σ) are chosen to be the same for all three networks.

VI. Results

Simulation results of the implementation of the SORBFN-based adaptive controller on a F-15 aircraft are presented in this section. The controller is simulated for the flight condition at altitude 25,000 ft and Mach 0.9 and a desired roll-rate input command. The performance of the SORBFN augmented dynamic inversion controller under different conditions of modeling uncertainties, disturbances, and actuator failures is discussed and compared with a dynamic inversion controller and a fixed structure RBF network augmented dynamic inversion controller. The fixed structure RBF network is implemented similar to the SORBFN in the three channels of the rotational dynamics. Each channel of the fixed network consists of 36 RBFs for which the centers are equally distributed on the state space that the flight dynamics are perceived to operate. The SORBFN is initialized to zero nodes, and the appropriate learning rates γ, node addition threshold N_a, and prune threshold N_d are chosen appropriately. It is desired that the controllers maintain an angular velocity tracking error below $1^\circ/\text{s}$.

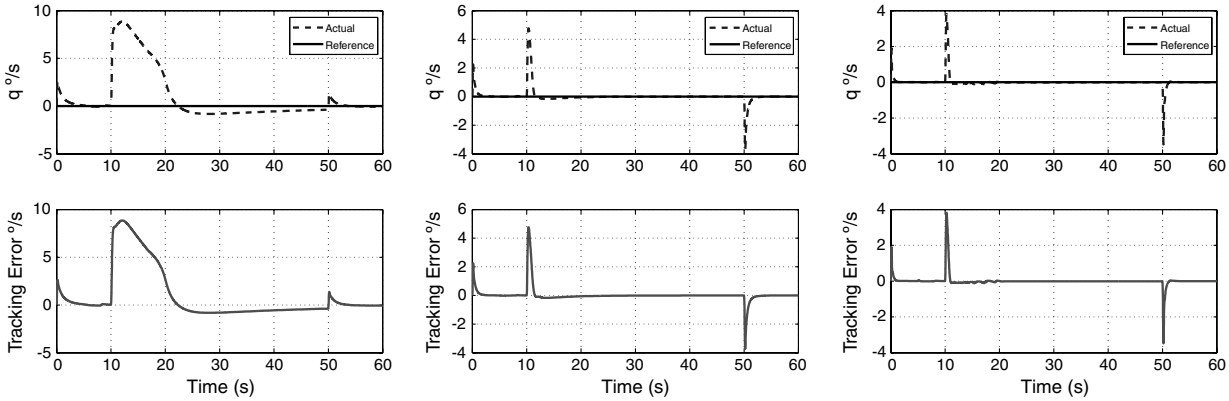
A. Control Surface Failure

In this section, locked effector-type faults are considered. From extensive simulation, locked effector faults of the stabilators are



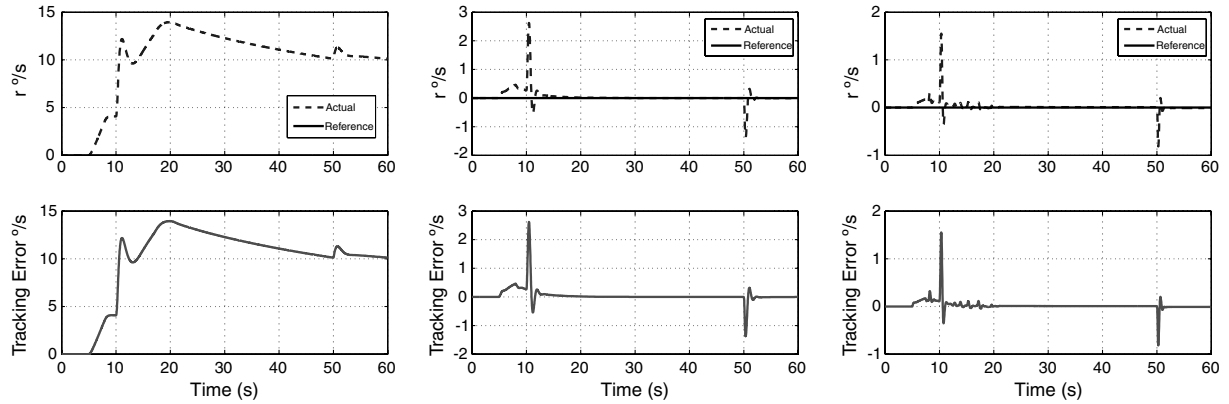
a) Dynamic inversion b) Dynamic inversion with fixed RBF c) Dynamic inversion with SORBFN

Fig. 2 Roll-rate tracking with right stabilator locked at 10° below trim, from 10 to 50 s.



a) Dynamic inversion b) Dynamic inversion with fixed RBF c) Dynamic inversion with SORBFN

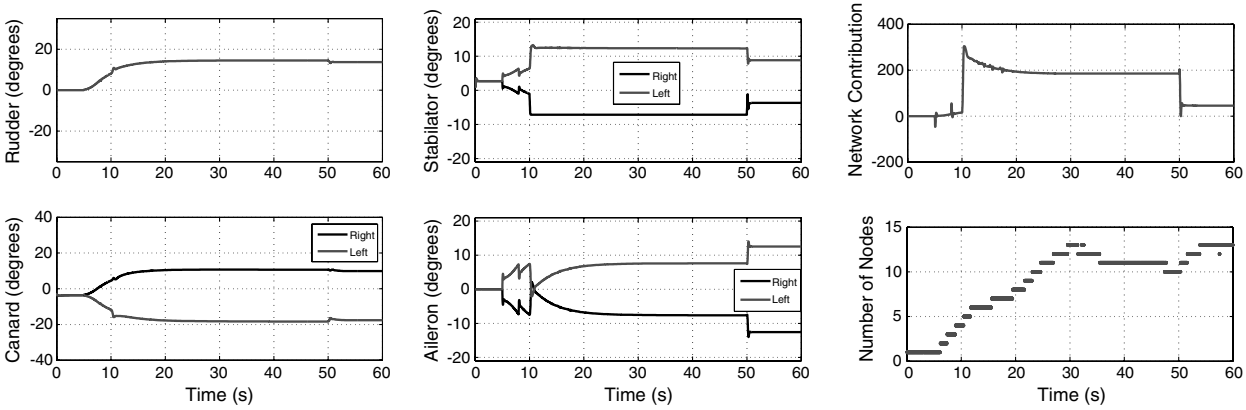
Fig. 3 Pitch-rate tracking with right stabilator locked at 10° below trim, from 10 to 50 s.



a) Dynamic inversion b) Dynamic inversion with fixed RBF c) Dynamic inversion with SORBFN

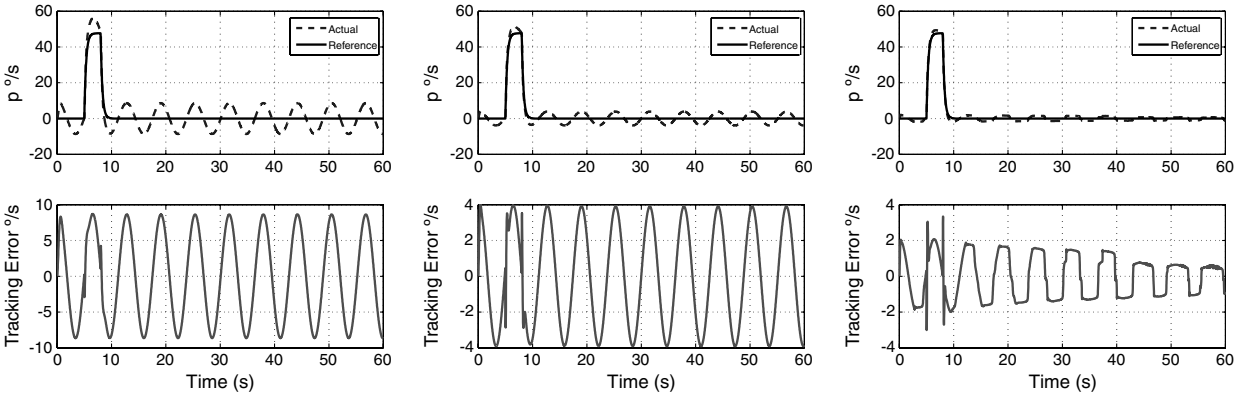
Fig. 4 Yaw-rate tracking with right stabilator locked at 10° below trim, from 10 to 50 s.

found to have the most adverse effect on the aircraft. Differential stabilators are used for roll control, and symmetric stabilators are used for pitch control on the modified F-15 aircraft. Hence, a stabilator failure will significantly affect both roll- and pitch-rate tracking. The right stabilator is considered to get locked at 10° below trim at 10 s into the simulation. The fault is assumed to be corrected at 50 s. The trim value of the stabilator for the given flight condition is 2.86°. Figures 2–4 show the angular velocity tracking performance. The dynamic inversion controller without an adaptive component suffers greatly due to the control surface failure at 10 s. Comparatively, both the controllers with the fixed RBF and SORBFN adaptive component recover from the failure quickly, and the error in the angular velocity tracking is very low in all three channels. While it is evident that both the fixed structure RBF network and the SORBFN perform equally well in compensating for the errors due to control surface failure, the maximum tracking error (even though not significant) is greater in the fixed RBF controller as compared with the SORBFN-based controller. Figure 5c shows the network contribution and the node allocation (addition and deletion) capabilities for the SORBFN as a function of time in the roll channel. It can be seen that the number of nodes required to compensate for the control surface failure is less than 15, as compared with the 36 prechosen nodes for the fixed structure network.



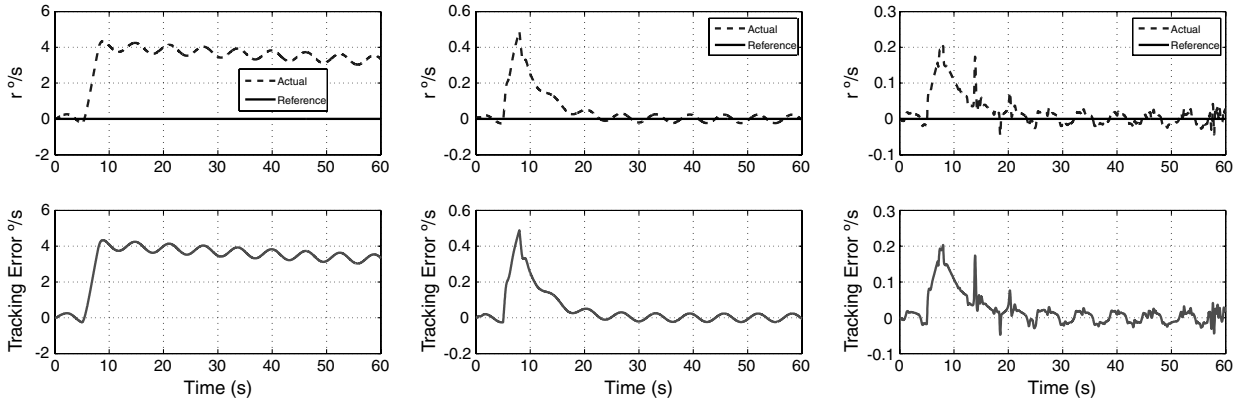
a) Rubber and Canard b) Stabilator and Aileron c) Roll channel SORBFN

Fig. 5 Control surface deflection and network contribution with right stabilator locked at 10° below trim, from 10 to 50 s.



a) Dynamic inversion **b) Dynamic inversion with fixed RBF** **c) Dynamic inversion with SORBFN**

Fig. 6 Roll-rate tracking with disturbance in roll channel.



a) Dynamic inversion **b) Dynamic inversion with fixed RBF** **c) Dynamic inversion with SORBFN**

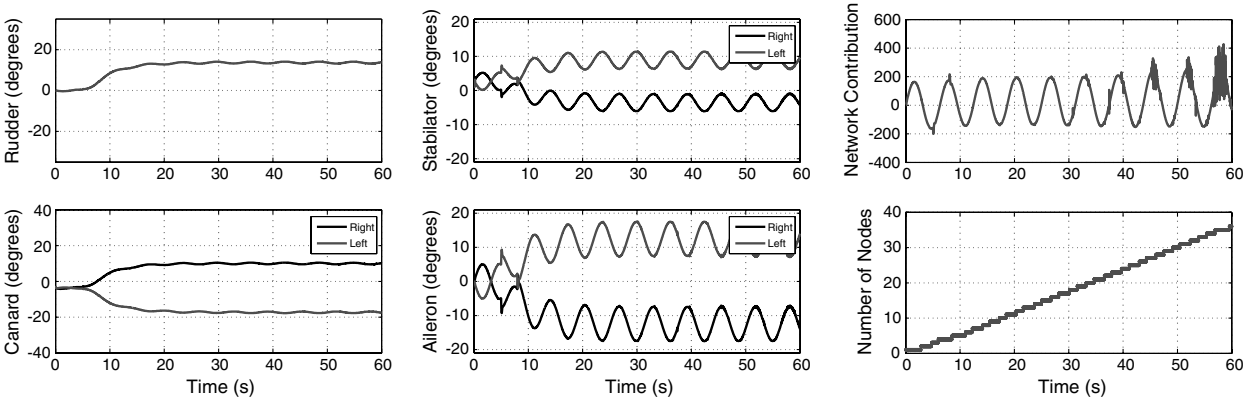
Fig. 7 Yaw-rate tracking with disturbance in roll channel.

B. Disturbance Rejection

The ability of the adaptive controllers to satisfactorily track the reference signal in the presence of a disturbance signal is now considered. It is assumed that a sinusoidal disturbance of a specified amplitude enters the rotational dynamics through the rolling moment equation. The fixed structure network has the same structure (36 nodes) as the ones used to compensate for the error due to control surface failure. The SORBFN has been reinitialized to have zero nodes at the start of simulation. Figures 6 and 7 show the roll- and yaw-rate tracking performance of the controllers. The pitch-rate tracking performance is not shown, since there is minimum coupling between the longitudinal and lateral dynamics in the linear model. The performance of the SORBFN controller is much better as compared with the controller with the fixed structure network and the dynamic inversion controller. The error in roll-rate tracking continues to decrease with the SORBFN controller and reaches a value below $1^\circ/s$, while with the fixed structure RBF network, the amplitude of the tracking error remains constant at $4^\circ/s$. Similar effects can also be seen in the yaw-rate tracking performance of the controllers. The control surface deflections for the SORBFN controller and roll channel network adaptation are shown in Fig. 8.

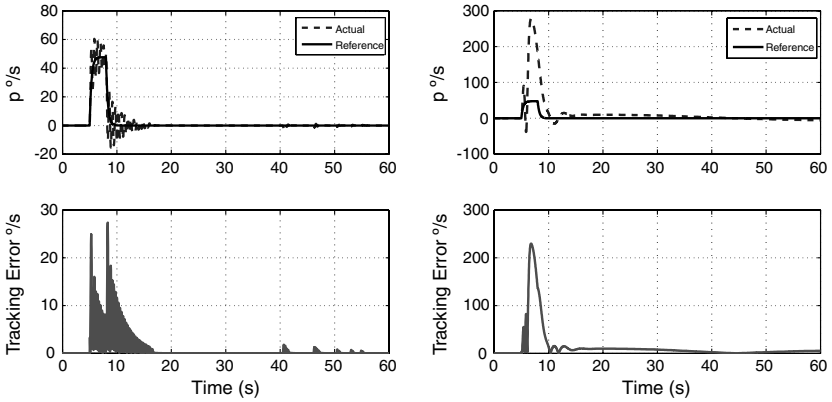
C. Uncertainty in Rolling Moment Stability Derivative

The ability of the SORBFN controller to compensate for modeling errors due to uncertainty in the stability derivatives is now discussed. In the dynamic inversion control law given by Eq. (54),



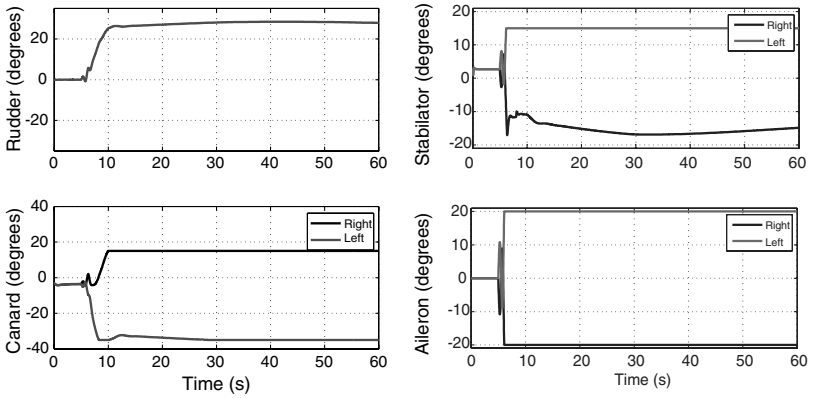
a) Rubber and Canard **b) Stabilator and Aileron** **c) Roll channel SORBFN**

Fig. 8 Control surface deflection and network contribution with sinusoidal disturbance in roll channel.



a) With SORBFN **b) With fixed RBF**

Fig. 9 Roll-rate tracking with uncertainty in base aerodynamic rolling moment.

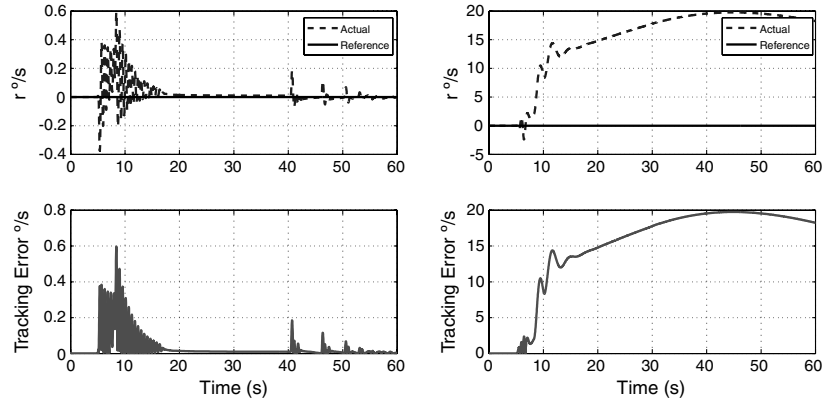


a) Rudder and Canard **b) Stabilator and Aileron**

Fig. 10 Control surface deflection with uncertainty in base aerodynamic rolling moment for fixed RBF controller.

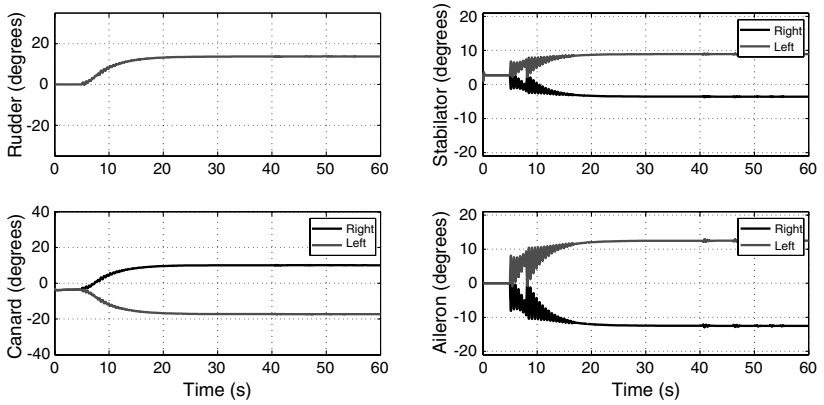
$$M_{\text{aero}} = \begin{pmatrix} L_{\text{BAE}} \\ M_{\text{BAE}} \\ N_{\text{BAE}} \end{pmatrix} = \begin{pmatrix} \bar{q}Sb\left(C_{l_\beta}\beta + \frac{b}{2V}C_{l_p}p + \frac{b}{2V}C_{l_r}r\right) \\ \bar{q}S\bar{c}\left(C_{m_\alpha}\alpha + \frac{\bar{c}}{2V}C_{m_q}q + C_{m_c}\delta_c\right) \\ \bar{q}Sb\left(C_{n_\beta}\beta + \frac{b}{2V}C_{n_p}p + \frac{b}{2V}C_{n_r}r\right) \end{pmatrix} \quad (61)$$

where $\bar{q}$ is the dynamic pressure, α is the angle of attack, β is the sideslip, S is the wing area, b is the wingspan, V is the velocity, and C_{l_*}, C_{m_*}, and C_{n_*} are the rolling, pitching, and yawing moment stability derivatives, respectively. Uncertainty is introduced into the base rolling moment L_{BAE} by changing the value stability derivative C_{l_p} used for dynamic inversion. Figure 9b shows the roll-rate tracking abilities of the adaptive controller with the fixed RBF under the effect of uncertainty in the rolling moment. The controller is unable to track the roll rate effectively. It can be seen that, when the desired roll rate changes during the time period from 5 to 10 s, the tracking error is significantly large. From the control surface deflections in Fig. 10b, it can be noted that the ailerons are saturated at the same time. The ailerons saturate because the fixed RBF is unable to cancel the effect of the uncertainty and the desired moment required to track the angular velocity as obtained from the dynamic inversion is incorrect. Because of this saturation, the control hedging



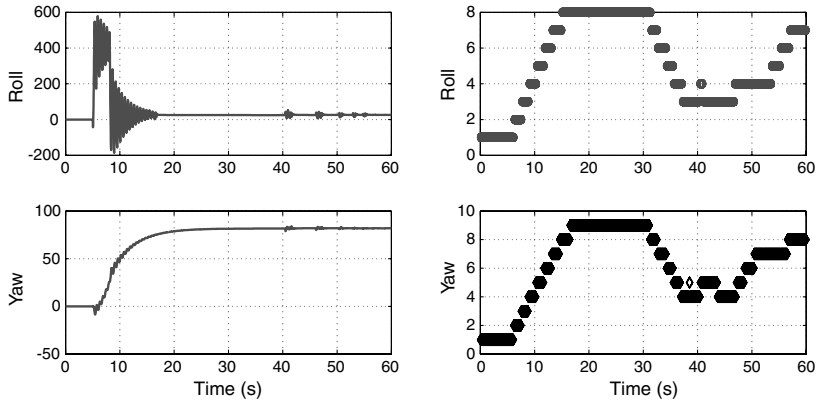
a) With SORBFN **b) With fixed RBF**

Fig. 11 Yaw-rate tracking with uncertainty in base aerodynamic rolling moment.



a) Rudder and Canard **b) Stabilator and Aileron**

Fig. 12 Control surface deflection with uncertainty in base aerodynamic rolling moment for SORBFN controller.



a) Network contribution **b) Number of nodes**

Fig. 13 SORBFN performance in roll and yaw channels, with uncertainty in base aerodynamic rolling moment.

component now modifies the reference model such that the adaptation of the weights of the fixed RBF is stopped. Since the fixed RBF cannot continue to adapt, the performance of the controller deteriorates, and the tracking error continues to be large. This can also be seen in the yaw-rate tracking performance in Fig. 11b. In comparison, the adaptive controller with SORBFN performs very well and is able to track the roll rate and yaw rate quite accurately, as seen in Figs. 9a and 11a. This can be attributed to the function approximation capabilities of the SORBFN, which cancels out the effect of the uncertainty in the model. Figures 10 and 12 show the control surface deflections for the two controllers. Figure 13 illustrates the SORBFN contribution and the number of nodes in the roll and yaw channels.

VII. Conclusions

The design of an adaptive controller that augments a dynamic inversion controller with a SORBFN is presented in this paper. The performance of the SORBFN-based controller is demonstrated with its application to the rotational dynamics of a highly maneuverable aircraft, where it is shown to compensate for errors in dynamic inversion that may be caused due to modeling uncertainties, disturbances, or control surface failures. The SORBFN-based controller is compared with an adaptive controller augmented with a fixed structure RBF network, which has been designed to compensate for control surface failures. The comparative simulations for both controllers, performed under uncertainties and external disturbances, show the superior ability of the SORBFN in achieving prespecified tracking error convergence.

References

[1] Farrell, J., and Polycarpou, M., "Adaptive Approximation Based Control: General Theory," *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*, Wiley-Interscience, New York, 2006, pp. 285–288.

[2] Hu, Q., "Neural Network-Based Adaptive Attitude Tracking Control For Flexible Spacecraft With Unknown High-Frequency Gain," *International Journal of Adaptive Control and Signal Processing*, Vol. 24, No. 6, 2009, pp. 477–489. doi:10.1002/acs.1140

[3] Huang, S., Tan, K., Lee, T., and Putra, A., "Adaptive Control Of Mechanical Systems Using Neural Networks," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 37, No. 5, 2007, pp. 897–903. doi:10.1109/TSMCC.2007.900660

[4] Lavretsky, E., and Hovakimyan, N., "Adaptive Dynamic Inversion For Nonaffine-in-Control Uncertain Systems via Time-Scale Separation," *Journal of Dynamical and Control Systems*, Vol. 14, No. 1, 2008, pp. 33–41. doi:10.1007/s10883-007-9033-5

[5] Li, C.-Y., Jing, W.-X., and Gao, C.-S., "Adaptive Backstepping-Based Flight Control System Using Integral Filters," *Aerospace Science and Technology*, Vol. 13, No. 2, 2009, pp. 105–113. doi:10.1016/j.ast.2008.05.002

[6] Wang, D., Huang, J., Lan, W., and Li, X., "Neural Network-Based Robust Adaptive Control Of Nonlinear Systems With Unmodeled Dynamics," *Mathematics and Computers in Simulation*, Vol. 79, No. 5, 2009, pp. 1745–1753. doi:10.1016/j.matcom.2008.09.002

[7] Stengel, R., "Toward Intelligent Flight Control," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 23, No. 6, Nov.–Dec. 1993, pp. 1699–1717. doi:10.1109/21.257764

[8] Pashilkar, A., Sundararajan, N., and Saratchandran, P., "Neuro Controller for Aircraft Auto Landing Under Severe Winds and Control Surface Failures," AIAA Paper 2005-0914, Jan. 2005.

[9] Shankar, P., Yedavalli, R., and Burken, J., "An Adaptive Flight Controller Using Growing and Pruning Radial Basis Function Network," AIAA Paper 2006-6415, Aug. 2006.

[10] Nguyen, N., Nejad, M., and Huang, Y., "Hybrid Adaptive Flight Control With Bounded Linear Stability Analysis," AIAA Paper 2007-6422, Aug. 2007.

[11] Napolitano, M., and Kincheloe, M., "On-Line Learning Neural-Network Controllers for Autopilot Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 5, Nov.–Dec. 1995, pp. 1008–1015.
doi:10.2514/3.21498

[12] Calise, A., "Development Of A Reconfigurable Flight Control Law for the X-36 Tailless Fighter Aircraft," AIAA Paper 2000-3940, Aug. 2000.

[13] McFarland, M., and Calise, A., "Adaptive Nonlinear Control of Agile Antiair Missiles Using Neural Networks," *IEEE Transactions on Control Systems Technology*, Vol. 8, No. 5, Sept. 2000, pp. 749–756.
doi:10.1109/87.865848

[14] Shin, D.-H., and Kim, Y., "Reconfigurable Flight Control System Design Using Adaptive Neural Networks," *IEEE Transactions on Control Systems Technology*, Vol. 12, No. 1, Jan. 2004, pp. 87–100.
doi:10.1109/TCST.2003.821957

[15] Kurnaz, S., Cetin, O., and Kaynak, O., "Adaptive Neuro-Fuzzy Inference System Based Autonomous Flight Control of Unmanned Air Vehicles," *Expert Systems with Applications*, Vol. 37, No. 2, 2010, pp. 1229–1234.
doi:10.1016/j.eswa.2009.06.009

[16] Platt, J., "A Resource-Allocating Network for Function Interpolation," *Neural Computation*, Vol. 3, No. 2, 1991, pp. 213–225.
doi:10.1162/neco.1991.3.2.213

[17] Sanner, R., and Slotine, J.-J., "Structurally Dynamic Wavelet Networks for the Adaptive Control of Uncertain Robotic Systems," *Conference on Decision and Control*, New Orleans, LA, IEEE Publ., Piscataway, NJ, Dec. 1995, pp. 2460–2467.

[18] Fabri, S., and Kadirkamanathan, V., "Dynamic Structure Neural Networks for Stable Adaptive Control of Nonlinear Systems," *IEEE Transactions on Neural Networks*, Vol. 7, No. 5, Sept. 1996, pp. 1151–1167.
doi:10.1109/72.536311

[19] Choi, J., and Farrell, J., "Nonlinear Adaptive Control Using Networks of Piecewise Linear Approximators," *IEEE Transactions on Neural Networks*, Vol. 11, No. 2, March 2000, pp. 390–401.
doi:10.1109/72.839009

[20] Nakanishi, J., Farrell, J., and Schaal, S., "A Locally Weighted Learning Composite Adaptive Controller with Structure Adaptation," *International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, IEEE Publ., Piscataway, NJ, Oct. 2002, pp. 882–889.

[21] Nakanishi, J., Farrell, J., and Schaal, S., "Composite Adaptive Control with Locally Weighted Statistical Learning," *Neural Networks*, Vol. 18, No. 1, 2005, pp. 71–90.
doi:10.1016/j.neunet.2004.08.009

[22] Yan, L., Saratchandran, P., and Sundararajan, N., "Nonlinear System Identification Using Lyapunov Based Fully Tuned Dynamic RBF Networks," *Neural Processing Letters*, Vol. 12, No. 3, 2000, pp. 291–303.
doi:10.1023/A:1026571426761

[23] Huang, G.-B., Saratchandran, P., and Sundararajan, N., "An Efficient Sequential Learning Algorithm for Growing and Pruning RBF (GAP-RBF) Networks," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 34, No. 6, Dec. 2004, pp. 2284–2292.
doi:10.1109/TSMCB.2004.834428

[24] Huang, G.-B., Saratchandran, P., and Sundararajan, N., "A Generalized Growing and Pruning RBF (GGAP-RBF) Neural Network for Function Approximation," *IEEE Transactions on Neural Networks*, Vol. 16, No. 1, Jan. 2005, pp. 57–67.
doi:10.1109/TNN.2004.836241

[25] Yan, L., Saratchandran, P., and Sundararajan, N., "MRAN Neuro-Flight Controller For Robust Aircraft Control," *Fully Tuned Radial Basis Function Neural Networks for Flight Control*, Kluwer Academic, Norwell, MA, 2002, pp. 127–139.

[26] Farrell, J., and Zhao, Y., "Self-Organizing Approximation-Based Control," *IEEE Transactions on Neural Networks*, Vol. 18, No. 4, July 2007, pp. 1220–1231.
doi:10.1109/TNN.2007.899217

[27] Farrell, J., "Persistence of Excitation Conditions in Passive Learning Control," *Automatica*, Vol. 33, No. 4, 1997, pp. 699–703.
doi:10.1016/S0005-1098(96)00203-8

[28] Park, J., and Sandberg, I. W., "Universal Approximation Using Radial-Basis-Function Networks," *Neural Computation*, Vol. 3, No. 2, 1991, pp. 246–257.
doi:10.1162/neco.1991.3.2.246

[29] Burken, J., Kaneshige, J., and Stachowiak, S., "Reconfigurable Control With Neural Network Augmentation for a Modified F-15 Aircraft," NASA TM 2006-213678, 2006.

[30] Williams-Hayes, P., "Flight Test Implementation of a Second Generation Intelligent Flight Control System," NASA TM 2005-213669, 2005.

[31] Rovithakis, G., "Nonlinear Adaptive Control in the Presence of Unmodelled Dynamics Using Neural Networks," *Conference on Decision and Control*, Phoenix, AZ, Vol. 3, IEEE Publ., Piscataway, NJ, Dec. 1999, pp. 2150–2155.

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

[33] Johnson, E., Calise, A., and Corban, J., "Adaptive Guidance and Control for Autonomous Launch Vehicles," *Aerospace Conference*, Big Sky, MT, IEEE Publ., Piscataway, NJ, March 2001, pp. 2669–2682.