JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 20, No. 5, September–October 1997

Analysis of Adaptive Neural Networks for Helicopter Flight Control

Jesse Leitner*
Air Force Research Laboratory, Kirtland Air Force Base, New Mexico 87117-5776
and
Anthony Calise† and J. V. R. Prasad‡
Georgia Institute of Technology, Atlanta, Georgia 30332-0150

The design of online adaptive neural networks for use in a nonlinear helicopter flight control architecture is treated. Emphasis is given to network architecture and the effect that varying the adaptation gain has on performance. Conclusions are based on a six-degree-of-freedom nonlinear evaluation model of an attack helicopter and a metric that measures the network's ability to cancel the effect of modeling errors for a complicated maneuver. The network is shown to provide nearly perfect tracking in the face of significant modeling errors and, additionally, to cancel the model inversion error after a short initial period of learning. Furthermore, it is shown that the performance varies gracefully and monotonically improves as the adaptation gain parameter is increased. The effect on control effort is modest and is mainly perceptible only during a short training episode that can be associated with transition from hover to forward flight.

I. Introduction

NONLINEAR control of helicopters by the method of feedback inversion has been extensively explored in the literature. Examples of studies based on simulated responses and actual flight tests may be found in Refs. 1 and 2, whereas the baseline controller structure presented here evolves from Ref. 3. Nonlinear control techniques require accurate modeling and employ complicated methods for inverting tabulated aerodynamic data. We follow the approaches detailed in Ref. 4 in which a nominal inverting controller is designed based on a single flight condition, with augmentation provided by an online adaptive neural network. The network functions to cancel the effects of inversion error caused by modeling errors and variations that occur throughout the flight envelope.

The focus is on developing a systematic approach to selecting the network architecture. The main contributions are as follows: 1) an analysis of the inversion error, which is used to decide the neural network architecture; 2) an evaluation of several classes of basis functions, which are derived from the inversion error analysis; and 3) a demonstration that performance varies gracefully and monotonically improves as the adaptation gain parameter is increased from zero to very high values. This last feature is important in that it provides a safe and reliable approach to flight test evaluation of online adaptive neural networks.

II. Model Inversion Control and Online Neural Networks

A. Linearizing Transformation

Consider a nonlinear system of n degrees of freedom in general form:

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

where $y(t)$ and $\dot{y}(t) \in R^n$ are the state variables and $\delta(t) \in R^n$ is the control variable. It is convenient to define a pseudocontrol variable $\boldsymbol{U}_s \in R^n$, such that

$$\ddot{\boldsymbol{y}} = \boldsymbol{U}_s \tag{2}$$

$$\boldsymbol{U}_s = f(\boldsymbol{y}, \dot{\boldsymbol{y}}, \boldsymbol{\delta}) \tag{3}$$

Then, if f is an invertible mapping with respect to the control $\boldsymbol{\delta}$, the inverse of Eq. (1) can be expressed as

$$\boldsymbol{\delta} = f^{-1}(\boldsymbol{y}, \dot{\boldsymbol{y}}, \boldsymbol{U}_s) \tag{4}$$

In practice the function f (or f^{-1}) is only known approximately. It may be an analytical or empirical model for the dynamics, and it may be a set of equations, a tabular representation, or a neural network model developed based on off-line training. Thus, the inverse is only approximate, in general, and results in an approximate model inversion control law

$$\hat{\boldsymbol{\delta}} = \hat{f}^{-1}(\boldsymbol{y}, \dot{\boldsymbol{y}}, \boldsymbol{U}_s) \tag{5}$$

so that the closed-loop dynamics can be represented as

$$\ddot{\boldsymbol{y}} = \boldsymbol{U}_s + \boldsymbol{\chi}(\boldsymbol{y}, \dot{\boldsymbol{y}}, \boldsymbol{U}_s) \tag{6}$$

where

$$\boldsymbol{\chi}(\boldsymbol{y}, \dot{\boldsymbol{y}}, \boldsymbol{U}_s) = f(\boldsymbol{y}, \dot{\boldsymbol{y}}, \hat{\boldsymbol{\delta}}) - \hat{f}(\boldsymbol{y}, \dot{\boldsymbol{y}}, \hat{\boldsymbol{\delta}}) \tag{7}$$

and $\boldsymbol{\chi}$ represents the inversion error.

B. Neural Network-Based Adaptive Control

The adaptive control architecture is chosen as

$$\boldsymbol{U}_s(t) = \boldsymbol{U}_{pd}(t) + \ddot{\boldsymbol{y}}_c(t) - \boldsymbol{U}_{\text{ad}}(t) \tag{8}$$

where

$$\boldsymbol{U}_{pd} = K_p(\boldsymbol{y}_c - \boldsymbol{y}) + K_d(\dot{\boldsymbol{y}}_c - \dot{\boldsymbol{y}}) \tag{9}$$

is a proportional-plus-derivative control law used to shape the response, $\ddot{\boldsymbol{y}}_c$ is the commanded acceleration vector, and $\boldsymbol{U}_{ad}$ is an adaptive signal whose sole purpose is to cancel the inversion error $\boldsymbol{\chi}$. Figure 1 shows a block diagram of the feedback structure. Based on Fig. 1, the tracking error is defined as $\tilde{\boldsymbol{y}} = \boldsymbol{y}_c - \boldsymbol{y}$, and the closed-loop dynamics can be expressed as

$$\ddot{\tilde{\boldsymbol{y}}} + K_d\dot{\tilde{\boldsymbol{y}}} + K_p\tilde{\boldsymbol{y}} = \boldsymbol{U}_{\text{ad}} - \boldsymbol{\chi} \tag{10}$$

Presented as Paper 95-3268 at the AIAA Guidance, Navigation, and Control Conference, Baltimore, MD, Aug. 7–10, 1995; received Sept. 19, 1996; revision received May 1, 1997; accepted for publication May 12, 1997.

*Project Leader, Guidance, Navigation, and Control Branch. Member AIAA.
†Professor, School of Aerospace Engineering. Fellow AIAA.
‡Associate Professor, School of Aerospace Engineering. Senior Member AIAA.

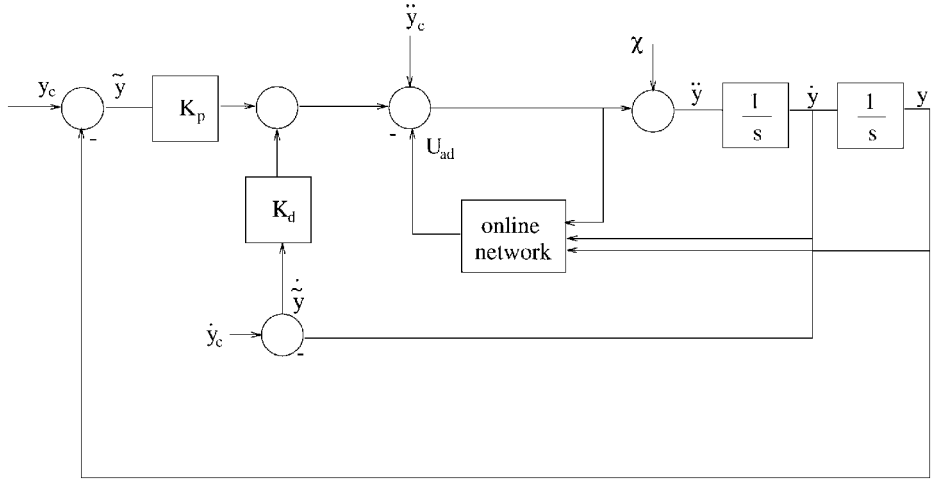

Fig. 1 Interface of neural network within inner loop controller structure.

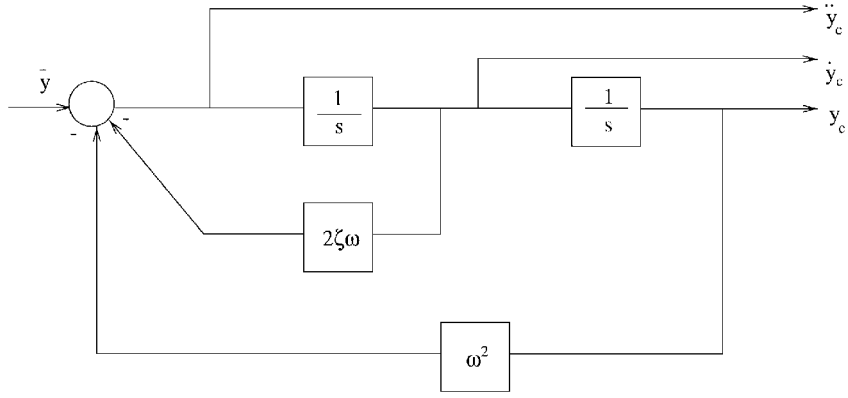

Fig. 2 Command filter simulation diagram.

III. Nominal Controller Synthesis

We consider a controller for the rotational dynamics of the helicopter. The controls for the rotational variables are lateral and longitudinal cyclic stick inputs and pedal input. The collective input is treated as a slow variable in the control law, decoupled from the moment equations. The vector $\boldsymbol{y}$ in Eq. (6) is the set of Euler angles $\boldsymbol{y} = \{\phi, \theta, \psi\}$.

In terms of the individual channels, the signal $\boldsymbol{U}_s$ is composed of the following scalar signals:

$$U_\phi = K_{p_\phi}(\phi_c - \phi) + K_{d_\phi}(\dot{\phi}_c - \dot{\phi}) + \ddot{\phi}_c - U_{\mathrm{ad}_\phi} \tag{11}$$

$$U_\theta = K_{p_\theta}(\theta_c - \theta) + K_{d_\theta}(\dot{\theta}_c - \dot{\theta}) + \ddot{\theta}_c - U_{\mathrm{ad}_\theta} \tag{12}$$

$$U_\psi = K_{p_\psi}(\psi_c - \psi) + K_{d_\psi}(\dot{\psi}_c - \dot{\psi}) + \ddot{\psi}_c - U_{\mathrm{ad}_\psi} \tag{13}$$

where the dynamics for the Euler angles in the transformed system are specified by the gains K_p and K_d in each channel in accordance with Eq. (10).

A. Rotational Dynamics Inversion

Consider the body axis rotational dynamics based on linearized aerodynamics:

$$\dot{\omega} = A_1 \Delta \boldsymbol{z} + A_2 \boldsymbol{\omega} + B \Delta \boldsymbol{\eta} + \boldsymbol{f}(\boldsymbol{\omega}) \tag{14}$$

where $\boldsymbol{\omega}$ is the body angular velocity vector $\{p, q, r\}$, $\boldsymbol{f}(\omega)$ represents the inertial nonlinear moment terms, $\boldsymbol{z}$ is the vector of translational states and the collective variable $\{u, v, w, \delta_{\mathrm{col}}\}$, $\boldsymbol{\eta}$ is the vector of moment controls $\{\delta_{\mathrm{lat}}, \delta_{\mathrm{lon}}, \delta_{\mathrm{ped}}\}$, and Δ represents a perturbation from trim value. The model inversion control law gives the required perturbation controls,

$$\Delta \boldsymbol{\eta} = B^{-1}\{\dot{\boldsymbol{\omega}}_c - A_1 \Delta \boldsymbol{z} - A_2 \boldsymbol{\omega} - \boldsymbol{f}(\boldsymbol{\omega})\} \tag{15}$$

where the body axis pseudocontrols $\dot{\omega}_c$ are determined by transforming the vector $\boldsymbol{U}_s$ from the Euler frame into the body frame. In terms of the individual components,

$$\begin{aligned} \dot{p}_c &= U_\phi - U_\psi \sin\theta - \dot{\psi}\dot{\theta}\cos\theta \\ \dot{q}_c &= U_\theta \cos\phi - \dot{\theta}\dot{\phi}\sin\phi + U_\psi \sin\phi\cos\theta \\ &\quad + \dot{\psi}\dot{\phi}\cos\phi\cos\theta - \dot{\psi}\dot{\theta}\sin\phi\sin\theta \\ \dot{r}_c &= -U_\theta \sin\phi - \dot{\theta}\dot{\phi}\cos\phi + U_\psi \cos\phi\cos\theta \\ &\quad - \dot{\psi}\dot{\phi}\sin\phi\cos\theta - \dot{\psi}\dot{\theta}\cos\phi\sin\theta \end{aligned} \tag{16}$$

The commanded rates and accelerations needed in Eqs. (11–13) can be generated by filtering the commanded positions $\boldsymbol{y}_c$ using a second-order filter as shown in Fig. 2. Note that, because only the rotational dynamics are inverted and regulated in this application, it is possible for the combined system of dynamics (translation plus rotation) to be unstable. Therefore, it is essential that the so-called zero dynamics are stable[5] or that an outer controller be added to stabilize the translational dynamics.

B. Analysis of the Inversion Error

In Sec. III.A, the dynamic inversion control laws were developed based on the assumption of linear aerodynamics. For this formulation to be realistic, the aerodynamic stability and control derivatives defined in the matrices A_1, A_2, and B, as well as the trim values from which the perturbation quantities are measured, must be continuously varying with the flight condition, thus producing a dynamic trim map[2] representing the changes in aerodynamics. We will assume that all of these quantities depend approximately on a polynomial function of forward and sideward velocities U and V defined as follows:

$$U = \dot{X}\cos\psi + \dot{Y}\sin\psi \tag{17}$$

$$V = -\dot{X}\sin\psi + \dot{Y}\cos\psi \tag{18}$$

where $\dot{X}$ and $\dot{Y}$ are the inertial northbound and eastbound speeds, respectively, and ψ is the body axis heading of the vehicle. In fact, in the simulation code used for this application, all aerodynamic quantities vary biquartically with the variables U and V. However, this may not be the case for the actual vehicle.[6] For simplicity, we will ignore the contribution of the quadratic inertia terms in the following analysis and assume that any associated inversion error is minimal. An objective in the present study is to demonstrate that the on-line neural network can be used to replace the need for scheduling A_1, A_2, B, and the state and control trim values as functions of U and V, as well as to account for other unmodeled variations in the dynamics. Therefore, we represent Eq. (14) in the following form:

$$\dot{\omega} = [\bar{A}_1 + \delta A_1(U, V)]\Delta z + [\bar{A}_2 + \delta A_2(U, V)]\omega + [\bar{B} + \delta B(U, V)]\Delta \eta \tag{19}$$

$$\Delta z = \widehat{\Delta z} - \delta z_{\text{trim}}(U, V) \qquad \Delta \eta = \widehat{\Delta \eta} - \delta \eta_{\text{trim}}(U, V)$$

where A_1, A_2, and B are matrices with constant coefficients representing estimated stability and control derivatives at some nominal flight condition. The nominal control laws are based on the model in Eq. (14), and the perturbation quantities in state and control are based on a nominal trim condition. Thus, Δz and $\Delta \eta$ are approximated as

$$\widehat{\Delta z} = z - \hat{z}_{\text{trim}} \tag{20}$$

$$\widehat{\Delta \eta} = \bar{B}^{-1}\{\dot{\omega}_c - \bar{A}_1 \widehat{\Delta z} - \bar{A}_2 \omega\} \tag{21}$$

where $\hat{z}_{\text{trim}}$ and $\hat{\eta}_{\text{trim}}$ are constant vectors that represent state and control trim values at the nominal flight condition. The δ terms in Eq. (19) represent the stability and control derivative and trim variations as the flight condition changes. Given the complexity in obtaining and storing such terms, we will neglect them in the nominal controller and let the on-line neural network account for them.

Applying the control law in Eq. (21) to the perturbed equations of motion in Eq. (19) results in the following equation for the closed-loop dynamics:

$$\dot{\omega} = \dot{\omega}_c + \left[\delta A_1 - \delta B \bar{B}^{-1} \bar{A}_1\right]\Delta \hat{z} + \left[\delta A_2 - \delta B \bar{B}^{-1} \bar{A}_2\right]\omega + \delta B \bar{B}^{-1} \dot{\omega}_c - \bar{A}_1 \delta z_{\text{trim}} - \bar{B}\delta \eta_{\text{trim}} - \delta A_1 \delta z_{\text{trim}} - \delta B \delta \eta_{\text{trim}} \tag{22}$$

or, in shorthand notation,

$$\dot{\omega} = \dot{\omega}_c + \tilde{\dot{\omega}} \tag{23}$$

The term $\tilde{\dot{\omega}}$ will be referred to as the body axis inversion error because, when this term is zero, the model inversion is perfect. Because the desired trajectories are Euler angles and rates, it is desirable to express Eq. (23) in an Euler angle frame. Defining the vector of Euler angles $\boldsymbol{y} = \{\phi, \theta, \psi\}$, we can transform body axis rotational accelerations to Euler angle second derivatives using the equation

$$\ddot{\boldsymbol{y}} = L(\phi, \theta)\dot{\omega} + \boldsymbol{g}(\boldsymbol{y}, \dot{\boldsymbol{y}}) \tag{24}$$

and, likewise, the commanded accelerations can be transformed using

$$\boldsymbol{U}_s = L(\phi, \theta)\dot{\omega}_c + \boldsymbol{g}(\boldsymbol{y}, \dot{\boldsymbol{y}}) \tag{25}$$

where

$$L(\phi, \theta) = \begin{bmatrix} 1 & \sin\phi \tan\theta & \cos\phi \tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi \sec\theta & \cos\phi \sec\theta \end{bmatrix} \tag{26}$$

and

$$\boldsymbol{g}(\boldsymbol{y}, \dot{\boldsymbol{y}}) = \begin{bmatrix} \dot{\psi}\dot{\theta} \sec\theta + \dot{\theta}\dot{\phi} \tan\theta \\ -\dot{\psi}\dot{\phi} \cos\theta \\ \dot{\theta}\dot{\phi} \sec\theta + \dot{\psi}\dot{\theta} \tan\theta \end{bmatrix} \tag{27}$$

Assuming the pitch attitude θ is not 90 deg, we can use Eqs. (24) and (25) in Eq. (23) to obtain

$$L^{-1}(\phi, \theta)[\ddot{\boldsymbol{y}} - \boldsymbol{g}(\boldsymbol{y}, \dot{\boldsymbol{y}})] = L^{-1}(\phi, \theta)[\boldsymbol{U}_s - \boldsymbol{g}(\boldsymbol{y}, \dot{\boldsymbol{y}})] + \tilde{\dot{\omega}} \tag{28}$$

which reduces to the convenient form

$$\ddot{\boldsymbol{y}} = \boldsymbol{U}_s + L(\phi, \theta)\tilde{\dot{\omega}} \tag{29}$$

Thus, the Euler angle definition for the inversion error is

$$\boldsymbol{\chi} = L(\phi, \theta)\tilde{\dot{\omega}} \tag{30}$$

The inversion error is then characterized by 1) terms that include a bilinear contribution from δA_1, δA_2, and δB; 2) terms that include a bilinear contribution from $\widehat{\Delta z}$, ω, and $\dot{\omega}_c$; 3) terms that are linear in δz_{trim} and $\delta \eta_{\text{trim}}$; 4) products of δ quantities; and 5) terms that include a bilinear contribution from each element of $L(\phi, \theta)$.

In the next section, we describe how the network is designed to reconstruct the inversion error in terms of the elements just defined.

IV. Adaptive Control Using On-Line Neural Networks

A. Development of the Network Equations

From Eq. (10),

$$\ddot{\tilde{y}}_i + K_{d_i}\dot{\tilde{y}}_i + K_{p_i}\tilde{y}_i = U_{\text{ad}_i} - \chi_i \tag{31}$$

where the tracking error $\tilde{\boldsymbol{y}} = \boldsymbol{y}_c - \boldsymbol{y}$. The subscript i varies from 1 to 3 and represents the ϕ, θ, and ψ channels, respectively. Note that the inversion is perfect when the network exactly reconstructs the inversion error. Thus, the quantity $U_{\text{ad}_i} - \chi_i$ is a good measure of how the network is performing and will be termed the adjusted inversion error.

It is helpful to rewrite Eq. (31) in state-space form. For each channel, the error dynamics are a second-order system of the form (dropping the i subscripts for convenience)

$$\dot{\boldsymbol{e}} = A\boldsymbol{e} + \boldsymbol{b}(\boldsymbol{U}_{\text{ad}} - \boldsymbol{\chi}) \tag{32}$$

where $\boldsymbol{e} = [\tilde{y}\ \ \dot{\tilde{y}}]^T$, $\boldsymbol{b} = [0\ \ 1]^T$, and

$$A = \begin{bmatrix} 0 & 1 \\ -K_p & -K_d \end{bmatrix} \tag{33}$$

The task now is to perform the reconstruction based on available measurements in the system. We define the adaptive control law in each channel as

$$U_{\text{ad}_i} = \boldsymbol{w}^T \boldsymbol{g} \tag{34}$$

$$\dot{w} = -ks\boldsymbol{g} \tag{35}$$

where the vector $\boldsymbol{g}$ is a set of basis functions used to approximate the uncertainty and the vector $\boldsymbol{w}$ is the set of coefficients of each basis function. The update law is designed based on Lyapunov stability of the error signals, the proof of which is shown in the next section. The s term is an error metric dependent on the tracking errors in the system, defined as follows:

$$s = (1/2K_p)\tilde{y} + (1/2\lambda K_p)\dot{\tilde{y}} \tag{36}$$

where λ is defined in terms of the Lyapunov equation used to prove stability. An outline of the proof of stability in the next section defines the relationship of the network parameters to the stability analysis.

Now define the optimal vector of weights[4] for each channel as $\boldsymbol{w}^*$ and let $\tilde{\boldsymbol{w}} = \boldsymbol{w} - \boldsymbol{w}^*$. This enables Eq. (32) to be rewritten as

$$\dot{\boldsymbol{e}} = A\boldsymbol{e} + b\tilde{\boldsymbol{w}}^T\boldsymbol{g} + b\left(\boldsymbol{w}^{*^T}\boldsymbol{g} - \boldsymbol{\chi}\right) \tag{37}$$

Let

$$\epsilon \equiv \sup_{\rho} \left|\boldsymbol{w}^{*^T}\boldsymbol{g}(\boldsymbol{\rho}) - \boldsymbol{\chi}(\boldsymbol{\rho})\right| \tag{38}$$

which represents a bound on the residual inversion error that is unmodeled by the neural network. The vector $\boldsymbol{\rho}$ consists of all independent variables of the inversion error. Equation (38) essentially defines ϵ as the worst-case difference between the inversion error and the best approximation for the inversion error for the given set of network inputs. Before moving forward to the stability analysis, there is an important assumption to make. Consider Eq. (34), which represents the computation of the adaptive component of the pseudocontrol. As shown in Fig. 1, the pseudocontrol is an input to the network and is, thus, an element of the vector $\boldsymbol{g}$ because of its explicit appearance in the inversion error in Eq. (22). This gives rise to a fixed point problem, which must have a solution in order to solve Eq. (34). Thus, the assumption is made here that the fixed point solution exists (thus, $\boldsymbol{U}_{\text{ad}}$ exists).[4]

B. Stability Analysis

The analysis is based on Lyapunov's direct method for determining stability.[7] Because the functional form of each channel is the same, consider the Lyapunov function candidate for any of the channels,

$$V = \begin{cases} V_p(\boldsymbol{e}) + \dfrac{\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{w}}}{2\gamma} & \text{when} \quad \sqrt{e^TP\boldsymbol{e}} > e_0 \\ V_p(\boldsymbol{e}_c) + \dfrac{\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{w}}}{2\gamma} & \text{when} \quad \sqrt{e^TP\boldsymbol{e}} \le e_0 \end{cases} \tag{39}$$

where e_0 is a scalar constant to be defined later,

$$V_p(\boldsymbol{e}) = \tfrac{1}{2}e^TP\boldsymbol{e} \qquad P > 0 \tag{40}$$

and $\boldsymbol{e}_c$ is a vector in R^2 that satisfies $\sqrt{(\boldsymbol{e}_c^TP\boldsymbol{e}_c)} = e_0$. This ensures continuity of V across the radial boundary defined by e_0. The time derivative of the candidate Lyapunov function for $\sqrt{(e^TP\boldsymbol{e})} > e_0$ is given by

$$\dot{V} = \frac{1}{2}\dot{e}^TP\boldsymbol{e} + \frac{1}{2}e^TP\dot{e} + \frac{\tilde{\boldsymbol{w}}^T\dot{\tilde{\boldsymbol{w}}}}{\gamma} \tag{41}$$

Substituting Eq. (37) results in

$$\dot{V} = \tfrac{1}{2}e^TA^TP\boldsymbol{e} + \tfrac{1}{2}e^TPA\boldsymbol{e} + \left\{\tilde{\boldsymbol{w}}^T\boldsymbol{g} + \left(\boldsymbol{w}^{*^T}\boldsymbol{g} - \boldsymbol{\chi}\right)\right\}^Tb^TP\boldsymbol{e} + e^TPb\left\{\tilde{\boldsymbol{w}}^T\boldsymbol{g} + \left(\boldsymbol{w}^{*^T}\boldsymbol{g} - \boldsymbol{\chi}\right)\right\} \tag{42}$$

Now, because A is Hurwitz,[7] there exists a symmetric, positive definite solution P to the Lyapunov equation

$$PA + A^TP = -Q \tag{43}$$

where Q is a symmetric, positive definite matrix. Equation (42) then becomes

$$\dot{V} = -\tfrac{1}{2}e^TQ\boldsymbol{e} + e^TPb\tilde{\boldsymbol{w}}^T\boldsymbol{g} + e^TPb\left(\boldsymbol{w}^{*^T}\boldsymbol{g} - \boldsymbol{\chi}\right) + (1/\gamma)\tilde{\boldsymbol{w}}^T\dot{\tilde{\boldsymbol{w}}} \tag{44}$$

Because e^TPb is a scalar quantity, it can be easily moved around to obtain

$$\dot{V} = -\tfrac{1}{2}e^TQ\boldsymbol{e} + e^TPb\left(\boldsymbol{w}^{*^T}\boldsymbol{g} - \boldsymbol{\chi}\right) + \tilde{\boldsymbol{w}}^T\{e^TPb\boldsymbol{g} + (1/\gamma)\dot{\tilde{\boldsymbol{w}}}\} \tag{45}$$

Choosing the update law

$$\dot{w} = \dot{\tilde{w}} = -\gamma e^TPb\boldsymbol{g} \tag{46}$$

results in

$$\dot{V} = -\tfrac{1}{2}e^TQ\boldsymbol{e} + e^TPb\left(\boldsymbol{w}^{*^T}\boldsymbol{g} - \boldsymbol{\chi}\right) \tag{47}$$

$$\le -\tfrac{1}{2}e^TQ\boldsymbol{e} + |e^TPb|\left|\boldsymbol{w}^{*^T}\boldsymbol{g} - \boldsymbol{\chi}\right| \tag{48}$$

$$\le -\tfrac{1}{2}e^TQ\boldsymbol{e} + \epsilon|e^TPb| \tag{49}$$

$$\le -\tfrac{1}{2}\|e\|_2^2\lambda_{\min}(Q) + \epsilon|e^TPb| \tag{50}$$

and because

$$e^TP\boldsymbol{e} \le \lambda_{\max}(P)\|e\|_2^2 \tag{51}$$

that leaves

$$\dot{V} \le -\frac{1}{2}\frac{e^TP\boldsymbol{e}}{\lambda_{\max}(P)}\lambda_{\min}(Q) + \epsilon|e^TPb| \tag{52}$$

$$\le -\frac{1}{2}\frac{e^TP\boldsymbol{e}}{\lambda_{\max}(P)}\lambda_{\min}(Q) + \epsilon|e^T\sqrt{P}||\sqrt{P}b| \tag{53}$$

$$= -\frac{1}{2}\frac{e^TP\boldsymbol{e}}{\lambda_{\max}(P)}\lambda_{\min}(Q) + \epsilon\sqrt{|e^T\sqrt{P}||\sqrt{P}\boldsymbol{e}|}|\sqrt{P}b| \tag{54}$$

$$\le -\frac{1}{2}\frac{e^TP\boldsymbol{e}}{\lambda_{\max}(P)}\lambda_{\min}(Q) + \epsilon\sqrt{e^TP\boldsymbol{e}}\sqrt{\lambda_{\max}(P)} \tag{55}$$

and, thus, the derivative of the Lyapunov function is strictly negative when

$$\sqrt{e^TP\boldsymbol{e}} > \frac{2\epsilon\lambda_{\max}(P)^{\frac{3}{2}}}{\lambda_{\min}(Q)} \equiv e_0 \tag{56}$$

Therefore, ultimate boundedness is guaranteed for the region where $\sqrt{(e^TP\boldsymbol{e})} > e_0$ because V is a strictly positive, strictly increasing function of e and $\tilde{\boldsymbol{w}}$ (Ref. 7). Now it is important to relate the update law in Eq. (46) to that of Eq. (35). This involves solving the Lyapunov equation in Eq. (43) for an arbitrary symmetric, positive definite matrix Q. It can be shown that the off-diagonal terms in Q have no effect on the update laws in this application. Additionally, because the final solution can simply be scaled by a scalar constant, only the relative magnitude of the diagonal elements of Q is relevant. Therefore, without loss of generality, the range of update laws of this form is covered by assuming

$$Q = \begin{pmatrix} q & 0 \\ 0 & 1 \end{pmatrix} \tag{57}$$

The solution of Eq. (43) can be found to be

$$P = \begin{pmatrix} \left(\dfrac{K_d}{2K_p} + \dfrac{1}{2K_d}\right)q + \dfrac{K_p}{2K_d} & \dfrac{q}{2K_p} \\ \dfrac{q}{2K_p} & \dfrac{q + K_p}{2K_pK_d} \end{pmatrix} \tag{58}$$

with some simple algebra. This reduces Eq. (46) to the form

$$\dot{w} = \gamma\left(\frac{q}{2K_p}\tilde{\boldsymbol{y}} + \frac{q + K_p}{2K_dK_p}\dot{\tilde{\boldsymbol{y}}}\right)\boldsymbol{g} \tag{59}$$

Thus, based on the preceding stability analysis,

$$\lambda = K_dq/(q + K_p) \tag{60}$$

which, by the constraint on q, implies that $0 < \lambda < K_d$, and

$$k = \gamma q \tag{61}$$

Now to choose a value of λ that gives a minimum dead zone, it is necessary to minimize the expression for e_0 in Eq. (56). For simplicity, we choose the value that minimizes the ratio $\lambda_{\max}(P)/\lambda_{\min}(Q)$, thus giving $Q = I$, or equivalently, $q = 1$ (Ref. 8). Thus, λ is chosen as $K_d/(1 + K_p)$.

The last step is to consider the situation where the tracking error moves within the boundary defined by e_0. The derivative of the Lyapunov function is then

$$\dot{V} = \frac{\dot{\tilde{\boldsymbol{w}}}^T\tilde{\boldsymbol{w}}}{\gamma} \tag{62}$$

for which the only choice is to set

$$\dot{w} = \dot{\tilde{w}} = 0 \tag{63}$$

which implies that $\dot{V} = 0$ when $e^T Pe < e_0$. Thus, for global stability, a dead zone is required with a radius of e_0, inside of which the adaptation is turned off. However, if $\epsilon = 0$, and thus there are sufficient basis functions to exactly cancel the inversion error, $e_0 = 0$ and no dead zone is required. For the application in this paper, the set of basis functions is assumed to entirely span the space of the inversion error, an assumption that may not be valid for a more comprehensive rotorcraft model.[6] Note that the tracking errors are ultimately bounded regardless of the network structure or inputs. However, an improper choice of structure and inputs may result in a required dead zone size that is unreasonably large.

C. Network Structure

The first step in determining the appropriate structure for the network involves identifying and classifying the network inputs. Based on the functions described in the list at the end of Sec. III.B, there are three basic categories of inputs: the unknown $\boldsymbol{\delta}$ functions representing trim variations; terms that enter linearly, such as perturbation states and body-axis pseudocontrols; and finally the terms in the $L(\phi, \theta)$ transformation, which represent the transformation of the pseudocontrols from the body rotational accelerations to Euler second time derivatives. The first category is the one involving the most judgment. The terms in this category represent the basis functions that best describe the unknown $\boldsymbol{\delta}$ functions. In the present case, we assume all $\boldsymbol{\delta}$ functions to depend on the projected speeds U and V in the same fashion, specifically a general biquadratic function of U and V with unknown coefficients. This assumption is valid for this application because all aerodynamic quantities depend explicitly on U and V in the simulation used to generate the results presented. However, this category of inputs may require more consideration when applied to a comprehensive model for the helicopter.[6] The second category of inputs is fairly straightforward, as inspection of the inversion error clearly shows solely a bilinear dependence on these terms in the inversion error. Finally, the last set of terms comes about in the transformation of the closed-loop dynamics. These are simply several trigonometric functions from the transformation matrix, which enter the error term bilinearly as well. We will ignore terms that are products of two $\boldsymbol{\delta}$ terms because it is believed terms of such high order will be negligible.

To form the vector of basis functions $\boldsymbol{g}$, we first preprocess the input data, normalizing all variables between -1 and 1. Next, the category I inputs are formed by expanding the normalized U and V inputs ($\bar{U}, \bar{V}$) to a biquadratic form along with a bias term. This results in a category I vector, $\boldsymbol{C}_1$, defined as

$$\boldsymbol{C}_1 = \{1, \bar{U}, \bar{V}, \bar{U}^2, \bar{V}^2, \bar{U}\bar{V}, \bar{U}^2\bar{V}, \bar{V}^2\bar{U}, \bar{U}^2\bar{V}^2\} \tag{64}$$

The category II vector simply consists of the normalized components of the vectors $\widehat{\Delta z}$, $\boldsymbol{\omega}$, and the previous time step value of $\dot{\boldsymbol{\omega}}_c$ along with a bias term:

$$\boldsymbol{C}_2 = \{1, \overline{\Delta u}, \overline{\Delta v}, \overline{\Delta w}, \overline{\Delta \delta_{\text{col}}}, \bar{p}, \bar{q}, \bar{r}, \dot{p}_c, \dot{q}_c, \dot{r}_c\} \tag{65}$$

The category III vector is composed of the nonzero terms of the $L(\phi, \theta)$ matrix. No normalization is required as long as the pitch attitude is reasonably small:

$$\boldsymbol{C}_3 = \{1, \sin\phi\tan\theta, \cos\phi\tan\theta, \cos\phi, \sin\phi, \sin\phi\sec\theta, \cos\phi\sec\theta\} \tag{66}$$

The vector of basis functions $\boldsymbol{g}$ is composed of all possible products of the elements of $\boldsymbol{C}_1, \boldsymbol{C}_2$, and $\boldsymbol{C}_3$ (Ref. 6). Using Kronecker products to represent the neuron interactions, $\boldsymbol{g}$ can be composed as

$$\boldsymbol{g} = \text{kron}[\text{kron}(\boldsymbol{C}_1, \boldsymbol{C}_2), \boldsymbol{C}_3] \tag{67}$$

where kron($\bullet, \bullet$) represents the Kronecker product and is defined as follows:

$$\text{kron}(x, y) = [x_1 y_1 \quad x_1 y_2 \quad \cdots \quad x_m y_n]^T \tag{68}$$

where $\boldsymbol{x} \in R^m$ and $\boldsymbol{y} \in R^n$.

The network structure is not minimal. A careful analysis will show that many of the terms of $\boldsymbol{g}$ can be eliminated depending on whether the pitch, roll, or yaw network is considered. However, in this study we chose to keep the same network structure in all channels although the design is slightly conservative.

V. Numerical Results

In this section we apply the theory and concepts presented to a nonlinear simulation model of the AH-64 helicopter. The simulation code used is a modified version of NASA's TMAN simulation code.[9] The aerodynamic coefficients in the simulation code were obtained at a grid of trimmed flight conditions specified by (U, V) pairs to cover the flight envelope based on linear models generated from McDonnell Douglas's FLYRT simulation.[10] The dynamic trim map is a biquartic function of U and V. The six-degree-of-freedom model includes all nonlinear kinematic and gravitational terms and is based on a quasistatic rotor representation. First-order actuator dynamics with 20-rad/s bandwidth are modeled in each channel. The gains in the nominal controller were chosen as $K_{p_i} = 7.0$ and $K_{d_i} = 4.6$ in all three channels. These gains correspond to natural frequency of 2.65 rad/s and damping ratio of 0.869 and, thus, a 1% settling time of 2.0 s and 8% overshoot in response to a step input. The network parameter λ was chosen as 0.57 based on Eq. (60) with $q = 1$ ($Q = I$). The command filters shown in Fig. 2 had a damping ratio of 1 and a natural frequency of 1 rad/s. The command used is an inner-loop-only version of the elliptical turn in which the helicopter starts from hover, banks to a 0-deg roll attitude, pitches forward to a -25-deg pitch attitude, pitches back to a 0-deg pitch attitude, then finally performs three revolutions in yaw attitude while maintaining 0-deg pitch and roll attitudes. The translational dynamics are uncontrolled. The purpose of the pitch command is to accelerate the vehicle from hover to forward flight. In Sec. V.A, results are presented based on the network structure described in Sec. IV.C and compared to a simpler network,[11] both at the same adaptation gain. Afterward, in Sec. V.B, the structure is fixed as described in Sec. IV.C and the adaptation gain is varied from low (10) to high (1000) to demonstrate the effect of varying the speed of adaptation as well as the relative insensitivity of the performance to changes in adaptation gain. For all runs presented, the nominal controller uses all parameters from the initial hover condition, including stability and control derivatives, as well as trim values for the states and controls.

Figure 3 shows the response of the system with the nominal controller only (no network), whereas Fig. 4 compares the pitch response without and with a network (gain of 200). Figure 5, like Fig. 3, shows the response of the system with the nominal controller only (no network). The dash-dotted line is with the neural network, whereas the dashed line is without the network. Because the roll response exhibits the largest errors, it is emphasized in the subsequent analysis.

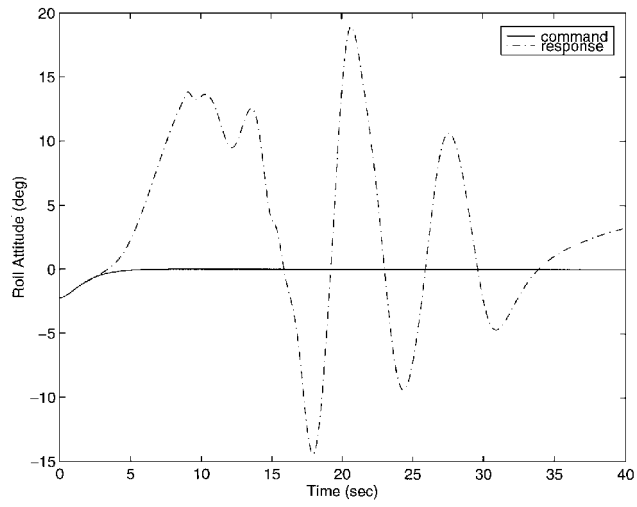

Fig. 3 Time history of roll attitude, no neural network.

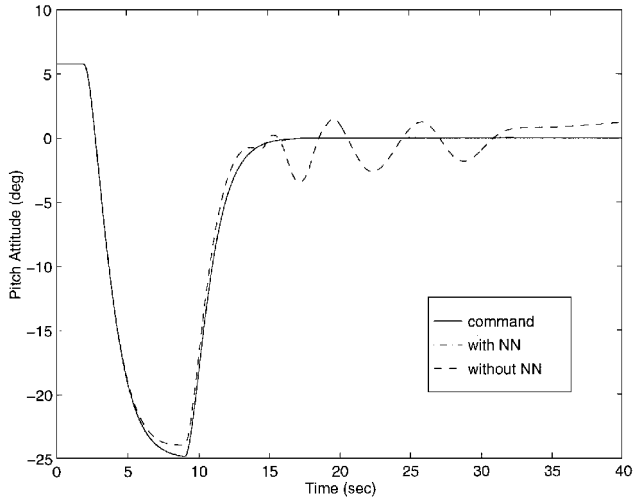

Fig. 4 Time history of pitch attitude, without and with neural network; $k = 200$.

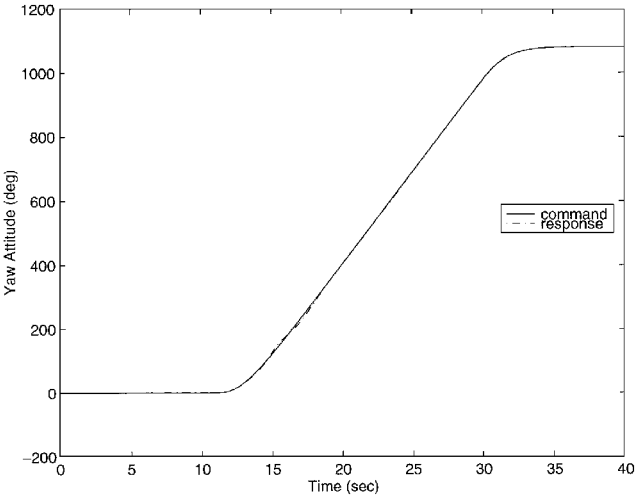

Fig. 5 Time history of yaw attitude, no neural network.

A. Comparison of Network Structures

In this section we compare results from a simple network presented in Ref. 11 with the results from the network designed in Sec. IV.C. The simple network includes the category I inputs as well as bilinear inputs of the axis pseudocontrol corresponding to each channel (U_ϕ, U_θ, and U_ψ, respectively). Figure 6 shows a comparison of the roll responses for each network at a fixed adaptation gain, $k = 1000$. Note the difference in scale of Fig. 6 relative to Fig. 3. Clearly, even the simple network provides good performance in the presence of the uncertainties. This is because the category I inputs (those representing the variations in the model) are the dominant sources of the inversion error. However, notice that the more complex network not only reduces the magnitude of the tracking errors but additionally reduces the activity of the response. This is because, in the absence of some of the required inputs, the bias term as well as other inputs work to account for the missing terms. The result is a design that is highly sensitive to the network parameters, in particular, adaptation gain. Figures 7 and 8 show two second width sliding window rms values of the adjusted roll inversion errors ($\boldsymbol{U}_{\text{ad}} - \boldsymbol{\chi} = \boldsymbol{U}_{pd} + \ddot{\boldsymbol{y}}_c - \ddot{\boldsymbol{y}}$) for each channel for the simple and complex networks, respectively.

Figures 7 and 8 give a good measure of the network's ability to reconstruct and cancel the errors caused by changes in the aerodynamics and the trim values.

B. Effects of Adaptation Gain Variations

Herein we fix the structure of the network to the complex structure defined in Sec. IV.C, and we consider variations in the adaptation gain k. In Figs. 9, 10, and 11 ($k = 10, 200$, and 1000, respectively), we see similarities in the shape of the tracking errors, yet,

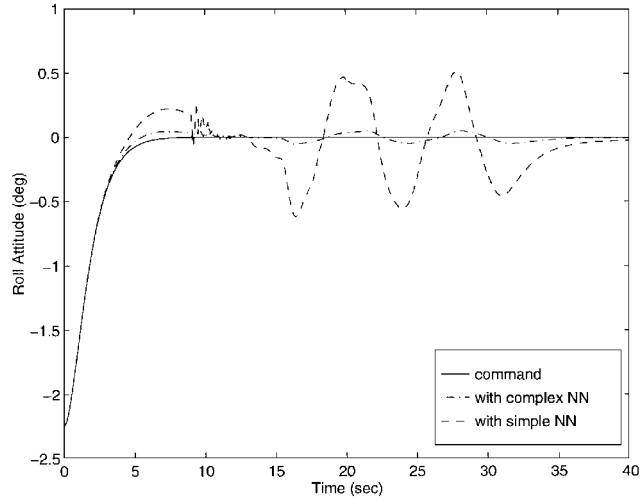

Fig. 6 Time history of roll attitude, $k = 1000$.

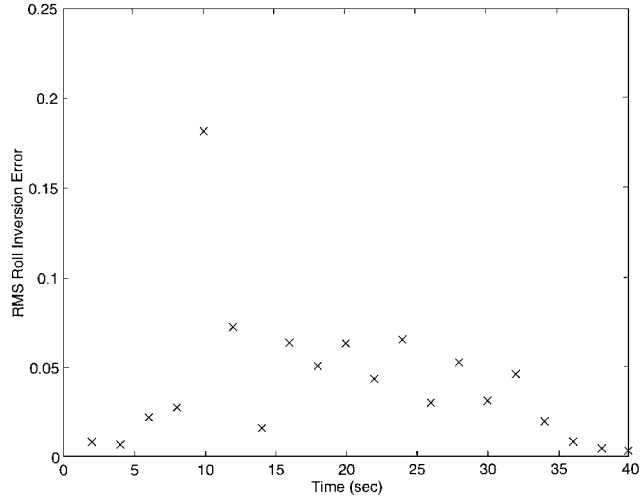

Fig. 7 Simple network rms adjusted roll inversion error, $k = 1000$.

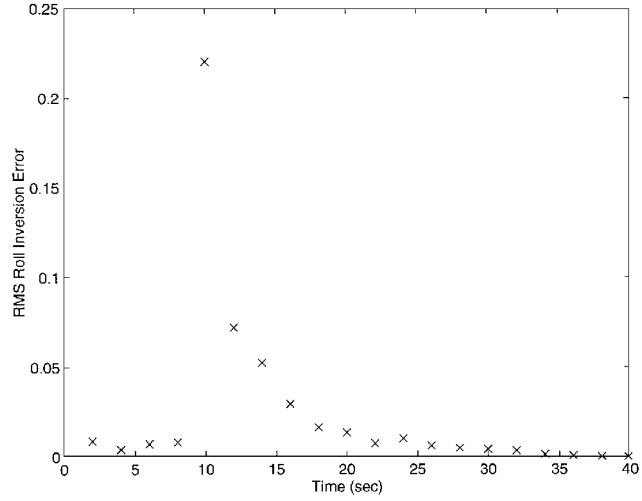

Fig. 8 Complex network rms adjusted roll inversion error, $k = 1000$.

as the adaptation gain is increased, the magnitude of the tracking errors is subsequently decreased proportionally. In Figs. 12–14, it is shown (not surprisingly) that control activity increases during the learning phases, particularly in the transition from hover to forward flight, where the most significant trim changes occur. In particular, at around 8 s, we see significant activity in the roll attitude plots, which occurs just after the maximum (negative) pitch attitude and just before the initiation of the yaw maneuver. This is indicative of a rapid change in the roll dynamics (which appears as nearly

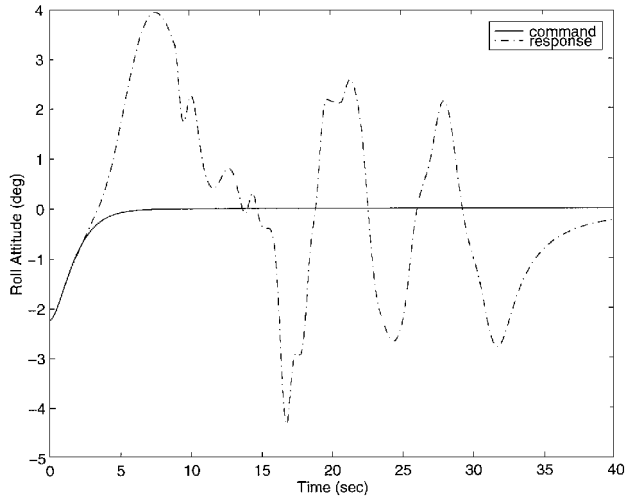

Fig. 9 Time history of roll attitude, $k = 10$.

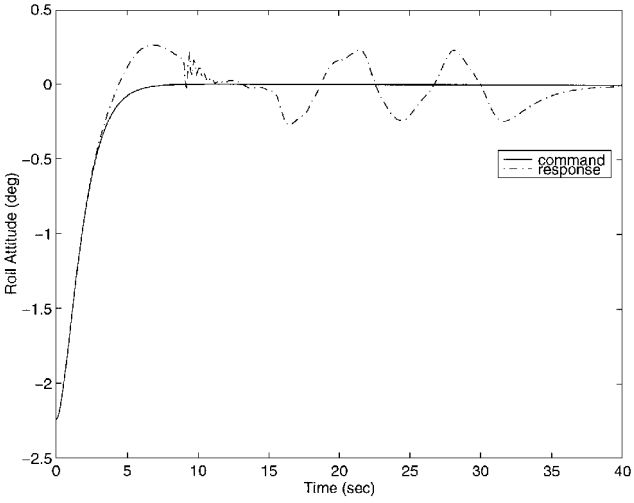

Fig. 10 Time history of roll attitude, $k = 200$.

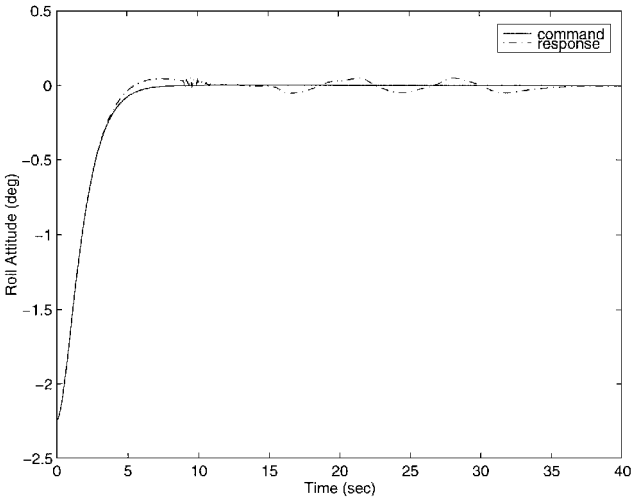

Fig. 11 Time history of roll attitude, $k = 1000$.

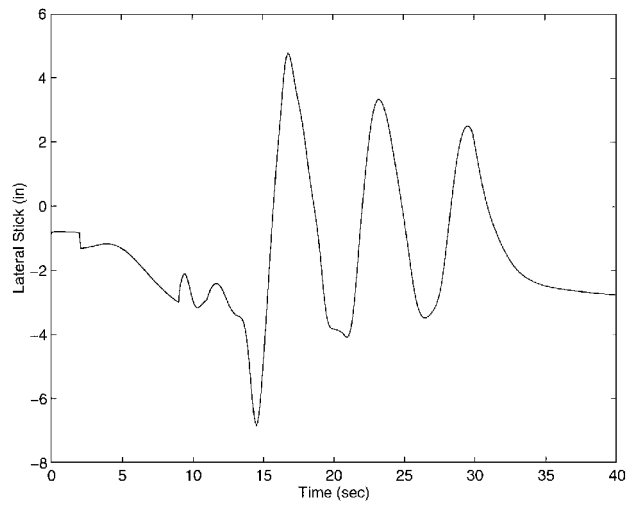

Fig. 12 Time history of lateral cyclic, $k = 10$.

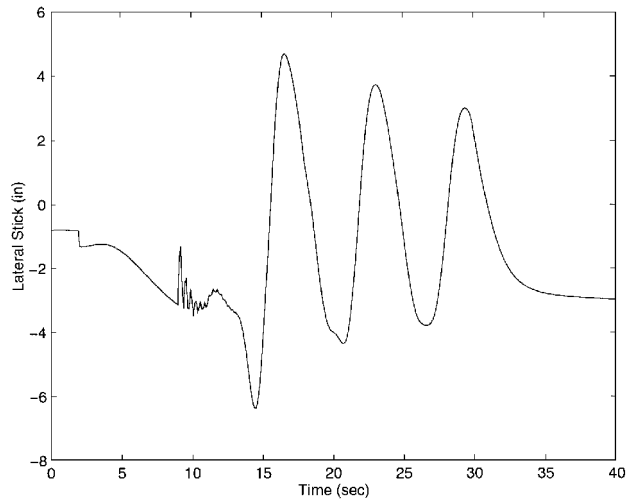

Fig. 13 Time history of lateral cyclic, $k = 200$.

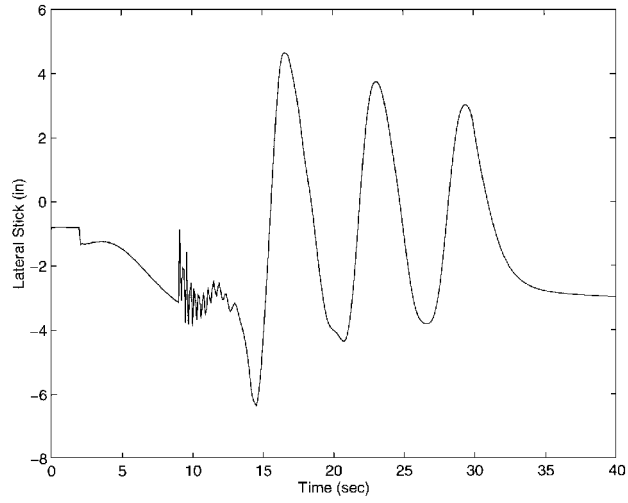

Fig. 14 Time history of lateral cyclic, $k = 1000$.

a discontinuity in the adjusted inversion error histories for higher adaptation rates). The interpretation of this is that the adaptation gain must be set high enough so that the network can learn the local adjusted inversion error, but adaptation rates too high may result in undesirable oscillations or unimplementable controls based on actuator bandwidths. This can be seen in Figs. 8, 15, and 16 where, for the high adaptation gain, we see a peak in adjusted inversion error at the beginning of each learning phase (corresponding to the different sections of the maneuver) followed by virtually zero adjusted inversion error, whereas for the low adaptation gain, complete learning never seems to occur. Note the larger scale in Fig. 15.

VI. Discussion

The results presented show much promise in dealing with the significant uncertainties present in the helicopter dynamics. However, the category I inputs to the network were chosen based on knowledge of the explicit functional dependence of the aerodynamic parameters within the simulation. In reality, the terms U and V in polynomial

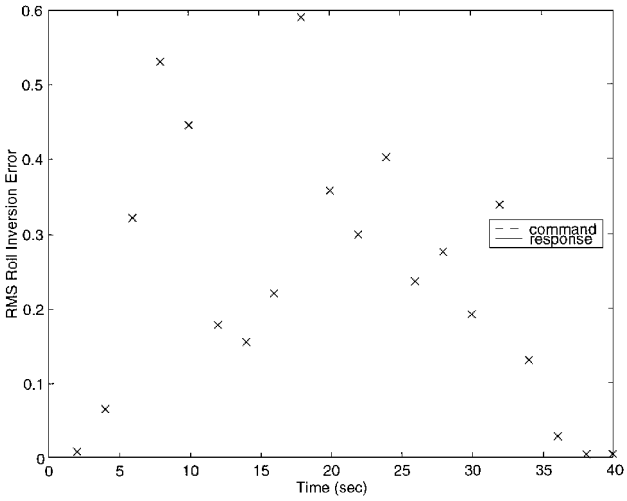

Fig. 15 Complex network rms adjusted roll inversion error, $k = 10$.

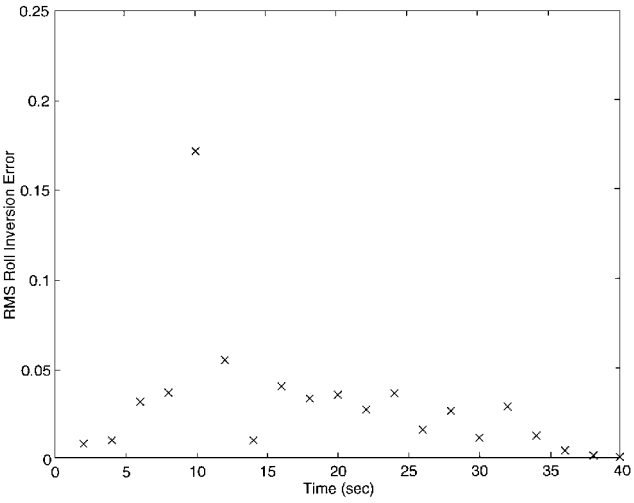

Fig. 16 Complex network rms adjusted roll inversion error, $k = 200$.

form may not be sufficient to characterize these model variations. Thus, it is important to realize that considerable thought must be placed into the selection of category I inputs, whereas the category II and III inputs are completely general for the given controller structure. This problem is addressed in Ref. 6, where the adaptive neural network presented is applied to a comprehensive rotorcraft model and the problem becomes a bit more complex. Furthermore, although actuator delays are included in the simulation model, the dynamic effect of the rotor is not addressed here. A more comprehensive evaluation that includes rotor dynamics may be found in Ref. 6.

VII. Conclusions

Neural networks can be used to enhance flight control designs based on feedback inversion. A crucial issue is the architecture of the network that permits on-line adaptation, together with accurate cancellation of the inversion error. This is enhanced by a careful analysis of the sources of inversion error. The application we have addressed confirms that rapid adaptation is achievable for a complex helicopter maneuver, with learning control activity confined to a short training interval. In addition, with a suitably chosen architecture for the network, performance varies gracefully and monotonically improves as the adaptation gain is varied from zero to a very high value. This is a desirable feature from both theoretical and practical viewpoints. The ultimate selection of an adaptation gain involves a tradeoff between the level of control activity during training episodes vs the magnitude of the resulting tracking errors during a maneuver.

Acknowledgments

This work was supported in part by the U.S. Army Research Office under Grant DAAH04-93-G-0002 and in part by the U.S. Air Force Phillips Laboratory under the Palace Knight program.

References

[1]Meyer, G., Hunt, R. L., and Su, R., "Design of a Helicopter Autopilot by Means of Linearizing Transformations," Guidance and Control Panel, 35th Symposium, AGARD CP 321, 1983, pp. 4-1–4-11; also NASA TM-84295, A-9091, NAS 1.15;84295, Oct. 1982.

[2]Smith, G. A., and Meyer, G., "Total Aircraft Flight Control System-Balanced Open and Closed Loop Control with Dynamic Trim Maps," *Challenge of the '80s; Proceedings of the Third Digital Avionics Systems Conference* (Fort Worth, TX), Inst. of Electrical and Electronics Engineers, New York, 1979, pp. 215–223.

[3]Heiges, M., "A Helicopter Flight Path Controller Design via a Nonlinear Transformation Technique," Ph.D. Thesis, Dept. of Aerospace Engineering, Georgia Inst. of Technology, Atlanta, GA, March 1989.

[4]Kim, B. S., and Calise, A. J., "Nonlinear Flight Control Using Neural Networks," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 1, 1997, pp. 26–33.

[5]Slotine, J. J. E., and Li, W., *Applied Nonlinear Control*, Prentice–Hall, Englewood Cliffs, NJ, 1991, pp. 267–271.

[6]Leitner, J., "Helicopter Nonlinear Control Using Adaptive Feedback Linearization," Ph.D. Thesis, Dept. of Aerospace Engineering, Georgia Inst. of Technology, Atlanta, GA, May 1995.

[7]Khalil, H., *Nonlinear Systems*, Macmillan, New York, 1992, pp. 98–113.

[8]Patel, R. V., and Toda, M., "Qualitative Measures of Robustness for Multivariable Systems," Proceedings of the Joint Automatic Control Conf., Rept. TP8-A, June 1980.

[9]Prasad, J., and Lipp, A., "Synthesis of a Helicopter Nonlinear Controller Using Approximate Model Inversion," *International Jounal on Mathematical and Computer Modeling*, Vol. 18, No. 3/4, 1993, pp. 89–100.

[10]Shanthakumaran, P., "The Application of Flight Simulation Models in Support of Rotorcraft Design and Development," *AGARD Conference Proceedings 513*, 1991.

[11]Calise, A., Kim, B.-S., Leitner, J., and Prasad, J. V. R., "Helicopter Adaptive Flight Control Using Neural Networks," *IEEE Conference on Decision and Control* (Lake Buena Vista, FL), Inst. of Electrical and Electronics Engineers, New York, 1994.