JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 32, No. 1, January–February 2009

Envelope Protection for Autonomous Unmanned Aerial Vehicles

Ilkay Yavrucuk*
Middle East Technical University, 06531 Ankara, Turkey
J. V. R. Prasad†
Georgia Institute of Technology, Atlanta, Georgia 30332
and
Suraj Unnikrishnan‡
Guided Systems Technologies, Inc., McDonough, Georgia 30253

DOI: 10.2514/1.35265

This paper describes the design, development, and testing of an automatic envelope protection system as implemented on Georgia Institute of Technology's unmanned helicopter GTMax. The envelope protection system makes use of online-learning adaptive neural networks to generate online dynamic models, which are used to estimate limits on controller commands. The system provides command capability up to the limit boundaries while preventing envelope exceedance. Simulation and flight-test results are provided for load factor and rotor stall limit protection during aggressive maneuvering.

Nomenclature

$\boldsymbol{A}, \boldsymbol{b}, b$	=	linearized dynamics
a	=	linear parameter or activation potential
a_c	=	outer-loop acceleration command
$\boldsymbol{a_m}$	=	filtered acceleration measurement
b_v, b_w	=	neural network bias
d_{ext}, w	=	external disturbance
$\boldsymbol{E}, e$	=	error state vector and parameter error
K	=	feedback gain
K_{ERITS}	=	rotor stall-limit smoothing-function gain
K_{NZ}	=	load factor limit smoothing-function gain
k_1, k_2	=	positive constants
$N_z, \hat{N}_z$	=	load factor and estimate
$p, \boldsymbol{P}$	=	position and positive-definite solution to the Lyapunov equation
$\boldsymbol{Q}$	=	positive-definite matrix
q_c	=	pitch rate command
r	=	relative degree
$\boldsymbol{u_c}, u_p$	=	controller command vector and element
$V, \boldsymbol{v}$	=	forward speed and velocity vector
V_{tip}	=	rotor tip speed
v_{cas}	=	calibrated airspeed
$\boldsymbol{W}, \boldsymbol{V}, v, w$	=	neural network input and output weights
$\boldsymbol{x}$	=	vector of aircraft states
$\boldsymbol{Y}_p, y_p$	=	limit vector and limit parameter
z_j	=	input to the jth hidden-layer neuron
Δ	=	modeling error
$\boldsymbol{\delta}_{\text{comb}}$	=	actuator control input
δ_f	=	collective control input
$\boldsymbol{\delta}_m$	=	longitudinal and lateral pedal-control input
θ_v, θ_w	=	neural network thresholds
$\boldsymbol{\mu}, \bar{\boldsymbol{\mu}}, \mu$	=	neural network inputs
$\boldsymbol{\nu}_{\text{ad}}, \nu_{\text{ad}}$	=	adaptive element signal
$\boldsymbol{\nu}_{\text{dc}}, \nu_{\text{dc}}$	=	linear dynamic compensator signal
ρ	=	air density
$\boldsymbol{\sigma}, \boldsymbol{\sigma}'$	=	neuron sigmoidal function, gradient

Subscripts

c	=	command
h	=	horizontal
lim	=	limit
marg	=	margin
mod	=	modified
nom	=	nominal
x, y, z	=	vector components

Superscripts

B	=	body frame
v	=	vehicle-carried frame
r	=	rth-order derivative

I. Introduction

BOTH manned and unmanned aircraft are constrained by many operational limits. Envelope protection is the task of monitoring and ensuring vehicle operation within these limits. Recent advances in flight control systems enable autonomous maneuvering that can challenge an unmanned aerial vehicle's (UAV) flight envelope [1,2]. In manned vehicles, the task of envelope protection is often delegated to the onboard human pilot. In the case of unmanned vehicles, the only options available for envelope protection are either to limit the vehicle commands to the automatic flight control system (AFCS) or to limit the actuator commands from the AFCS. Typically, conservative hard limits are set in a UAV's flight control channels as maximum- and minimum-allowable command inputs [1]. In reality, true command limits are complex functions of highly nonlinear aircraft dynamics and therefore vary with flight condition and/or vehicle configuration. The goal of automatic envelope protection is to not only enable the aircraft to safely operate within its envelope, but to also do so without restricting the aircraft to a smaller region of its operational envelope, hence making use of the aircraft's full flight envelope. Therefore, an effective automatic envelope protection system will reduce the compromise between safety and performance, thus improving the overall confidence of safe operations of UAVs, especially during aggressive maneuvering close to their operational limits.

Received 22 October 2007; revision received 28 July 2008; accepted for publication 28 July 2008.

*Faculty Member, Department of Aerospace Engineering.
†Professor, School of Aerospace Engineering. Associate Fellow AIAA.
‡Research Scientist. Member AIAA.

Recent studies have proposed various envelope protection systems for manned vehicles. Typically, there are two approaches that have been pursued for envelope protection in manned vehicles. The first approach [3] relies on appropriate modifications and redesign of the vehicle flight control system, which uses sensor data feedback to prevent limit exceedance. The advantage of this approach is that envelope limiting becomes automatic, resulting in a reduced pilot workload. However, emergency situations have to be handled separately in such an approach. The second approach [4–7] relies on predicting impending limits ahead of time through limited sensing and providing enhanced cues to the pilot such as tactile cueing via a variable-force-feel system. The advantage is that the pilot can override the system during emergencies. However, the disadvantage is that the workload may increase if pilot attention is required to monitor the cueing device. Although previous studies have mostly addressed envelope protection in manned vehicles, similar studies for unmanned vehicles are somewhat sparse, mostly due to the vast number of types of unmanned vehicles and the operational envelopes associated with such vehicles. Although the first preceding approach of automatic envelope protection through a redesign of the vehicle flight control system is plausible for unmanned vehicles, it unnecessarily complicates the already-complex controller design process. Further, it is often difficult to a priori design for all possible limit behaviors resulting from vehicle modifications. Hence, a modular architecture is desired wherein an envelope protection system monitors the commands to the flight control system and limits them in case an envelope violation is predicted.

The development of envelope protection methods for unmanned flight vehicles, especially in the context of adaptive flight control systems, is a new area that needs to be fully explored. In [8], two limit-avoidance architectures (viz., limit avoidance via control limiting and limit avoidance via command limiting) are proposed for unmanned aerial vehicles. In both cases, they involve the use of previously developed adaptive limit-detection algorithms [9,10] for prediction of the future response of limit parameters and the corresponding control/command margins. In [11], limit avoidance via the command-limiting approach is evaluated in flight for the case of rotor stall limit avoidance on the Georgia Institute of Technology's R-Max (GTMax) unmanned-helicopter testbed. This paper considers further development of envelope protection via command-limiting algorithms, their integration with the low-level adaptive flight control algorithms, and software-in-the-loop (SITL) and flight-test evaluations on the GTMax executing an aggressive e-turn maneuver, where the helicopter is commanded a sharp 180 deg turn from a forward-flight condition.

A Yamaha R-Max unmanned helicopter was modified under the Defense Advanced Research Projects Agency's Software Enabled Control (SEC) program at the Georgia Institute of Technology to serve as a testbed for UAV control research, called the GTMax. The development and flight evaluation of an automatic envelope protection system (EPS) was one of the objectives in this program. The EPS was evaluated as part of the GTMax rotary-wing experiments during the initial, midterm, and final demonstration of the SEC program [12]. The final flight demonstration of advanced technologies developed during the SEC program was conducted at the McKenna military operations on urban terrain site in Fort Benning, Georgia. Here, the EPS was integrated into the flight control system as a standalone software module within Boeing's software infrastructure called the open control platform (OCP), also developed under the SEC program. The OCP aims to provide a software environment to facilitate design, evaluation, and fielding of advanced technologies such as the EPS into future complex architectures. The OCP is designed to coordinate and support distributed interaction among diverse components and support dynamic reconfiguration and customization in real time of flight components such as the EPS.

The GTMax uses an adaptive neural-network-based low-level controller receiving trajectory commands from a high- and/or midlevel guidance controller [1]. The EPS was implemented as a midlevel controller component and designed to modify nominal, mid-, and high-level guidance commands before passing them to the low-level controller. The modifications ensure that the aircraft remains within its specified flight envelope while performing routine or aggressive maneuvers.

A common challenge in envelope protection systems is to translate aircraft limits into command limits. As they will be nonlinear functions of the aircraft dynamics, model approximations are used for estimation. In [4,5], offline-trained neural networks are used to store this information. A disadvantage is that large amounts of simulation or flight-test data are required for training the neural network for model representation. Even so, estimations will suffer from system uncertainties that are not represented in the data. In [9,10], online-learning neural networks are used to approximate selected aircraft dynamics that are then inverted to estimate command margins. As offline training of network weights is not required, the method has the advantage of adapting to varying flight conditions and different vehicle configurations. The technique was evaluated in simulation for the envelope protection of the XV-15 tilt rotor by combining an artificial pilot model and an active-force-feel side-stick model into the Generalized Tilt Rotor Simulation program. GTMax's EPS makes use of similar adaptive algorithms for the estimation of aircraft dynamics.

The key benefit of the EPS system is that it allows the aircraft to fly exactly at its limit boundary by constantly updating the controller command limits during flight. Previously, conservatively set hard limits were used in the controller to guarantee a flight in the vehicle's flight envelope. Another benefit of the proposed system is that the flight controller design does not need to be altered, because the commands to the controller are modified; therefore, closed-loop stability is still maintained. The adaptive nature of the EPS needs minimal a priori knowledge of the plant dynamics to function, which is another key advantage over systems that use approximate models or previously obtained data.

This paper provides a theoretical development of the adaptive limit detection and avoidance methodology as implemented into the GTMax, followed by a detailed description of the implementation. Finally, simulation and flight-test results are presented. The system is demonstrated for two limit parameters: namely, the load factor limit and the rotor stall limit during aggressive maneuvering of the GTMax.

II. Preliminaries

The proximity of a limit parameter to its limit boundary is called the limit margin (Fig. 1). The limit margin is easily determined when the instantaneous value of the limit parameter can be calculated directly using sensor measurements. However, a challenge in developing a flight envelope protection system is the determination of *future* limit margins to prevent limit violations due to the dynamic nature of the limit-parameter response. The time horizon of prediction has been a significant factor in determining the effectiveness of envelope protection systems, particularly in manned applications [7]. The choice of the prediction horizon is usually determined based on the accuracy of the approximate model chosen for approximating limit-parameter dynamics. In many instances, low-fidelity and static models constrain the prediction horizon to lower values that necessitate selecting additional safety margins near the envelope. When dealing with limit-parameter dynamics of a relatively short time scale (a settling time in seconds), it is desirable to have the maximum prediction horizon to compensate in cases of impending limit violation. Therefore, the first task in envelope protection is to design an approach to obtain a model of the limit-

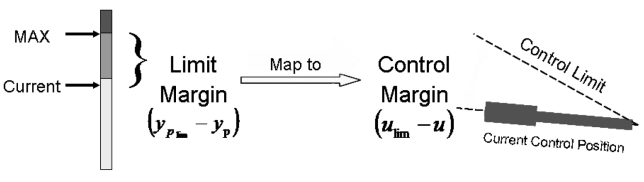

Fig. 1 Limit margin and control margin.

parameter dynamics for the estimation of the future limit margin with reasonable accuracy. This problem is known as *limit detection*. The model accuracy should be sufficient to provide either the maximum available prediction horizon (the model can accurately predict the maximum future response value) or a future value with a prediction horizon sufficient to compensate for the dynamic nature of the limit-parameter response.

An extension of the limit-detection problem is the translation of the future limit margin into an equivalent value in the control channel referred to as the control margin (Fig. 1). The control margin is the difference between the control limit and the current control input. The prediction horizon is a key component of the control-margin calculations. The control limit is defined as the step control input value that will result in the future response (future here means within the selected prediction horizon window) of the limit parameter at the limit boundary. If a prediction horizon is chosen to be Δt, then the control limit is calculated to be the control input that will result in a limit-parameter response to be at the limit boundary in Δt time. It is important to note that for small values of prediction horizon, the control margins will change rapidly, making effective envelope protection challenging and difficult unless continuous modifications are made to the control channel.

The final task in envelope protection is to avoid the limit parameter from exceeding its prescribed limits based on the control margins. This is called *limit avoidance*. Hence, for a single-input/single-output system, if $y_p \in \Re$ represents a limit parameter and $u \in \Re$ is the effective control input, future limit margins are obtained by subtracting the future value of the limit parameter associated with a given control input from the upper or lower limits of the limit parameter. Therefore, the estimated future limit margin can be defined as

$$\hat{y}_{p_{\text{marg.future}}} = y_{p_{\text{lim}}} - \hat{y}_{p_{\text{future}}} \tag{1}$$

where $y_{p_{\text{lim}}}$ is the value of the limit parameter at the limit boundary; $\hat{y}_{p_{\text{future}}}$ is the estimated future value of the limit parameter that will be reached in Δt time, provided that the control inputs are held constant at some value u; and the hat symbol denotes the estimated values.

Control margins are found by subtracting the current control input u from the estimated control limit $\hat{u}_{\text{lim}}$:

$$\hat{u}_{\text{marg}} = \hat{u}_{\text{lim}} - u \tag{2}$$

Most of the recent research in envelope protection systems focuses on the estimation of $\hat{y}_{p_{\text{future}}}$ and $\hat{u}_{\text{lim}}$. The ideal case would be when the estimated future limit-parameter response value would correspond to the maximum value that will be reached with the current control input. An estimation of maximum response is difficult in most cases. However, if such an estimate is available, this information can be translated into a control-margin estimation of maximum prediction horizon, and the excessive control input that would cause the aircraft to violate a limit could be subtracted from the applied control input instantaneously. Therefore, the following two aspects are identified in limit detection:

1) The limit and control margins have to be identified *before* the aircraft reaches a limit boundary. This will require estimation algorithms for approximating limit-parameter dynamics.

2) The limit margins must be translated into control margins.

The estimation of the proximity of the aircraft state to its envelope limits and the calculation of control margins therefore requires the building of dynamic relationships representing the limit parameters as functions of aircraft states and control inputs. In general, developing such relationships requires accurate model parameters that are difficult to determine. Simplified, even linearized, models have been used to approximate those relationships [13,14]. However, the estimation of limit and control margins are easily degraded when models no longer match with the actual aircraft dynamics. In this paper, an adaptive architecture is used to generate an adaptive neural-network-based estimate of the limit-parameter dynamics on the vehicle in real time.

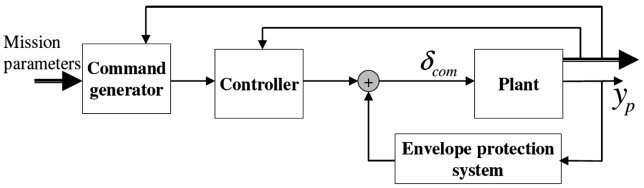

Fig. 2 Control-limiting architecture.

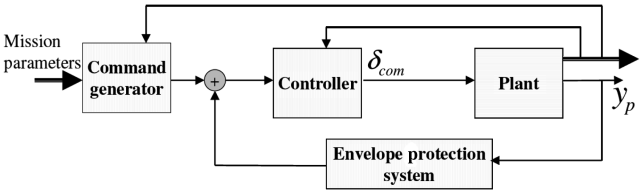

Fig. 3 Command-limiting architecture.

In an unmanned vehicle application in which envelope protection is to be automated, the system can be set up in various ways. Depending on the desired level of interaction with the flight controller, two fundamental architectures to incorporate a modular envelope protection design into a UAV's existing flight control system are identified: control limiting and command limiting.

In control limiting, the allowable control travel of each control channel (actuator command) is calculated and the actuator command inputs are limited artificially. Here, the model is constructed between the controller output δ_{com} and the limit parameter y_p (Fig. 2). Problems associated with the use of control limiting are similar to those of actuator saturation. In particular, issues related to closed-loop stability need to be reassessed.

A preferred method for an automatic EPS is the use of command limiting. In this architecture, the model is viewed as a combination of the low-level controller and the vehicle, instead of the vehicle dynamics only. A block-diagram representation of this approach is shown in Fig. 3. Then the higher-level commands to the controller $\boldsymbol{u_c}$ can be modified for envelope protection while guaranteeing that the closed-loop stability analysis for the combined vehicle controller system remains valid.

III. Theoretical Development

The following is a development of the method for command limiting for adaptive limit detection. Let us represent the closed-loop dynamics of an aircraft with the following nonlinear state equation:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u_c}), \qquad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \qquad \boldsymbol{x} \in \Re^n, \qquad \boldsymbol{u_c} \in \Re^i \tag{3}$$

where $\boldsymbol{x}$ is the state vector with initial condition $\boldsymbol{x}_0$, $\boldsymbol{u_c}$ is the controller command vector, and function $\boldsymbol{f}$ is assumed to be continuous and to satisfy the global Lipschitz condition. Moreover, for any finite initial condition, the solution of Eq. (3) does not escape in finite time.

Consider $y_p \in \Re$ to be a limit parameter that, in general, will be a nonlinear function of the system states:

$$y_p = h(\boldsymbol{x}) \tag{4}$$

The relative degree of an output is defined as the minimum number of differentiations of the output required for the control variable to appear explicitly in the dynamic relationship. If r is the relative degree of the limit parameter y_p, Eq. (4) is differentiated r times until a control appears:

$$y_p^{(r)} = h_r(\boldsymbol{x}, y_p, y_p^{(1)}, \ldots, y_p^{(r-1)}, u_p) \tag{5}$$

In Eq. (5), u_p is an element within the control vector $\boldsymbol{u_c}$ and, for simplicity, is assumed to be either the only control that appears after r differentiations or that it is a linear combination of multiple controls. The following additional assumptions are made:

Assumption 1: The limit parameter has a well-defined and known relative degree.

Assumption 2: The limit-parameter value is available or can be calculated from the available sensor measurements.

Assumption 3: The sign of the limit-parameter control sensitivity is known and the magnitude is also known to a reasonable upper bound.

First, an approximate linear model is chosen for the limit parameter based upon the knowledge of the relative degree and dynamics of the system. Let $\hat{y}_p$ be the estimate of the limit parameter available from this model. Then,

$$\hat{y}_p^{(r)} = \hat{h}_r(\hat{y}_p, \hat{y}_p^{(1)}, \ldots, \hat{y}_p^{(r-1)}, u_p) = \sum_{i=0}^{r-1} a_i \hat{y}_p^{(i)} + b u_p \tag{6}$$

Equation (7) is the state-space form of the approximate model dynamics in Eq. (6) using $\hat{\boldsymbol{Y}}_p = [\hat{y}_p \ \hat{y}_p^{(1)} \ \cdots \ \hat{y}_p^{(r-1)}]^T$ as the state vector:

$$\dot{\hat{\boldsymbol{Y}}}_p = \boldsymbol{A}\hat{\boldsymbol{Y}}_p + b\boldsymbol{b}_{r,r}u_p \tag{7}$$

In Eq. (7), $\boldsymbol{b}_{i,j}$ is defined as a unit vector representing the ith dimension in a j-dimensional space and

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_1 & a_2 & a_3 & a_4 & \cdots & a_{r-1} \end{bmatrix}, \qquad \boldsymbol{b}_{r,r} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{8}$$

A. Adaptive Element

The linear approximate model in Eq. (6) will not match the true nonlinear limit-parameter dynamics of Eq. (5) in all regimes of the flight envelope. Therefore, this approximate model is augmented with an additional adaptive element that can capture the modeling uncertainty Δ resulting from the choice of the static linear approximate model such that

$$\Delta(\boldsymbol{x}, y_p, y_p^{(1)}, \ldots, y_p^{(r-1)}, y_p^r) \triangleq h^{(r)}(\boldsymbol{x}, y_p, y_p^{(1)}, \ldots, y_p^{(r-1)}, u_p) - \hat{h}^{(r)}(y_p, y_p^{(1)}, \ldots, y_p^{(r-1)}, u_p) \tag{9}$$

and therefore,

$$y_p^{(r)} = \sum_{i=0}^{r-1} a_i y_p^{(i)} + b u_p + \Delta \tag{10}$$

Neural networks (NN) are powerful function approximators and have been previously used in similar situations encountered in control problems [15–18]. NN parameters can be tuned to approximate a continuous function in a compact domain to any desired level of accuracy.

In Fig. 4, a generic structure of a single-hidden-layer neural network (SHL-NN) is presented. A SHL-NN has three layers of neurons: namely, an input layer, a hidden layer, and an output layer. The output from the preceding layers feed into the adjacent forward layer along weighted interconnections. The values of interconnections between the neurons in the input layer and the neurons in the hidden layer, represented in a matrix form, is referred to as the hidden-layer weight matrix. Similarly, the output-layer weight matrix represents the value of interconnections between the output-layer neurons and the output of the basis-function blocks. The dimension of these weight matrices depend upon the number of neurons that are present in the individual layers.

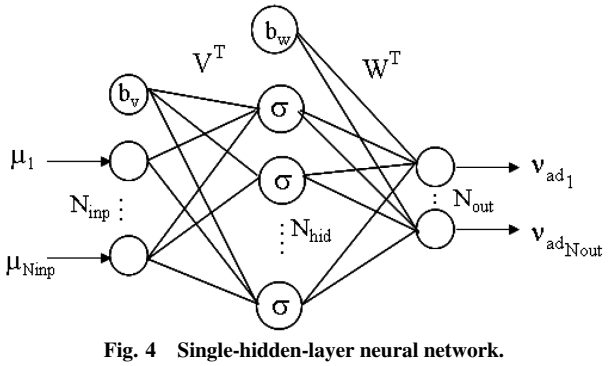

Fig. 4 Single-hidden-layer neural network.

Let N_{inp}, N_{hid}, and N_{out} be the number of neurons in the input, hidden, and output layers, respectively. Then $\boldsymbol{V} \in \Re^{(N_{\text{inp}}+1)\times N_{\text{hid}}}$ and $\boldsymbol{W} \in \Re^{(N_{\text{hid}}+1)\times N_{\text{out}}}$ are the hidden-layer and output-layer weight matrices, respectively. The output of the SHL-NN is given by

$$\boldsymbol{\nu}_{\text{ad}}(\bar{\boldsymbol{\mu}}) = \boldsymbol{W}^T\boldsymbol{\sigma}(\boldsymbol{V}^T\bar{\boldsymbol{\mu}}) \tag{11}$$

where $\bar{\boldsymbol{\mu}}$ is the normalized input vector constructed by concatenating the input-layer bias with the NN input vector. The vector $\boldsymbol{\sigma}$ consists of the outputs from the hidden-layer sigmoidal activation function of each neuron, where the jth neuron in the hidden layer is

$$\sigma_j(z_j) = 1/(1 + e^{-az_j}) \tag{12}$$

where a is called the activation potential. Let the input to the jth neuron be denoted by

$$z_j = b_v\theta_{vj} + \sum_{i=1}^{N_{\text{inp}}} v_{ij}(\mu_i) \tag{13}$$

where b_v is the input-layer bias and θ_{vj} the jth threshold. Finally, the input–output relationship for the kth element of the output layer is

$$\nu_{\text{ad}_k} = b_w\theta_{w_k} + \sum_{j=1}^{N_{\text{hid}}} w_{jk}\sigma_j(z_j) \tag{14}$$

Note that the threshold θ_{v_j} is part of the hidden-layer weight matrix $\boldsymbol{V}$ and θ_{w_k} is part of the output-layer weight matrix $\boldsymbol{W}$.

The neural network weights are tuned online using the following weight update laws:

$$\dot{\hat{\boldsymbol{W}}} = -[(\boldsymbol{\sigma} - \boldsymbol{\sigma}'\hat{\boldsymbol{V}}^T\bar{\boldsymbol{\mu}})(\hat{\boldsymbol{E}}^T\boldsymbol{P}) + \kappa \parallel \hat{\boldsymbol{E}} \parallel \hat{\boldsymbol{W}}]\Gamma_w \tag{15}$$

$$\dot{\hat{\boldsymbol{V}}} = -\Gamma_v[\bar{\boldsymbol{\mu}}((\hat{\boldsymbol{E}}^T\boldsymbol{P})\hat{\boldsymbol{W}}^T\boldsymbol{\sigma}') + \kappa \parallel \hat{\boldsymbol{E}} \parallel \hat{\boldsymbol{V}}] \tag{16}$$

where $\boldsymbol{P}$ is the solution of the following Lyapunov equation:

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} = -\boldsymbol{Q} \tag{17}$$

and $\boldsymbol{Q} > 0$. The matrices Γ_w and Γ_v are positive-definite matrices that influence the update rate of the NN weights and are therefore referred to as the output-layer and hidden-layer learning rates, respectively. The parameter κ is used in the NN update equations to ensure that the modeling error $\hat{\boldsymbol{E}}$ and the NN weights are ultimately bounded. The parameter $\boldsymbol{\sigma}'$ refers to the vector derivative of the vector sigmoidal function. Hence, the outputs of the derivatives of the sigmoidal function are used to create the $\boldsymbol{\sigma}'$ matrix. A Lyapunov-based proof of the convergence and boundedness of the NN response is given in [1].

Physical insight into the system limit-parameter dynamics is required to determine the measurable system variables that influence the modeling uncertainty. The NN input vector consists of a normalized set of these system variables. In some cases, measurements or estimates for some of these variables may not be available. For example, in the case of limit parameters with a relative degree greater than one, higher-order derivatives of the limit parameter are usually not available. Therefore, an adequate number of delayed values of limit-parameter measurement, as given in Eq. (18), are used within the NN input vector [17]:

$$\boldsymbol{u} = [\,y_p(t) \quad y_p(t-d) \quad \cdots \quad y_p(t-nd)\,]^T \tag{18}$$

B. Model Approximation

The estimate of limit-parameter dynamics obtained by augmenting the linear approximate model with a neural network is referred to as the adaptive estimate of limit-parameter dynamics. A block-diagram representation of the adaptive estimation architecture is presented in Fig. 5. The final differential equation form of the adaptive estimate is given in Eq. (19):

$$\begin{aligned}\hat{y}_p^{(r)} &= \hat{h}^{(r)}(\hat{y}_p, \hat{y}_p^{(1)}, \ldots, \hat{y}_p^{(r-1)}, u_p) + \nu_{\text{ad}}(\bar{\boldsymbol{u}}) - \nu_{\text{dc}} \\ \hat{y}_p^{(r)} &= \sum_{i=0}^{r-1} a_i \hat{y}_p^{(i)} + b u_p + \nu_{\text{ad}}(\bar{\boldsymbol{u}}) - \nu_{\text{dc}}\end{aligned} \tag{19}$$

The design of nonlinear adaptive observers in the form of Eq. (19) is an active area of research. Some recent results on the properties of the nonlinear dynamic systems that can be approximated using adaptive neural networks and results on the domain over which the approximations are valid can be found in [19,21] and, more recently, in [22].

Note that in addition to the neural network, the linear approximate model is also augmented by a linear dynamic compensator. The linear dynamic compensator, given by Eq. (20), is designed to provide additional stability to the error dynamics by attenuating the effect of NN modeling uncertainty and external disturbances (d_{ext} and w) on the limit-parameter estimation-error dynamics:

$$\dot{\eta} = \boldsymbol{A}_l \eta + \boldsymbol{B}_l e, \qquad \nu_{\text{dc}} = \boldsymbol{C}_l \eta + \boldsymbol{D}_l e \tag{20}$$

Equation (21) provides the definition for the limit-parameter estimation error:

$$e \triangleq \hat{y}_p - y_p \tag{21}$$

The limit-parameter estimation-error dynamics are obtained by subtracting Eq. (10) from Eq. (19):

$$e^{(r)} = \sum_{i=0}^{r-1} a_i e^{(i)} + (\nu_{\text{ad}}(\bar{\mu}) - \Delta) - \nu_{\text{dc}} \tag{22}$$

The error dynamics in Eq. (22), transformed into state-space representation using the state vector $\boldsymbol{E} \triangleq [e\ e^{(1)}\ \cdots\ e^{(r-1)}]^T$, are given in Eq. (23):

$$\dot{\boldsymbol{E}} = \boldsymbol{A}\boldsymbol{E} + \boldsymbol{b}_{r,r}(\nu_{\text{ad}} - \Delta) - \boldsymbol{b}_{r,r}\nu_{\text{dc}} \tag{23}$$

In some cases, measurements of limit-parameter derivatives are not available. Therefore, an error observer is used to obtain an estimate of the true error vector [18]. This estimate of error vector $\hat{\boldsymbol{E}}$ is then used in NN adaptation laws (Fig. 5). A proof of the convergence and boundedness of the error signals in the presence of the adaptation law of Eqs. (15) and (16) is also given in [1].

The well-known problem of persistency of excitation (PE) is difficult to characterize with the nonlinearities in the system [20]. However, in a practical sense, the PE condition plays a minimal role in the applications considered, because the neural network adaptation is very fast, compared with the aircraft dynamics.

Once a functional relationship in the form of Eq. (19) is established, it is up to the limit detection and avoidance methods to use this information for effective envelope protection. The following is a description of the use and implementation of these relationships onto the GTMax.

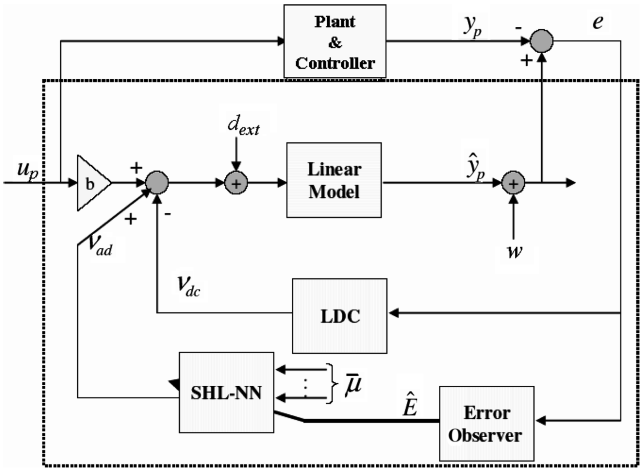

Fig. 5 Block-diagram representation of the adaptive estimation architecture (LDC denotes the linear dynamic compensator).

IV. GTMax Envelope Protection System

The GTmax is a modified Yamaha R-Max helicopter testbed at the Georgia Institute of Technology (Fig. 6). The autonomous helicopter is equipped with various hardware and software components for the testing of advanced control algorithms. Part of the recent effort involved the development, integration, and flight testing of an automatic EPS. The EPS is integrated as an add-on to the nominal flight control system architecture. During testing, it was desired to preserve the closed-loop system properties and to not change the low-level flight controller architecture. As a result, the EPS was designed as a self-contained and independent midlevel controller module interacting with the baseline controller commands (trajectory commands) of the existing GTMax flight controller using command-limiting techniques.

A. Baseline Flight Controller

A simplified block diagram of the GTMax baseline (low-level) flight controller is shown in Fig. 7. It includes a trajectory generator, an adaptive neural-network-based low-level flight controller, and command hedging to account for actuator nonlinearities such as rate and control saturation. The trajectory-generator inputs are heading commands and set points in space through which a trajectory is desired. The trajectory generator then generates corresponding acceleration, velocity, and position commands. The flight control system consists of adaptive outer and inner loops. The position, velocity, and acceleration commands from the trajectory generator are passed on to an outer-loop reference model, which filters the trajectory commands to generate smooth position and velocity commands for the outer loop. The outer-loop reference-model outputs are combined with proportional and derivative position-error feedback terms and the outer-loop adaptive neural network output to arrive at the outer-loop pseudocontrols. Model inversion of the outer-loop pseudocontrols is used to generate the collective stick input δ_f and desired attitude commands. The modified attitude commands are then given as input to the inner loop, along with user-specified angular rate commands. Similar to the outer loop, the inner loop computes three pseudocontrols, which are inverted to obtain longitudinal and lateral pedal-control inputs $\boldsymbol{\delta}_m$. The adaptive neural network output adds to the outer-loop and inner-loop pseudocontrols to minimize the error between reference-model states and the corresponding vehicle states. To prevent neural network adaptation

Fig. 6 GTMax autonomous helicopter testbed.

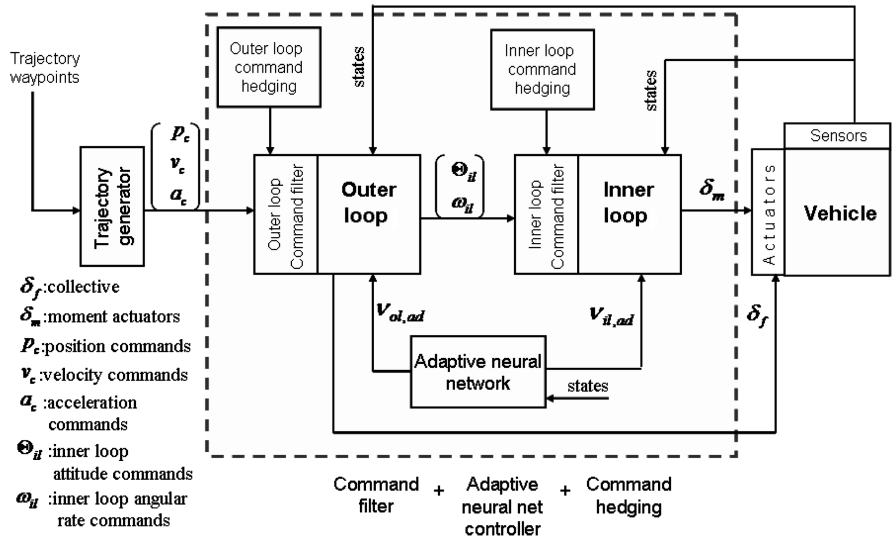

Fig. 7 GTMax adaptive neural-network-based controller architecture.

during control saturation, pseudocontrol hedging is incorporated into the control system, which hedges the outer and inner-loop reference models appropriately. More details of this controller are given in [1].

B. Envelope Protection System Architecture

During aggressive maneuvering of the GTMax, two critical limits were identified: load factor limit and rotor stall limit. The EPS was then designed to provide protection for both of the limits. The following is a description of the actual implementation.

1. Load Factor Protection

An effective way to prevent load factor limit exceedance in forward flight is to limit pitch rate and roll attitude response of the vehicle by limiting commands to the flight controller. Although roll attitude commands are directly limited in the controller using instantaneous measurements, the EPS is set up to compute the available pitch rate command margin and use it to limit pitch rate command and the corresponding trajectory and pitch attitude command inputs to the flight controller. Similar to Eq. (19), the following adaptive limit-parameter estimate is used for the load factor dynamics:

$$\dot{\hat{N}}_z = a(\hat{N}_z - N_{z_{\text{nom}}}) + bq_c + \nu_{\text{ad}}(\bar{\boldsymbol{\mu}}) + K(\hat{N}_z - N_z) \tag{24}$$

where q_c is the pitch rate command, and the real constants a and b were chosen to be -3.5 and 4, respectively. Sensor measurements of vehicle accelerations are used in the calculation of the load factor N_z. Noise is filtered from the acceleration sensor measurements by using a second-order low-pass digital Butterworth filter with a cutoff frequency of 0.2 Hz. The load factor response of the vehicle is then computed from the filtered acceleration measurements $\boldsymbol{a_m}$ using the following expression:

$$N_z = \frac{\|\boldsymbol{a_m}\|}{g} \tag{25}$$

where $\|\cdot\|$ represents the 2-norm, and the constant g represents the acceleration due to gravity.

Note that the load factor dynamics are approximated using a linear model and a nonlinear augmenting function ν_{ad}, representing the output of the adaptive SHL-NN. The network is selected to have a sigmoidal activation function as its basis function and the following input vector:

$$\bar{\boldsymbol{\mu}} = \begin{bmatrix} q_c & \frac{N_z}{2} & \frac{v_x^B}{80} & \frac{v_z^B}{50} \end{bmatrix}^T \tag{26}$$

More details of the parameters used are shown in Table 1.

During a maneuver, a load factor approaches its maximum value asymptotically. Therefore, the maximum value will occur during its steady state when $\dot{N}_z = 0$. Using this property, future limit and command margins can be calculated for a given load factor limit $N_{z_{\text{lim}}}$, when Eq. (24) is used and the following equations are solved for $\hat{N}_{z_{\max}}$ and $\hat{q}_{c_{\text{lim}}}$, respectively:

$$0 = a(\hat{N}_{z_{\max}} - N_{z_{\text{nom}}}) + bq_c + \nu_{\text{ad}}(\bar{q}_c, \bar{\hat{N}}_{z_{\max}}, \bar{V}) + Ke \tag{27}$$

$$0 = a(N_{z_{\text{lim}}} - N_{z_{\text{nom}}}) + b\hat{q}_{c_{\text{lim}}} + \nu_{\text{ad}}(\bar{\hat{q}}_{c_{\text{lim}}}, \bar{N}_{z_{\text{lim}}}, \bar{V}) + Ke \tag{28}$$

Note that the neural network output ν_{ad} is also a function of $\hat{N}_{z_{\max}}$ and $\hat{q}_{c_{\text{lim}}}$. The solutions to these equations are obtained by guaranteed fixed-point solutions of the neural networks [23]. The following iterations are used:

$$\hat{N}_{z_{\max_{i+1}}} = -a^{-1}[bq_c + \nu_{\text{ad}}(\bar{q}_c, \bar{\hat{N}}_{z_{\max_i}}, \bar{V}) + Ke] + N_{z_{\text{nom}}} \tag{29}$$

$$\hat{q}_{c_{\text{lim}_{i+1}}} = -b^{-1}[a(N_{z_{\text{lim}}} - 1) + \nu_{\text{ad}}(\bar{\hat{q}}_{c_{\text{lim}_i}}, N_{z_{\text{lim}}}, \bar{V}) + Ke] \tag{30}$$

In practice, about 5 iterations at each time step were enough for the convergence of the preceding equations.

Note that $\hat{N}_{z_{\max}}$ is an estimated future value of the limit parameter that will be reached with current controls q_c, whereas $\hat{q}_{c_{\text{lim}}}$ is the estimated control input that would be required to drive the maximum load factor response to the limit boundary $N_{z_{\text{lim}}}$. Hence, limit margin and control margins can be estimated as

$$\hat{N}_{z_{\text{marg}}} = N_{z_{\text{lim}}} - \hat{N}_{z_{\max}} \tag{31}$$

$$\hat{q}_{c_{\text{marg}}} = \hat{q}_{c_{\text{lim}}} - q_c \tag{32}$$

Table 1 Design parameters for the load factor EPS

Number of neurons in hidden layer	8
Output-layer learning rate Γ_w	4.0
Hidden-layer learning rate Γ_v	0.1
Sigma-mod parameter κ	0.02
Sigmoid activation potential	1.0
Compensator gain	8.0

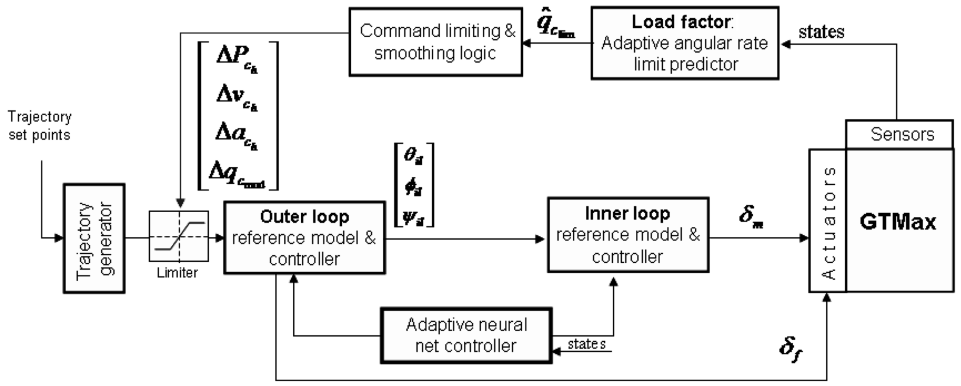

Fig. 8 Block-diagram representation of load factor limit-protection architecture.

A block-diagram representation of the integration of the load factor limit-protection system with the flight controller is shown in Fig. 8. The limiting is smoothed out to avoid chattering at the limit boundary, and modifications to the pitch rate command are provided at the onset of the approaching limit. The pitch rate command and the nominal pitch attitude command from the trajectory generator are modified and limited as follows for a positive upper pitch rate command limit:

$$q_{c_{\text{mod}}} = q_c + K_{Nz}(q_c - \hat{q}_{c_{\text{lim}}})\,\text{sign}(q_c - \hat{q}_{c_{\text{lim}}}) \tag{33}$$

where

$$K_{Nz} = \begin{cases} \exp(-k_2(q_c - \hat{q}_{c_{\text{lim}}})) & \text{if } q_c > \hat{q}_{c_{\text{lim}}} \\ 1.0 & \text{if } q_c \le \hat{q}_{c_{\text{lim}}} \end{cases} \tag{34}$$

and

$$\Delta\theta_{\text{lim}} \approx (q_{c_{\text{mod}}} - q_c)\Delta t \tag{35}$$

Although the angular rate and attitude commands are limited in the inner loop, a tracking error will develop in the outer loop because the trajectory is no longer followed, due to the limits imposed on the inner loop in the EPS. As a result, the adaptive neural network flight controller would recognize the resulting trajectory tracking errors as model uncertainties and try to adjust the pseudocontrols in the outer loop. This would lead to an inconsistency between the limitations imposed on the angular rate commands by the EPS and the attitude commands generated by the outer-loop controller. To prevent this, acceleration commands computed in the trajectory generator are also modified to reflect the changes being made by the EPS to the outer-loop controller commands. Corrections to the acceleration commands are calculated by reasoning that tracking a pitch rate command corresponds to rotating the desired force vector accordingly. Therefore, corrections to the nominal acceleration commands are calculated based on corrected pitch rate commands:

$$\Delta a^B_{c_x} = a^B_c q_{c_{\text{mod}}} \Delta t \tag{36}$$

The changes in acceleration commands are propagated via integration to changes in velocity and position commands. These command limits are used to adjust the limiters of Fig. 8.

2. *Rotor Blade Stall Protection*

The rotor blade stall typically gives rise to main rotor pitch change-link loadings and is identified as another limit parameter. Because it is not practical to measure rotor blade stall, it is approximated using its correlation with the expected retreating indicated tip speed (ERITS). The ERITS factor is an estimation of the air tip speed of the retreating side of the helicopter blade scaled by the load factor. It is concluded in [24] that an ERITS number less than approximately 300 ft/s is an indication of rotor blade stall:

$$\text{ERITS} = \frac{V_{\text{tip}}(\rho/\rho_s) - v_{\text{cas}}}{N_z} \tag{37}$$

where V_{tip} is the rotor tip speed, v_{cas} is the calibrated airspeed, and ρ/ρ_s is the relative air density.

An effective way to avoid rotor stall is to limit the horizontal-acceleration commands a_{c_h} in the outer loop of the adaptive flight controller:

$$a_{c_h} = \sqrt{(a^v_{c_x})^2 + (a^v_{c_y})^2} \tag{38}$$

Hence, the following dynamic model is constructed:

$$\dot{\text{ERITS}} = a(\text{ERITS} - \text{ERITS}_{\text{nom}}) + b a_{c_h} + \nu_{\text{ad}}(\bar{\mu}) + Ke \tag{39}$$

where the parameters a and b are selected as -3 and 55, respectively. The following input vector was used for the neural network:

$$\bar{\mu} = \begin{bmatrix} \frac{\text{ERITS}}{500} & \frac{a_{c_h}}{25} & \frac{v^B_x}{80} & \frac{v^B_z}{50} \end{bmatrix}^T \tag{40}$$

More parameter details are provided in Table 2.

Similar to the load factor, the ERITS factor also exhibits a first-order behavior to acceleration inputs. A block-diagram representation of the integration of the rotor stall protection system with the flight controller is shown in Fig. 9.

The limit-protection algorithm computes the available horizontal-acceleration command margin and uses it to limit the horizontal-acceleration commands to the outer loop of the flight controller. Similar to the load factor case, a command-limit smoothing logic is used to prevent any undesirable chattering in the command inputs at the limit boundary. Note that the limit on the ERITS number is a lower limit, and hence a lower bound on the horizontal acceleration is imposed:

$$a_{c_{h_{\text{mod}}}} = a_{c_h} + K_{\text{ERITS}}(a_{c_h} - \hat{a}_{c_{h_{\text{lim}}}})\,\text{sign}(a_{c_h} - \hat{a}_{c_{h_{\text{lim}}}}) \tag{41}$$

where

$$K_{\text{ERITS}} = \begin{cases} \exp(-k_1(a_{c_h} - \hat{a}_{c_{h_{\text{lim}}}})) & \text{if } a_{c_h} > \hat{a}_{c_{h_{\text{lim}}}} \\ 1.0 & \text{if } a_{c_h} \le \hat{a}_{c_{h_{\text{lim}}}} \end{cases} \tag{42}$$

Note that although the purpose of the smoothing logic is only to avoid chattering at the limit boundary, its parameters can be set such that the system would become active well before the command limit is reached, resulting in an overly conservative EPS design. In the following examples, it is believed that it was set appropriately, yet it

Table 2 Design parameters for the ERITS EPS

Number of neurons in hidden layer	8
Output-layer learning rate Γ_w	10.0
Hidden-layer learning rate Γ_v	0.4
Sigma-mod parameter κ	0.02
Sigmoid activation potential	1.0
Compensator gain	5.0

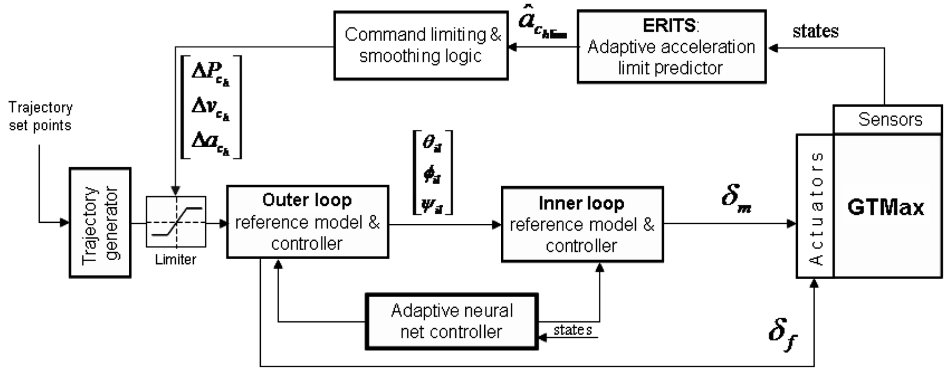

Fig. 9 Block-diagram representation of ERITS limit-protection architecture.

probably could be fine-tuned to even further reduce its activation threshold.

In this approach, the EPS requires an accurate estimate of the limit-parameter response behavior only when it approaches its limit value. This information is used to activate the system through the smoothing function and to calculate an accurate command limit; hence, the predictions need to be accurate near the limit boundary. This is precisely what the adaptive estimation model is expected to do: the approximate model dynamics will become more accurate locally when the system approaches the limit boundary. On the other hand, the EPS will not interfere with the closed-loop behavior of the system, because the EPS modifies the command to the controller, hence still preserving closed-loop dynamic behavior.

Although SHL-NNs are universal approximators when chosen to be sufficiently large, the actual set over which the approximations are valid is reduced when the network is of limited size. Therefore, the validity of the approximation is local. However, because the horizon over which the prediction is needed (prediction horizon) is a user-

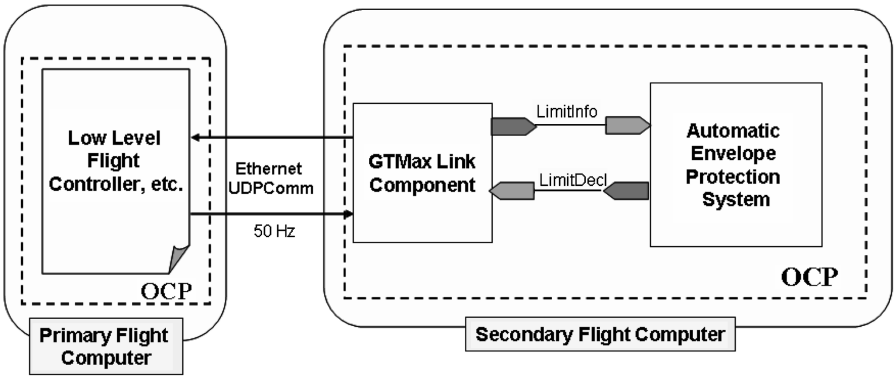

Fig. 10 EPS integration using OCP.

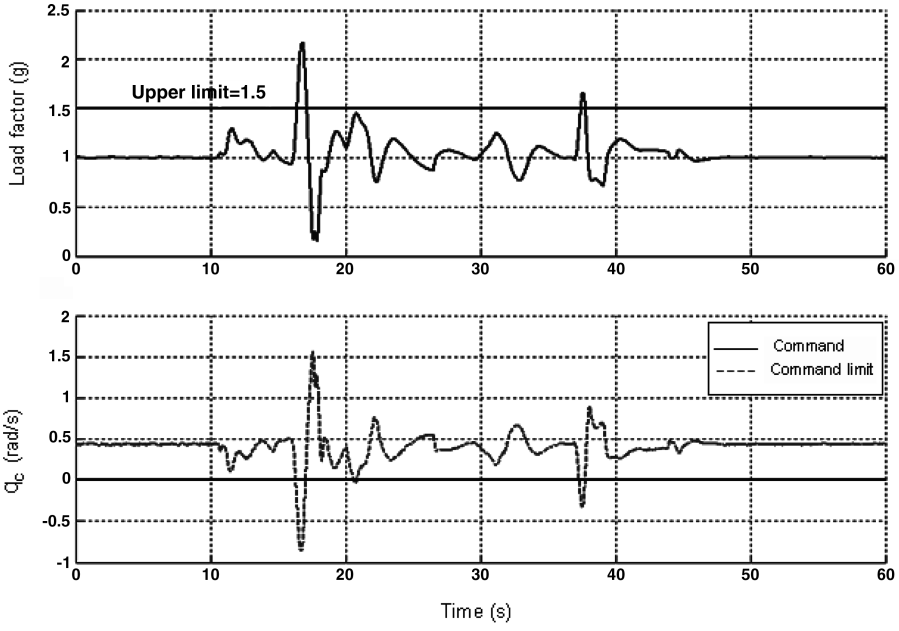

Fig. 11 Load factor response with EPS off.

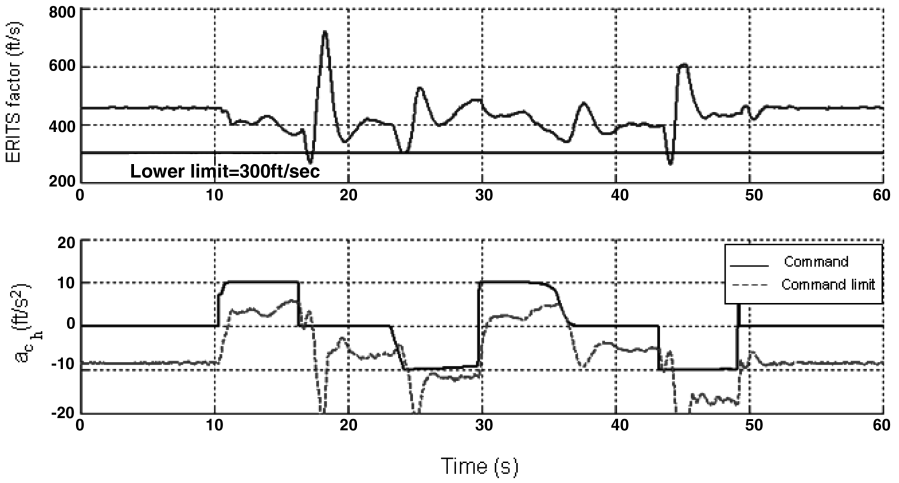

Fig. 12 ERITS-factor response with EPS off.

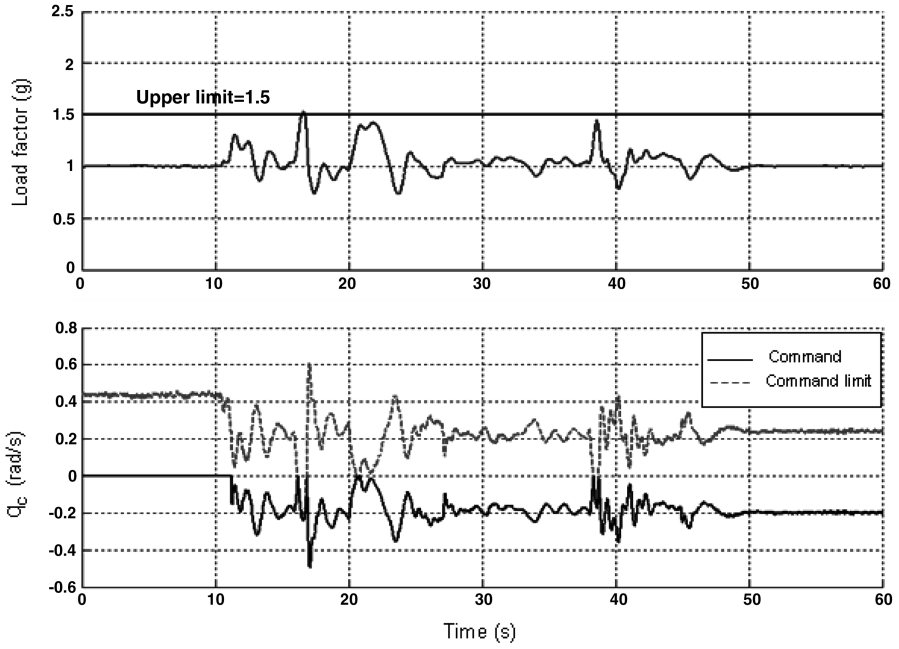

Fig. 13 Load factor Limiting with EPS on.

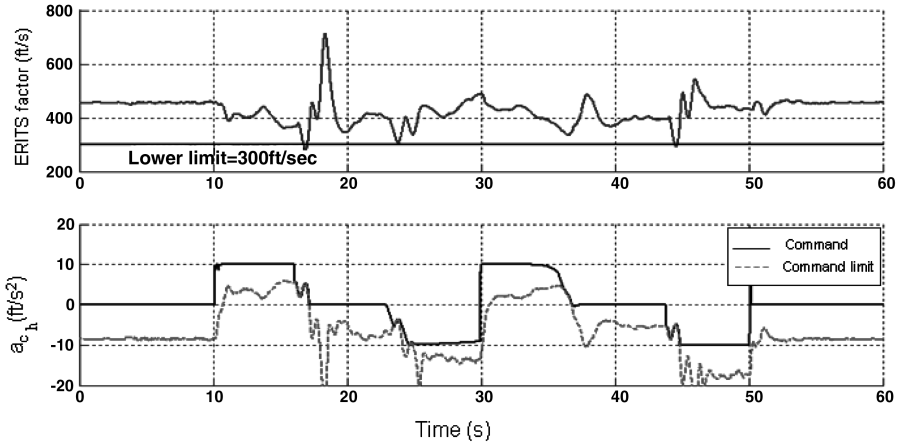

Fig. 14 ERITS-factor limiting with EPS on.

defined quantity, it is always possible to trade the prediction error with the complexity of the network needed for fidelity for an appropriate choice of prediction horizon. Here, the prediction horizon needed is arrived at using simulations.

All data required to calculate the ERITS factor are available from sensor measurements onboard the GTMax.

C. Switching Algorithms

When multiple limit parameters associated with multiple controls are present, the coupling between the limit-parameter dynamics might result in nonunique, or even no, solutions. For example, whereas reducing one control might protect one of the limit parameters, it might trigger the exceedance of another one. To address this problem, switching logic is designed to turn the various channels of the EPS on and off based on the importance of each limit parameter at the given flight condition.

The physical interpretation of this switching in the GTMax EPS is as follows. At low speeds, preventing rotor stall (lower limit on ERITS) and high values of load factor (upper limit on load factor) are generally not in conflict. A rapid deceleration to hover may give rise to significant slowing of the rotor and may result in exceedance of the assumed ERITS limit. As a result, at low speeds, all limit avoidance is done through manipulating the acceleration limit. As the forward velocity increases, the linear acceleration is no longer fast enough and the angular velocity control has to be used to prevent load factor violations (the critical limit factor in high speed). In the unlikely case of having both limits active, switching logic is implemented to avoid the conflicting case in which the acceleration command would be increased to avoid stall, which in turn would increase the load factor, therefore reducing the ERITS factor further. At high speeds, the switching logic gives priority to the angular velocity command, which ultimately avoids both the load factor and the rotor stall limits.

D. System Integration

The EPS component is integrated into the GTMax control architecture as a midlevel component through the OCP. The OCP is designed to allow features such as adaptability of individual components for optimal system performance, interoperability between different software and hardware platforms, and plug-and-play extensibility.

The GTMax avionics design incorporates two onboard computers. The primary computer runs the low-level flight controller with the

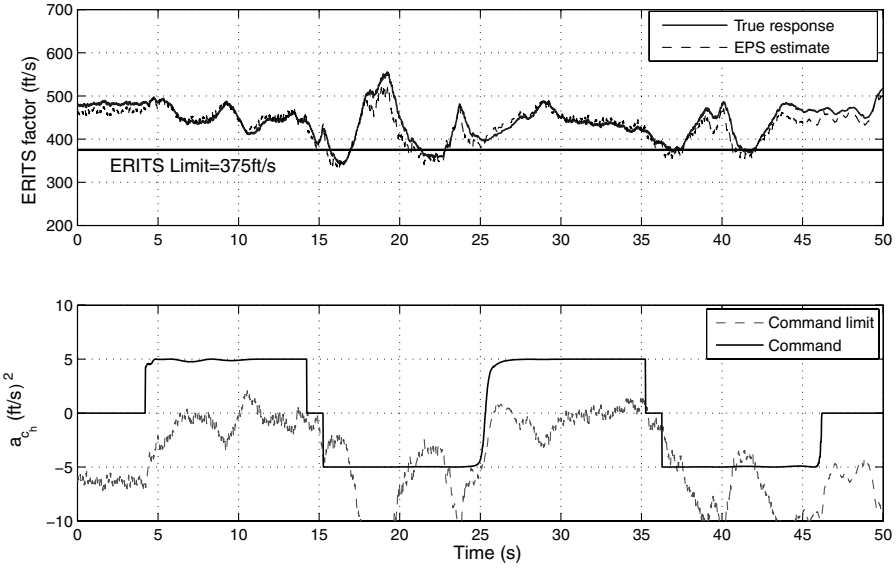

Fig. 15 ERITS-factor response for acceleration and deceleration with EPS off.

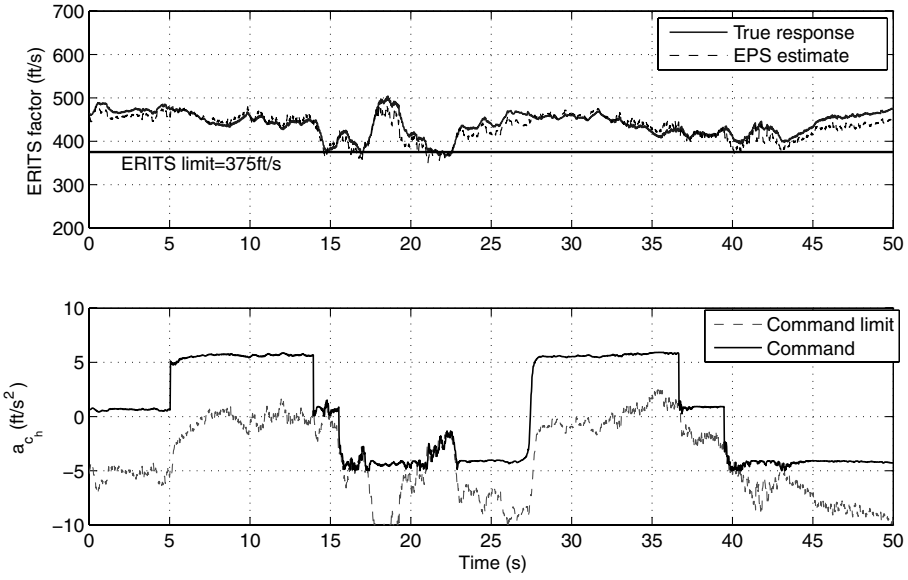

Fig. 16 ERITS-factor limiting for acceleration and deceleration with EPS on

trajectory generator and the secondary computer runs other software components, such as the midlevel controller components. In the implementation, the EPS is run on the secondary flight computer. The OCP data-link component was used to exchange data between the ethernet UDP ports of the primary and secondary flight computers. These signals are decoded within the OCP data link. Although the signal LimitInfo contains data required for limit detection, LimitDecl provides command limits to the low-level flight controller. The rate of communication between the primary flight computer and the secondary flight computer through the OCP data link component was set at 50 Hz (Fig. 10).

V. Simulation Results

The integration of a new software component into the GTMax flight software involves extensive testing. First, the onboard software source code is compiled for SITL simulations into a software environment called GUST (Georgia Tech Unified Simulation Tool). Here, all software and hardware is simulated. The GUST simulation includes models of all hardware and software signals, including a dynamic model of the GTMax with engine and rotor dynamics, sensor models, aircraft interfaces, actuator models, etc. It can inject environmental disturbances and time delays into the simulation. The GUST simulation is used frequently in the development of advanced control technologies for the GTMax, and for this purpose, the modeling fidelity is known to be satisfactory [25].

A. Aggressive E-Turn Maneuver

An aggressive maneuver called the e-turn maneuver was used to demonstrate the EPS in simulation. Here, the aircraft was commanded to accelerate from a hovering condition to about 85 ft/s

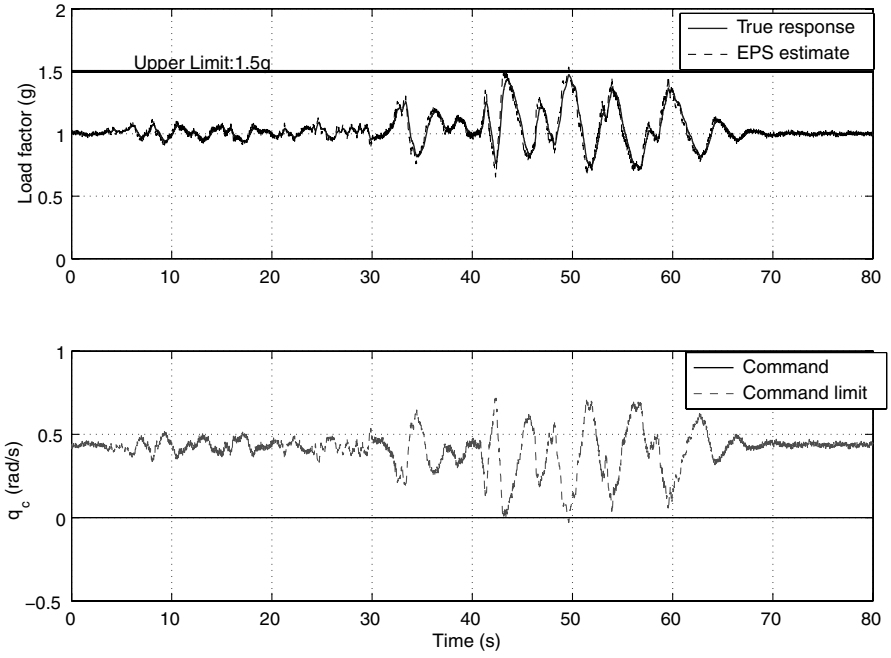

Fig. 17 Load factor response with EPS off.

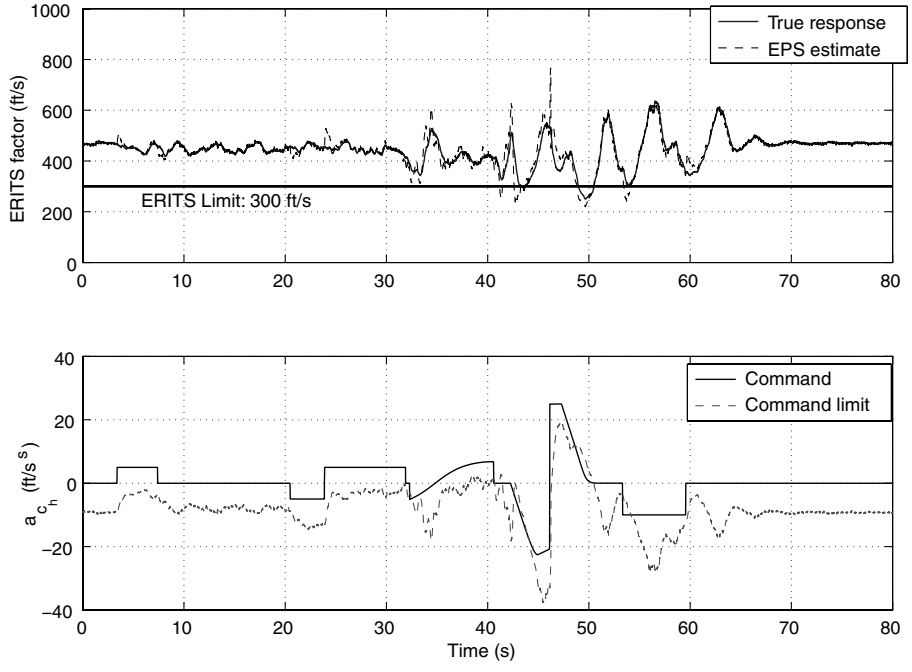

Fig. 18 ERITS response with EPS off.

forward-flight velocity, pull up to reach a point of zero velocity, and then make a 180 deg sharp turn and fly back to its original location at a maximum velocity of 85 ft/s and hover at the starting point. This maneuver was both structurally and aerodynamically one of the most demanding maneuvers performed during the flight tests of the GTMax.

The EPS was required to keep the aircraft within the prescribed load factor and ERITS limits. The upper limit for the load factor was set to 1.5 g and the lower limit for ERITS to 300 ft/s. These limits do not reflect the true limits of the GTMax, but are rather chosen to be able to run similar flight tests later without risking the aircraft. To be able to compare and demonstrate the effectiveness of the EPS, the maneuvers were run first for EPS switched off and then with EPS on.

The load factor response when the EPS is off is shown in Fig. 11. The pitch rate command and the pitch rate command limit (as predicted by the load factor limit-protection system) are also shown in the lower plot of Fig. 11. There is no explicit pitch rate command when EPS is off. All pitch rate change results from the translational trajectory commands. Whenever the load factor response exceeds the prescribed upper limit of 1.5 g, there is a corresponding violation in the commanded pitch rate, as seen in Fig. 11. The ERITS response with the rotor stall protection off is shown in Fig. 12. The horizontal-acceleration commands are also shown in Fig. 12. Whenever the ERITS response drops below the set lower limit of 300 ft/s, there is a corresponding violation in the horizontal-acceleration command.

The load factor response for the same maneuver with envelope protection on, along with the commanded pitch rate as modified by the envelope protection system, is shown in Fig. 13. Observe that the load factor violations are prevented by maintaining the commanded pitch rate within the predicted limits. The ERITS response with the rotor stall protection on is shown in Fig. 14. Observe that limit violations are prevented by maintaining the horizontal-acceleration command within the predicted limits.

VI. Flight-Test Results

Various maneuvers were flight tested to demonstrate the effectiveness of the EPS. Results for two of the maneuvers are presented.

A. Hover-to-Hover Acceleration–Deceleration Maneuver

First, an acceleration and deceleration maneuver was set up for the testing of the rotor stall limit protection only. The aircraft was commanded from the current hovering position to a forward position

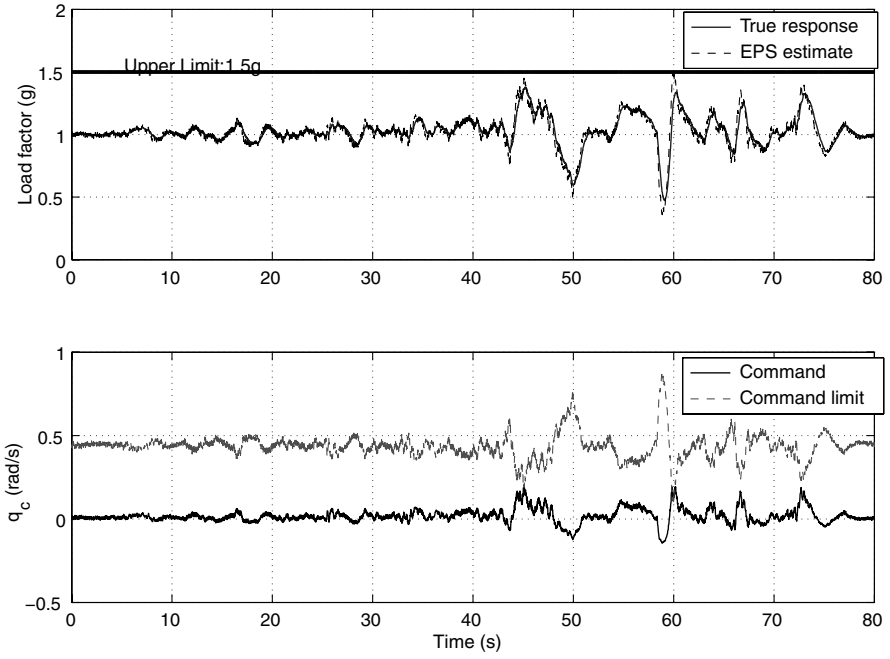

Fig. 19 Load factor limiting with EPS on.

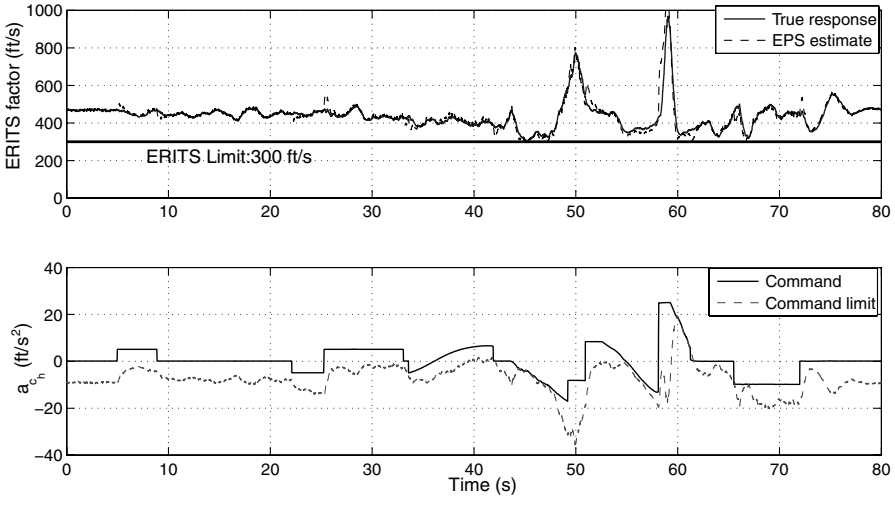

Fig. 20 ERITS limiting with EPS on.

of 600 ft through a sequence of acceleration–deceleration commands such that the vehicle came to hover at the commanded distance of 600 ft. The same was repeated for a backward flight (with the vehicle nose trailing the tail). These maneuvers required accelerations and decelerations that resulted in exceedance of the more conservatively set rotor stall limit (ERITS limit set at 375 ft/s).

Two flight tests were conducted. First, the limit-avoidance routine was off and the limit detection was on. During this maneuver, the vehicle response was seen to violate the selected ERITS limit boundary. Figure 15 shows the results of the future ERITS response estimation (shown as the EPS estimation in dotted lines) and the acceleration command-limit prediction. A horizontal-acceleration command-limit violation occurred whenever the future prediction response exceeded the limit. Next, the EPS was switched on and the same maneuver was executed (Fig. 16). The vehicle was successful in staying within the prescribed limits. Comparing Figs. 15 and 16, it is observed that with the EPS on, the EPS limited the acceleration commands to the command-limit value whenever a limit violation was foreseen. As a result, the ERITS stayed within the selected limit value.

B. Aggressive E-Turn Maneuver

Next, the aggressive e-turn maneuver was tested. The e-turn maneuver was performed at a forward velocity of 60 ft/s (Note that this lower value was used in contrast to the 85 ft/s previously stated in simulation evaluations so that the vehicle did not quickly go outside the visual range of the safety pilot.) The upper limit for the load factor was set to 1.5 g and an ERITS limit was set at 300 ft/s.

In Figs. 17 and 18, the load factor and ERITS-factor responses to the e-turn maneuver are presented with the EPS off. In this test case, the prediction of future limit and command margins were still enabled (dotted lines in the figure), but the EPS would not attempt to avoid any limit violations. Hence, it would be expected that the command inputs would violate the estimated command limits whenever the predicted limit-parameter response exceeded its prescribed limit. The estimated pitch rate command limit and the actual pitch rate command are shown in the lower plot of Fig. 17. Similarly, the estimated horizontal-acceleration command limit and the actual horizontal command are shown in the lower figure of Fig. 18.

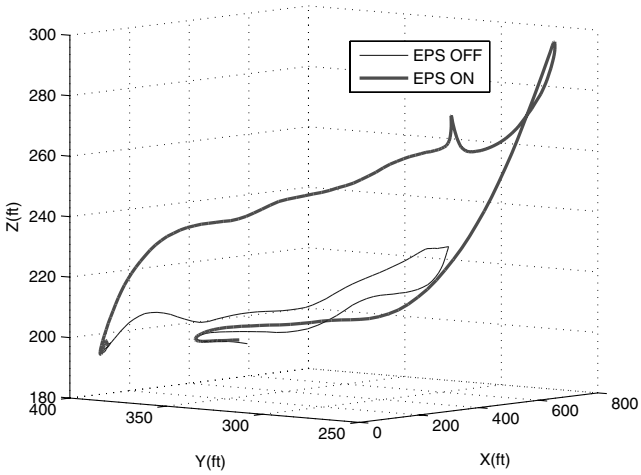

Fig. 21 Trajectory comparison.

Figures 19 and 20 show flight-test results with the EPS on for the e-turn maneuver case. It is clear from these figures that the commands to the flight controller were adjusted by the EPS when a violation of the command margins was predicted. The upper figures depict the limit factors and their future predictions. Note that with EPS on, the commands are not allowed to violate their respective estimated command limits. When the limit boundary is hit, the command rides along the estimated constrained boundary. As a result, both the load factor and the ERITS factor stay within their prescribed limits

A comparison of the resulting trajectories with EPS on and EPS off is shown in Fig. 21. The trajectory is altered to protect the vehicle from envelope exceedance. A comparison of the inertial velocities for both cases are presented in Fig. 22. The maneuver with EPS on is about 10 s slower. Note that in reality, a case of EPS off would be ill-defined, because if one were to use the true limit boundaries, the vehicle could not have survived or damage to the vehicle could have occurred.

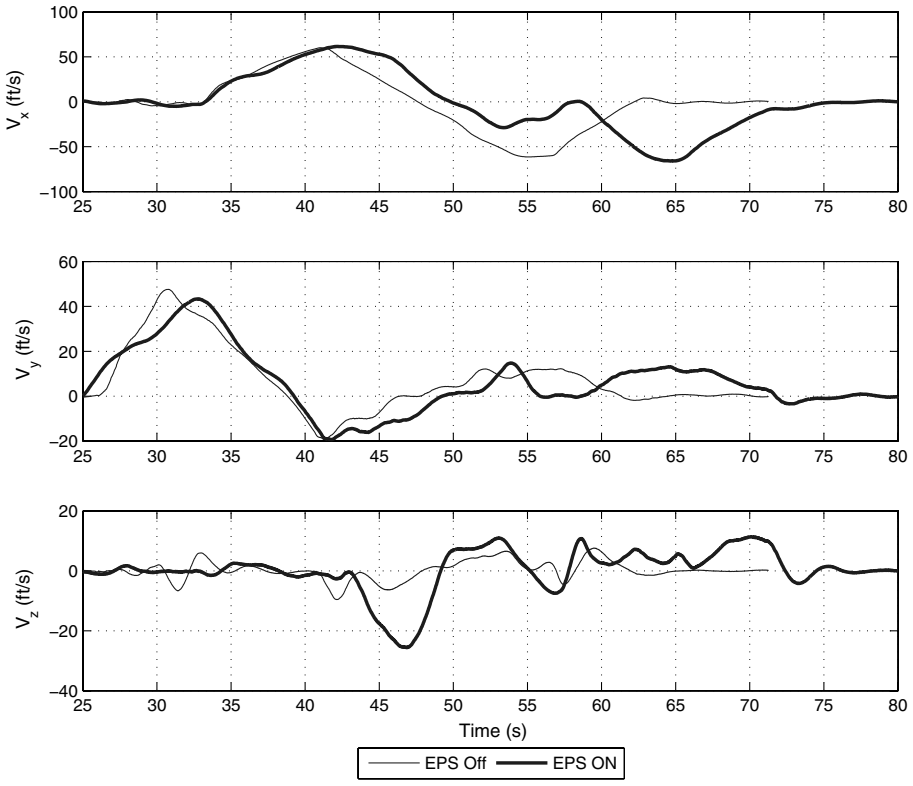

Fig. 22 Inertial velocity comparison.

VII. Conclusions

An automatic envelope protection system (EPS) is developed for the Georgia Institute of Technology's unmanned-helicopter testbed, GTMax. The EPS system is integrated into the GTMax simulation (GUST) and the onboard flight controller within the Open Control Platform (OCP) architecture. The modular architecture used retains the low-level adaptive flight control system and its architecture, previously developed and validated on the GTMax. This EPS design and integration approach can be applied across various vehicles and controller platforms without requiring any significant modifications to their respective nominal controller architectures.

The EPS system is designed to prevent large structural loads on the GTMax by limiting maximum values of the load factor and to avoid rotor stall by limiting lower values of the expected retreating indicated tip speed (ERITS). The effectiveness of the EPS system is demonstrated using high-fidelity simulation evaluations conducted using flight-verified simulation software for multiple aggressive maneuvers. Flight-test results have demonstrated that the envelope protection systems can provide realistic approximations of vehicle command limits. The adaptive estimations were able to successfully predict limit and command margins. It was observed that care should be taken when tackling multiple limits associated with multiple controls, as the coupling between the limit parameters and associated controls could result in nonunique, or no, solutions.

Procedures followed before flight tests increased system confidence and cut development time. The capabilities of the OCP were effective in the ease of integration and the flexibility in changing, developing, and interconnecting software components.

Acknowledgments

This study was carried out under the National Rotorcraft Technology Center (NRTC) Center of Excellence in Rotorcraft Technology (CERT) program and the Software Enabled Control program at Georgia Institute of Technology, sponsored by the Defense Advanced Research Projects Agency.

References

[1] Johnson, E., and Kannan, S., "Adaptive Trajectory Control for Autonomous Helicopters," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 3, May 2005, pp. 524–538.

[2] Gravilets, V., Mettler, B., and Feron, E., "Human-Inspired Control Logic for Automated Maneuvering of Miniature Helicopter," *Journal of Guidance Control and Dynamics*, Vol. 27, No. 5, 2004, pp. 752–759.

[3] Jeram, G. J., "Open Platform for Limit Protection with Carefree Maneuver Applications," Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, 2004.

[4] Horn, J. F., Calise, A. J., and Prasad, J. V. R., "Flight Envelope Limit Detection and Avoidance for Rotorcraft," *Journal of the American Helicopter Society*, Vol. 47, No. 4, Oct. 2002, pp. 253–262.

[5] Horn, J. F., Calise, A. J., and Prasad, J. V. R., "Flight Envelope Cueing on a Tilt-Rotor Aircraft Using Neural Network Limit Prediction," *Journal of the American Helicopter Society*, Vol. 46, No. 1, Jan. 2001, pp. 23–31.

[6] Menon, P. K., Iragavarapu, V. R., and Whalley, M. S., "Estimation of Rotorcraft Limit Envelopes Using Neural Networks," *Proceedings of the American Helicopter Society 52nd Annual Forum*, AHS International, Alexandria, VA, June 1996, pp. 1423–1431.

[7] Bateman, A. J., Ward, D. G., Barron, R. L., and Whalley, M. S., "Piloted Simulation Evaluation of a Neural Network Limit Avoidance System for Rotorcraft," AIAA Paper 99-4252, Aug. 1999.

[8] Prasad, J. V. R., Yavrucuk, I., and Unnikrishnan, S., "Adaptive Limit Prediction and Avoidance for Rotorcraft," *Proceedings of the 28th European Rotorcraft Forum*, Royal Aeronautical Society, London, Sept. 2002, pp. 88.1–88.7.

[9] Yavrucuk, I., Prasad, J. V. R., Calise, A. J., and Unnikrishnan, S., "Adaptive Limit Margin Prediction and Limit Avoidance," *Proceedings of the American Helicopter Society 58th Annual Forum*, Vol. 2, AHS International, Alexandria, VA, June 2002, pp. 856–866.

[10] Yavrucuk, I., Prasad, J. V. R., and Calise, A. J., "Carefree Maneuvering Using Adaptive Neural Networks," AIAA Paper 2002-4495, Aug. 2002.

[11] Yavrucuk, I., Unnikrishnan, S., and Prasad, J. V. R., "Envelope Protection in Autonomous Unmanned Aerial Vehicles," *Proceedings of the American Helicopter Society 59th Annual Forum*, Vol. 2, AHS International, Alexandria, VA, May 2003, pp. 2000–2010.

[12] Bay, J., and Heck, B., "Software Enabled Control," *IEEE Control Systems Magazine*, Vol. 23, No. 1, Feb. 2003, pp. 19–20.

[13] Horn, J. F., and Sahani, N., "Detection and Avoidance of Main Rotor Hub Moment Limits on Rotorcraft," *Journal of Aircraft*, Vol. 41, No. 2, 2004, pp. 372–379.

[14] Sahasrabudhe, V., Horn, J. F., Sahani, N., and Faynberg, A., "Simulation Investigation of a Comprehensive Collective-Axis Tactile Cueing System," *Journal of the American Helicopter Society*, Vol. 51, No. 3, 2006, pp. 215–224.

[15] Rysdyk, R. T., and Calise, A. J., "Adaptive Model Inversion Flight Control for Tiltrotor Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 3, 2001, pp. 402–407.

[16] Calise, A. J., Lee, S., and Sharma, M., "Development of a Reconfigurable Flight Control Law for Tailless Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 5, 2001, pp. 896–902.

[17] Lavretsky, E., Hovakimyan, N., and Calise, A., "Upper Bounds for Approximation of Continuous-Time Dynamics Using Delayed Outputs and Feedforward Neural Networks," *IEEE Transactions on Automatic Control*, Vol. 48, No. 9, 2003, pp. 1606–1610.

[18] Hovakimyan, N., Nardi, F., Calise, A. J., and Kim, N., "Adaptive Output Feedback Control of Uncertain Systems using Single Hidden Layer Neural Networks," *IEEE Transactions on Neural Networks*, Vol. 13, No. 6, Nov. 2002, pp. 1420–1431.

[19] Kim, Y. H., Lewis, F. L., and Abdallah, C. T., "A Dynamic Recurrent Neural-Network-based Adaptive Observer for a Class of Nonlinear Systems," *Automatica*, Vol. 33, No. 8, 1997, pp. 1539–1543.

[20] Lu, S., and Basar, T., "Robust Nonlinear System Identification Using Neural Network Models," *IEEE Transactions Neural Networks*, Vol. 9, No. 3, May 1998, pp. 407–429.

[21] Hovakimyan, N., Nardi, F., and Calise, A. J., "A Novel Error Observer-based Adaptive Output Feedback Approach for Control of Uncertain Systems," *IEEE Transactions on Automatic Control*, Vol. 47, No. 8, Aug. 2002, pp. 1310–1314.

[22] Lavretsky, E., Hovakimyan, N., and Calise, A., "Upper Bounds for Approximation of Continuous-Time Dynamics Using Delayed Outputs and Feedforward Neural Networks," *IEEE Transactions on Automatic Control*, Vol. 48, No. 9, 2003, pp. 1606–1610.

[23] Calise, A. J., Hovakimyan, N., and Idan, M., "Adaptive Output Feedback Control of Nonlinear Systems Using Neural Networks," *Automatica*, Vol. 37, No. 8, Aug. 2001, pp. 1201–1211.

[24] Whalley, M. S., Hindson, W. S., and Thiers, G. G., "A comparison of Active Sidestick and Conventional Inceptors for Helicopter Flight Envelope Tactile Cueing," *Proceedings of the American Helicopter Society 56th Annual Forum*, AHS International, Alexandria, VA, May 2000, pp. 181–204.

[25] Kannan, S. K., Koller, A. A., and Johnson, E. N., "Simulation and Development Environment for Multiple Heterogeneous UAVs," AIAA Paper 2004-5041, Providence, RI, 2004.