JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 29, No. 5, September–October 2006

Fault-Detection Design for Uninhabited Aerial Vehicles

Héctor P. Rotstein,* Ryan Ingvalson,† Tamás Keviczky,‡ and Gary J. Balas§
University of Minnesota, Minneapolis, Minnesota 55455

Fault Detection (FD) plays a vital role in ensuring the safety of a flight-control system, especially that of an uninhabited aerial vehicle. An FD algorithm is designed to detect a situation in which a faulty condition has occurred in the system. The main theoretical contribution of this work is a new residual threshold function, which is input dependent and enhances the FD capabilities of highly uncertain systems. The combined FD algorithm and new threshold function were simulated in the laboratory, in a high-fidelity hardware-in-the-loop environment, and flight tested as part of the Defense Advanced Research Projects Agency (DARPA) Software Enabled Control (SEC) Program. The DARPA SEC program is a research initiative designed to provide flight-control engineers with a reusable interface for the implementation of flight-control algorithms and flight management software on embedded systems.

Nomenclature

d	=	norm bounded model uncertainty excitation signal
E	=	residual energy
F	=	fault-detection filter
f	=	norm bounded fault model excitation signal
$\hat{f}$	=	fault model output signal
G_m	=	identified linear time-invariant plant model
g	=	threshold function
K_p	=	model uncertainty tuning gain
n	=	norm bounded model uncertainty excitation signal
r	=	fault detection filter residual
$\check{r}$	=	threshold function residual
$\tilde{r}$	=	residual error
S	=	running norm operator with a forgetting factor
T	=	projection operator
$\mathcal{T}_{\text{rw}}$	=	transfer matrix for $\mathcal{H}_\infty$ optimization
T_s	=	sampling rate of the fault detection algorithm
t	=	time variable/simulation time, s
u	=	heading rate command
$\boldsymbol{u}$	=	command vector of the true plant, or the nonlinear plant model
$\grave{u}$	=	corrupted (faulty) heading rate command
W	=	weighting function
$\boldsymbol{w}$	=	extended input for $\mathcal{H}_\infty$ optimization
y	=	heading rate output of the linear time-invariant plant model
$\boldsymbol{y}$	=	measurement vector of the true plant, or the nonlinear plant model
y_χ	=	heading output of the true plant, or the nonlinear plant model
$y_{\dot{\chi}}$	=	differentiated heading output of the true plant, or the nonlinear plant model
$\grave{y}$	=	faulty heading rate output
$\hat{y}$	=	heading rate output of the robust plant model
$\tilde{y}$	=	model error
z	=	discrete-time transfer function variable
$\hat{\beta}$	=	threshold function noise tuning parameter
Δ	=	perturbation matrix
Δt	=	input time delay of nonlinear plant model
κ	=	threshold function forgetting factor

Subscripts

f	=	fault model association
m	=	plant model association
n	=	noise model association

Presented as Paper 2005-6251 at the AIAA Guidance, Navigation and Control Conference, San Francisco, CA, 15–18 August 2005; received 30 July 2005; revision received 10 December 2005; accepted for publication 12 December 2005.

*Visiting Professor; on sabbatical from Rafael and Technion, Department of Electrical Engineering, Haifa, 32000, Israel; hector@ee.technion.ac.il.

†Graduate Student, Department of Aerospace Engineering and Mechanics; currently Research Engineer, Honeywell International, Navigation Communication and Control, Minneapolis, MN, 55418; ryan.ingvalson@honeywell.com. Member AIAA.

‡Postdoctoral Associate, Department of Aerospace Engineering and Mechanics, 100 Union Street, SE; keviczky@aem.umn.edu.

§Professor, Department of Aerospace Engineering and Mechanics, 100 Union Street, SE; balas@aem.umn.edu. Associate Fellow AIAA.

I. Introduction

THE Software Enabled Control (SEC) program[1] was a research initiative undertaken by the Defense Advanced Research Projects Agency (DARPA) to leverage recent developments in software and computing technologies for applications to control systems. The application domain for the technology is uninhabited aerial vehicles (UAV). The program culminated in Edwards, California, at NASA's Dryden Flight Research Center (DFRC) with a flight test of the Boeing T-33/UCAV flight test bed, which simulated a fixed-wing UAV. University of Minnesota (UMN) researchers, in collaboration with researchers from the University of California at Berkeley (UCB), implemented and flight tested a receding horizon control (RHC) and a fault-detection (FD) algorithm that made use of an adaptive customizable program interface. This interface is the open control platform (OCP).[2] It provided the software infrastructure to the T-33/UCAV for implementation of real-time adaptive control algorithms.

The T-33/UCAV aircraft was chosen because its avionics platform could be fitted with the same package as the X-45 uninhabited combat aerial vehicle (UCAV). Both aircraft are pictured in Fig. 1. This provided the SEC program with a single aircraft characterized by realistic UAV dynamics as well as being fitted with an avionics interface of a true UAV.

To understand the challenge involved in designing a FD filter for the SEC fixed-wing program, consider the block diagram of the plant as shown in Fig. 2. The usual assumption in FD design is that the control signals (i.e., the signals sent to the actuators) and the measurements of the dynamic response are all available to the FD algorithm. In this way, the FD design can be carried out on an open-loop system, that is, independent from having to consider the feedback from the stability augmentation and autopilot systems.

From an FD perspective, the advantage of using open-loop signals is clear. One of the very fundamental roles of a feedback loop is

a)

b)

Fig. 1 Images of the two aircrafts of interest for the SEC fixed-wing flight demonstration: a) the T-33 Jet Aircraft and b) the X-45 UCAV. (The T-33 photo in this figure was obtained from a online photo database. The UCAV photo was obtained from the Boeing Company's corporate web site: www.boeing.com.)

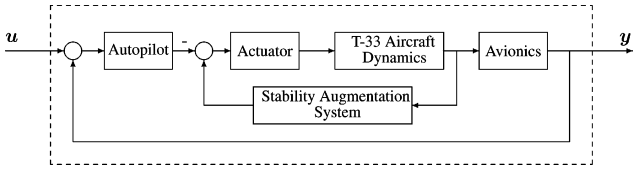

Fig. 2 Plant setup for the SEC fixed-wing experiment. Notice that only the vectors *u* and *y* are available for processing.

to desensitize the system to plant variations and uncertainty; hence, faults can be masked by the closed-loop control action of such systems. One could also claim that faults effectively masked by the control system are less important than those that reflect themselves in the closed-loop behavior. In such a situation, FD is still important because it directly affects safety, risk, and mission success assessments, but FD also becomes more difficult. Constraints such as these were similar to those imposed by the SEC program, yet our results show that a certain level of FD can still be achieved. This should *not* be understood as an argument in favor of using input-output or external signals even if the internal signals are available, because one of the conclusions of this project is that FD is indeed challenging without internal signals.

As alluded to in the preceding paragraph, the plant model available for the design of both the control and FD algorithms did not provide access to its internal, closed-loop signals. The plant model provided to SEC researchers was DEMOSIM. (This simulation was provided by the Boeing Company as part of the SEC program.) It is a nonlinear simulation of the T-33 dynamics, including models of the UCAV avionics, an autopilot, and a stability augmentation system. In other words, DEMOSIM is a black-box model with only the input vector $\boldsymbol{u}$ and measurement vector $\boldsymbol{y}$ available.

Almost every approach to FD assumes that either a valid, though possibly inexact, model of the internal dynamics of the plant of interest exists or that learning a model from data is possible (see the comprehensive books of Chen and Patton[3] and Gertler[4]). For the SEC program, a model was available, but the internal dynamic models were not. Because of this, the effect of a fault could not be internal introduced into DEMOSIM. The challenge was to develop an FD system that could be designed and tested, in hardware-in-the-loop (HIL) and flight test, given a limited set of information and actuation. This challenge was addressed with a number of tools, such as $\mathcal{H}_\infty$ fault detection, fault model estimation using a full nonlinear aircraft simulation, and system-identification techniques.

Given these limitations, an FD algorithm was developed for the T-33/UCAV flight test bed. The strategy employed for addressing the closed-loop FD problem is based primarily on the foundations of robust control theory, specifically that of $\mathcal{H}_\infty$ optimization. It was also necessary to synthesize an input-dependent threshold function, for which a novel approach was developed using ideas found in the model invalidation literature. The FD algorithm with the threshold function was a primary component of the OCP. Once integrated into the OCP, the algorithm was tested in an offline simulation, a HIL simulation, and flight tested as part of the SEC fixed-wing capstone demonstration. This paper focuses on the development and testing FD algorithm and threshold detection filter for the SEC program.

The outline of the paper is as follows. A brief background to FD is presented in Sec. II. Section III provides a detailed introduction to the SEC FD problem as well as a description of the setup and design of the FD filter. Section IV contains the theoretical contribution of the paper: the development of the input-dependent threshold function. This section also contains a discussion of a number of tuning parameters that were introduced in order to compensate for the conservative nature of the threshold function's decision criterion. Section V contains further modeling details and shows the considerations involved in designing the FD filter and in tuning the threshold function. This section presents performance results of the FD algorithm in an offline simulation. An overview of the full SEC fixed-wing capstone demonstration scenario is given in Sec. VI. The SEC capstone demonstration was the experiment plan for an offline simulation and was also the flight plan for both the HIL test performed at the Boeing Company and for the flight test flown at NASA DFRC in June of 2004. The results of these two tests are given in Secs.VII and VIII, respectively. Lastly, Sec. IX contains conclusions and a summary of results.

II. Fault Detection

Fault detection is understood as the ability to recognize unexpected changes in the response of a system, usually resulting from physical failures or breakdowns. Research efforts in FD started during the early 1970s, and a relatively large body of knowledge is now available. For a review of many results found in the literature, the reader is again referred to Chen and Patton[3] and Gertler.[4]

Arguably the most straightforward method of dealing with faults is to use redundant subsystems. For instance, a typical commercial aircraft's navigation sensing suite can contain triple-redundant inertial references plus double-redundant air data sensors. A voting scheme can be implemented to check the performance of the individual sensors and detect abnormal behavior.

Hardware redundancy is expensive and usually limited to high-end applications. This stimulates the drive to replace hardware redundancy by "analytic" redundancy whereby additional knowledge of the system is leveraged instead of actual redundancy. Earlier work in FD application and design, such as that by Massoumnia,[5] concentrated on the conditions for detecting and identifying faults for highly idealized models. The focus of this work was on the tradeoff

between the ability to detect faults and the level of noise in the measurements. Recently, paralleling the developments in control theory, the effect of model uncertainty has been recognized as a major factor affecting fault detection. This has given rise to the idea of robust fault detection, namely, the ability to detect faults in the presence of model uncertainty and other disturbances. Initially, robustness has been addressed indirectly by first designing a FD filter and then based upon the assumed uncertainty level applying a threshold filter to the *residual* (the output signal of the FD filter).[6,7] In particular, Emami-Naeini et al.[6] attempted to estimate the smallest size of the failure, which is detectable in spite of sensor noise and model uncertainty. In a later work, model uncertainty was explicitly taken into account, and several robust FD filters were proposed.[3,8–10] Preliminary applications of these techniques have also been reported in the literature, for example, see Marcos et al.[11]

A FD scheme usually consists of two stages: construction of a filter for generating residuals and a decision stage for analyzing the residuals and deciding if a fault has actually occurred. Relatively little has been done in combining robust FD filters with the synthesis of a robust threshold strategy. For example, in Stoustrup et al.[12] an optimal threshold function was investigated, where optimality was understood in terms of false alarm and missed detection rates. This approach provides a practical solution when the basic tradeoff is between fault detection and measurement noise, but becomes less convenient when measurement noise is small as compared to model uncertainty. The main tradeoff when working on *real* systems, as was the case for the SEC program, is between fault detection and model uncertainty. Thus, robust synthesis techniques are required for both the FD filter design and the threshold strategy.

When model uncertainty is large, the residuals generated by any FD filter are significant, and the design of a threshold function capable of handling model uncertainty becomes critical. This paper presents a solution to this problem using energy-like arguments. The proposed approach has similarities to the work in Shim and Sznaier,[13] where a model invalidation argument was used to decide whether a fault has occurred or not. The main disadvantage of the model invalidation approach was that it cannot be implemented online because it involves the solution of an optimization problem of monotonically increasing size. By exploiting the structure of the SEC problem, an alternative criterion is formulated for the FD threshold filter, which dramatically reduces the computation cost and enables real-time implementation.

III. $\mathcal{H}_\infty$ Fault-Detection Filter

The basic setup for this FD problem is the single-input, single-output (SISO) system shown in Fig. 3. One familiar with robust control theory will immediately recognize it as a typical "uncertain" system. The block G_m represents the nominal model of the plant, and the W blocks and the Δ blocks are used to represent the uncertain aspects of the system and are known as the weighting functions and perturbation matrices, respectively. The diagram is set up such that the response of the true plant (i.e., DEMOSIM or the T-33/UCAV) will be bounded by the set of $\hat{y}$ responses for all variations in the Δ blocks. Lastly, F is the filter to be designed. The primary difference in this use of $\mathcal{H}_\infty$ optimization is that problem proposed is open loop, that is, the transfer function to be designed does *not* lie in a feedback path, whereas in a standard $\mathcal{H}_\infty$ control problem the design block always lies in a feedback path. Even though our formulation is not a typical $\mathcal{H}_\infty$ optimization problem, it is well suited for it because the performance objectives and uncertainty descriptions fit well within the $\mathcal{H}_\infty$ framework.

As shown in uncertainty diagram of Fig. 3, there are two Δ blocks, also known as uncertainty descriptions. These uncertainty blocks are unknown and are in general linear-time-invariant transfer functions. They are used to account for unmodeled dynamics and can also be used represent modeling errors, such as static nonlinearities. Additionally, it is usually assumed that each Δ is norm bounded by 1, that is, $\|\Delta\|_\infty < 1$. With this assumption, the transfer functions W_f and W_m must be modeled appropriately to represent the magnitude and frequency content of the signals.

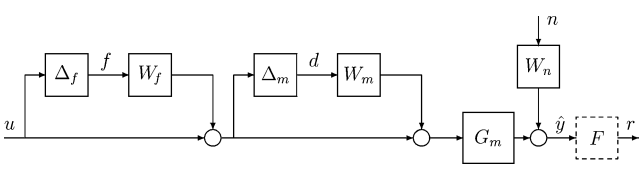

Fig. 3 Robust plant model for the $\mathcal{H}_\infty$ design of the FD filter F. The fault disturbance is represented by the Δ_f and W_f blocks, the model uncertainty by the Δ_m and W_m blocks, and the noise disturbance by the W_n block.

A. $\mathcal{H}_\infty$ Problem Formulation

Block F of Fig. 3 represents the FD filter to be designed, and in this section we outline the $\mathcal{H}_\infty$ formulation used to solve this problem. The first uncertainty block Δ_f of Fig. 3 is associated with the perturbations of the fault model. The second uncertainty block Δ_m represents the uncertainty involved in describing a physical system via a mathematical model, that is, modeling errors. Block W_n models exogenous disturbances in the system, for example, noise. The weighting functions W_f and W_m allow the introduction of a priori knowledge and frequency conttent about the nature of the fault and plant uncertainty. As mentioned in the Introduction, the objective of the FD algorithm is to detect a fault during a closed-loop operation, and these two uncertainty blocks are an input-output effort at modeling fault dynamics and plant uncertainty. G_m is a linear-time-invariant model of DEMOSIM (see Sec. V.A for details about G_m), and the mismatch between G_m and DEMOSIM is accounted for by the $W_m\Delta_m$ term. The fault model is represented by the $W_f\Delta_f$ term, and the fault dynamics were designed to represent an aileron actuator fault (see Sec. V.B for details). The setup in Fig. 3 highlights that if there is no frequency separation between the fault $W_f\Delta_f$ and the model uncertainty $W_m\Delta_m$, one cannot distinguish their individual effects using input-output signals. Therefore, the proposed approach requires that the fault and modeling errors lie in different frequency bands.

Following the $\mathcal{H}_\infty$ norm-based fault-detection approach,[11] the diagram in Fig. 3 is redrawn as shown in Fig. 4. In this figure, the uncertain blocks Δ_f and Δ_m have been removed, and the signals f and d have been isolated. Because the fault must be detected in the presence of any input signal u, the filter design setup must be independent of u. This implies that u must be eliminated from the block diagram of Fig. 3. Because u and G_m are known, their product $y = G_m u$ can be subtracted from the robust plant response $\hat{y}$. After simplifying the diagram, u is eliminated, and the new input to the filter F becomes $\tilde{y} := \hat{y} - y$.

A standard $\mathcal{H}_\infty$ optimization problem is formulated based on Fig. 4. This formulation assumes that the signals f and d are of size 1, so that the early assumption—that Δ_f and Δ_m are also norm bounded by 1—remains true. The objective of the $\mathcal{H}_\infty$ fault-detection design is to synthesize a filter F such that the transfer matrix $\mathcal{T}_{\text{rw}}$ between the extended input $\boldsymbol{w} := [f \;\; d \;\; n]^T$ and the residual error $\tilde{r}$ is small in an $\mathcal{H}_\infty$ sense. In particular, this would imply that r the residual of the FD filter tracks the fault $\hat{f} = W_f f$. Notice that this particular uncertainty formulation also covers disturbances at the plant input. In this form, standard $\mathcal{H}_\infty$ design tools can be used to design the filter. For alternative approaches to $\mathcal{H}_\infty$ fault detection, see Chen and Patton[3] (Chap. IX).

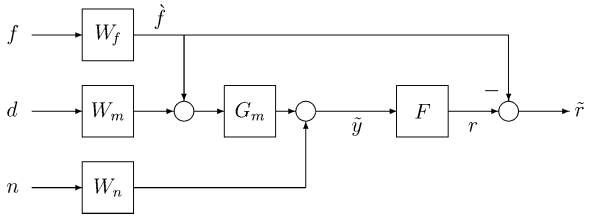

Fig. 4 Setup for the $\mathcal{H}_\infty$ fault-detection filter design. If the transfer function from f to $\tilde{r}$ is "small," then the disturbances d and n are rejected, and the residual r tracks the fault signal $\hat{f}$.

B. $\mathcal{H}_\infty$ Design Tradeoffs

To better understand the tradeoff involved in the $\mathcal{H}_\infty$ optimization problem, a closer look at the effect of the system inputs on the residual r is needed. These relations can be obtained from the system interconnection of Fig. 4. From the diagram, the residual is related to the inputs by

$$r = FG_mW_f f + FG_mW_m d + FW_n n \tag{1}$$

Equation (1) shows the tradeoff involved in the proposed fault-detection problem. In order that the residual r tracks the fault $W_f f$, the filter F should invert the plant G_m in the bandwidth of the fault (as determined by W_f). At the same time, the filter F should minimize the effect of the noise $W_n n$ and the plant-uncertainty/input-disturbance $G_m W_m d$. If there is adequate frequency separation between these two disturbances, then a good solution for F would be to approximate G_m^{-1} in the main frequency band of W_f and roll off at higher frequencies to prevent the disturbances from affecting the residual.

Notice that if W_m or W_n significantly overlap the main frequency band of W_f, then no FD filter will be able to isolate a fault adequately. This is a major concern in the design of an FD filter for an uncertain, closed-loop system. However, in many cases of practical importance the assumption of adequate frequency separation will hold, and a $\mathcal{H}_\infty$ FD filter produces the best compromise in terms of the $\mathcal{H}_\infty$ norm. This observation motivates the topic of the next section: the design of the FD decision criterion, that is, the threshold function.

IV. Is There a Fault?

A. Input-Dependent Threshold Function

In the absence of good frequency separation, plant uncertainty will result in a nonnegligible residual even if no fault is present. The use of a simple thresholding strategy will give rise to many of false alarms if the threshold is set too low, or increased missed detections if the threshold is set too high. This section describes a solution to this problem; an input-dependent threshold function is developed that exploits the additional information assumed about the system. Referring back to Fig. 3, r can be written as

$$r = F\{[G_m(I + \Delta_m W_m)(I + \Delta_f W_f) - G_m]u + W_n n\} \tag{2}$$

Following the model invalidation paradigm (e.g., see Smith et al.[14]), a fault will not be declared if there exists a stable Δ_m and n such that $\|\Delta_m\|_\infty \le 1$, $\|n\|_2 \le 1$, and

$$r = F(G_m \Delta_m W_m u + W_n n) \tag{3}$$

That is, there exists a linear-time-invariant uncertainty and a noise signal consistent with the problem that can explain the observed data.

In the context of fault detection, Smith et al.[14] shows that given $[u^T(\tau)\ r^T(\tau)]^T$ for $\tau = 0, \ldots, t$, the problem of verifying the existence of a norm-bounded uncertainty Δ_m subject to Eq. (3) can be transformed into an optimization problem with a linear matrix inequality constraint. This fact has interesting consequences for offline fault detection, but is not suitable for online fault detection because it involves the solution of a monotonously increasing optimization problem. The objective of this section is to present an alternative, although weaker, criterion suitable for real-time applications.

Consider the projection operator:

$$\left(T_{t_1}^{t_2} u\right)(\tau) = \begin{cases} u(\tau) & t_1 \le \tau < t_2 \\ 0 & \text{elsewhere} \end{cases} \tag{4}$$

with the simplifying notation $T^t = T_0^t$. Equation (3) can be replaced by the stronger condition

$$\|T^t r\|_2^2 \le \|T^t F G_m \Delta_m W_m u\|_2^2 + \|T^t F W_n n\|_2^2 \tag{5}$$

for each time instant t. Given a causal operator G, one has $\|T^t G u\|_2 \le \|G T^t u\|_2$, and so Eq. (5) implies that

$$\|T^t r\|_2^2 \le \|F G_m \Delta_m W_m T^t u\|_2^2 + \|F W_n T^t n\|_2^2 \tag{6}$$

Given the assumptions $\|\Delta_m\|_\infty \le 1$, $\|n\|_2 \le 1$,

$$\|T^t r\|_2^2 \le \|F G_m W_m\|_\infty^2 \|T^t u\|_2^2 + \|F W_n\|_\infty^2 \tag{7}$$

Because the preceding transfer matrices are constant for a given design, condition (7) can be rewritten as

$$\|T^t r\|_2^2 \le \alpha^2 \|T^t u\|_2^2 + \beta^2 \tag{8}$$

where

$$\alpha := \|F G_m W_m\|_\infty \tag{9}$$

$$\beta := \|F W_n\|_\infty \tag{10}$$

Condition (8) can be used for fault detection in real-time applications. Indeed, one can compute the threshold signal

$$g(t) = \|T^t r\|_2^2 - \alpha^2 \|T^t u\|_2^2 - \beta^2 \tag{11}$$

for each time t and declare a fault if $g(t) > 0$ at some time instant t.

Notice that if Eq. (5) holds for each time t, then Eq. (3) will also hold. The opposite is not necessarily the case for linear-time-invariant uncertainties Δ_m, and in general the former condition is much more restrictive. Also, the use of the triangular inequality and the norm-bounding properties make condition (8) a sufficient, but far from necessary, condition for Eq. (3). Therefore, condition (8) must be relaxed to make it useful in practice.

B. Relaxing the Threshold Condition

Condition (8) is relaxed by introducing two design parameters. In addition to bridging the gap between Eqs. (3) and (6), there are other reasons that motivate the introduction of the design parameters. One reason is that the weighting functions W_m and W_n were modified in the $\mathcal{H}_\infty$ design phase to achieve desirable behavior in the FD filter, such as a specific roll-off rate. These modifications might have resulted in slightly unrealistic disturbance models. This is especially true for the noise signal n, which is often stochastic in nature and can only be approximately modeled in the $\mathcal{H}_\infty$ framework. Another reason is that the FD filter and the threshold function are ill suited—because $\mathcal{H}_\infty$ is part of a worst-case criteria—for the critical FD design task of trading off false alarm and missed detection rates.

Two design parameters are introduced in condition (8) in an attempt to compensate for these difficulties. First, the running norm

$$\|T^t u\|_2^2 = \sum_{\tau=0}^{t-1} \|u(\tau)\|^2 \tag{12}$$

is modified by introducing κ a forgetting factor, with $\kappa < 1$

$$S^t u(t)^2 \doteq \sum_{\tau=0}^{t-1} \|\kappa^{t-1-\tau} u(\tau)\|^2 \tag{13}$$

This exponential decay on the influence of "old" data can be used for both norms in Eq. (11). The usage of the forgetting factor has two main consequences:

1) The threshold strategy in Eq. (11) tends to become insensitive to faults if W_m is relatively large as compared to the actual model mismatch observed in practice. This is especially true when the function is computed over long time intervals. The resulting large negative threshold values of g can give rise to *missed detection*. The forgetting factor will combat this by allowing g to return to normal levels.

2) At some points during the operation of the system, one might want to allow for relatively large model mismatch that is not captured by the uncertainty bound W_m. In these instances, Eq. (11) can result in a *false alarm* because the mismatch excites the FD filter in the same way as the fault. This difficulty can be overcome by temporarily ignoring the value achieved by g while the large mismatch is present and then allowing the exponential weight of the forgetting factor to bring g back to less than zero.

Second, the noise level β is replaced by a tuning parameter $\hat{\beta}$ that can be used to reduce the false alarm rate. This parameter can be tuned by analyzing $[u(t)\, r(t)]^T$ data records under benign conditions, for example, operating points where model mismatch is small.

V. Fault Detection for the SEC Program

A. Plant Model

As mentioned earlier, the fault-detection design for the SEC flight test was based upon DEMOSIM, a black-box nonlinear simulation of the T-33/UCAV aircraft. To summarize our knowledge of DEMOSIM, one can provide input commands, modify simulation parameters, and observe output data, but there is no access to the internal signals, dynamics, or logic. In addition to standard model uncertainty, one also needs to address the fact that DEMOSIM's autopilot implementation and internal discrete logic (saturation levels, limiters, etc.) are unknown. To make things even more difficult from a fault-detection perspective, the inputs to DEMOSIM are actually autopilot commands. Thus, only guidance-level (i.e., kinematic) control of the vehicle is possible.

The components of the input and output vectors used for control design are shown in Table 1. The fault-detection scheme described in the preceding sections was applied to a subsystem of DEMOSIM. Specifically, the lateral-directional dynamics were of interest, that is, the heading rate to heading channel, or simply denoted as the heading channel. This subsystem was selected because simulation analysis of DEMOSIM showed that the heading channel was effectively decoupled from the other two reference commands (see Table 1) if the commands are restricted to lie within certain tolerable limits. This simplified the fault-detection problem to a SISO system.

A linear model was identified from input/output data obtained from the DEMOSIM's heading channel. (For identification and simulation, numerically differentiated heading output of DEMOSIM was used as the measurement. Correspondingly, the identified plant model was actually a heading rate to heading rate transfer function.) The model identified was based upon the response of DEMOSIM to a 0.5-deg/s step command. A third-order autoregressive exogenous model was identified using the MATLAB® System Identification Toolbox.[15] With a sampling rate of 10 Hz, the discrete-time transfer function for this model was

$$G_m = \frac{2.48 \cdot 10^{-3} z^3}{(z - 0.98)(z^2 - 1.89z + 0.90)} \tag{14}$$

This is the model used in all design and analysis to follow. The frequency response of this model is shown in Fig. 5.

The linearized model of Eq. (14) was validated for use in control and FD design by extensive simulations comparing it with the nonlinear DEMOSIM. The sampling frequency of 10 Hz was selected as a tradeoff between fidelity, DEMOSIM constraints, and computational constraints. (Note that this model was also used to design a RHC, which requires online optimization.) No degradation in performance was observed when comparing the 10 Hz linear model with others models computed at higher sampling rate.

B. Fault Model

DEMOSIM did not provide a way of internally simulating a fault. This made it necessary to simulate a fault by corrupting either the input or output channel of DEMOSIM in such a way that the resulting output resembled a faulty system. There are many ways in which this can be done. The approach used in this paper involved insertion

Table 1 Input command and output measurement vectors for the T-33/UCAV test bed

Command vector $\boldsymbol{u}$	Measurement vector $\boldsymbol{y}$
Velocity, ft/s	Velocity, ft/s
Heading rate, deg/s	Heading, deg
Altitude rate, ft/s	Flight-path angle, deg

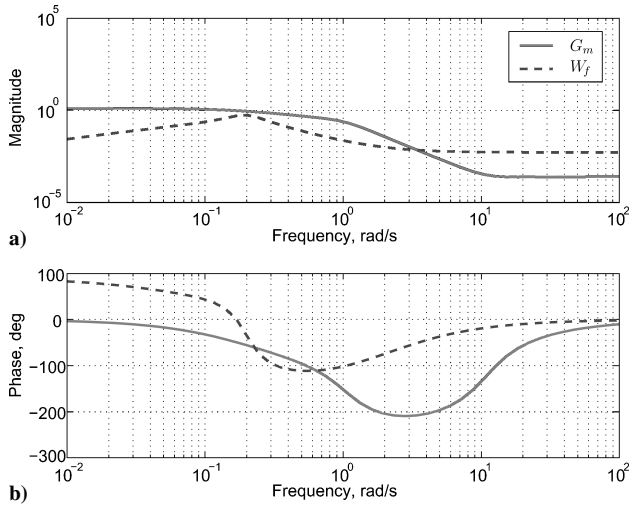

Fig. 5 Frequency responses for the plant model G_m and the fault model W_f. Note the frequency at which peak in magnitude occurs for W_f, as frequency separation of this model from the other input disturbances was critical during the FD design.

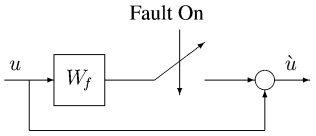

Fig. 6 Multiplicative fault input. The basic setup use for the implementation and identification of the fault model.

of a multiplicative input fault, shown in Fig. 6. In this figure, u is the actual or true command sent to DEMOSIM, and $\grave{u}$ is a "corrupted" command that will produce a "faulty" output. The no-fault scenario corresponds to the case when the Fault On switch is open, that is, $\grave{u} = u$.

Because only lateral motion is being considered in this FD problem, it is necessary to only look at faults that have strong coupling to this channel. One such fault would be an aileron actuator fault, and so W_f was designed such that the overall system (i.e., with the multiplicative fault input) would behave as if a true aileron actuator fault had occurred. At this point, it is important to stress that because the FD algorithm was designed by corrupting the closed-loop behavior of the lateral axis, it will in practice only detect a degraded lateral axis control condition, which does not necessarily imply an aileron actuator fault. Other conditions such as faults in a flap or wing damage can also degrade the lateral axis control.

Physically, an aileron actuator fault can result in changed dynamics of the actuator for example, a change in its damping or natural frequency. These dynamical changes—*faults*—could be caused by physical damage to the system, such as a loss of hydraulic pressure or damage to the aileron control surface. It is based upon these dynamical changes that we designed the fault model as will be seen next.

Identification of W_f

Identification of the fault filter W_f was performed using a Boeing 747 (B747) aircraft simulation. Unlike DEMOSIM, this simulation allowed internal parameters of the actuators to be varied, so that with it both nominal responses and faulty responses could be generated. The following paragraphs describe how the fault filter W_f was identified using these responses and how it was generalized and scaled according to the dynamics of DEMOSIM's heading channel.

Assuming a multiplicative input fault as in Fig. 6, the response $\grave{y}$ of the true faulty system was approximated by

$$\grave{y} = G_m(1 + W_f)u \tag{15}$$

From here, W_f is to be identified such that the preceding equation is a good model of the faulty system. The block diagram for this equation is shown on the left side of Fig. 7. If we assume that all signal levels remain within reasonable bounds, the nonlinear plant is

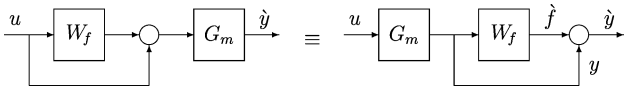

Fig. 7 Equivalent fault identification diagrams. The left side shows the initial setup, whereas the right side shows an equivalent representation that allowed the identification of W_f.

well approximated by G_m, and the left side of Fig. 7 can be redrawn with G_m on the input side. This equivalent representation is shown on the right side of Fig. 7. From this figure, it is clear that $\dot{y} = y + \dot{f}$, where $\dot{f}$ is the fault signal. The relationship can then be written as $\dot{f} = \dot{y} - y$, and because both $\dot{y}$ and y can be observed from the simulation the frequency response of W_f is calculated as

$$W_f = \frac{\text{FFT}\,(\dot{y} - y)}{\text{FFT}\,y} \tag{16}$$

where FFT denotes the discrete-time fast Fourier transform.

In a true system, any given instance of a fault can manifest itself with a slightly different frequency response for each occurrence. Thus, we did not characterize W_f with only one faulty response $\dot{y}$. Instead, we used a family of $\dot{y}$ responses to find W_f. This was accomplished by overbounding the collection of responses with a single "hand-fit" weighting function $\hat{W}_f$. For $\hat{W}_f$ to be an accurate model of the true fault, it was important that a large domain of faulty aileron dynamics was used in generating the family of $\dot{y}$ responses. The resulting $\hat{W}_f$ upper bound was used as the final weighting function. Future references to $\hat{W}_f$ will be denoted simply as W_f.

Generation of Nominal and Faulty Responses

As mentioned earlier, the fault simulated in the B747 simulation was an aileron actuator fault. In the simulation, the actuator is modeled as a second-order system with a nominal natural frequency $\omega_0 = 16.4$ rad/s and damping ratio $\zeta_0 = 0.67$. Using these parameter values, the nominal response [i.e., y of Eq. (16)] was generated with the simulation.

The family of the faulty responses was generated by running the simulation with a variety of natural frequency ω and damping ratio ζ combinations, where $\omega \in [3, 15]$ rad/s and $\zeta \in [0.7, 1.5]$. These sets of parameter values caused the actuator to exhibit a slower response, which is reasonable for a faulty or degraded actuator. Equation (16) was used to generate a frequency response for each of the faulty cases. A bounding transfer function for this set was found and used as the weighting function W_f in the FD design. The final transfer function identified was

$$W_f = \frac{2.6 \cdot 10^{-3} s(s+10)(s+5)(s+0.3)}{(s+0.9)(s+0.2)(s^2+0.416s+0.64)} \tag{17}$$

The frequency response of W_f is shown Fig. 5.

Because the dynamics of the B747 simulation were not the same as the dynamics of DEMOSIM, it was necessary to shift the frequency of W_f so that its response lied within the bandwidth of DEMOSIM's linear model G_m. The gain was also adjusted and was later used as a parameter to vary the intensity of the fault. This parameter provided control directly related to the missed detection rate of the filter. A higher gain for the function W_f meant more energy was now entering the system through the fault. This led to slower response filter designs with higher missed detection rates. [Note that it is not suggested that the procedure described in Sec. V.B is an ideal method for generating the fault filter W_f. On the contrary, whenever possible the filter should be found by introducing actual faults in the system and observing their input-output behavior, but because of the limitations of the system at hand (see the discussion about DEMOSIM in Sec. I), a reasonable alternative was necessary. In summary, our alternative approach consisted of the following: 1) basing the fault model on a high-fidelity nonlinear B747 simulation introduced with aileron faults; 2) computing a set of faulty closed-loop responses; 3) bounding the faulty responses by the function W_f using standard robust control arguments; and 4) scaling the frequency of W_f by comparing the bandwidth relation between the T-33/DEMOSIM and B747 simulations. This approach was not only physically motivated but also seemed reasonable, even though it meant that no claims of fidelity nor safety in practice could be made.]

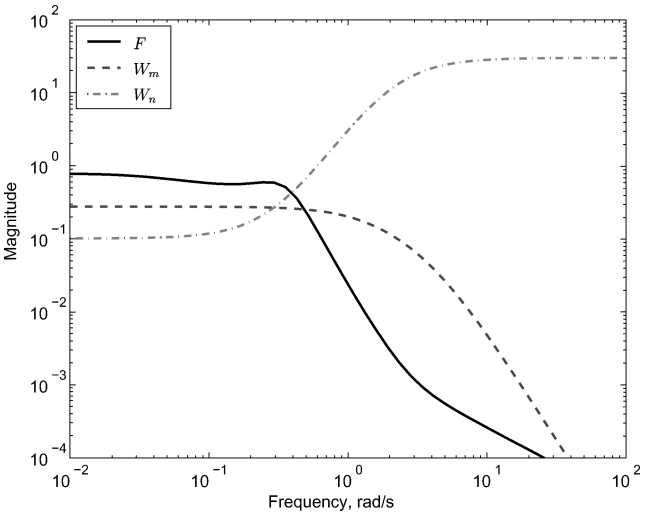

Fig. 8 Frequency response of the $\mathcal{H}_\infty$ filter design and weighting functions. These plots should be compared to the magnitude plots of Fig. 5, noting the interaction of the responses in the frequency range around 0.2 rad/s.

C. Filter Design Results

The characteristics of the synthesized $\mathcal{H}_\infty$ filter and its expected performance are presented in this section. Figure 8 shows a magnitude Bode plot of the $\mathcal{H}_\infty$ optimal FD filter F, corresponding to the problem formulation in Sec. III. Also included in this figure are the final weighting functions W_m and W_n used in the design.

From Fig. 8, it is clear that F is essentially a low-pass filter. This in not surprising because the problem has been formulated as a disturbance rejection problem. Also, the roll off occurs at a frequency that is expected, near the point an which the noise W_n begins to increase. In the sections to follow, the performance of this filter is discussed in multiple test environments and integration levels. First the FD filter without the threshold function is integrated with DEMOSIM and the OCP and tested by simulation. These results are generated from a priori command signals and assist in demonstrating the fundamental performance of the stand-alone FD filter. Following that, the FD filter with the threshold function is integrated with the other UMN/UCB SEC technologies into the entire SEC capstone demonstration flight-test plan, and its performance is assessed in various test situations: offline simulation, HIL simulation, and flight testing.

D. Offline FD Filter Performance

A simulation environment to test the fault-detection filter is shown in Fig. 9. This setup was based on the $\mathcal{H}_\infty$ design interconnection of Fig. 4 and the fault model setup of Fig. 6.

In this simulation, two 0.5-deg/s magnitude heading rate step commands are issued to DEMOSIM: a positive step at $t = 0$ s followed by a negative step at $t = 100$ s. The fault is inserted at $t = 70$ s. The simulation results in Fig. 10 include the heading rate command u, the faulty command $\dot{u}$, the model error $\tilde{y}$, the residual r, and fault signal $\dot{f}$, respectively. (In the simulation and flight-test results to follow, the model error is calculated with the true plant output $y_{\dot{\chi}}$ instead of the plant model estimate $\hat{y}$ used in the design phase.)

Because the fault was inserted at $t = 70$ s, the response to the first step command is a nominal response, and the response to the second step is a faulty response. Note from Fig. 5 that W_f rolls off at low frequency. This means that the fault is only excited during a transient phase of the input command u. This can be seen in Fig. 10b by observing that $\dot{u}$ remains constant even after that fault is turned

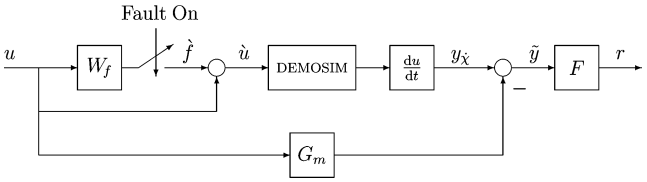

Fig. 9 Fault detection offline simulation diagram. This is the simulation setup that was implemented and used in the UMN laboratory to test and analyze the FD algorithm when integrated with the nonlinear plant DEMOSIM.

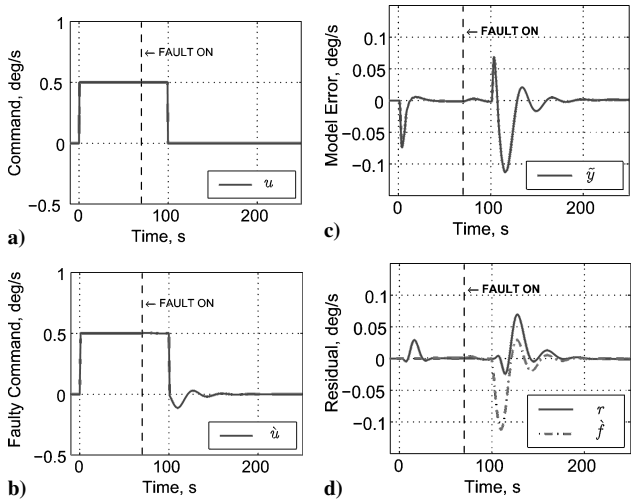

Fig. 10 Offline simulation results of the FD algorithm and DEMOSIM. Plot a) shows the heading rate reference command u, plot b) shows the "corrupted" command $\grave{u}$ that is used to generate the faulty plant response, plot c) shows the observed model error, and plot d) compares the residual r and the fault $\hat{f}$.

on. It is only after the second step (i.e., $t > 100$ s) that $\grave{u}$ becomes different from u.

As expected, the residual shown in Fig. 10d indicates that the FD filter is excited more highly during the faulty response than the nominal one. The output $\hat{f}$ of W_f is shown in Fig. 9d to indicate the tracking performance between these two signals. Recall that in the $\mathcal{H}_\infty$ optimization one of the performance objectives was that r was to track the fault $\hat{f}$. Although the figure clearly shows that that the tracking error is not always small, the residual does eventually converges towards the desired response $\hat{f}$.

One reason why the residual tracking error is not always small is model mismatch. To see this better, an additional run of the simulation was made. It was identical to that just described, except that the fault was not turned on. The main results of this run are shown in Fig. 11. The nominal (i.e., nonfaulty) heading rate responses $y_{\dot{\chi}}$ and y from this run are shown in Fig. 11a. Shown in Fig. 11b is the model error $\tilde{y} = y_{\dot{\chi}} - y$. Because the fault was not turned on, $\tilde{y}$ is a direct measure of the model mismatch over the entire run. Because the model mismatch is a component of the signal sent to the FD filter, there must be frequency separation between it and the fault in order its effect is filtered out. Such is not the case for this problem, and this fact indicates a tradeoff in the FD design between the tracking performance (i.e., of the residual r following $\hat{f}$) and the model mismatch rejection performance (which is related to the ability of the filter to detect a fault). On one extreme, heavily weighting the filter's tracking performance in the $\mathcal{H}_\infty$ design results in a filter that will track both the fault and the model mismatch. On the other extreme, attempting to only reject model mismatch results in an FD filter that is benign even to the fault. In both cases the fault is indistinguishable from the model mismatch.

This tradeoff was addressed by considering the energy of the residual. Shown in Fig. 12 is a 5-s running integral of the energy of the residual r of Fig. 10d. Clearly based upon the integral, there

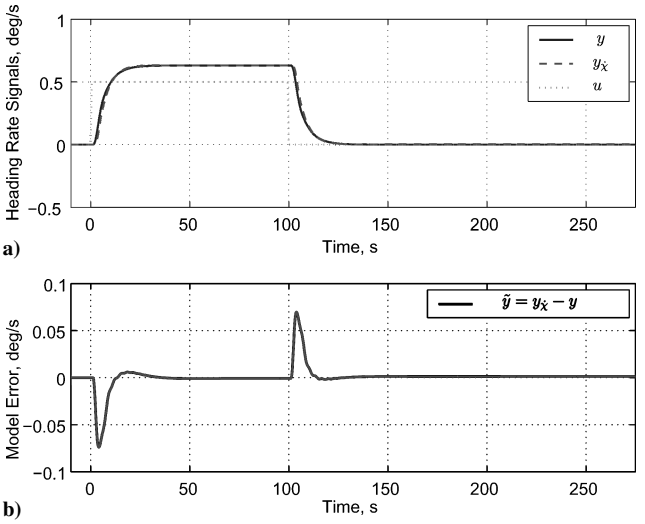

Fig. 11 Offline simulation heading rate responses and model mismatch. Plot a) compares the heading rate responses $y_{\dot{\chi}}$ and y to the command u. Plot b) shows the model error between DEMOSIM and the model G_m.

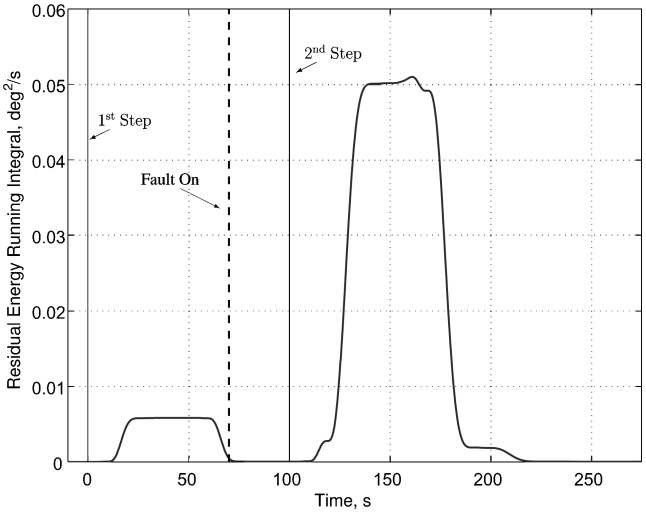

Fig. 12 Running integral of the residual's energy with a 5-s window. This plot gives insight into the residual's behavior in the presence of a fault. It shows clearly how the energy content of the signal is increased during a fault.

is much more energy stored in the residual after the second step compared to the first step. We denote the peak energy level after the first step (and prior to the second step) as E_1 and the peak energy level after the second step as E_2. Then we can define the difference as $\Delta E = E_2 - E_1$. Not surprisingly filter designs with good tracking performance had large values for E_1 and E_2 but small ΔE values, and filter design that exhibited good model mismatch rejection also had small ΔE values but with small peak energy levels. In trying to weigh the tradeoff, rather than looking directly at the tracking or model mismatch rejection performance, the energy difference ΔE was instead used as a criterion. Specifically, the filter design with the largest value of ΔE was chosen as the final FD filter.

VI. Software-Enabled Control Capstone Demonstration

A. Software-Enabled Control Capstone Demonstration Overview

The SEC fixed-wing capstone demonstration involved a flight test, which took place on 21 and 24 June 2004 at NASA DFRC in Edwards, California, of a Boeing T-33/UCAV aircraft that was

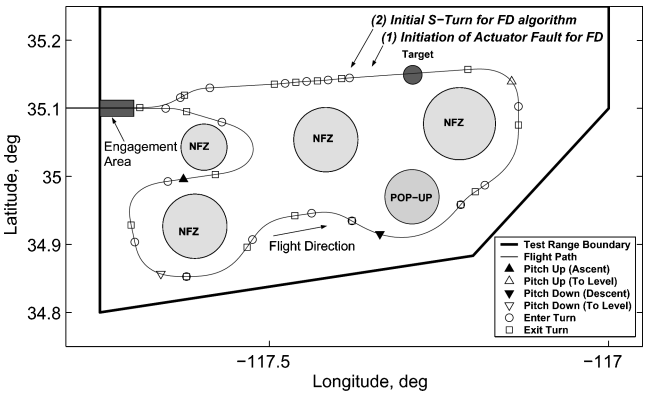

Fig. 13 UMN/UCB SEC Capstone Demonstration Experiment plan. This figure shows the RHC engagement area, the planned reference trajectory and maneuvers, no-fly-zones (NFZs), the pop-up obstacle, the target location, and finally the fault-detection segment.

integrated with the UMN/UCB SEC technologies. These technologies included a RHC algorithm, a RHC application program interface (RHC API), and a combined FD algorithm and threshold function. Prior to being flight tested, the flight plan associated with the SEC UMN/UCB capstone demonstration was also simulated offline and in HIL. The results of these tests are presented in the following sections.

Outlined next are the events of the SEC UMN/UCB flight demonstration experiment plan. Figure 13 is a diagram of the flight demonstration test range and the UMN/UCB flight plan. The flight experiment is divided into four main segments: initialization; phase I—pop-up threat avoidance; phase II—target engagement; and phase III—fault insertion and detection.

The first segment is the initialization. It is a set of conditional procedures that the pilot must undertake to ensure a safe transition of control from the autopilot to the RHC algorithm. The objective is to ensure the T-33/UCAV is flying within tolerable limits and is at the proper altitude and speed at the time the RHC is engaged. Because the trajectory is defined relative to the point of RHC engagement, care had to be taken to engage the RHC in a timely manner; otherwise, the trajectory might fall outside the test range boundary. To ensure that this would not happen, an RHC engagement region near the ingress point for the T-33/UCAV was determined. This area is denoted as the engagement area in Fig. 13. The initialization determines if the flight test will proceed, and a successful initialization occurs if the aircraft at the time of RHC engagement is within the engagement area and is flying a westerly track.

After initialization, the flight test starts with phase I. This phase involves trajectory tracking and obstacle avoidance while proceeding towards a predefined target. In this phase the reference trajectory passes around static no-fly-zone obstacles (NFZs, see Fig. 13) and towards the vicinity of a potential pop-up obstacle, which might or might not appear depending upon the decision of the flight experiment manager. In the case when the pop-up is inserted, the flight path will be replanned to avoid the obstacle, and the T-33/UCAV will continue towards the target.

Phase II begins after the aircraft has passed the pop-up obstacle. This phase includes positioning the T-33/UCAV for target engagement. The T-33/UCAV also maneuvers around additional NFZs and continues towards the target for engagement. Target engagement occurs when the T-33/UCAV flies over the target. This event signifies the end of phase II.

Lastly, phase III is the flight segment of most interest for fault detection. In this segment of the flight plan, the aileron actuator fault is inserted, and the combined FD filter and the threshold function of Sec. IV is used to detect the fault. Within 30 s of target engagement, the fault is turned on, following which is a series of S turns designed to excite the fault model. During target engagement, the aircraft should be in straight and level flight, and because the fault model is not responsive except during a transient the fault should not be detected until the onset of the S turns.

Table 2 FD threshold function design parameters

Design parameter	Description	Value
κ	Forgetting factor	0.95
$\hat{\beta}$	System noise level	1.9×10^{-3}
K_p	Model uncertainty tuning gain	0.5

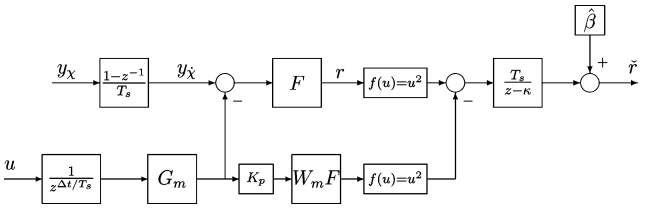

Fig. 14 Setup for the combined FD algorithm and threshold function. This is the full FD algorithm that was used in all SEC capstone demonstration experiments. This setup includes the full implementation of the input-dependent threshold function and its three tuning parameters: κ, $\hat{\beta}$, and K_p.

Testing the performance of the FD algorithm required a trajectory that would result in heading rate commands within the range of validity for the FD algorithm, $u \leq |0.5|$ deg/s. (This range was determined from standalone testing of the FD algorithm and DEMOSIM. This limitation was, in effect, a direct result of the fact that the internal dynamics of DEMOSIM were unavailable to the SEC researchers. This caused an inability to identify a high-fidelity model of DEMOSIM, which resulted in significant model mismatch.) Based upon the expected velocity, a way-point trajectory was designed such that the heading rate would fall within this range. In addition, the S-turn segment of the FD trajectory was designed so that the heading rate remained $\leq |0.2|$ deg/s. This limit ensured that the model mismatch would not dominate the filter residual. The onset of the S turns is shown in Fig. 13. For further details on the complete UMN/UCB flight demonstration experiment, see Keviczky et al.[16]

B. Simulation Environment of the UMN/UCB Capstone Demonstration

Figure 14 shows the combined FD filter and threshold function setup (i.e., the full FD algorithm) for testing of the SEC UMN/UCB capstone demonstration in offline simulation, HIL simulation, and flight. The only notable differences for each test situation is the source of the signal y_χ. The source of y_χ is DEMOSIM for the offline simulations, a high-fidelity T-33/UCAV model (a proprietary model used by the Boeing Company) for the HIL simulations, and the T-33/UCAV test bed for the flight tests. Additionally, in the offline simulations the RHC and the FD algorithms have been implemented in a MATLAB/Simulink environment that interfaces with the OCP and DEMOSIM, instead of being directly integrated into the OCP C++ code.

The inputs to the full FD algorithm are y_χ and u, the plant's heading output and heading rate RHC command, respectively. The output of the algorithm is $\check{r}$ the threshold residual. The signal $\check{r}$ is not to be confused with the FD filter residual r. (Both signals are denoted in Fig. 14.) Table 2 contains a description of the design parameters in the threshold function of Fig. 14 and their final design values.

C. Capstone Demonstration Offline Simulation Results

Figures 15 and 16 show the offline simulation results of the FD segment of the SEC UMN/UCB capstone demonstration flight plan with DEMOSIM as the plant. The timescales are shifted relative to time t_{on}, which denotes the time the command is switched from u to $\grave{u}$. This occurs just before the series of S turns designed to excite the fault for detection. These turns begins at $t_{\text{turn}} = 20$ s. The heading rate commands and response of DEMOSIM for the simulation are shown in Fig. 15.

The threshold function residual $\check{r}$ for this simulation is shown in Fig. 16. Because the trajectory was designed such that DEMOSIM

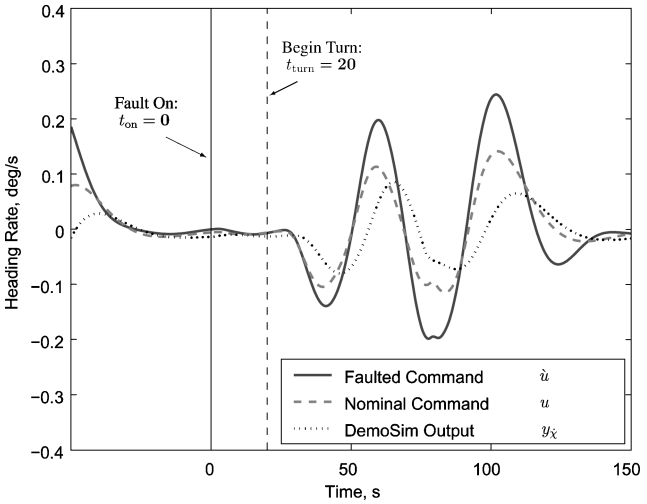

Fig. 15 Heading rate signals from the offline simulation of the SEC capstone demonstration with DEMOSIM and the combined RHC and full FD algorithms. Shows the response $y_{\dot{\chi}}$ of DEMOSIM to the faulty command $\grave{u}$. The nominal command u is also shown for comparison.

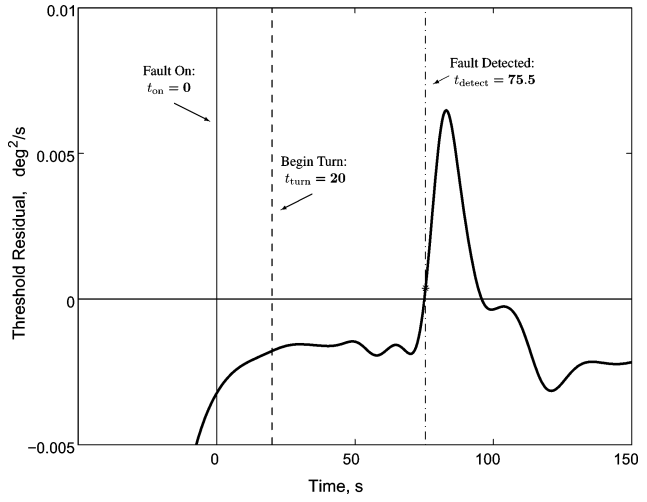

Fig. 16 Threshold residual $\check{r}$ from the offline simulation of the SEC capstone demonstration. The relevant events—fault insertion, onset of S turns, and fault detection—with their corresponding times are denoted.

would be in straight and level flight at t_{on}, no activity is expected from the FD algorithm until it is excited by the S turns, which begin at $t_{\text{turn}} = 20$ s. From Fig. 16 one can see that excitation of the residual occurred sometime after 20 s. Then at $t = 75.5$ s the residual $\check{r}$ became positive, and thus a fault was detected. Although there is delay in the response of the residual (because of the running integral of the threshold function), it nevertheless accomplished its goal and detected the fault.

VII. Hardware-in-the-Loop Test Results

HIL testing was performed by the Boeing Company as part of flight-test preparations for the final capstone demonstration. All of the UMN/UCB algorithms to be tested were implemented in C++ code and integrated into the OCP. The real-time operating system used for all HIL and flight tests was QNX.

The results shown in this section are based upon the experiment plan as outlined in Sec. VI.A. The HIL environment involved the same interface that would be in the T-33/UCAV cockpit during the flight test. This interface gave the pilot ability to monitor the aircraft's tracking performance, insert the pop-up obstacle, engage the target, and insert the fault. For the HIL testing a Boeing scientist, acting as the pilot, monitored the aircraft's performance and position via this interface and at the appropriate time engaged the fault simulator for fault insertion.

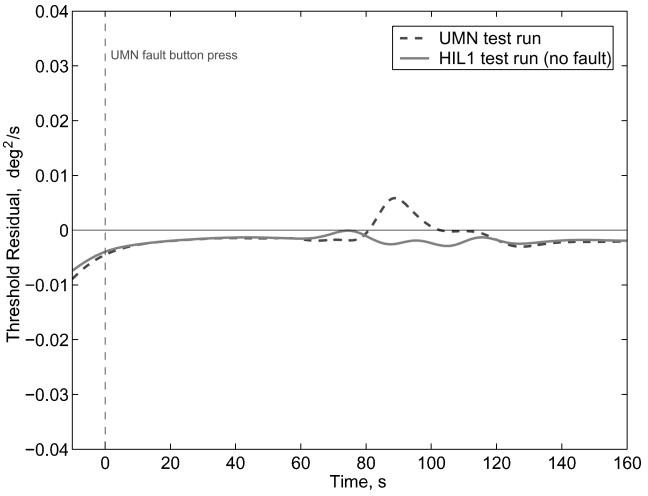

Fig. 17 Comparison of the threshold residual $\check{r}$ for the HIL1 simulation and the offline UMN lab simulation. The HIL1 simulation did not have a fault inserted.

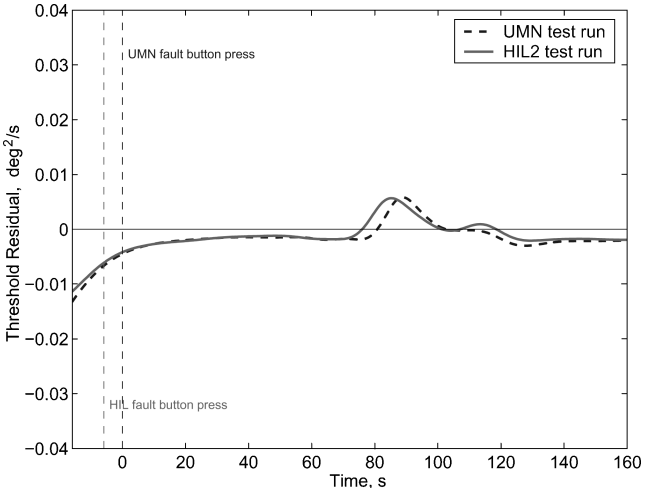

Fig. 18 Comparison of the threshold residual $\check{r}$ for the HIL2 simulation and the offline UMN lab simulation. In this case, the fault was inserted during the HIL2 simulation.

The results of the HIL simulations are shown in Figs. 17 and 18. These figures show the evolution of the threshold function residual $\check{r}$ in the time around fault insertion and detection. Each figure corresponds to one of the two successful simulation runs. The first HIL simulation was one in which the fault was not inserted, that is, the fault on switch was not closed. This simulation is referred to as HIL1, and the results are shown in Fig. 17. The second successful HIL simulation performed is referred to as HIL2, and it only differed from HIL1 in that fault was inserted. The results of this run are shown in Fig. 18. Also shown in these figures for comparison is the threshold residual from the offline UMN laboratory simulation (Fig. 16). The UMN laboratory results are denoted as UMN. The two HIL scenarios provide insight into the filter's model mismatch rejection performance because the only excitation in HIL1 is caused by model mismatch, whereas in HIL2 both model mismatch and the fault excite the FD filter.

One important note about the results shown in Figs. 17 and 18 is the similarity between the HIL and UMN runs. The coherency of the results validate that the FD algorithm was integrated properly into the embedded system and that it performed correctly. Also, because the HIL environment was a high-fidelity, realistic simulation, it also confirmed the validity of the $\mathcal{H}_\infty$ FD filter design and the theoretical foundations of the input-dependent threshold function. Note that the filter performed as expected in both HIL runs: a fault was not detected in HIL1 and was detected in HIL2. This fact confirms that

the FD filter was tuned correctly and also that it did a proper job in rejecting a certain level of model mismatch. This is encouraging because it also validates the model uncertainty characteristic chosen for synthesizing the FD filter.

VIII. Flight-Test Results

The final flight-test demonstration for the DARPA SEC fixed-wing program occurred in June 2004 at NASA Dryden Flight Research Center. The final demonstration for UMN/UCB team involved two flight tests. During the flight tests, the velocity loop was closed by a human pilot following a velocity command from the RHC software. Because the pilot was a part of the actuation, a model of the pilot was necessary during the RHC design. A highly simplified model was used to account for the human behavior of the pilot, and as a consequence the RHC achieved poor velocity tracking during the flight test. The effects were large velocity transients that led to degraded position keeping—observed as overshoot and oscillation around the reference trajectory. These difficulties were further compounded by high winds encountered during both flight-test runs. These combined effects were detrimental to the FD algorithm's performance. The oscillations around the reference trajectory were, in essence, viewed by the FD algorithm as model mismatch. Because the FD algorithm does not completely reject all levels of mismatch, the input-dependent threshold function integrated up a significant amount of residual energy prior to fault insertion. This was more energy than the threshold function's "forgetting factor" was able to eliminate. Because of this, at fault insertion the threshold residual was already positive, and thus a fault was immediately declared. This was a false alarm because it is expected (as seen in the offline and HIL simulations) that the fault should instead be detected at some time after the onset of the S turns.

Although the flight test did not result in a valid fault detection at the proper time, the flight test was not completely unsuccessful. Given the conditions encountered—poor velocity trajectory tracking, high winds, velocity oscillations—the FD algorithm performed just as expected. It was designed knowing there were limitations in its model mismatch rejection performance, caused by the intrinsic lack of frequency separation with the fault. Additionally, the threshold function was designed with a forgetting factor, which adds a time delay in the threshold function's response. Because of this delay and the poor position keeping, the threshold residual was not allowed to reach stability during the flight prior to fault insertion, as assumed in the flight test-planning. In such a situation, we understand that the FD algorithm can issue a false alarm. In addition, the behavior of the threshold residual is an indicator that the aircraft was outside the validity range of the linear model of the aircraft and thus also outside the allowable flight regime for the FD algorithm.

Last, we point the reader back to the HIL simulations and the results thereof, noting that the HIL simulations were in many ways much more representative of a true UAV in comparison with the T-33/UCAV test bed. This is especially true in regard to the velocity control. The HIL environment contained fully automatic velocity control loop, whereas the T-33/UCAV test bed's velocity control required a pilot in the loop. The former is clearly more representative of a real UAV, and therefore the results of the HIL should be of utmost significance when evaluating the overall success of the proposed FD algorithm and threshold function performance.

IX. Conclusions

This work was accomplished by University of Minnesota (UMN) and California-Berkeley (UCB) researchers as part of DARPA Software Enabled Control (SEC) Fixed-Wing Capstone Demonstration program. The fault-detection (FD) design for the SEC program was accomplished by an application of well-known $\mathcal{H}_\infty$ design methods. Also using robust theory, a novel approach to thresholding the FD residual was also presented. This new threshold function is uncertainty conscious and input dependent. This function was combined with the $\mathcal{H}_\infty$ FD filter and was successfully implemented in an offline simulation, a hardware-in-the-loop simulation, and a flight test. The effectiveness of this combined algorithm—also being coupled with the other SEC UMN/UCB technologies and a full-nonlinear aircraft model—was verified by a successful detection of a fault in the offline simulation. The results were later validated by another successful fault detection in the high-fidelity hardware-in-the-loop simulation performed by the Boeing Company. Lastly, the full FD algorithm was successfully implemented on the T-33/UCAV test bed and performed in real time as expected.

In summary, the results of the combined FD algorithm and threshold function presented in this paper show promising results by the successful detection of an aileron actuator fault in numerous testing environments. It is hoped that the application of $\mathcal{H}_\infty$ FD and the results and insights of the input-dependent threshold function will provide a basis for further research and investigation into robust fault detection.

Acknowledgments

This work was funded in part by the Defense Advanced Research Projects Agency under the Software Enabled Control program, with John Bay as program manager. The contract number is USAF/AFMC F33615-99-C-1497; Dale Van Cleave was the technical contract monitor. Support was also provided under NASA Langley Cooperative Agreement No. NCC-1-337, and Christine Belcastro was the technical contract monitor. Last, the authors would like to acknowledge the contribution of the reviewers who helped improve the quality of this paper.

References

[1]Samad, T., and Balas, G. (eds.), *Software-Enabled Control*, Wiley Interscience, Hoboken, NJ, 2003, Chap. 1.

[2]Paunicka, J. L., Mendel, B. R., and Corman, D. E., "Software Architectures for Real-Time Control," *Software-Enabled Control*, edited by T. Samad and G. Balas, Wiley Interscience, 2003, Chap. 4.

[3]Chen, J., and Patton, R. J., *Robust Model-Based Fault Diagnosis for Dynamic Systems*, Kluwer Academic Publishers, Boston, 1999.

[4]Gertler, J. J., *Fault Detection and Diagnosis in Engineering Systems*, Marcel Dekker, New York, 1998.

[5]Massoumnia, M. A., "A Geometric Approach to the Synthesis of Failure Detection Filters," *IEEE Transactions on Automatic Control*, Vol. 31, No. 9, 1986, pp. 839–846.

[6]Emami-Naeini, A., Akhter, M. M., and Rock, S. M., "Effect of Model Uncertainty on Failure Detection: The Threshold Selector," *IEEE Transactions on Automatic Control*, Vol. 33, No. 12, 1988, pp. 1106–1115.

[7]Gertler, J. J., "Survey of Model-Based Failure Detection and Isolation in Complex Plants," *IEEE Control Systems Magazine*, Vol. 8, No. 6, 1988, pp. 3–11.

[8]Chung, W. H., and Speyer, J. L., "A Game Theoretic Fault Detection Filter," *IEEE Transactions on Automatic Control*, Vol. 43, No. 2, 1998, pp. 143–161.

[9]Mangoubi, R., Appelby, B., and Farrell, J., "Robust Estimation in Fault Detection," *Proceedings of IEEE Conference on Decision and Control*, Vol. 2, IEEE Publications, Piscataway, NJ, 1992, pp. 2317–2322.

[10]Niemann, H., and Stoustrup, J., "Filter Design for Failure Detection and Isolation in the Presence of Modeling Errors and Disturbances," *Proceedings of IEEE Conference on Decision and Control*, Vol. 2, IEEE Publications, Piscataway, NJ, 1996, pp. 1155–1160.

[11]Marcos, A., Ganguli, S., and Balas, G., "Application of $\mathcal{H}_\infty$ Fault Detection and Isolation to a Transport Aircraft," *Control Engineering Practice*, Vol. 13, No. 1, 2005, pp. 105–119.

[12]Stoustrup, J., Niemann, H., and la Cour Harbo, A., "Optimal Threshold Functions for Fault Detection and Isolation," *Proceedings of the American Control Conference*, Vol. 2, IEEE Publications, Piscataway, NJ, 2003, pp. 1782–1787.

[13]Shim, D.-S., and Sznaier, M., "A Caratheodory-Fejer Approach to Simultaneous Fault Detection and Isolation," *Proceedings of the American Control Conference*, Vol. 4, IEEE Publications, Piscataway, NJ, 2003, pp. 2979–2984.

[14]Smith, R., Dullerud, G., Rangan, S., and Poolla, K., "Model Validation for Dynamically Uncertain Systems," *Mathematical Modelling of Systems*, Vol. 3, No. 1, 1997, pp. 43–58.

[15]Ljung, L., *System Identification Toolbox User's Guide*, ver. 6, The MathWorks, Natick, MA, 2004.

[16]Keviczky, T., Ingvalson, R., Rotstein, H., Packard, A., Natale, O. R., and Balas, G. J., "An Integrated Multi-Layer Approach to Software Enabled Control: Mission Planning to Vehicle Control," Univ. of Minnesota, Technical Report, Minneapolis, Nov. 2004.