JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 31, No. 4, July–August 2008

Fault-Tolerant Subspace Predictive Control Applied to a Boeing 747 Model

R. Hallouzi[*] and M. Verhaegen[†]
Delft University of Technology, 2628 CD Delft, The Netherlands

DOI: 10.2514/1.33256

This paper presents a fault-tolerant control system based on a combination of predictive control and subspace identification called subspace predictive control. In this control method, the mathematical model used by conventional predictive controllers to predict the future output is replaced by a subspace predictor. Because the subspace predictor is continuously updated in a closed-loop setting based on new input–output data, it can naturally adapt the controller after a fault has occurred. This property is very useful for fault-tolerant control because faults might be unanticipated. A novel feature of the presented subspace predictive control algorithm is that the predictor is recursively updated in a computationally efficient way. A multiple-model fault classification scheme is used to distinguish between anticipated and unanticipated faults. For anticipated faults the control system is configured such that it can accommodate these faults more quickly. The proposed fault-tolerant control system is evaluated on a detailed model of a Boeing 747.

Nomenclature

A, B, C	=	state matrix, input matrix, and measurement matrix of the state-space system, respectively
D_θ, D_ϕ	=	derivative gains for the pitch angle and roll angle reference command, respectively
$\boldsymbol{e}_k$	=	noise vector at time instant k
F_m	=	fault mode
f	=	dimension of future window
h, h_{ref}	=	altitude and reference signal for altitude, respectively
I_m	=	$m \times m$ identity matrix
I_θ, I_ϕ	=	integral gains for the pitch angle and roll angle reference command, respectively
J	=	predictive control objective function
K	=	process noise matrix
N_c	=	control horizon
N_p	=	prediction horizon
P_θ, P_ϕ	=	proportional gains for the pitch angle and roll angle reference command, respectively
p	=	dimension of past window
Q_a	=	tracking error weighting matrix
R	=	result of RQ factorization of matrix containing input–output data
R_a	=	input effort weighting matrix
R_a^Δ	=	incremental input effort weighting matrix
$\boldsymbol{r}_f$	=	future reference vector
$\boldsymbol{r}_k$	=	reference vector at time instant k
$\boldsymbol{u}_f$	=	future input vector
$\boldsymbol{u}_k$	=	input vector at time instant k
$V_{\text{TAS}}, V_{\text{TAS,ref}}$	=	true airspeed and reference signal for true airspeed, respectively
$\boldsymbol{w}_p$	=	vector with past input–output data
$\boldsymbol{x}_k$	=	state vector at time instant k
$\hat{\boldsymbol{y}}_f$	=	predicted future output vector
$\boldsymbol{y}_k$	=	output vector at time instant k
α	=	angle of attack
β	=	sideslip angle
Γ_r	=	predictor matrix that incorporates past input–output data
$\Delta\boldsymbol{u}_f$	=	incremental future input vector
$\theta, \theta_{\text{ref}}$	=	pitch angle and reference signal for pitch angle, respectively
Λ_r	=	predictor matrix that incorporates future inputs
λ	=	forgetting factor
Φ	=	closed-loop transfer matrix of state-space model in innovation form
ϕ, ϕ_{ref}	=	roll angle and reference signal for roll angle, respectively
ψ, ψ_{ref}	=	yaw angle and reference signal for yaw angle, respectively

I. Introduction

SUBSPACE identification is a technique that can be used for identification of state-space models from input–output data. This technique has drawn considerable interest in the last two decades [1,2], especially for linear time-invariant systems. A reason for this is the efficient way models are identified for systems of high order and with multiple inputs and outputs. Subspace identification can be used to form a subspace predictor for prediction of future outputs using past input–output data and a future input sequence. This subspace predictor can be computed without realizing the actual state-space models, which significantly reduces computational requirements. Favoreel and de Moor [3] combined the subspace predictor with model predictive control [4], resulting in a control algorithm that was given the name subspace predictive control (SPC). In SPC, the output predicted by the subspace predictor is part of the cost function of the predictive controller.

SPC has, since its establishment [3], been modified to fit application-specific requirements by different researchers [5,6]. For example, Kadali et al. [6] used a predictive control objective function with incremental inputs instead of the regular inputs. Furthermore, they added constraint handling and a feedforward term to the basic SPC algorithm [3]. However, the implementation of Kadali et al. [6] did not take advantage of the ability of SPC to adapt to changes in the system. If SPC is implemented such that the subspace predictor is updated online, based on new input–output data, it can be considered to be an adaptive controller. Woodley et al. [5] implemented SPC adaptively and also extended it to include $\mathcal{H}_\infty$ cost functions. Furthermore, they demonstrated the adaptive implementation of SPC

Presented as Paper 6665 at the AIAA Guidance, Navigation, and Control Conference and Exhibit, Hilton Head, SC, 20 August–23 August 2007; received 8 July 2007; revision received 27 November 2007; accepted for publication 27 November 2007.

[*]Doctor of Philosophy Candidate, Delft Center for Systems and Control; r.hallouzi@tudelft.nl. Member AIAA.

[†]Professor, Delft Center for Systems and Control.

on a simple single-input, single-output model. An important suggestion made by Woodley et al. [5] is that, in the near future, falling cost of computation would make the algorithm also suitable for more complex situations. In this paper, SPC is applied to a much more complex aircraft model.

Two novel features of the SPC algorithm presented in this paper are the use of a subspace predictor that is derived in a closed-loop setting, and the computationally efficient updating scheme of the predictor. In previous papers in which SPC was used [3,5,6], the subspace predictor was derived using open-loop subspace identification techniques. However, when the SPC algorithm is active, the data gathered to update the predictor are inherently closed-loop data. It has been proven that using closed-loop data for subspace identification results in a biased predictor [7]. Therefore, a number of different methods have appeared in the literature to deal with this issue [7–9]. Most of these methods require explicit knowledge of the controller or are based on (overly) stringent assumptions that limit their applicability. Recently, a practically applicable closed-loop subspace identification method that does not require explicit knowledge of the controller was developed by Chiuso [10]. Based on this method, a subspace predictor under closed-loop conditions can be derived [11], which is also used in this paper. Another novel feature of the SPC algorithm presented in this paper is the way that the predictor is updated in a recursive manner. This updating scheme differs from others that are based on the “receding horizon” principle, such as, for example, the scheme proposed by Woodley et al. [5]. In the receding horizon updating scheme, the predictor is based on input–output data from a fixed time window lagging behind the current time sample. In the recursive updating scheme, new data are appended to the old data, which are discounted with an exponential forgetting factor. This scheme has the advantage that it can be implemented in a computationally efficient manner by using Givens rotations [12].

The implementation of SPC as an adaptive controller makes it very suitable for fault-tolerant control (FTC) of aircraft. Most FTC systems deal with faults by using predesigned or parameter-dependent controllers depending on the type of fault that has occurred [13]. These systems require that the faults be either known in advance or be modeled by a variation of parameters [14,15]. In this way, control designs can be made for each anticipated fault. Besides the fact that this approach can be very involved, unanticipated faults or faults that cannot be modeled by parameter changes, such as severe structural damage, can occur. An advantage of SPC is that it can adapt online to these types of faults. This property is the result of the subspace predictor that is continuously updated using new input–output data. The main contribution of this paper is to display the usefulness of SPC for realistic FTC problems. The developed SPC-based FTC system is applied to a detailed simulation model of a Boeing 747 [16]. This model is used as a benchmark in the Action Group (AG) 16 on FTC of the Group for Aeronautical Research and Technology in Europe (GARTEUR). The objective in the simulations is to fly a predefined flight trajectory even after the occurrence of a number of critical actuator faults. The considered faults are stuck control surfaces, which might be either anticipated or unanticipated.

This paper is organized as follows. First, the architecture of the FTC system is explained in Sec. II. Subsequently, the closed-loop SPC algorithm is described in Sec. III. In Sec. IV, the mechanism that (re-)configures the SPC-based FTC system is explained. The simulation results of this system applied to the Boeing 747 model are given in Sec. V. Finally, concluding remarks are provided in Sec. VI.

II. Architecture of the Fault-Tolerant Control System

The architecture of the SPC-based FTC system consists of two control loops. The task of the outer control loop is to provide reference signals for the manipulated variables to be tracked by the inner loop. The manipulated variables are roll angle ϕ, pitch angle θ, and true airspeed V_{TAS}, each of which is a function of one of three controlled variables. These variables are the altitude h, the yaw angle ψ, and the true airspeed V_{TAS}, respectively. A desired flight trajectory can be generated by choosing appropriate reference signals for the controlled variables. The architecture of the SPC-based FTC system is depicted in Fig. 1. In this figure, it can be seen that, besides the two control loops, a fault classification system is present. Both the control loops and the fault classification system are explained in more detail in the following.

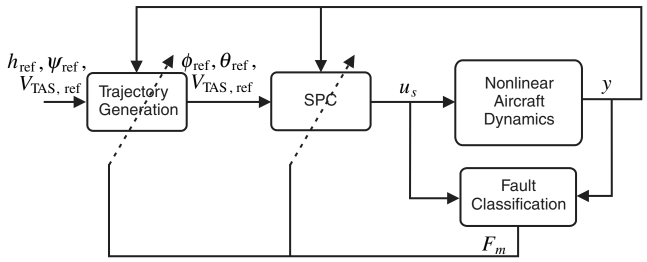
$h_{\mathrm{ref}}, \psi_{\mathrm{ref}}, V_{\mathrm{TAS,\,ref}}$
Trajectory Generation
$\phi_{\mathrm{ref}}, \theta_{\mathrm{ref}}, V_{\mathrm{TAS,\,ref}}$
SPC
u_s
Nonlinear Aircraft Dynamics
y
Fault Classification
F_m

Fig. 1 Architecture of the SPC-based FTC system.

A. Control Loops

The outer loop is implemented by means of a straightforward proportional integral derivative (PID) scheme. To track a desired altitude h_{ref}, a pitch angle command is generated as follows:

$$\theta_{\mathrm{ref}} = P_\theta(h - h_{\mathrm{ref}}) + I_\theta \int (h - h_{\mathrm{ref}})\,\mathrm{d}t + D_\theta \frac{\mathrm{d}(h - h_{\mathrm{ref}})}{\mathrm{d}t} + \alpha \quad (1)$$

where α is the angle of attack and P_θ, I_θ, and D_θ are design parameters that determine the behavior of the outer loop. The desired yaw angle ψ_{ref} is tracked by issuing a roll angle command to the inner loop. This command is generated as follows:

$$\phi_{\mathrm{ref}} = P_\phi(\psi - \psi_{\mathrm{ref}}) + I_\phi \int (\psi - \psi_{\mathrm{ref}})\,\mathrm{d}t + D_\phi \frac{\mathrm{d}(\psi - \psi_{\mathrm{ref}})}{\mathrm{d}t} \quad (2)$$

where P_ϕ, I_ϕ, and D_ϕ are the design parameters. An antiwindup scheme is implemented for both Eqs. (1) and (2) to prevent the integrators from continuing to integrate in case of saturated control signals. The command for true airspeed is generated in the outer loop by directly issuing the true airspeed command to the inner loop. Next, it can be seen in Fig. 1 that the outer loop is dependent on the fault mode F_m. This is done to adapt the commands issued to the inner loop in case of faults. The reason why the commands should be changed is that the inner loop can have degraded performance due to faults. The inner loop is implemented using SPC, which is explained in detail in Sec. III.

B. Multiple-Model Fault Classification

An important requirement for fault detection and isolation (FDI) systems commonly used for FTC is that the faults should be estimated with a certain accuracy, because they are directly used by the FTC system [14,15,17]. If these faults are not estimated accurately enough, poor performance of the FTC system may result. When SPC is used for FTC, in principle, no fault information is required because SPC can adapt itself to the situation at hand. However, this adaptation process can take some time. In case of anticipated faults, the adaptation can be sped up by using prior knowledge of the fault. This prior knowledge includes information on which controls should be used to accommodate the anticipated fault. The requirement for the fault classification scheme used in this paper is therefore to obtain this information by determining which controls cannot be used anymore due to faults. This requirement is more easily achieved than the requirements for FDI systems commonly used for FTC.

Fault classification is implemented by using multiple-model estimation. A multiple-model system consists of a model set that contains local models, each corresponding to a specific condition of the system. In an FDI setting, the local models usually represent different fault conditions of the monitored system [18]. Besides fault models, the model set also contains the nominal fault-free model of the system. When the system is in its fault-free operation mode, the model corresponding to the nominal case has maximum activation, which corresponds to a model weight of one, and all other models in the model set have a model weight of zero (minimum activation). In case of a fault, one or more of the local models corresponding to faults have model weights greater than zero. Anticipated faults result in a characteristic response of the model weights. If the model weights do not correspond to one of these characteristic responses, then the fault is declared as an unanticipated fault.

III. Subspace Predictive Control

The SPC algorithm [3] elegantly combines a subspace predictor with a generalized predictive control law. When the subspace predictor is updated recursively, SPC has the ability to adapt to unanticipated conditions. In this section, it is first explained how the subspace predictor is derived in a closed-loop setting and how it can be updated recursively, then it is explained how the predictor is integrated with a predictive controller.

A. Closed-Loop Subspace Predictor

Contrary to previous papers in which SPC was used [3,5,6], the subspace predictor in this paper is derived using closed-loop identification techniques. In these previous papers, open-loop identification techniques were used under closed-loop conditions. This results in a biased predictor due to correlation between inputs and measurement noise [7]. Favoreel et al. [8] did present an SPC method in which the subspace predictor was based on a closed-loop identification method, but this method is based on explicit controller knowledge and can be shown to be unable to cope with nonlinearities (e.g., due to saturation) of the controller [11]. Therefore, the subspace predictor used in this paper is derived using the closed-loop identification techniques developed by Chiuso [10], which do not require controller knowledge. Dong and Verhaegen [11] give a complete explanation of how these identification techniques can be used to derive a subspace predictor that can be integrated with a predictive control law. In this paper, only the elementary steps are treated.

1. *Derivation of the Subspace Predictor*

The model considered for deriving the subspace predictor is a state-space model in innovation form:

$$\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + K\boldsymbol{e}_k \tag{3}$$

$$\boldsymbol{y}_k = C\boldsymbol{x}_k + \boldsymbol{e}_k \tag{4}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ is the state of the system, $\boldsymbol{u}_k \in \mathbb{R}^m$ is the input of the system, $\boldsymbol{y}_k \in \mathbb{R}^l$ is the output of the system, and $\boldsymbol{e}_k$ is assumed to be a zero-mean white noise sequence. The matrices A, B, C, and K are the state-space matrices describing the system. The model described by Eqs. (3) and (4) can also be written as

$$\boldsymbol{x}_{k+1} = \Phi\boldsymbol{x}_k + B\boldsymbol{u}_k + K\boldsymbol{y}_k \tag{5}$$

where $\Phi = A - KC$. Subspace identification is based on relations between matrices that are systematically filled with input–output data. Two of such data matrices that are required for the derivation of the subspace predictor are created as follows:

$$Y_k = [\boldsymbol{y}_k \;\; \boldsymbol{y}_{k+1} \;\; \cdots \;\; \boldsymbol{y}_{k+j-1}] \tag{6}$$

$$Z_{[k-p,k)} = \begin{bmatrix} \boldsymbol{u}_{k-p} & \boldsymbol{u}_{k-p+1} & \cdots & \boldsymbol{u}_{k-p+j-1} \\ \boldsymbol{y}_{k-p} & \boldsymbol{y}_{k-p+1} & \cdots & \boldsymbol{y}_{k-p+j-1} \\ \boldsymbol{u}_{k-p+1} & \boldsymbol{u}_{k-p+2} & \cdots & \boldsymbol{u}_{k-p+j} \\ \boldsymbol{y}_{k-p+1} & \boldsymbol{y}_{k-p+3} & \cdots & \boldsymbol{y}_{k-p+j} \\ \vdots & \vdots & \cdots & \vdots \\ \boldsymbol{u}_{k-1} & \boldsymbol{u}_k & \cdots & \boldsymbol{u}_{k+j-2} \\ \boldsymbol{y}_{k-1} & \boldsymbol{y}_k & \cdots & \boldsymbol{y}_{k+j-2} \end{bmatrix} \tag{7}$$

where p denotes the "past" time horizon, the subscript $[k-p,k)$ denotes the range of the time indices of the first column of $Z_{[k-p,k)}$, and j denotes the number of columns that are used to create the data matrix $Z_{[k-p,k)}$. Usually, it holds that $j \gg p$. Let f denote the "future" time horizon, then the following matrix relation can be derived [10,11]:

$$\begin{bmatrix} Y_k \\ Y_{k+1} \\ \vdots \\ Y_{k+f-1} \end{bmatrix} = \begin{bmatrix} C\Phi^{s-1}[B \;\; K] & C\Phi^{s-2}[B \;\; K] & \cdots & \cdots & \cdots & C[B \;\; K] \\ 0 & C\Phi^{s-1}[B \;\; K] & \cdots & \cdots & \cdots & C\Phi[B \;\; K] \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & C\Phi^{s-1}[B \;\; K] & \cdots & C\Phi^{f-1}[B \;\; K] \end{bmatrix} Z_{[k-p,k)}$$
$$+ \begin{bmatrix} 0 & 0 & \cdots & 0 \\ C[B \;\; K] & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ C\Phi^{f-2}[B \;\; K] & \cdots & C[B \;\; K] & 0 \end{bmatrix} Z_{[k,k+f)} + \begin{bmatrix} E_k \\ E_{k+1} \\ \vdots \\ E_{k+f-1} \end{bmatrix} \tag{8}$$

where E_{k+i} and Y_{k+i}, $\forall\ i \in \{0, 1, \ldots, f-1\}$ are defined in a similar manner as Y_k in Eq. (6). Note that an important property of Eq. (8) is that there is no correlation between E_{k+i} and $Z_{[k,k+i)}$, $\forall\ i \in \{0, 1, \ldots, f-1\}$. It is this property that allows for an unbiased estimate of the system matrices. To estimate the predictor, it suffices to only consider the first block row, which can be written in the compact form

$$Y_k = \Xi_0 Z_{[k-p,k)} + E_k \tag{9}$$

Subsequently, Ξ_0 can be estimated by solving the least-squares problem

$$\hat{\Xi}_0 = \min_{\Xi_0} \| Y_k - \Xi_0 Z_{[k-p,k)} \|_F^2 \tag{10}$$

This least-squares problem can be solved by performing an RQ factorization [12]

$$\begin{bmatrix} Z_{[k-p,k)} \\ Y_k \end{bmatrix} = \overbrace{\begin{bmatrix} R_{11} & 0 \\ R_{21} & R_{22} \end{bmatrix}}^{R} \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} \tag{11}$$

from which the estimate $\hat{\Xi}_0$ can be computed as

$$\hat{\Xi}_0 = R_{21} R_{11}^{-1} \tag{12}$$

Let t denote the current time instant, then based on the estimate $\hat{\Xi}_0$, a subspace predictor of the following form can be derived:

$$\begin{bmatrix} \hat{\boldsymbol{y}}_{t+1} \\ \hat{\boldsymbol{y}}_{t+2} \\ \vdots \\ \hat{\boldsymbol{y}}_{t+f-1} \end{bmatrix} = \overbrace{\begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \vdots \\ \Gamma_{f-1} \end{bmatrix}}^{\Gamma_r} \overbrace{\begin{bmatrix} \boldsymbol{u}_{t-p} \\ \boldsymbol{y}_{t-p} \\ \vdots \\ \boldsymbol{u}_{t-1} \\ \boldsymbol{y}_{t-1} \end{bmatrix}}^{w_p} + \overbrace{\begin{bmatrix} \Lambda_1 & 0 & \cdots & 0 \\ \Lambda_2 & \Lambda_1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ \Lambda_{f-1} & \Lambda_{f-2} & \cdots & \Lambda_1 \end{bmatrix}}^{\Lambda_r} \begin{bmatrix} \boldsymbol{u}_t \\ \boldsymbol{u}_{t+1} \\ \vdots \\ \boldsymbol{u}_{t+f-2} \end{bmatrix} \tag{13}$$

where Γ_r and Λ_r are the desired subspace predictor matrices and the parameters Γ_i and Λ_i can be constructed from $\hat{\Xi}_0$ as

$$\Gamma_i = \hat{\Xi}_i + \sum_{j=0}^{i-1} \hat{C}\,\hat{\Phi}^{i-j-1} \hat{K} \Gamma_j \tag{14}$$

$$\Lambda_i = \hat{C}\hat{\Phi}^{i-1}\hat{B} + \sum_{j=1}^{i-1} \hat{C}\,\hat{\Phi}^{i-j-1} \hat{K} \Lambda_j \tag{15}$$

with $\Gamma_0 = \hat{\Xi}_0$ and $\Lambda_1 = \hat{C}\,\hat{B}$. The parameters $\hat{\Xi}_i$, $\forall\ i \in \{1, \ldots, f-1\}$ can be constructed from $\hat{\Xi}_0$ by using the relation

$$\begin{bmatrix} \hat{C}\hat{\Phi}^{s-1}[\hat{B}\ \hat{K}] & \hat{C}\hat{\Phi}^{s-2}[\hat{B}\ \hat{K}] & \cdots & \cdots & \cdots & \hat{C}[\hat{B}\ \hat{K}] \\ 0 & \hat{C}\hat{\Phi}^{s-1}[\hat{B}\ \hat{K}] & \cdots & \cdots & \cdots & \hat{C}\,\hat{\Phi}[\hat{B}\ \hat{K}] \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \hat{C}\hat{\Phi}^{s-1}[\hat{B}\ \hat{K}] & \cdots & \hat{C}\hat{\Phi}^{f-1}[\hat{B}\ \hat{K}] \end{bmatrix} = \begin{bmatrix} \hat{\Xi}_0 \\ \hat{\Xi}_1 \\ \vdots \\ \hat{\Xi}_{f-1} \end{bmatrix} \tag{16}$$

where the matrix on the left-hand side of Eq. (16) is an estimate of the corresponding matrix from Eq. (8).

2. *Recursive Implementation of R Update*

For the construction of the data matrices Y_k and $Z_{[k-p,k)}$ explained in the previous section, it was assumed that input–output data were present from time instants: $k-p, k-p+1, \ldots, k+j-1$. For an adaptive implementation of the subspace predictor, the predictor matrices should be recomputed again each time new data become present, i.e., at each sample time. In the case of the receding horizon scheme, this would mean that new data matrices Y_{k+1} and $Z_{[k-p+1,k+1)}$ would be generated using data from time instants: $k-p+1, k-p+2, \ldots, k+j$. Subsequently, a new estimate for the predictor matrices could be obtained by computing the RQ factorization from Eq. (11) based on the new data matrices. However, computing such an RQ factorization at each sample time can become computationally expensive for large data matrices. This computation can be prevented by using Cholesky updating and downdating of the R matrix [5]. The principle of this method is that old data are removed in the downdating step and new data are included in the updating step. These two steps combined require much less computational effort than computing the whole RQ factorization. A drawback of using Cholesky updating and downdating is that matrix RR^T is required to be positive definite at any time. However, this cannot be guaranteed. Therefore, in this paper, a recursive updating scheme of the R matrix, similar to the one developed by Lovera et al. [19] is used. This recursive updating scheme differs from the receding horizon scheme by the fact that it does not use a fixed window of data. Instead, new data are appended to the old R matrix, which is discounted with an exponential forgetting factor. The recursive updating scheme is explained in the following.

Let the upper-left and bottom-left block matrix of R at time instant $t-1$ $[R(t-1)]$ be denoted by $R_{11}(t-1)$ and $R_{21}(t-1)$, respectively. If new data become available at time instant t, a new vector $[\boldsymbol{w}_p^T\ \ \boldsymbol{y}_t^T]^T$ can be created. This vector can be used to update the already available matrix. The updating step consists of first appending $[\boldsymbol{w}_p^T\ \ \boldsymbol{y}_t^T]^T$ to $[R_{11}(t-1)^T\ \ R_{21}(t-1)^T]^T$. Subsequently, by applying a sequence of orthogonal Givens rotations [12], the matrix is made lower triangular, i.e., updated. This sequence of manipulations is described in the following equation:

$$\begin{bmatrix} \sqrt{\lambda} R_{11}(t-1) & \boldsymbol{w}_p \\ \sqrt{\lambda} R_{21}(t-1) & \boldsymbol{y}_t \end{bmatrix} \Omega = \begin{bmatrix} R_{11}(t) & 0 \\ R_{21}(t) & \tilde{\boldsymbol{y}}_t \end{bmatrix} \tag{17}$$

where Ω denotes the sequence of orthogonal transformations and $R_{11}(t)$ (which is lower triangular) and $R_{21}(t)$ are the matrices from which an updated $\hat{\Xi}_0$ can be computed according to Eq. (12). Note that R_{33} is not considered in the updating process because it does not influence the computation of $R_{11}(t)$ and $R_{21}(t)$. Also, in Eq. (17), a forgetting factor $\lambda \in [0, 1]$ is implemented to discount old data. The smaller the value of λ that is chosen, the more old data are discounted.

B. Integration of Subspace Predictor with Predictive Control

The predictive control problem can be formulated as follows. Let t be the current time step. Given a future reference output $\boldsymbol{r}_f = [\boldsymbol{r}_{t+1}\ \boldsymbol{r}_{t+2}\ \ldots\ \boldsymbol{r}_{t+N_p}]$ and a prediction of the outputs $\hat{\boldsymbol{y}}_f = [\hat{\boldsymbol{y}}_{t+1}\ \hat{\boldsymbol{y}}_{t+2}\ \ldots\ \hat{\boldsymbol{y}}_{t+N_p}]$, find an input sequence $\boldsymbol{u}_f = [\boldsymbol{u}_t\ \boldsymbol{u}_{t+1}\ \ldots\ \boldsymbol{u}_{t+N_c-1}]$ such that the following quadratic cost function is minimized:

$$\begin{aligned} J &= \sum_{k=1}^{N_p} (\hat{\boldsymbol{y}}_{t+k} - \boldsymbol{r}_{t+k})^T Q_c (\hat{\boldsymbol{y}}_{t+k} - \boldsymbol{r}_{t+k}) + \sum_{k=0}^{N_c-1} \boldsymbol{u}_{t+k}^T R_c \boldsymbol{u}_{t+k} \\ &= (\hat{\boldsymbol{y}}_f - \boldsymbol{r}_f)^T Q_a (\hat{\boldsymbol{y}}_f - \boldsymbol{r}_f) + \boldsymbol{u}_f^T R_a \boldsymbol{u}_f \end{aligned} \tag{18}$$

where N_p is the prediction horizon, N_c is the control horizon, $Q_c \in \mathbb{R}^{l\times l}$, and $R_c \in \mathbb{R}^{m\times m}$ are the weighting matrices for the tracking error and the input effort, respectively. The matrices $Q_a \in \mathbb{R}^{N_p l\times N_p l}$ and $R_a \in \mathbb{R}^{N_c m\times N_c m}$ are formed from Q_c and R_c as follows:

$$Q_a = \begin{bmatrix} Q_c & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & Q_c \end{bmatrix}, \qquad R_a = \begin{bmatrix} R_c & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & R_c \end{bmatrix} \tag{19}$$

The main idea of SPC is to use a subspace predictor to compute $\hat{\boldsymbol{y}}_f$ rather than to use a mathematical model. The cost function used by Favoreel and de Moor [3] is equal to Eq. (18). However, this cost function does not permit a zero steady-state tracking error in the case of a nonzero constant reference. Kadali et al. [6] therefore proposed to replace the input signal in the cost function by incremental inputs $\Delta \boldsymbol{u}_f$, where $\Delta = (1 - z^{-1})$ and z^{-1} is the backshift operator of one time step. To also penalize large control deflections, a cost function is used with both incremental inputs and the regular input signals

$$J = (\hat{\boldsymbol{y}}_f - \boldsymbol{r}_f)^T Q_a (\hat{\boldsymbol{y}}_f - \boldsymbol{r}_f) + \boldsymbol{u}_f^T R_a \boldsymbol{u}_f + \Delta \boldsymbol{u}_f^T R_a^\Delta \Delta \boldsymbol{u}_f \tag{20}$$

where R_a^Δ has matrices R_c^Δ on its diagonal and is constructed in a similar way as R_a.

A control signal $\boldsymbol{u}_f$ can be obtained by minimizing the cost function defined in Eq. (20) with respect to $\boldsymbol{u}_f$. Before this can be done, $\hat{\boldsymbol{y}}_f$ and $\Delta\boldsymbol{u}_f$ should be written as a function of $\boldsymbol{u}_f$. Using the subspace predictor derived in Eq. (13), $\hat{\boldsymbol{y}}_f$ can be written as

$$\hat{\boldsymbol{y}}_f = \Gamma_r \boldsymbol{w}_p + \Lambda_r \overbrace{\begin{bmatrix} I_m & 0 & \cdots & 0 \\ 0 & I_m & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & I_m \\ 0 & \cdots & 0 & I_m \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & I_m \end{bmatrix}}^{E} \boldsymbol{u}_f \tag{21}$$

where $I_m \in \mathbb{R}^{m\times m}$ denotes an identity matrix and the same notation is used in the remainder of the paper. The matrix E ensures that the input remains constant after the control horizon N_c. Next, $\Delta\boldsymbol{u}_f$ can be written as a function of $\boldsymbol{u}_f$ as follows:

$$\Delta\boldsymbol{u}_f = \overbrace{\begin{bmatrix} I_m & 0 & 0 & \cdots & 0 \\ -I_m & I_m & 0 & & 0 \\ 0 & -I_m & I_m & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & -I_m & I_m \end{bmatrix}}^{S_\Delta} \boldsymbol{u}_f - \overbrace{\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & I_m & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 \end{bmatrix}}^{S_w} \boldsymbol{w}_p \tag{22}$$

When relations (21) and (22) are substituted into Eq. (20) and the terms that do not depend on $\boldsymbol{u}_f$ are discarded, the following cost function results:

$$J(\boldsymbol{u}_f) = \boldsymbol{u}_f^T (E^T \Lambda_r^T Q_a \Lambda_r E + S_\Delta^T R_a^\Delta S_\Delta + R_a) \boldsymbol{u}_f + 2(\boldsymbol{w}_p^T \Gamma_r^T Q_a \Lambda_r E - \boldsymbol{r}_{f+1}^T Q_a \Lambda_r E - \boldsymbol{w}_p^T S_w^T R_a^\Delta S_\Delta)\boldsymbol{u}_f \tag{23}$$

Constraints should be placed on $\boldsymbol{u}_f$, $\Delta\boldsymbol{u}_f$, and $\hat{\boldsymbol{y}}_f$ according to the physical limitations of the aircraft. These constraints can be formulated as follows

$$U_{\min} \le \boldsymbol{u}_f \le U_{\max} \tag{24}$$

$$\Delta U_{\min} \le \Delta\boldsymbol{u}_f \le \Delta U_{\max} \tag{25}$$

$$Y_{\min} \le \hat{\boldsymbol{y}}_f \le Y_{\max} \tag{26}$$

where $U_{\min} = [\boldsymbol{u}_{\min}^T \ \cdots \ \boldsymbol{u}_{\min}^T]^T$, $\Delta U_{\min} = [\Delta\boldsymbol{u}_{\min}^T \ \cdots \ \Delta\boldsymbol{u}_{\min}^T]^T$, $Y_{\min} = [\boldsymbol{y}_{\min}^T \ \cdots \ \boldsymbol{y}_{\min}^T]^T$, and the same notation also holds for the parameters with subscript max. Because the considered optimization variable is $\boldsymbol{u}_f$, relations (21) and (22) are used to convert constraints (25) and (26). This results in the inequality constraint

$$A_{\text{ineq}} \boldsymbol{u}_f \le \boldsymbol{b}_{\text{ineq}} \tag{27}$$

with

$$A_{\text{ineq}} = [I_{N_c m} \ \ -I_{N_c m} \ \ S_\Delta^T \ \ -S_\Delta^T \ \ (\Lambda_r E)^T \ \ -(\Lambda_r E)^T]^T \tag{28}$$

$$\boldsymbol{b}_{\text{ineq}} = [U_{\max}^T \ \ -U_{\min}^T \ \ (\Delta U_{\max} + S_w \boldsymbol{w}_p)^T \ \ (-\Delta U_{\min} - S_w \boldsymbol{w}_p)^T \times (Y_{\max} - \Gamma_r \boldsymbol{w}_p)^T \ \ (-Y_{\min} + \Gamma_r \boldsymbol{w}_p)^T]^T \tag{29}$$

The predictive control law can now be formulated as a solution of the following quadratic programming (QP) problem at each sample time:

$$\min_{\boldsymbol{u}_f} J(\boldsymbol{u}_f) \qquad \text{s.t. } A_{\text{ineq}} \boldsymbol{u}_f \le \boldsymbol{b}_{\text{ineq}} \tag{30}$$

Solving a QP problem with linear constraints is a convex problem for which several efficient methods exist [4]. Note that at each sample time, only the first input vector from $\boldsymbol{u}_f$, i.e., $\boldsymbol{u}_t$, is used for control.

The control law (30) is derived for linear time-invariant systems of the form in Eqs. (3) and (4). However, in this paper, it is applied to a nonlinear aircraft model. This usage is justified because the nonlinear aircraft model can be well approximated by a linear parameter-varying (LPV) model [20], which has the same structure as Eqs. (3) and (4) but with time-varying system matrices. The variation of the time-dependent parameters is relatively small most of the time. In this case, SPC can easily adapt to the time-varying system. During fast variations of the time-dependent parameters or during strong nonlinear behavior of the aircraft, SPC can be less accurate. However, this inaccuracy is only temporary because the aircraft naturally tends to operate in modes that can be dealt with by SPC, even in case of faults.

IV. SPC (Re-)Configuration

SPC is a control method that can adapt itself to the system for which it is used. To fully exploit these capabilities, preferably, all relevant available inputs and outputs should be used to estimate the subspace predictor. Because the Boeing 747 model has 30 control inputs (see the Appendix) and even more outputs, a selection from these inputs and outputs should be made to reduce the computational burden of updating the subspace predictor. The SPC-based FTC system is configured such that it uses different sets of control inputs for different fault conditions. For anticipated faults, a specific set of inputs is chosen, and for unanticipated faults, a more general set is chosen. In this way, the changed dynamics in case of anticipated faults can be captured quicker than purely relying on adaptation of SPC. Both sets of control inputs are chosen such that sufficient control redundancy is available to perform "elementary maneuvers" after the occurrence of a fault. By elementary maneuvers, three basic abilities of the aircraft are meant. These are the ability to descend or ascend, the ability to change heading, and the ability to decelerate or accelerate.

In this paper, the SPC-based FTC system is demonstrated for two fault types, both of which are also used as benchmark faults in GARTEUR AG-16 [16]. These two faults are an anticipated elevator lock-in-place and an unanticipated rudder runaway. Lock-in-place is characterized by the freezing of a control surface at a certain position, regardless of the actuator commands. Runaway of a control surface is characterized as when the surface suddenly deflects to its maximum or minimum deflection position and stays locked at that position. These faults can have drastic consequences because they make further operation of the aircraft extremely difficult. The considered rudder runaway fault affects both the upper and lower rudder. The elevator lock-in-place fault affects all four elevator surfaces. The two faults are classified using the multiple-model method described in Sec. II.B. For this purpose, a model set is designed according to the method described by Hallouzi et al. [21].

In the nominal case, the aforementioned maneuvers can be performed using SPC with an input vector $\boldsymbol{u}_k$ with only four inputs, which are listed in Table 1. Each input can, however, drive more than one of the control surfaces listed in the Appendix. This is because it is assumed that these surfaces are symmetrically actuated (or asymmetrically, in the case of the ailerons and spoilers). The control surfaces that are not directly driven by SPC are chosen constant and

Table 1 **SPC input allocation**

Nominal case	Elevator lock-in-place	Unanticipated faults
Aileron (controls 1–4)	aileron (controls 1–4)	aileron (controls 1–4)
Elevator (controls 17–20)	stabilizer (control 21)	spoiler (controls 5–8/13–16)
Rudder (controls 22–23)	rudder (controls 22–23)	elevator (controls 17–20)
Engine (controls 26–29)	engine (controls 26–29)	stabilizer (control 21)
		rudder (controls 22–23)
		engine left (controls 26–27)
		engine right (controls 28–29)

Table 2 **Outputs used for the SPC**

Output	Symbol	Unit	Noise (standard dev.)
Roll angle	ϕ	deg	$1.73 \cdot 10^{-4}$
Pitch angle	θ	deg	$1.73 \cdot 10^{-4}$
Yaw angle	ψ	deg	$1.73 \cdot 10^{-4}$
True airspeed	V_{TAS}	m/s	$1.00 \cdot 10^{-1}$
Angle of attack	α	deg	$1.73 \cdot 10^{-3}$
Sideslip angle	β	deg	$1.73 \cdot 10^{-3}$
Altitude	h	m	$1.00 \cdot 10^{-1}$

equal to a value that is valid for a trimmed situation at the beginning of the flight simulation. For an elevator lock-in-place fault, the SPC-based FTC system uses the stabilizer instead of the elevator surfaces for control of the longitudinal motion. For unanticipated faults, the engine controls are subdivided into a control input that controls the left engines and one that controls the right engines such that differential engine thrust can be used when necessary. Furthermore, spoilers are added for increased control authority. Note that for anticipated conditions, the input set can be chosen smaller. This has the additional benefit that SPC can be implemented more computationally efficient.

Besides the input vector $\boldsymbol{u}_k$, the SPC-based FTC system also requires a number of measurements from the aircraft to be used in the output vector $\boldsymbol{y}_k$. A selection is made from the many available measurements taking into consideration three issues. The first issue is the size of the output vector $\boldsymbol{y}_k$, which should be chosen as small as possible to reduce computational requirements. The second issue is concerned with the quality of the subspace predictor. For this purpose, the chosen outputs should capture the relevant dynamics of the system. Finally, the third issue is concerned with the manipulated variables. The control objective of the SPC-based FTC system is for the reference trajectory $\boldsymbol{r}_f$ to be tracked by the predicted output vector $\hat{\boldsymbol{y}}_f$ [see Eq. (20)]. Therefore, the output vector $\boldsymbol{y}_k$ should include the measurements of the physical quantities to be manipulated. With the previous considerations in mind, seven outputs are chosen, which are listed in Table 2 together with their noise levels. These realistic noise levels correspond to those of conventional aircraft sensors [22].

The SPC-based FTC system should be initialized such that it does not start identifying the system from scratch when a switch is made from nominal operation to an operation mode corresponding to a fault or when the simulation starts from $T = 0$ s. Therefore, matrix R is initialized using input–output data obtained from simulation of the open-loop aircraft. In the case of anticipated faults, open-loop data of the model with the anticipated fault are used to initialize the R matrix. And, in the case of unanticipated faults, open-loop data of the *nominal* model are used to initialize the R matrix.

V. Simulation Results

The Boeing 747 model used for evaluation of the proposed FTC method was originally developed for aircraft simulation and analysis on a MATLAB/Simulink platform by van der Linden [23], after which it was adapted by Smaili [24]. Next, it was modified by Marcos and Balas [25] such that the model could be used as a benchmark for FTC and FDI [26,27]. The final modifications of this model are performed within GARTEUR AG-16 to include a flight scenario and a number of specific faults to be considered by its participants [16]. In this paper, two faults defined from the final model are considered. These faults are elevator lock-in-place and rudder runaway. The flight scenario consists of an initial straight and level flight at an altitude of 980 m and a true airspeed of 92.6 m/s. During this first flight phase, faults can be inserted and must also be classified. Next, a second phase consisting of a heading change is initiated. The third and final flight phase of the trajectory consists of a descent to an altitude of 100 m. In this section, results from three simulations are presented, all of which are based on the described flight scenario. In the first simulation, the flight scenario is simulated without any faults. In the second and third simulation, a fault is injected during the first flight phase. In the second simulation, an anticipated lock-in-place fault of the elevators is injected, and in the third simulation, an unanticipated rudder runaway fault is injected.

Before the actual simulation results are presented, the choices for the simulation settings and tuning parameters are described first. The aircraft model is simulated with a frequency of 100 Hz. The operation frequency of the SPC-based FTC system is 10 Hz, which is fast enough relative to the aircraft dynamics. The SPC parameters are chosen as $p = 20$, $f = 20$, $\lambda = 0.995$, $N_p = f$, and $N_c = 5$. The subspace predictor parameters p and f are chosen relative to the aircraft dynamics. The parameter λ is tuned such that the predictor is modified just enough at each sample time to cope with the varying dynamics. The weights Q_a, R_a, and R_a^Δ are tuned relative to each other based on a combination of simulation experience and "rules of thumb" [4]. These weights are tuned differently for the different fault settings described in Table 1. Furthermore, weight Q_a only contains nonzero entries on its diagonal for the entries that are manipulated by SPC, i.e., ϕ, θ, V_{TAS}, and β. The tuning procedure for the outer loop parameters P_θ, I_θ, D_θ, P_ϕ, I_ϕ, and D_ϕ is based on simulation experience, similar to the weighting matrices. Parameter j, which determines the number of columns in the data matrices in Eqs. (6) and (7), is chosen to have a value of 1000. This means that the data matrices contain $1000/10$ Hz $= 100$ s of data. Note that these large data matrices are created only once for each condition. Once an R matrix is computed based on these two data matrices, only the R matrix is used and updated in SPC. The R matrix is generally much smaller than the data matrices because its dimensions do not depend on j. All simulations have been performed under closed-loop conditions with measurement noise levels as described in Table 2. Moreover, turbulence that is modeled according to the Dryden turbulence model is added to the simulated aircraft.

A. Trajectory Following for the Nominal Case

In this section, the simulation results for the nominal condition are presented. The flight trajectory starts with a straight and level flight. During this flight phase, the control objective is to maintain a constant altitude, yaw angle, velocity, and sideslip angle. Next, at $T = 100$ s a change in yaw angle from 180 to 300 deg is initiated. Finally, at $T = 200$ s, a descent is initiated to an altitude of 100 m. In Fig. 2, the references for the manipulated variables are represented by dashed lines. It can be seen that the reference signals are generally tracked very well, especially when the fact is considered that no model information is used. During the descent, the true airspeed is not tracked very well, because the aircraft gains speed when descending. Also, the sideslip angle has a minimal tracking error during the heading change maneuver. The flight trajectory is depicted in Fig. 3 as well as the angle of attack, yaw angle, and altitude. The actuator deflections and the engine commands are

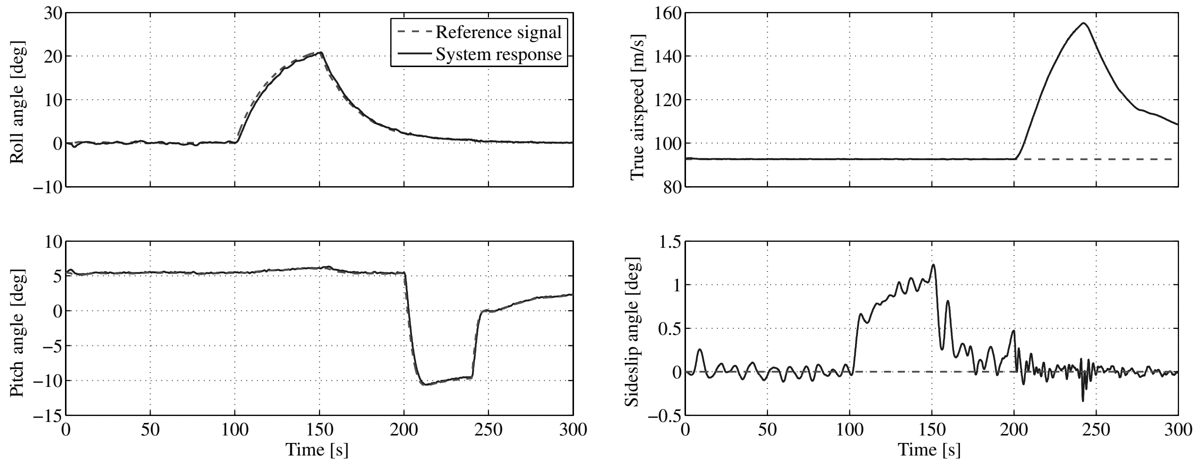

Fig. 2 Roll angle, pitch angle, true airspeed, and sideslip angle of the aircraft for the nominal condition. The dashed signals correspond to the control reference signals.

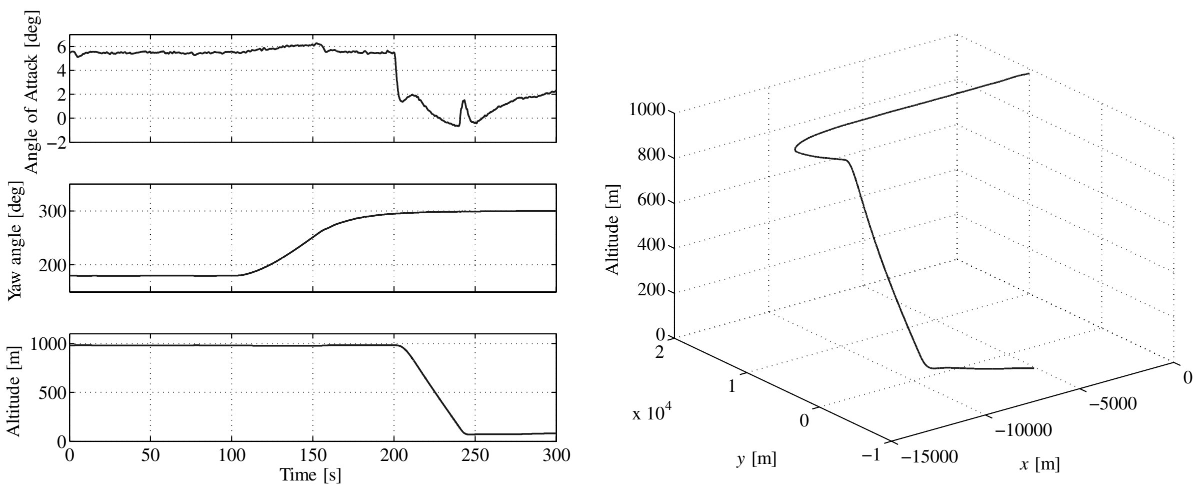

Fig. 3 Angle of attack, yaw angle, altitude, and trajectory of the aircraft for the nominal condition.

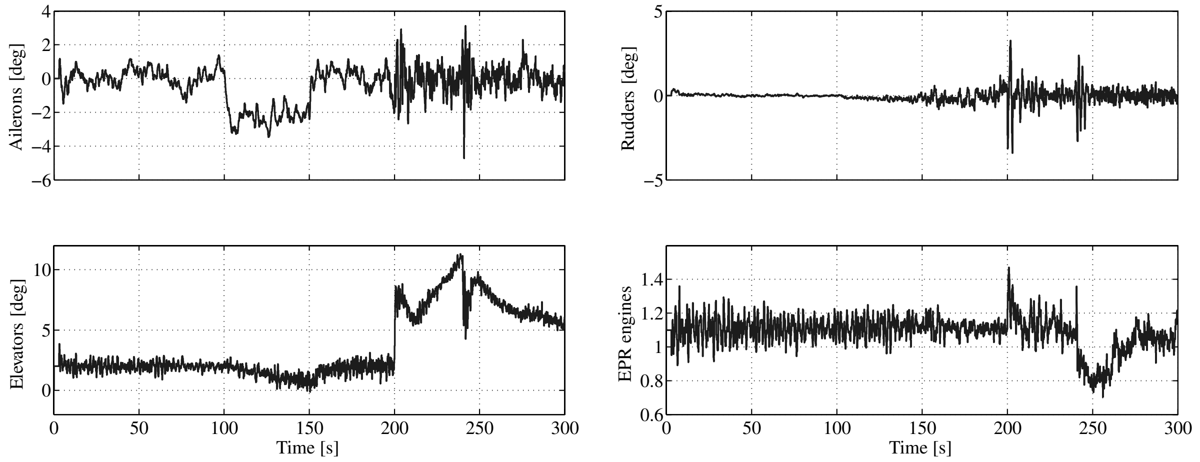

Fig. 4 Actuator deflections and engine commands for the nominal condition.

depicted in Fig. 4. The engine commands are expressed in engine pressure ratio (EPR). It can be seen that the control signals are quite smooth, which is a result of the constraints on $\boldsymbol{u}_f$.

B. Trajectory Following for Elevator Lock-In-Place

In this section, the simulation results for elevator lock-in-place are presented. The elevator lock-in-place fault is injected at $T = 18$ s at a deflection of 2.33 deg. The fault is correctly classified at $T = 28$ s. It can be seen in Fig. 5 that the reference signal for the true airspeed has been increased just after classification of the fault. This has been done to increase the effectiveness of the stabilizer surface to allow sufficient control authority. Furthermore, it can be seen that tracking of the reference signals is performed satisfactorily. In Fig. 6, the angle of attack, yaw angle, and altitude are depicted together with the

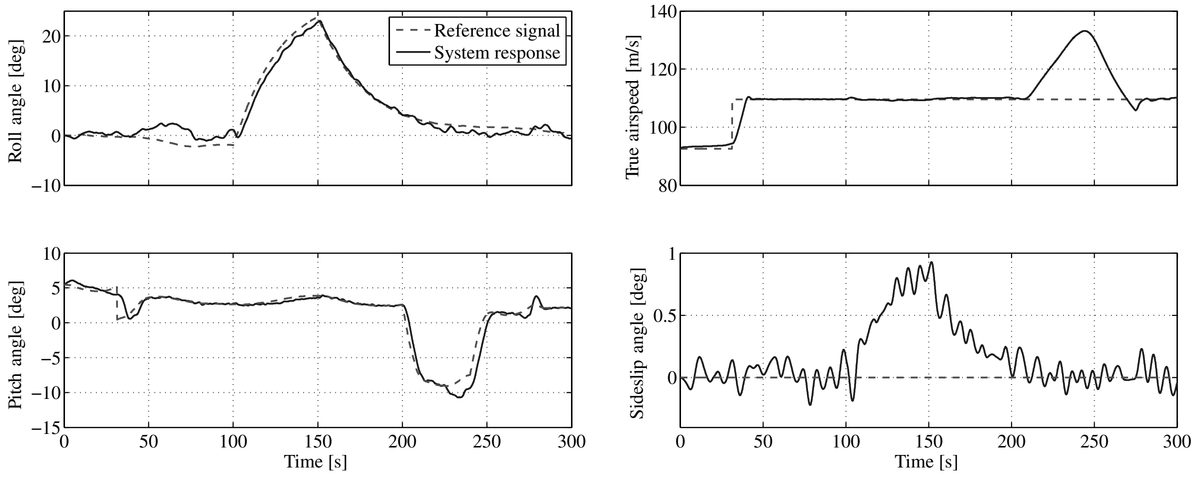

Fig. 5 Roll angle, pitch angle, true airspeed, and sideslip angle of the aircraft for elevator lock-in-place. The dashed signals correspond to the control reference signals.

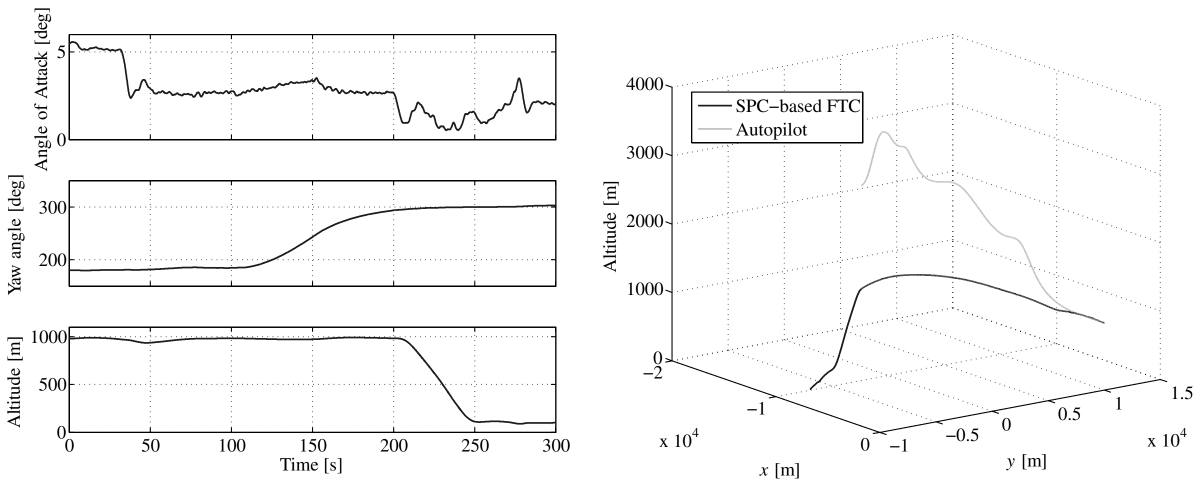

Fig. 6 Angle of attack, yaw angle, altitude, and trajectory of the aircraft for elevator lock-in-place. In the trajectory plot, the light gray line corresponds to the trajectory flown by the autopilot.

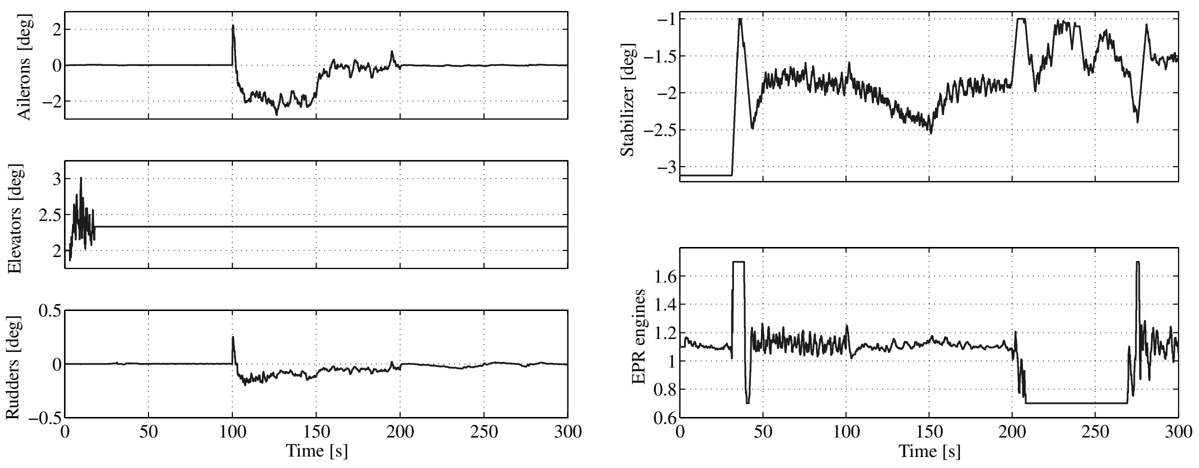

Fig. 7 Actuator deflections and engine commands for elevator lock-in-place.

flight trajectory. For comparison purposes, the same trajectory is also flown using the autopilot from the GARTEUR AG-16 benchmark, the result of which is indicated by a light gray signal in the figure. It can be seen that the result of the fault is an upward pitching moment which can not counteracted by the autopilot because it does not have control over the stabilizer. Therefore, when the autopilot is used, human pilot intervention is required to accommodate this fault. Because the elevator lock-in-place fault does not affect lateral motion, the heading change maneuver is still performed adequately by the autopilot. In Fig. 7, the actuator deflections and engine commands are shown. It can be seen that the elevator deflection remains constant after the fault is injected and that the stabilizer takes

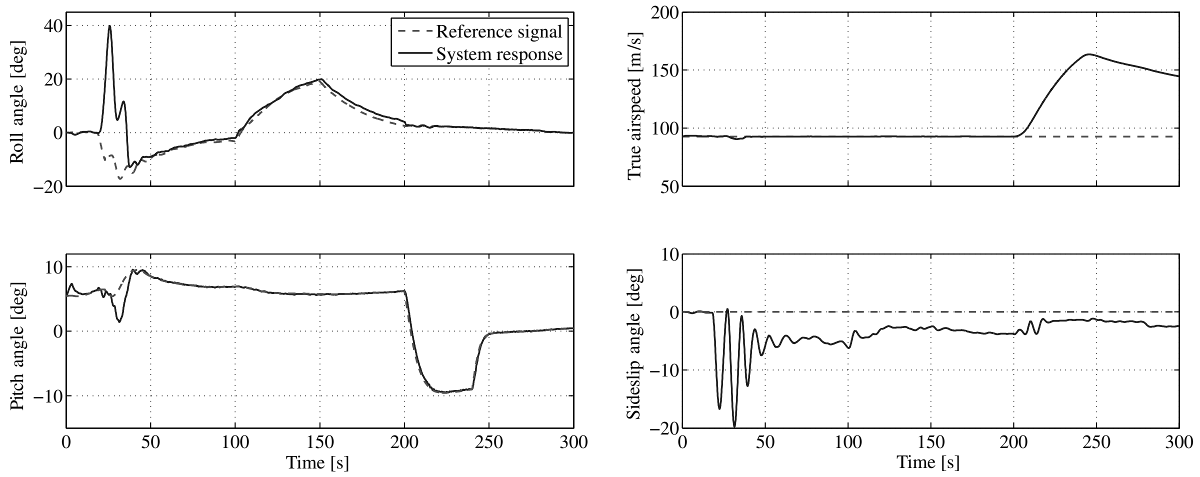

Fig. 8 Roll angle, pitch angle, true airspeed, and sideslip angle of the aircraft for rudder runaway. The dashed signals correspond to the control reference signals.

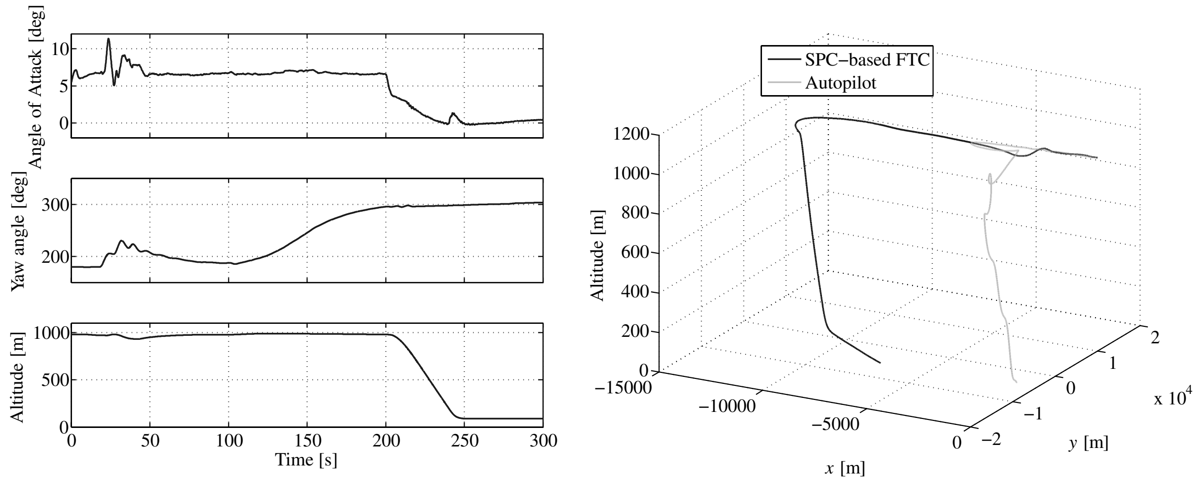

Fig. 9 Angle of attack, yaw angle, altitude, and trajectory of the aircraft for rudder runaway. In the trajectory plot, the light gray line corresponds to the trajectory flown by the autopilot.

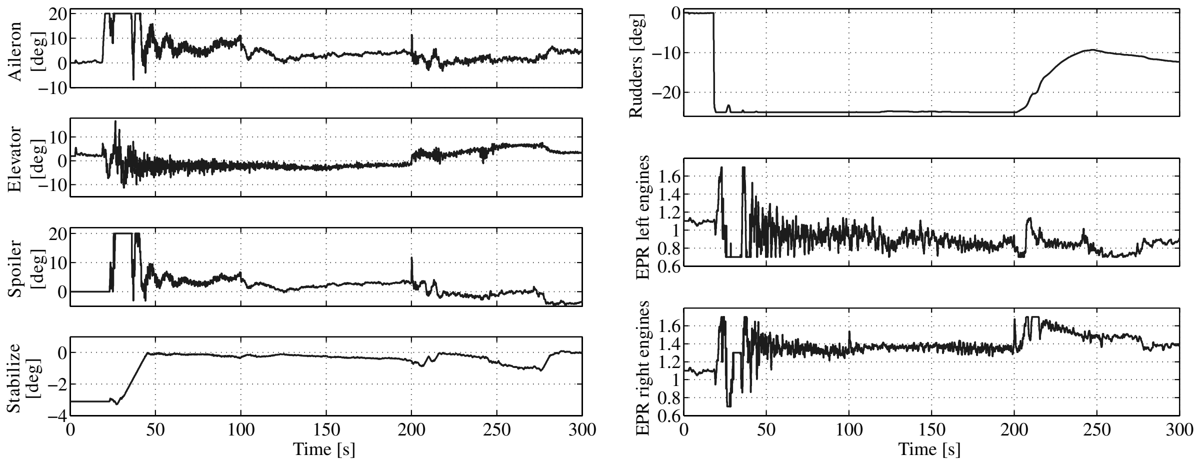

Fig. 10 Actuator deflections and engine commands for rudder runaway.

over after the fault is classified. Note also that the rate of change of the stabilizer input is small when compared with the other surfaces. The reason for this is that the stabilizer surface has a maximum deflection rate of 0.5 deg/s, which is about 100 times smaller than the other surfaces. Also, it can be observed that the stabilizer is used in a limited range to prevent overly large altitude fluctuations due to the slow operation of this surface. Generally, it can be concluded from these simulation results that the reaction on the fault is performed quickly and adequately as a result of the available prior knowledge that has significantly reduced adaptation time.

C. Trajectory Following for Rudder Runaway

In this section, the simulation results for unanticipated rudder runaway are presented. The rudder runaway fault is injected at $T = 18$ s. After this, the upper and lower rudder surfaces start moving with a rate of 50 deg/s from their position at $T = 18$ s to the minimum deflection position of -25 deg. The rudder runaway fault is declared to be an unanticipated fault at $T = 23.2$ s. It can be seen in Fig. 8 that the aircraft starts to roll immediately after insertion of the fault and that the reference signals are not tracked very well just after the fault. This is because SPC needs some time to gather data for adapting to the faulty condition. After this has been done, the reference signals are tracked satisfactorily again. Also note that the aircraft has a sideslip angle, which cannot be completely controlled toward zero due to the fault. At $T = 100$ s, the heading change is initiated. Subsequently, at $T = 200$ s, a descent to 100 m is initiated. In Fig. 9, it can be seen that both the heading change and the descent maneuver are performed adequately. Furthermore, it can be observed that the autopilot is unable to counteract the yawing moment resulting from the rudder runaway fault, not even with a full deflection of the spoilers and ailerons. It is therefore clear that the human pilot must intervene to try to accommodate the fault. In Fig. 10, it can be seen that, after the fault, some time is required before the control signals become smooth again, which is a result of the adaptation process. Also, it can be seen how the ailerons work together with the engines (providing differential thrust) and the spoilers to counteract the yawing moment resulting from the rudder runaway fault. In the time interval $T = 200$–300 s, the rudders have moved away from their minimum deflection position of -25 deg because the aircraft picks up speed resulting in a reduced blowdown limit, which means that the rudders are forced back toward their neutral position.

D. Concluding Remarks on Simulation Results

The presented simulation results show that by using the proposed methodology, it is possible to design a controller for the nominal and faulty aircraft using only input–output data. This conclusion is remarkable, especially when the complexity of the aircraft model is considered. Although the performance of the designed controllers might not be on the same level as that of advanced model-based controllers, the proposed control design methodology has two important advantages:

1) Modeling of the system to be controlled takes up a large part of the design process of model-based controllers. Because the proposed methodology provides a framework to derive a controller using only input–output data, a significant amount of time can be saved in the design process.

2) For fault-tolerant control, it is often required to have a model of the postfault system. This requirement results in the impossibility of providing fault-tolerant control for all possible faults because not all possible faults can be anticipated. However, the proposed methodology can even deal with unanticipated faults by adapting online to faults using input–output data. Therefore, it is a very suitable method for fault-tolerant control.

VI. Conclusions

A reconfigurable fault-tolerant control system is presented that is able to adapt online to faults. This system consists of a subspace predictor, derived in a closed-loop setting, combined with predictive control. The subspace predictor, which does not require knowledge of a mathematical model, is continuously updated online using new input–output data. It is this property that gives the proposed system its ability to adapt to faults. These faults may be either anticipated or unanticipated, which is determined by a multiple-model fault classification system. In case of anticipated faults, prior knowledge of the faults allows the changed dynamics to be captured faster than purely relying on adaptation. A special setting for unanticipated faults has been designed that uses more control inputs than for anticipated faults to fully exploit the adaptation capabilities. The proposed fault-tolerant control system is evaluated in simulation on a detailed model of a Boeing 747. In the simulation of an unanticipated fault, it can be seen that the controller requires some time to adapt to the new fault situation. This is an inevitable consequence of the data-driven adaptation concept. However, in general, it can be concluded from the simulations that the system performs satisfactorily in both nominal and faulty conditions.

Appendix: Control Inputs for the Boeing 747

The 30 different controls that are available on the Boeing 747 are listed in Table A1. The locations of these controls on the aircraft are depicted in Fig. 11.

Table A1 Controls of the Boeing 747

Control no.	Description
1.	right inner aileron
2.	left inner aileron
3.	right outer aileron
4.	left outer aileron
5.	spoiler panel # 1
6.	spoiler panel # 2
7.	spoiler panel # 3
8.	spoiler panel # 4
9.	spoiler panel # 5
10.	spoiler panel # 6
11.	spoiler panel # 7
12.	spoiler panel # 8
13.	spoiler panel # 9
14.	spoiler panel # 10
15.	spoiler panel # 11
16.	spoiler panel # 12
17.	right inner elevator
18.	left inner elevator
19.	right outer elevator
20.	left outer elevator
21.	stabilizer angle
22.	upper rudder surface
23.	lower rudder surface
24.	outer flaps
25.	inner flaps
26.	thrust engine # 1
27.	thrust engine # 2
28.	thrust engine # 3
29.	thrust engine # 4
30.	landing gear

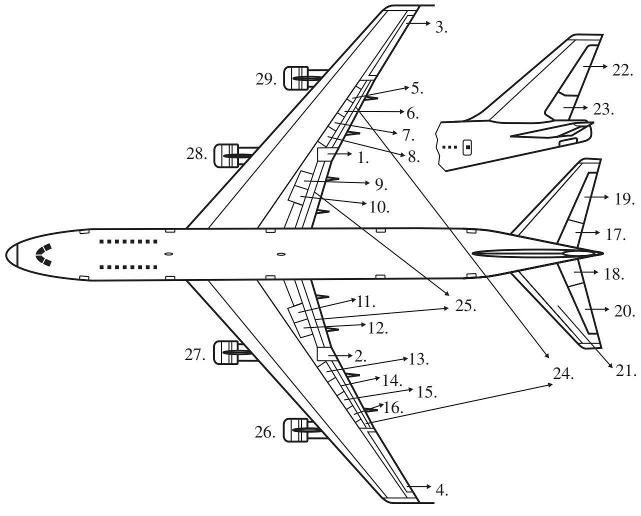

Fig. 11 Locations of controls on the Boeing 747.

Acknowledgments

This research was supported by the Dutch Technology Foundation under project number 04506. We would like to thank M. H. Smaili and J. Breeman from National Aerospace Laboratory (The Netherlands) for managing the Boeing 747 benchmark. Furthermore, we would like to thank the other participants of GARTEUR AG-16 for valuable discussions during project meetings. Finally, we would like to thank J. Rice from the Delft Center for Systems and Control.

References

[1] van Overschee, P., and de Moor, B., *Subspace Identification for Linear Systems: Theory, Implementation, Applications*, Kluwer Academic, Dordrecht, The Netherlands, 1996, pp. 1–30.

[2] Verhaegen, M., and Dewilde, P., "Subspace Identification, Part 1: The Output-Error State Space Model Identification Class of Algorithms," *International Journal of Control*, Vol. 56, No. 5, 1992, pp. 1187–1210. doi:10.1080/00207179208934363

[3] Favoreel, W., and de Moor, B., "SPC: Subspace Predictive Control," *Proceedings of the 14th IFAC World Congress* [CD-ROM], Vol. H, International Federation of Automatic Control, July 1999, pp. 235–240.

[4] Maciejowski, J. M., *Predictive Control with Constraints*, Prentice–Hall, Upper Saddle River, NJ, 2002, pp. 36–107.

[5] Woodley, B. R., How, J. P., and Kosut, R. L., "Subspace Based Direct Adaptive $\mathcal{H}_\infty$ Control," *International Journal of Adaptive Control and Signal Processing*, Vol. 15, No. 5, 2001, pp. 535–561. doi:10.1002/acs.688

[6] Kadali, R., Huang, B., and Rossiter, A., "Data Driven Subspace Approach to Predictive Controller Design," *Control Engineering Practice*, Vol. 11, No. 3, 2003, pp. 261–278. doi:10.1016/S0967-0661(02)00112-0

[7] Ljung, L., and McKelvey, T., "Subspace Identification from Closed Loop Data," *Signal Processing*, Vol. 52, No. 2, 1996, pp. 209–215. doi:10.1016/0165-1684(96)00054-0

[8] Favoreel, W., de Moor, B., Gevers, M., and van Overschee, P., "Closed-Loop Model-Free Subspace-Based LQG-Design," *Proceedings of the 7th Mediterranean Conference on Control and Automation*, Inst. of Electrical and Electronics Engineers [CD-ROM], June 1999, pp. 1926–1939.

[9] Jansson, M., "New Subspace Identification Method for Open and Closed Loop Data," *16th IFAC World Congress* [CD-ROM], International Federation of Automatic Control Paper Tu-E02-TO/2, July 2005.

[10] Chiuso, A., "Role of Vector Autoregressive Modeling in Predictor-Based Subspace Identification," *Automatica*, Vol. 43, No. 6, 2007, pp. 1034–1048. doi:10.1016/j.automatica.2006.12.009

[11] Dong, J., and Verhaegen, M., "Closed-Loop Subspace Predictive Control for Fault Tolerant MPC Design," *17th IFAC World Congress* (accepted for presentation), 2008.

[12] Golub, G. H., and Loan, C. F. V., *Matrix Computations*, 3rd ed., John Hopkins Univ., Baltimore, MD, 1996, pp. 206–274.

[13] Hajiyev, C., and Caliskan, F., *Fault Diagnosis and Reconfiguration in Flight Control Systems*, Kluwer Academic, Dordrecht, The Netherlands, 2003, pp. 283–326.

[14] Shin, J.-Y., and Belcastro, C. M., "Performance Analysis on Fault Tolerant Control System," *IEEE Transactions on Control Systems Technology*, Vol. 14, No. 5, Sept. 2006, pp. 920–925. doi:10.1109/TCST.2006.876911

[15] Song, Y., Campa, G., Napolitano, M., Seanor, B., and Perhinschi, M. G., "Online Parameter Estimation Techniques Comparison within a Fault Tolerant Flight Control System," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 3, 2002, pp. 528–537.

[16] Smaili, M. H., Breeman, J., Lombaerts, T. J. J., and Joosten, D. A., "Simulation Benchmark for Integrated Fault Tolerant Flight Control Evaluation," *Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit*, AIAA Paper 2006-6471, Aug. 2006.

[17] Pachter, M., and Huang, Y.-S., "Fault Tolerant Flight Control," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 1, 2003, pp. 151–160.

[18] Zhang, Y., and Li, X. R., "Detection and Diagnosis of Sensor and Actuator Failures Using IMM Estimator," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No. 4, Oct. 1998, pp. 1293–1313. doi:10.1109/7.722715

[19] Lovera, M., Gustafsson, T., and Verhaegen, M., "Recursive Subspace Identification of Linear and Non-Linear Wiener State-Space Models," *Automatica*, Vol. 36, No. 11, 2000, pp. 1639–1650. doi:10.1016/S0005-1098(00)00103-5

[20] Marcos, A., and Balas, G. J., "Development of Linear-Parameter-Varying Models for Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 27, No. 2, March 2004, pp. 218–228.

[21] Hallouzi, R., Verhaegen, M., and Kanev, S., "Model Weight Estimation for FDI Using Convex Fault Models," *6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes* [CD-ROM], International Federation of Automatic Control ThB4.1, Aug. 2006.

[22] Breeman, J., "Quick Start Guide to AG 16 Benchmark Model," National Aerospace Lab. (The Netherlands) Technical Rept., 2006.

[23] van der Linden, C. A. A. M., *DASMAT: Delft University Aircraft Simulation Model and Analysis Tool*, Delft Univ. Press, Delft, The Netherlands, 1998, pp. 4–50.

[24] Smaili, M. H., "FLIGHTLAB 747 Benchmark for Advanced Flight Control Engineering v4.03." Delft Univ. of Technology, Technical Rept., Delft, The Netherlands, 1999.

[25] Marcos, A., and Balas, G. J., "Boeing 747-100/200 Aircraft Fault Tolerant and Fault Diagnostic Bench-Mark," Univ. of Minnesota, TR AEM-UoM-2003-1, Minneapolis, MN, 2003.

[26] Marcos, A., Ganguli, S., and Balas, G. J., "Application of $\mathcal{H}_\infty$ Fault Detection and Isolation to a Transport Aircraft," *Control Engineering Practice*, Vol. 13, No. 1, 2005, pp.105–119. doi:10.1016/j.conengprac.2004.02.006

[27] Szászi, I., Marcos, A., Balas, G. J., and Bokor, J., "Linear Parameter-Varying Detection Filter Design for a Boeing 747-100/200 Aircraft," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 3, 2005, pp. 461–470.