Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 48-23 (2015) 166–171

Predictive Flight Control with Active Diagnosis and Reconfiguration for Actuator Jamming [⋆]

Laura Ferranti [*] Yiming Wan [*] Tamás Keviczky [*]

[*] *Delft Center for Systems and Control, Delft University of Technology, Delft, 2628 CD, The Netherlands, (e-mails:{l.ferranti,y.wan,t.keviczky}@tudelft.nl).*

Abstract: Actuator jamming in flight control applications may be attributed to a temporary stall load due to large aerodynamic forces, or even a permanently stuck faulty control surface. These two root causes of actuator jamming have different consequences on available control authority and fault duration. As an important consequence, any reconfiguration strategy that may be applied to handle these types of jamming faults needs to distinguish between them in order to take appropriate measures. However, the similarity of fault phenomena between the two root causes makes this problem challenging (i.e., not possible using only passive fault diagnosis). We propose to integrate reconfigurable model predictive control (MPC) with active fault diagnosis (FD) to address this problem. After detecting actuator jamming, a sequence of reconfiguration strategies (relying on actuator constraint updates and a soft constraint formulation) is adopted to aid in the FD decision process and take appropriate measures after obtaining the FD results. The proposed reconfiguration strategies allow the FD module to distinguish the two root causes of the jamming and also detect the end of stall load. The MPC then implements a suitable controller reconfiguration depending on the FD results. A nonlinear Airbus civil aircraft simulator is used to illustrate the effectiveness of our proposed approach.

Keywords: Fault-tolerant control, predictive control, model-based control, active diagnosis, reconfigurable control, flight control, aerospace.

1. INTRODUCTION

Actuator jamming has long been investigated in the field of fault-tolerant flight control (Edwards et al., 2010). For such actuator faults, model predictive control (MPC) provides a well-recognized technique for fault tolerance (De Almeida and Leißling, 2010; Kale and Chipperfield, 2005; Maciejowski and Jones, 2003; Lew, 2013). On one hand, MPC without reconfiguration has some inherent *self-reconfiguration* properties to reallocate control effort in the presence of actuator faults (Maciejowski, 1998). On the other hand, reconfigurable MPC further improves fault tolerance capacity by exploiting extra fault information in a flexible and systematic manner, especially when it comes to dealing with constraints (Maciejowski, 1998). Reconfigurable MPC has to be integrated with a FD module to obtain fault information. Robustness and guaranteed fault tolerance of this integrated fault-tolerant MPC (FTMPC) scheme was analyzed with set theoretic methods in Stoican and Olaru (2013); Yetendje et al. (2013).

Actuator jamming in most literature is attributed to a permanent stuck fault, during which the actuator is locked at a certain position. The study of temporary jamming due to heavy aerodynamic forces, however, is much less common. This temporary malfunction, known as stall load, leads to more stringent control limits, but can be recovered once the aerodynamic forces become smaller (Goupil et al., 2014). Although both stuck fault and stall load lead to a jammed actuator, their consequences on the control limits and fault duration are different. Therefore, we need to distinguish the two root causes of actuator jamming and determine the end of stall load, in order to apply suitable reconfiguration strategies. Conventional FD cannot achieve this goal because the fault phenomena of a stuck actuator and stall load have high similarity.

We propose to integrate reconfigurable MPC with active FD to address the challenge above. Instead of passively monitoring actuator behaviors, we exploit a sequence of reconfiguration strategies to assist the FD module, not only to distinguish the two root causes of the jamming, but also to detect the end of stall load. Then, the MPC adopts suitable successive reconfigurations, leading to improved control performance. All these improvements from both FD and control perspective cannot be achieved without using active reconfiguration to assist FD.

The use of active FD in the context of FTMPC has been rather limited so far and considered only permanent faults (Puncochar et al., 2015; Raimondo et al., 2013; Xu et al., 2014). In contrast, our contribution lies in discriminating between a permanent stuck fault and temporary stall load that share highly similar fault symptoms, and illustrating

[⋆] This research is supported by the European Union's Seventh Framework Programme FP7/2007-2013 under grant agreement n° AAT-2012-RTD-2314544 entitled "Reconfiguration of Control in Flight for Integral Global Upset Recovery (RECONFIGURE)".

the effectiveness of our approach using a nonlinear Airbus civil aircraft simulator.

In the following, Section 2 presents the Airbus simulator used to evaluate our design. Section 3 describes our fault-tolerant control architecture. Section 4 introduces the proposed detection and diagnosis strategy and highlights the interactions between the FD module and the MPC. Section 5 compares the behavior of the MPC with and without the reconfiguration when multiple faults occur on the elevators. Finally, Section 6 concludes this paper.

2. BENCHMARK MODEL AND SCENARIO DEFINITION

This section describes our benchmark model, a high-fidelity nonlinear Airbus civil aircraft simulator, and the fault scenarios we focus on. For sake of brevity, in the following, we describe the linearized model of the aircraft used to build the MPC prediction model.

2.1 The aircraft longitudinal model

This work focuses on the control of the longitudinal motion of the aircraft. Specifically, the linearized and discretized longitudinal dynamics are described as follows:

$$x(t+1) = Ax(t) + B_{\mathrm{u}}u(t) \tag{1a}$$

$$y(t) = Cx(t) + D_{\mathrm{u}}u(t), \tag{1b}$$

where $x := [q\ p\ v\ \alpha\ \theta\ h]^{\mathrm{T}} \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the state vector, which includes the pitch rate, roll rate, ground speed, angle of attack, pitch angle, and altitude, respectively, $u := [\delta_{\mathrm{e_{li}}}\ \delta_{\mathrm{e_{ri}}}\ \delta_{\mathrm{e_{lo}}}\ \delta_{\mathrm{e_{ro}}}\ \delta_{\mathrm{ths}}] \in \mathcal{U} \subseteq R^{n_{\mathrm{u}}}$ is the control input with $\delta_{\mathrm{e_{li}}}, \delta_{\mathrm{e_{ri}}}, \delta_{\mathrm{e_{lo}}}$, and $\delta_{\mathrm{e_{ro}}}$ representing the left inner, right inner, left outer and right outer elevator deflections, δ_{ths} representing the trimmed horizontal stabilizer deflection, and $y := [\mathrm{nz}\ x^{\mathrm{T}}]^{\mathrm{T}} \in \mathcal{Y} \subseteq \mathbb{R}^{p}$ is the output vector with nz representing the vertical load factor. All the states are measurable. These measurements are, however, affected by delays that we must compensate in our design (Section 3).

Concerning the five available control surfaces, the elevators are modeled as third-order linear systems, while the horizontal stabilizer does not have associated dynamics. The following model describes the actuator dynamics:

$$x_{\mathrm{e}}(t+1) = A_{\mathrm{e}}x_{\mathrm{e}}(t) + B_{\mathrm{e}}u_{\mathrm{e}}(t) \tag{2a}$$

$$y_{\mathrm{e}}(t) = C_{\mathrm{e}}x_{\mathrm{e}}(t) + D_{\mathrm{e}}u_{\mathrm{e}}(t) \tag{2b}$$

$$y_{\mathrm{ths}}(t) = u_{\mathrm{ths}}, \tag{2c}$$

where $x_{\mathrm{e}} \in \mathcal{X}_{\mathrm{e}} \in \mathbb{R}^{n_{\mathrm{e}}}$, $u_{\mathrm{e}} \in \mathcal{U}_{\mathrm{e}} \subseteq \mathbb{R}^{n_{\mathrm{u}}-1}$, $u_{\mathrm{ths}} \in \mathcal{U}_{\mathrm{ths}} \subseteq \mathbb{R}$ and $[y_{\mathrm{e}}^{\mathrm{T}}\ y_{\mathrm{ths}}]^{\mathrm{T}} \equiv u$. In the following, we use $u_{\mathrm{MPC}} := [u_{\mathrm{e}}^{\mathrm{T}}\ u_{\mathrm{ths}}]^{\mathrm{T}}$ to indicate the actuator input.

Finally, we assume that $\mathcal{X}, \mathcal{U}, \mathcal{Y}, \mathcal{X}_{\mathrm{e}}$, and $\mathcal{U}_{\mathrm{e}}$ are polyhedral sets that contain the origin in their interior. Furthermore, in the following, we use $\overline{\delta}_{\mathrm{e}_i}$ and $\underline{\delta}_{\mathrm{e}_i}$ to indicate the upper and the lower bounds of the i-th elevator output δ_{e_i}.

2.2 Fault Description

In this paper, the considered fault is elevator jamming, i.e., one or more elevators remain fixed at an unpredictable value $\delta_{\mathrm{e}_i}^{\mathrm{f}}$ ($i \in \mathcal{I} := \{\mathrm{li, ri, lo, ro}\}$), before they could reach their normal saturation limits. The elevator jamming can be attributed to two different root causes:

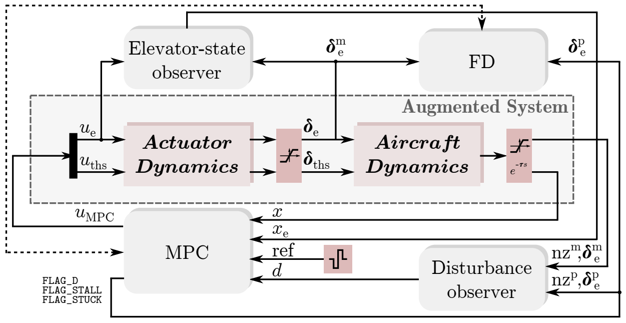


Fig. 1. Proposed control architecture.

- *Stuck Fault.* The elevator is permanently locked at a certain position $\delta_{\mathrm{e}_i}^{\mathrm{f}}$. This effect can be modelled by modifying both the upper and lower control limits equal to the jammed position $\delta_{\mathrm{e}_i}^{\mathrm{f}}$.
- *Stall load* (Goupil et al., 2014). The elevator is temporarily jammed during an aggressive manoeuvre, because strong aerodynamic forces prevent it to achieve its expected control surface deflection. In this situation, the elevator can still move within its reduced control limits $[-\underline{\delta}_{\mathrm{e}_i}, \delta_{\mathrm{e}_i}^{\mathrm{f}}]$ or $[-\delta_{\mathrm{e}_i}^{\mathrm{f}}, \overline{\delta}_{\mathrm{e}_i}]$, determined by the jammed position $\delta_{\mathrm{e}_i}^{\mathrm{f}}$. The stall load ends if either the manoeuvre becomes less aggressive, or the aerodynamic forces become smaller.

Considering their different consequences on the control limits and fault duration, a stuck elevator and stall load need to be distinguished and require adopting different reconfiguration strategies in FTC. Nevertheless, because of the high similarity in the fault phenomena, these two root causes are very difficult to be distinguished. Hence, our proposed integrated FTC approach actively modifies the control strategies to help the FD module discriminate between the two root causes, as detailed in Section 4.

3. FTC ARCHITECTURE

In the following, we describe our FTC architecture. In particular, highlighted in Figure 1, are the actuator dynamics, the aircraft dynamics, the constraints (depicted as saturation blocks), and the sensor delays. Our fault-tolerant controller consists of the following components (depicted in light grey in Figure 1):

Elevator-state observer By using the elevator model (2a)-(2b), four Luenberger observers, characterized by a gain L, are constructed. The gain L is the same for all the operating points, given that the elevators are LTI systems. Each observer independently monitors one elevator. On one hand, the elevator-state estimates are needed to exploit the elevator dynamics in the MPC problem formulation. On the other hand, these elevator-state estimates are used to compute predicted elevator outputs $\delta_{\mathrm{e}}^{\mathrm{p}}$ for the disturbance observer and the FD module.

Disturbance observer The objective of our disturbance observer is twofold: *(i)* compensate for delays and sensor dynamics that are absent in the MPC prediction model by monitoring $e_{\mathrm{nz}} := \mathrm{nz}^{\mathrm{m}} - \mathrm{nz}^{\mathrm{p}}$, i.e., the mismatch between the measured and the predicted load factor, and *(ii)* compensate for $e_{\delta_{\mathrm{e}_i}} := \delta_{\mathrm{e}_i}^{\mathrm{m}} - \delta_{\mathrm{e}_i}^{\mathrm{p}}$, i.e., the mismatch between the measured and the predicted elevator outputs.

The disturbance observer computes the estimated $d := [d_{\mathrm{nz}}\ d_{\mathrm{e}}^{\mathrm{T}}]^{\mathrm{T}}$ as follows:

$$d(t+1) = d(t) + \begin{bmatrix} e_{\mathrm{nz}} \\ e_{\delta_{\mathrm{e}_i}} \end{bmatrix}. \tag{3}$$

This estimated disturbance $d \in \mathbb{R}^{n_{\mathrm{d}}}$ ($n_{\mathrm{d}} = 5$) affects the predicted elevator outputs, the aircraft states, and the aircraft outputs. Hence, we must consider this disturbance as an additional state in the MPC formulation.

FD The Fault Diagnosis module relies on using the elevator-output prediction error $e_{\delta_{\mathrm{e}_i}}$ to compute the residual signal. The generated residual for each elevator is evaluated by its root mean square (RMS) value

$$J_i(t) = \sqrt{\frac{1}{N_{\mathrm{eval}}} \sum_{k=t-N_{\mathrm{eval}}+1}^{t} e^2_{\delta_{\mathrm{e}_i}}(k)} \ \ (i \in \mathcal{I})$$

over a sliding window $[t - N_{\mathrm{eval}} + 1, t]$. The fault detection decision is made by comparing each residual evaluation value $J_i(t)$ with the related threshold J_i^{th}, i.e.,

$$\begin{cases} J_i(t) \le J_i^{\mathrm{th}} \Rightarrow \text{fault-free in elevator } i \\ J_i(t) > J_i^{\mathrm{th}} \Rightarrow \text{jamming in elevator } i. \end{cases} \tag{4}$$

After fixing the length of the sliding evaluation window, the thresholds $\{J_i(t)\}$ are determined by the plant-model mismatch of the elevator model (2). In practice, each threshold J_i^{th} can be selected as the peak value of $J_i(t)$ in a large set of fault-free scenarios. In this work, we determine the thresholds by using an aggressive fault-free manoeuvre, i.e., when stall loads might be more likely to occur.

Note that the detection logic (4) is insufficient to identify the root cause of jamming by itself. In Section 4, we combine the detection logic (4) with different active reconfigurations to capture more detailed fault information.

MPC The model predictive controller uses the augmented aircraft model described below for the prediction. In particular, given (1), (2), and the control increment

$$\Delta u(t) = u_{\mathrm{MPC}}(t) - u_{\mathrm{MPC}}(t-1), \tag{5}$$

the augmented aircraft model is given by:

$$x_{\mathrm{MPC}}(t+1) = A_{\mathrm{MPC}} x_{\mathrm{MPC}}(t) + B_{\mathrm{MPC}} \Delta u(t) \tag{6a}$$
$$y_{\mathrm{MPC}}(t) = C_{\mathrm{MPC}} x_{\mathrm{MPC}}(t) + D_{\mathrm{MPC}} \Delta u(t), \tag{6b}$$

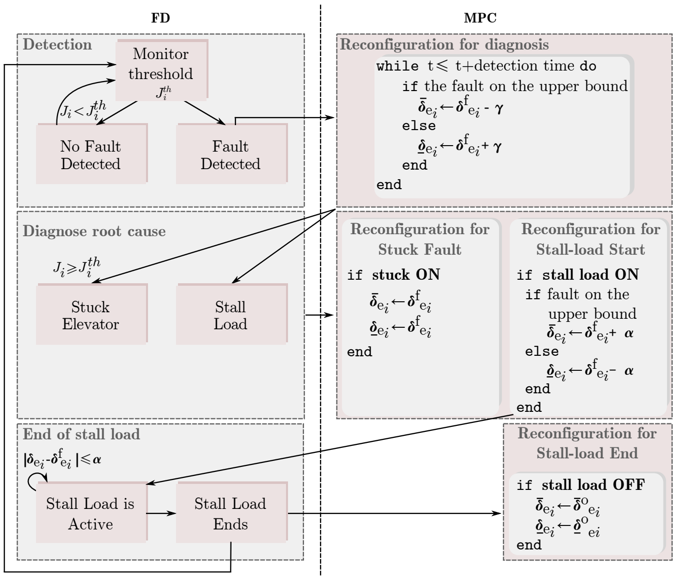
where $x_{\mathrm{MPC}} := [x^{\mathrm{T}}\ x_{\mathrm{e}}^{\mathrm{T}}\ u_{\mathrm{MPC}}^{\mathrm{T}}\ d^{\mathrm{T}}]^{\mathrm{T}} \in \mathcal{X}_{\mathrm{MPC}} \subseteq \mathbb{R}^{n_{\mathrm{MPC}}}$ ($n_{\mathrm{MPC}} := n_{\mathrm{x}} + n_{\mathrm{u}} + n_{\mathrm{w}} + n_{\mathrm{d}}$), $\Delta u \in \Delta\mathcal{U} \in \mathbb{R}^{n_{\mathrm{u}}}$, and $y_{\mathrm{MPC}} := [y^{\mathrm{T}}\ y_{\mathrm{e}}^{\mathrm{T}}]^{\mathrm{T}} \in \mathcal{Y}_{\mathrm{MPC}} := \mathcal{Y} \times \mathcal{U} \subseteq \mathbb{R}^{p+n_{\mathrm{u}}}$. The structure of A_{MPC}, B_{MPC}, C_{MPC}, and D_{MPC} follows from the choice of the state, input, and output for the cascade depicted in Figure 1. Finally, we formulate our control problem:

$$\underset{x_{\mathrm{MPC}} \in \mathcal{X}_{\mathrm{MPC}}, \Delta u \in \Delta\mathcal{U}}{\text{minimize}} \ \mathbf{V}(x_{\mathrm{MPC}}, \Delta u, \mathrm{ref}, x_{\mathrm{init}}) \tag{7a}$$
$$\text{s.t.: } x_{\mathrm{MPC}_0} = x_{\mathrm{init}}, \tag{7b}$$
$$x_{\mathrm{MPC}_{t+1}} = A_{\mathrm{MPC}} x_{\mathrm{MPC}_t} + B_{\mathrm{MPC}} \Delta u_t, \tag{7c}$$
$$y_{\mathrm{MPC}} \in \mathcal{Y}_{\mathrm{MPC}} \tag{7d}$$

where we use x_{MPC_t} ($t = 0, \ldots, H_{\mathrm{p}}$ and H_{p} is the prediction horizon) to indicate the t-step-ahead prediction. The cost (7a) is given by $\mathbf{V} := \frac{1}{2}\sum_{t=0}^{H_{\mathrm{p}}} (\mathrm{nz}_t - \mathrm{ref})^{\mathrm{T}} Q (\mathrm{nz}_t - \mathrm{ref}) + \frac{1}{2}\sum_{t=0}^{H_{\mathrm{u}}} \Delta u_t^{\mathrm{T}} R \Delta u_t$, where $H_{\mathrm{u}} \le H_{\mathrm{p}}$ is the control horizon, $Q \in \mathbb{S}_{\ge 0}^{n_{\mathrm{MPC}}}$, $R \in \mathbb{S}_{>0}^{n_{\mathrm{u}}}$, and ref is the desired reference trajectory generated by the pilot stick, assumed constant along the length of the prediction horizon. By using the dynamic equations (6a) and softening the constraints (7d), we can reformulate Problem (7) as a condensed QP with $\Delta\mathbf{u} \in \mathbb{R}^{n_{\mathrm{u}} H_{\mathrm{u}}}$ and $\epsilon \in \mathbb{R}_{\ge 0}$ (ϵ indicates the slack variable used to soften the constraints) as decision variables:

$$\underset{\Delta\mathbf{u}, \epsilon}{\text{minimize}} \ \frac{1}{2}\Delta\mathbf{u}^{\mathrm{T}} \mathcal{H} \Delta\mathbf{u} + h^{\mathrm{T}} \Delta\mathbf{u} + \rho\epsilon + \mathbf{c} \tag{8a}$$
$$\text{s.t.: } G\Delta\mathbf{u} + E x_{\mathrm{init}} + g - \epsilon\mathbf{1} \le 0. \tag{8b}$$

For details on Problem (8) we refer to Maciejowski (2002).


FD
MPC
Detection
Monitor threshold J_i^{th}
$J_i < J_i^{th}$
No Fault Detected
Fault Detected
Reconfiguration for diagnosis
Diagnose root cause
$J_i \geqslant J_i^{th}$
Stuck Elevator
Stall Load
Reconfiguration for Stuck Fault
Reconfiguration for Stall-load Start
End of stall load
$|\boldsymbol{\delta}_{\mathrm{e}_i} - \boldsymbol{\delta}^{\mathrm{f}}_{\mathrm{e}_i}| \leqslant \alpha$
Stall Load is Active
Stall Load Ends
Reconfiguration for Stall-load End

```
while t ⩽ t+detection time do
    if the fault on the upper bound
        $\bar{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \boldsymbol{\delta}^{\mathrm{f}}_{\mathrm{e}_i} - \gamma$
    else
        $\underline{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \boldsymbol{\delta}^{\mathrm{f}}_{\mathrm{e}_i} + \gamma$
    end
end
```

```
if stuck ON
    $\bar{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \boldsymbol{\delta}^{\mathrm{f}}_{\mathrm{e}_i}$
    $\underline{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \boldsymbol{\delta}^{\mathrm{f}}_{\mathrm{e}_i}$
end
```

```
if stall load ON
  if fault on the
     upper bound
     $\bar{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \boldsymbol{\delta}^{\mathrm{f}}_{\mathrm{e}_i} + \alpha$
  else
     $\underline{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \boldsymbol{\delta}^{\mathrm{f}}_{\mathrm{e}_i} - \alpha$
  end
end
```

```
if stall load OFF
   $\bar{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \bar{\boldsymbol{\delta}}^{\mathrm{o}}_{\mathrm{e}_i}$
   $\underline{\boldsymbol{\delta}}_{\mathrm{e}_i} \leftarrow \underline{\boldsymbol{\delta}}^{\mathrm{o}}_{\mathrm{e}i}$
end
```

Fig. 2. Description of the interaction FD-MPC.

4. INTERACTION FD-MPC

This section aims to describe the close interactions between the FD module and the MPC in our proposed integrated FTMPC approach summarized in Figure 2.

Detection As Figure 2 shows, during the detection phase, the FD module constantly monitors the residual signal. If the residual evaluation signal J_i at time t_{f_i} exceeds the predefined threshold J_i^{th}, the FD module detects jamming of the i-th elevator ($i \in \mathcal{I}$). At this stage, the root cause of jamming is still unknown and the FD module sends a message to the MPC controller to activate the first reconfiguration (reconfiguration for diagnosis in Figure 2).

Reconfiguration for diagnosis The aim of this reconfiguration is to help the FD module understand the root cause of jamming. The MPC checks the sign of $e_{\delta_{\mathrm{e}_i}}$ at time t_{f_i} to decide whether to modify $\bar{\delta}_{\mathrm{e}_i}$ or $\underline{\delta}_{\mathrm{e}_i}$, i.e., the upper or the lower bounds of the i-th elevator. The idea is to temporarily set the faulty-elevator bound to a tightened value $\delta^{\mathrm{f}}_{\mathrm{e}_i} \pm \gamma$, where $\delta^{\mathrm{f}}_{\mathrm{e}_i}$ is the measured value of the elevator at time t_{f_i} and γ is a positive constant that should be tuned sufficiently small to preserve the performance of the controller, but, at the same time, large enough to allow the size of residual signal exceed the predefined threshold J_i^{th} for a stuck elevator. Note that the $\pm$ sign depends on the bound that the MPC modifies, according to the description in Figure 2. The MPC maintains this new γ-tightened bound for τ samples.

Diagnosis of the root cause During the reconfiguration above, the FD module diagnoses the root cause of the jamming by monitoring J_i. If $J_i(t_{\mathrm{f}_i}+\tau) < J_i^{\mathrm{th}}$ during the diagnosis period, the FD module confirms the root cause as a *stall load*, because (thanks to the reconfiguration) the faulty elevator can still move within its reduced bounds. If $J_i(t_{\mathrm{f}_i}+\tau) \geq J_i^{\mathrm{th}}$, the FD module confirms the root cause as a *stuck elevator*, because the faulty elevator was unable to reach the tightened bound.

Reconfiguration for stuck fault As soon as the FD module communicates the root cause of jamming, the MPC performs the second reconfiguration. If the diagnosis is that the elevator is stuck, the MPC performs the reconfiguration for the stuck elevator by setting both $\underline{\delta}_{\mathrm{e}_i}$ and $\overline{\delta}_{\mathrm{e}_i}$ to $\delta^f_{\mathrm{e}_i}$, as Figure 2 shows. This second reconfiguration is also the last one for the stuck elevator.

Reconfiguration for stall-load start If the diagnosis is stall load on the i-th elevator, the MPC performs the following reconfiguration to allow detecting the end of the stall load. It sets the previously modified bound ($\overline{\delta}_{\mathrm{e}_i}$ or $\underline{\delta}_{\mathrm{e}_i}$ depending on the sign of $e_{\delta_{\mathrm{e}_i}}$ at time t_{f_i}) to the new value $\delta^f_{\mathrm{e}_i} \pm \alpha$, i.e., the controller allows a $\alpha > 0$ larger feasible region for the i-th elevator, but does not restore yet the original bound ($\overline{\delta}^{\mathrm{o}}_{\mathrm{e}_i}$ or $\underline{\delta}^{\mathrm{o}}_{\mathrm{e}_i}$). This new limit allows detecting the elevators moving either beyond or below the temporarily jammed position at the end of the stall load.

Remark 1. At the end of a stall load, the elevator is free to move within its original bounds. Setting $\alpha = 0$ prevents the FD module to monitor the end of the stall load if the elevator, when the actual stall load ends, needs to exceed its reduced bound, because the limits in the MPC (with $\alpha = 0$) do not allow the elevator to exceed them, leading to a more conservative behavior.

End of stall load During the reconfiguration for stall-load start, the FD module constantly monitors the mismatch $|\delta^{\mathrm{m}}_{\mathrm{e}_i} - \delta^f_{\mathrm{e}_i}|$. If $|\delta^{\mathrm{m}}_{\mathrm{e}_i} - \delta^f_{\mathrm{e}_i}| \leq \alpha$, the FD module communicates that the stall load is still active on the i-th elevator and the MPC maintains its current formulation. When this condition is violated, the FD module communicates the end of the stall load to the controller and returns to monitor the residual value.

Reconfiguration for stall-load end When the stall load ends, the MPC must restore the original saturation limit, which is the last reconfiguration for the stall load.

Remark 2. The MPC reconfiguration can handle more than one elevator fault at a time, thanks to the decoupled structure of the FD module, which monitors each elevator independently. Furthermore, delays in the communication of the root cause of jamming are partially compensated by the disturbance observer, while possible infeasibility issues during the reconfiguration are handled by using soft constraints, as explained in Section 3.

5. SIMULATION RESULTS

This section presents some illustrative results of our integrated control strategy on a nonlinear Airbus simulator. In this respect, we compared the following designs:

MPC This design relies only on the information provided by the disturbance observer. The MPC controller handles the faults as additional disturbances, but does not reconfigure itself to compensate for the faults.

MPC + SED This design relies on a FD module unable to discriminate between a stuck elevator and a stall load. When a jamming occurs, the FD module assumes that it is a stuck fault and communicates the information to the MPC for the reconfiguration, according to the strategy presented in Section 4 for the stuck fault. The acronym SED stands for Stuck-Elevator Detection.

MPC + SED + SLD This design also relies on the FD module to perform the reconfiguration of the controller. Compared to the previous one, the FD module is able to understand whether the elevators are stuck or only temporarily jammed, using the information provided by the MPC controller during the diagnosis phase (Section 4). The acronym SLD stands for Stall-Load Detection.

The baseline for the comparison is the behavior of the system controlled by the MPC, in the fault-free case. We maintained the same control tuning parameters to allow a fair comparison with the **MPC** design. Furthermore, we selected the threshold J_i^{th} in the FD module, according to the guideline of Section 3. In addition, we selected the time required for the diagnosis of the root cause of the jamming $\tau_{\mathrm{f}} = 30\ T_s$ ($T_s := 40$ ms indicates the sampling time), compromising between the performance of the controller and the accuracy of the detection. Another parameter that requires a trade-off between performance and accuracy is γ, used to tighten the faulty-elevator constraints during the *reconfiguration-for-diagnosis* phase. We adopted the following strategy:

$$\gamma_i := \frac{t - t_{\mathrm{f}_i}}{\tau_{\mathrm{f}}}[\delta^{\mathrm{f}}_{\mathrm{e}_i} \pm \varepsilon - \delta^{\mathrm{p}}_{\mathrm{e}_i}(t_{\mathrm{f}_i})] - \delta^{\mathrm{p}}_{\mathrm{e}_i}(t_{\mathrm{f}_i}), \qquad (9)$$

where t is the current time, $\varepsilon > 0$ sufficiently small (the $\pm$ depends on the bound that is affected by the stall load), and $\delta^{\mathrm{p}}_{\mathrm{e}_i}(t_{\mathrm{f}_i})$ is the predicted value of the i-th elevator when the fault is detected. This strategy allows the diagnosis of the root cause of jamming and prevents undesired oscillations on the elevator output. Finally, we selected α large enough to avoid false alarms in the detection of the stall load end right after the diagnosis phase.

We trimmed the aircraft at an altitude of 12,500 feet and calibrated airspeed of 335 knots (inside the flight envelope). The MPC updates its prediction model based on the actual operating point in the flight envelope by interpolating over a set of available linearized models using Mach number and altitude as scheduling parameters. The goal of the controller is to track a doublet signal on the vertical load factor nz produced by the pilot, which causes the aircraft to deviate from its initial operating point. The doublet starts at 5 sec and ends at 20 sec. We compare the performance of the three designs in two different scenarios:

- Stall load that prevents the outer elevators to exceed 0 deg, i.e., the outer elevators can only move within $[\underline{\delta}_{\mathrm{e}_o} 0]$.
- Stuck outer elevators at 0 deg at 7.76 sec since the beginning of the simulation.

Note that the nonlinear benchmark only allows to simulate a stuck elevator at 0 deg. The stall load value had been selected accordingly for comparison. Furthermore, these fault scenarios are chosen to avoid instability issues. In general, instability of the system in the presence of faults might occur, based on the severity of the faults themselves.

However, our main goal, when focusing on the given fault scenarios, is to show how our technique is able to detect and handle actuator faults in an integrated fashion.

Figure 3 shows the results obtained using the aforementioned three different designs. In particular, the left column compares them when the stall load occurs on the outer elevators, while the right column compares them when the outer elevators remain stuck. The first row of Figure 3 shows the vertical load factor, the second and the fourth rows depict the elevator behaviors, and the third row presents the residual evaluation values.

Our proposed approach (solid blue line) largely improves the nz tracking, confirming that the diagnosis of the root cause of the jamming is fundamental for the performance of the overall control system. In particular, note the behavior of the vertical load factor using the **MPC** design (solid orange line) in both scenarios. The disturbance observer alone is not able to compensate for the faults, confirming that control reconfiguration is necessary to preserve the performance of the aircraft. Furthermore, note the behavior of the system controlled using the **MPC+SED** design (solid grey line), when the stall load occurs on the outer elevators (left column). When the stall load ends (at $\approx$ 20 sec), there is a performance loss on the nz tracking obtained using the **MPC+SED** design caused by the activation of the slack variable used to soften the outer elevator outputs, which violate the constraints imposed by the controller after the stuck elevator detection. Finally, note that the relatively long detection time does not affect the performance of the controller. Finally, as Figure 3 shows, the **MPC+SED** and the **MPC+SED+SLD** show comparable performance in the stuck elevator scenario (right column).

In order to illustrate our strategy, consider the second and third rows of the left column, which represent the outer elevators in stall load and the associated residual evaluation values. When $J_{\mathrm{e}_{lo}}$ and $J_{\mathrm{e}_{ro}}$ violate the threshold (at $t_{\mathrm{f}_o} \approx 10$ sec), the FD module activates the first reconfiguration. Given that the elevators can still move within their reduced bounds, the residuals decrease as soon as the elevator output reacts to the change in the bounds. Hence, the FD module diagnoses a stall load on the outer elevators. After the end of the diagnosis, the MPC relaxes the upper bound (at $\approx$ 11 sec) and the FD module waits for the end of the stall load. When the FD module detects the end of the stall load (at $\approx$ 22 sec), the MPC restores the original bounds. Finally, note the behavior of the healthy inner elevators that react accordingly to compensate the loss of the outer ones during the fault.

Consider the second and third rows of the right column, which represent the behaviors of the stuck outer elevators and the associated residual evaluation values. The FD module detects the occurrence of a jamming as soon as $J_{\mathrm{e}_{lo}}$ and $J_{\mathrm{e}_{ro}}$ violate the threshold (at $t_{\mathrm{f}_o} \approx 10$ sec). As a consequence, the MPC controller updates the upper bound according to the reconfiguration-for-diagnosis strategy for τ_{f} samples. Given that $J_{\mathrm{e}_{lo}}$ and $J_{\mathrm{e}_{ro}}$ remain above the threshold, the FD module diagnoses that the elevators are stuck and the MPC controller updates both the upper and the lower bounds according to the reconfiguration-for-stuck strategy, causing $J_{\mathrm{e}_{lo}}$ and $J_{\mathrm{e}_{ro}}$ to return below the threshold, as Figure 3 shows.

6. CONCLUSIONS AND FUTURE WORK

We presented a novel fault-tolerant controller tailored to aerospace applications. Our approach relies on the close interaction between FD module and MPC controller. The FD module exploits the controller to diagnose the root cause of the elevator jamming and the MPC exploits the information provided by the FD module to better handle the jamming. We showed on a nonlinear Airbus simulator the benefits that our strategy can bring to the performance of the control system. As part of our future work, we will perform more extensive tests on the nonlinear simulator at different operating points.

REFERENCES

De Almeida, F.A. and Leißling (2010). Fault-tolerant model predictive control with flight-test results. *Journal of Guidance, Control, and Dynamics*, 33, 363–375.

Edwards, C., Smaili, H., and Lombaerts, T. (2010). *Fault Tolerant Flight Control: A Benchmark Challenge.* Springer-Verlag.

Goupil, P., Boada-Bauxell, J., Marcos, A., Cortet, E., Kerr, M., and Costa, H. (2014). AIRBUS efforts towards advanced real-time fault diagnosis and fault-tolerant control. In *Proc. of the 19th IFAC World Congress*, 3471–3476. Cape Town, South Africa.

Kale, M.M. and Chipperfield, A.J. (2005). Stabilized MPC formulations for robust reconfigurable flight control. *Control Engineering Practice*, 13, 771–788.

Lew, J. (2013). Robust predictive control for structures under damage condition. *Journal of Guidance, Control, and Dynamics*, 36, 1824–1829.

Maciejowski, J.M. (1998). The implicit daisy-chaining property of constrained predictive control. *Applied Mathematics and Computer Science*, 8, 695–712.

Maciejowski, J.M. (2002). *Predictive control: with constraints.* Pearson education.

Maciejowski, J.M. and Jones, C.N. (2003). MPC fault-tolerant flight control case study: flight 1862. In *Proc. of IFAC Safeprocess Sympoisum*, 119–124. Washington, DC, USA.

Puncochar, I., Siroky, J., and Simandl, M. (2015). Constrained active fault detection and control. *IEEE Transactions on Automatic Control*, 60, 253–258.

Raimondo, D.M., Marseglia, G.R., Braatz, R.D., and Scott, J.K. (2013). Fault-tolerant model predictive control with active fault isolation. In *Proc. of 2013 Conference on Control and Fault-Tolerant Systems.* Nice, France.

Stoican, F. and Olaru, S. (2013). *Set-theoretic Fault-tolerant Control in Multisensor Systems.* John Wiley & Sons, Inc.

Xu, F., Olaru, S., Puig, V., Ocampo-Martnez, C., and Niculescu, S. (2014). Sensor-fault tolerance using robust MPC with set-based state estimation and active fault isolation. In *Proc. of 53rd Conference on Decision and Control*, 4953–4958. Los Angeles, CA.

Yetendje, A., Seron, M.M., and De Doná, J.A. (2013). Robust multiactuator fault-tolerant MPC design for constrained systems. *International Journal of Robust and Nonlinear Control*, 23, 1828–1845.

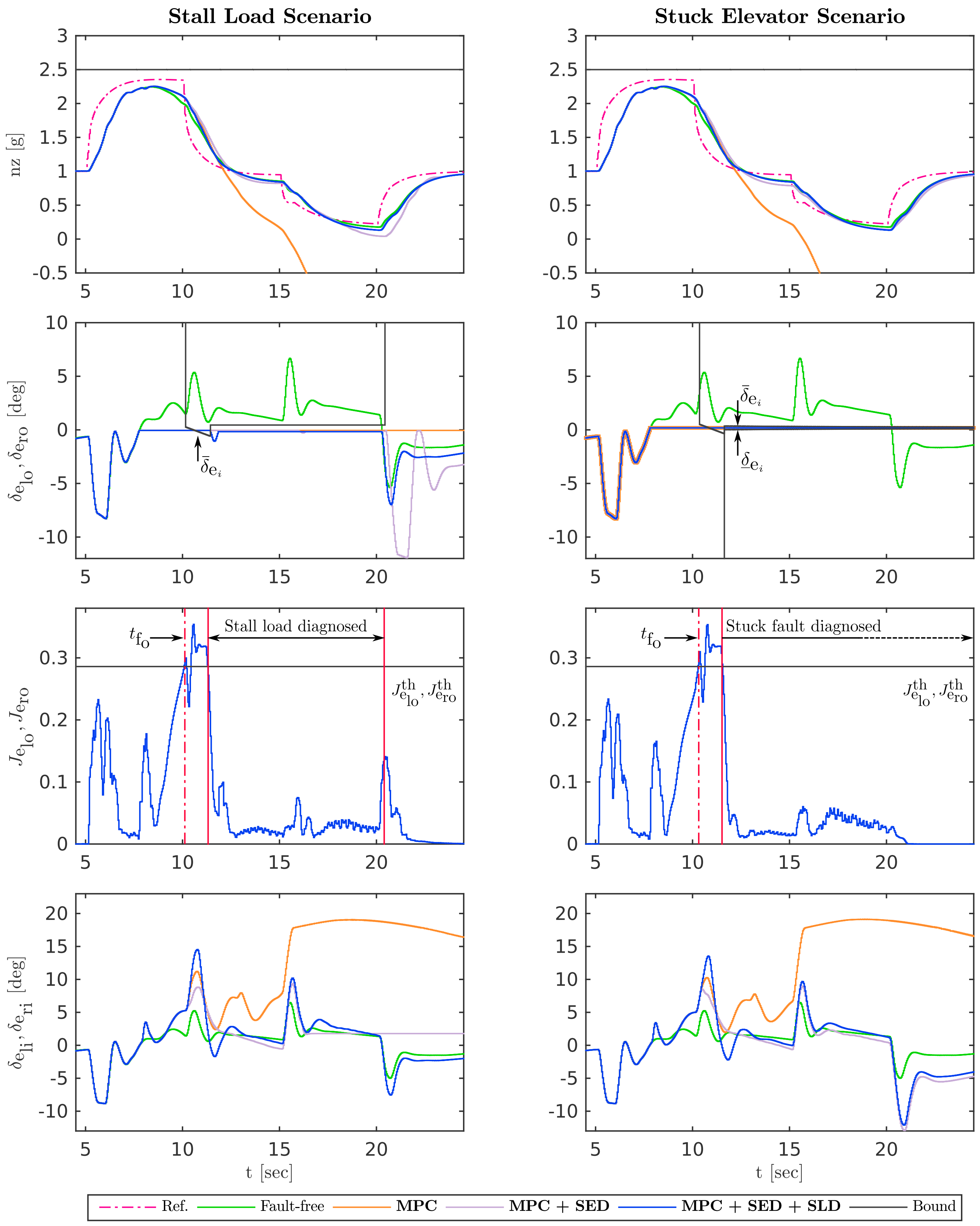


Fig. 3. Evaluation of three designs to handle the elevator jamming. The left column shows the behavior of the system when a stall load occurs on the outer elevators. The right column depicts the behavior of the system when the outer elevators are stuck at their position.