Gain-scheduled two-loop autopilot for an aircraft ⋆

L. Ravanbod * **D. Noll** **

* *Université Paul Sabatier, Institut de Mathématiques, Toulouse, France (Tel: 0033-561558626; e-mail: lalehravanbod@yahoo.fr).*
** *Université Paul Sabatier, Institut de Mathématiques, Toulouse, France (Tel: 0033-561558624; e-mail: dominikus.noll@math.univ-toulouse.fr)*

Abstract: We present a new method to compute output gain-scheduled controllers for non-linear systems. We use structured H_∞-control to pre-compute an optimal controller parametrization as a reference. We then propose three practical methods to implement a control law which has only an acceptable loss of performance with regard to the optimal reference law. Our method is demonstrated in longitudinal flight control, where the dynamics of the aircraft depend on the operational conditions velocity and altitude. We design a structured controller consisting of a PI-block to control vertical acceleration, and another I-block to control the pitch rate.

1. INTRODUCTION

We design a vertical acceleration hold system for longitudinal flight control of an aircraft, which consists in a gain-scheduled autopilot combining a PI-block to control vertical acceleration in the outer loop with a I-block to control the pitch rate in the inner loop. The nonlinear dynamics of the aircraft are represented as a parameter-varying family of linearizations at a large number of trimmed flight conditions, forming the flight envelope $\mathcal{E}$. Aerodynamic flight conditions $\mathbf{e} \in \mathcal{E}$ may either be classified by altitude/velocity, $\mathbf{e} = (h, V)$, or by Mach/dynamic pressure, $\mathbf{e} = (M, \bar{q})$.

The way in which we construct gain-scheduled PI-I-controllers $K(\mathbf{e})$ is original in so far as it introduces the H_∞-control paradigm into the realm of PI-I control, a domain where controllers are generally *tuned* using heuristics, not optimized. We proceed as follows. We introduce a suitable closed-loop performance channel $w \to z$, which reflects the imposed performance and robustness specifications. Then we pre-compute the H_∞-optimal structured PI-I-controller at every flight point $\mathbf{e} \in \mathcal{E}$, using a plant $P(\mathbf{e})$ representing the linearized open-loop system at flight point $\mathbf{e} \in \mathcal{E}$. In other words, for every $\mathbf{e} \in \mathcal{E}$ we pre-compute a solution $K^*(\mathbf{e})$ to the structured H_∞-control problem

$$\begin{array}{ll} \text{minimize} & \|T_{w\to z}\,(P(\mathbf{e}), K)\,\|_\infty \\ \text{subject to} & K \text{ is a PI-I-controller} \\ & K \text{ stabilizes } P(\mathbf{e}) \text{ internally} \end{array} \tag{1}$$

Roughly, $K^*(\mathbf{e})$ stands for the *best* way to control the system at flight conditions $\mathbf{e} \in \mathcal{E}$ instantaneously. Another explanation is as follows: if we could compute an optimal H_∞-controller $K^*(\mathbf{e})$ with the required PI-I-structure in real time t, then we would do this at the flight point $\mathbf{e}(t)$ and apply $K^*(\mathbf{e}(t))$ to $P(\mathbf{e}(t))$ at that instant.

In a second step we use this theoretical control law $K^*(\mathbf{e})$, ($\mathbf{e} \in \mathcal{E}$) as a reference to construct a more practical scheduled PI-I-controller $K(\mathbf{e})$. This controller should be convenient to embed and to store, and yet should not fall back behind $K^*(\mathbf{e})$ in H_∞-performance by more than a fixed percentage. In other words, an admissible parametrization $K(\mathbf{e})$ has to satisfy

$$\|T_{w\to z}\,(P(\mathbf{e}), K(\mathbf{e}))\|_\infty \le (1+\alpha)\,\|T_{w\to z}\,(P(\mathbf{e}), K^*(\mathbf{e}))\|_\infty \tag{2}$$

for every $\mathbf{e} \in \mathcal{E}$, where for instance $\alpha = 10\%$. We present three methods to compute such a practical gain-scheduled PI-I autopilot $K(\mathbf{e})$, referred to as (a) by triangulation, (b) by the greedy method, and (c) by fitting.

It is interesting to compare our philosophy to existing techniques in parameter-varying control. A widely used approach computes full-order LPV controllers via quadratic stability [4] and LMIs. This gives a stability certificate and allows criteria like H_∞ or H_2, see [1, 2]. A limitation is that PI-I controllers in a complex closed loop control configuration (as in our example) are not available as long as one wishes to stay with LMIs. More seriously, however, is the fact that this approach is intrinsically conservative due its worst-case point of view. Namely, the smallest γ with

$$\max_{\mathbf{e}\in\mathcal{E}} \|T_{w\to z}\,(P(\mathbf{e}), K(\mathbf{e}))\|_\infty \le \gamma \tag{3}$$

is sought, whereas the idea in $K^*(\mathbf{e})$, respectively in (2), is to perform as good as possible for *every* $\mathbf{e} \in \mathcal{E}$. Our study will show that (2) may indeed have huge advantages over (3).

Switching LPV control has been considered an alternative, as it uses multiple parameter-dependent Lyapunov functions [9, 10], reducing conservatism. But even then one has to accept that the LPV approach within PID control has

⋆ Funding by Fondation de Recherche pour l'Aéronautique et l'Espace under contract *Survol*, and by Fondation EADS under contract *Technicom* is gratefully acknowledged.

strong limitations. For example: variable parameters are measured precisely, but are not included in the state space [11, 5, 9]. The output matrix is parameter independent and full row rank [12].

On the practical side there exists a large variety of techniques to tune PID controllers and PID architectures, both for LTI and parameter-varying systems. Since the 1960s empirical gain-scheduling control has been used for nonlinear and time varying systems. This achieves closed loop stability for slowly varying parameters, but in contrast with H_2 and H_∞ techniques, no optimality in any sense is achieved.

The paper is structured as follows. In Section 2 we discuss the non-linear open-loop model. In Section 3 we explain how the system is linearized at the flight points $\mathbf{e}$ in the flight envelope $\mathcal{E}$ and then the H_∞-synthesis scheme taking into account the control law specifications at each flight point $\mathbf{e}$ is determined. Then the pointwise optimal structured H_∞ controller is constructed. Practical scheduled PI-I-controllers are constructed in Sections 4 – 6.

2. NONLINEAR AIRCRAFT MODEL

We have used a nonlinear aircraft model available in the file $rct\_airframe1$ of simulink used within MatlabR2010b. This is a 3 degree-of-freedom model in longitudinal mode. Compared to the 6 degree-of-freedom model it is assumed that $p = v = r = \Phi = \psi = Y_e = 0$. For a description of the complete model see [16, 15].
Equations of motion are:

$$\begin{aligned} \dot{u} &= -g\sin(\theta) - qw + a_x + F_T/m \\ \dot{w} &= g\cos(\theta) + qu + a_z \\ \dot{\theta} &= q \\ \dot{q} &= \bar{M}/I_y \\ \dot{X}_e &= u\cos(\theta) + w\sin(\theta) \\ \dot{Z}_e &= -u\sin(\theta) + w\cos(\theta) \end{aligned} \tag{4}$$

where

$$\begin{aligned} a_x &= \frac{\bar{q}S}{m}\left(C_{x\alpha}(\alpha, M) + C_{xl0}\cdot\delta_e\right), \\ a_z &= \frac{\bar{q}S}{m}\left(C_{z\alpha}(\alpha, M) + C_{zl0}\cdot\delta_e\right), \end{aligned}$$

and X_e, Z_e $[m]$: x, z-position w.r.t. earth, $h = -Z_e$ altitude, u, w $[m/s]$: longitudinal and normal velocities, V $[m/s]$: total aircraft velocity, θ $[rad]$: pitch angle, q $[rad/s]$: pitch rate, $\delta_{fin} = \delta_e$ $[rad]$ elevator angle, I_y: moment of inertia about y body axis, M: Mach number, $\bar{M}$: aerodynamic moment, m: mass, S : wing surface area, $\bar{q}$: dynamic pressure, C_{xl0} and C_{zl0}: the constants, C_{xt} and C_{zt} : aerodynamic coefficients. In the present study the thrust F_T is held constant at 1000 N.
For synthesis the system (4) is linearized at the various trimmed flight points $\mathbf{e} = (h, V) \in \mathcal{E}$. The system is considered to be stabilized via $K^\sharp$, which is for simplicity chosen independent of $\mathbf{e} \in \mathcal{E}$. The PI part of $K^\sharp$ (az control in the lower image of Figure 1) is $k_p + \frac{k_i}{s} = 0.003 + \frac{0.01}{s}$, the static q-gain is $k_g = 1.5$.
This system is then reduced to a family of second-order models for the short-period longitudinal motion

$$\begin{bmatrix} d\dot{V} \\ d\dot{q} \end{bmatrix} = \begin{bmatrix} A_{11}(h,V) & A_{12}(h,V) \\ A_{21}(h,V) & A_{22}(h,V) \end{bmatrix} \begin{bmatrix} dV \\ dq \end{bmatrix} + \begin{bmatrix} B_{21}(h,V) \\ B_{22}(h,V) \end{bmatrix} d\delta_e$$

indexed by the flight points $\mathbf{e} = (h, V)$, where $V(t) = V + dV(t)$, $q(t) = q(h,V) + dq(t)$, $\delta_e(t) = \delta_e(h,V) + d\delta_e(t)$ represent offsets about nominal values at (h, V). Outputs are $a_z(t) = a_z(h,V) + da_z(t)$ and $q(t) = q(h,V) + dq(t)$. In our study we use a rectangular grid in the (h, V)-plane

$$h \in [1500, 12000]\, m, \Delta h = 525\,(21 \text{ steps})$$
$$V \in [700, 1150]\, m/s^2, \Delta V = 15\,(31 \text{ steps})$$

leading to a total of $21 \cdot 31 = 651$ flight points $\mathbf{e} = (h, V)$ forming the flight envelope $\mathcal{E}$ (see Figure 2 left).

3. H_∞ CONTROL

For synthesis the parameter-varying model (5) has to be completed into a plant $P(\mathbf{e}) = P(h, V)$ by adding disturbances (wind gusts), reference input signals, and performance and robustness channels. This parameter-varying plant $P(\mathbf{e})$ will be described in the following section and used to synthesize a scheduled PI-I controller.
The H_∞-control scheme used to synthesize a gain-scheduled controller is shown in Figure 1 (b). In this architecture, the tunable elements include the two PI controller gains ("az Control" block) and the pitch-rate gain ("q Gain" block). The autopilot must respond to a step command $a_{z,\text{ref}}$ in about 1 second with minimal overshoot.

In view of the response time requirement, the target crossover frequency ω_c is set to 2 rad/s and the target loop shape $LS(s) = \frac{1+0.001\frac{s}{\omega_c}}{0.001+\frac{s}{\omega_c}}$ is used. It can be shown that if the peak gain of the closed-loop transfer from w to z is close to 1, then

- The open-loop response approximately matches the target loop shape $LS(\cdot)$;
- The worst-case sensitivity is close to 1, which ensures good stability margins for the outer loop;
- The overshoot in the response to an $a_{z,\text{ref}}$ step command is small;
- The gain from d to a_z does not exceed $m = 1000$.

To fix the filter m we have used the specific flight point $h = 3050\ m$ and $V = 984\ m/s$, where the peak gain from d to a_z is 60 dB, meaning that m should be at least 60 dB, or $m = 1000$. In order to satisfy these control law specifications, the closed-loop performance channel $w \to z$ with $w = (a_{z,\text{ref}}, md, LS(s)n)$ and $z = (LS(s)e, a_z)$ is chosen.

This is now where our new control strategy sets in. For each of the 651 points $\mathbf{e} = (h, V)$ in the flight envelope $\mathcal{E}$ we compute an H_∞-optimal PI-I controller $K^*(\mathbf{e})$ using the optimization program (1). To solve (1) we use the Matlab function HINFSTRUCT [MatlabR2010b], which is based on the fundamental work [3]. The rationale of HINFSTRUCT relies on non-smooth optimization and can be found in [3] or [13]. For details on the use of HINFSTRUCT see [MatlabR2010b].

The closed-loop performance graph

$$(h, V) \mapsto \|T_{w\to z}\left(P(h,V), K^*(h,V)\right)\|_\infty$$

is shown on the right of Figure 2.

Using the optimal controller $K^*(\mathbf{e})$ would require storing 651×5 numerical values (3 gains for each (h, V) and h, V

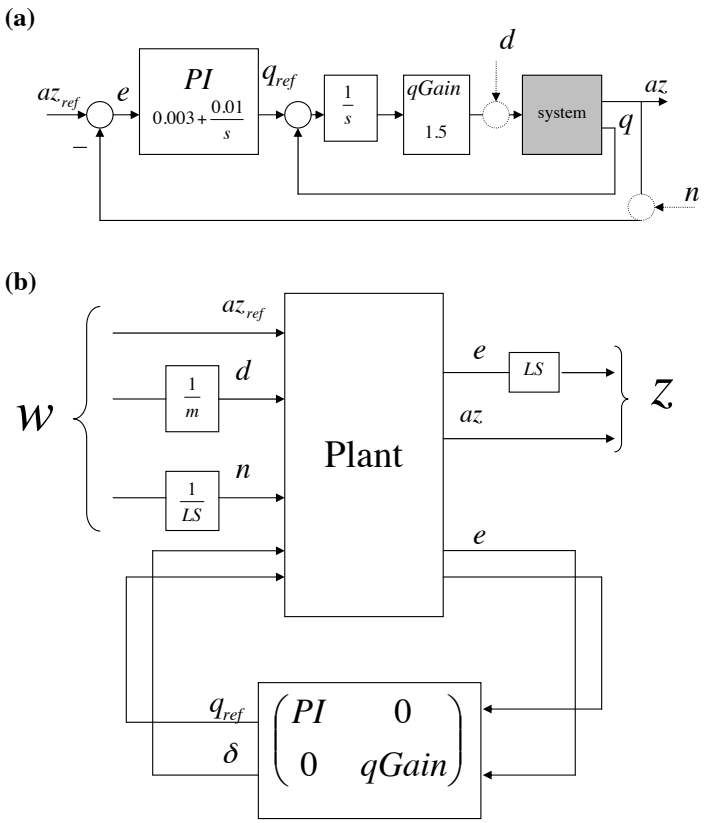

Fig. 1. Schemes used for (a): linearizing the non linear aircraft, (b) H_∞ synthesis

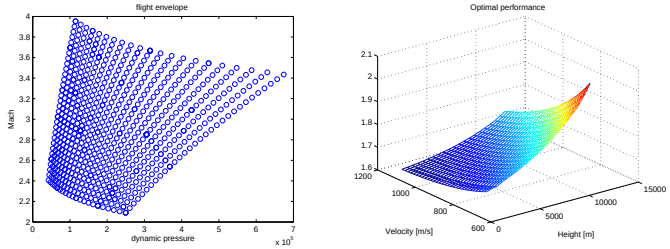

Fig. 2. Left image shows flight envelope $\mathcal{E}$ in the geometry $(M, \bar{q})$. Right image plots optimal H_∞ performance over $\mathcal{E}$, now in the geometry $\mathbf{e} = (h, V)$.

themselves), plus the rules to look values up in the table. Our goal is therefore to find parameterizations $K(\mathbf{e})$ which are easier to handle and need less storage, but at the same time show acceptable performance in the sense of rule (2), where our experiments use $\alpha = 10\%$. We subsequently describe the three approaches, termed (a) by triangulation, $K_{\text{tri}}(\mathbf{e})$, (b) by the greedy method, $K_{\text{greedy}}(\mathbf{e})$, and (c) by interpolation of gains $K_{\text{int}}(\mathbf{e})$.

4. TRIANGULATION

In this approach one constructs a triangulation of the flight envelope $\mathcal{E}$ such that every node $\mathbf{e}_i$ is in $\mathcal{E}$ and the triangulated controller $K_{\text{tri}}(\mathbf{e}_i)$ coincides with $K^*(\mathbf{e}_i)$ at the $\mathbf{e}_i$. Within each triangle Δ_{ijk} with corners $\mathbf{e}_i = (h_i, V_i)$, $\mathbf{e}_j = (h_j, V_j)$, $\mathbf{e}_k = (h_k, V_k)$ oriented clockwise and $\mathbf{e} = (h, V) \in \mathcal{E} \cap \Delta_{ijk}$ the function $k_p(\mathbf{e})$ is defined as $k_p(h, V) = ah + bV + c$, where

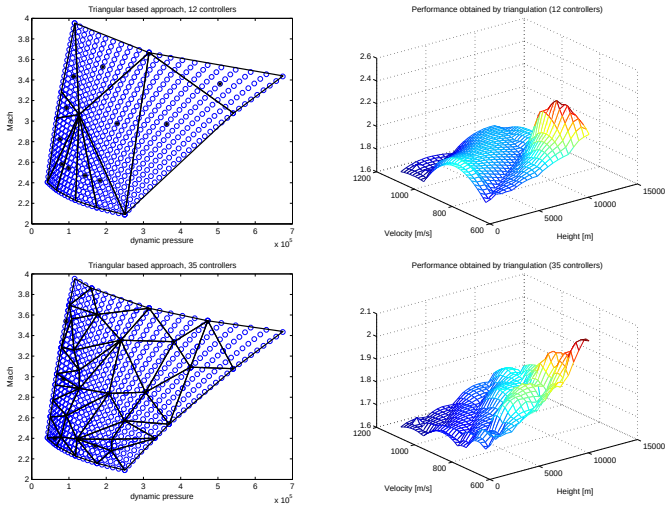

Fig. 3. Two triangulations together with the performance graphs $(h, V) \mapsto \|T_{w\to z}(P(h, V), K(h, V))\|_\infty$ are shown. The controller $K(h, V)$ is linear on each triangle and coincides with $K^*(h, V)$ at the nodes $\mathbf{e}_i$. The finer triangulation satisfies (1), the coarser fails.

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = Q^{-1} \begin{bmatrix} k_p(\mathbf{e}_i) \\ k_p(\mathbf{e}_j) \\ k_p(\mathbf{e}_k) \end{bmatrix} \text{ with } Q = \begin{bmatrix} h_i & V_i & 1 \\ h_j & V_j & 1 \\ h_k & V_k & 1 \end{bmatrix}. \qquad (5)$$

Computation of $k_i(h, V)$ and $k_g(h, V)$ is analogous. In this way $K_{\text{tri}}(\mathbf{e})$ is piecewise affine on the triangles Δ_{ijk} and continuous as a function of (h, V). The triangulation is acceptable if, according to (2), the closed-loop performance of $K_{\text{tri}}(\mathbf{e})$ does not exceed 110% of the performance of $K^*(\mathbf{e})$ at each of the 651 flight points $\mathbf{e} \in \mathcal{E}$. Figure 3 shows two examples. The controller constructed by the upper triangulation with 11 triangles (and 12 nodes) is not acceptable, because the performance exceeds the 110%. The lower triangulation needs 35 triangles (with 45 nodes) and leads to an acceptable controller. For each node 5 informations (3 for controller parameters and 2 for h, V) must be stored.

In [8] a related approach based on a triangulation of the flight envelope is developed. The authors treat the problem by a receding horizon model-based predictive control method. To conclude, for each point (h, V) in the flight envelope, the controller $K_{\text{tri}}(h, V)$ can be found if $45 \cdot 5 = 225$ data and an algorithm to find to which triangle h, V belongs are stored.

5. THE GREEDY METHOD

A very natural way to construct a controller parametrization $K_{\text{greedy}}(\mathbf{e})$ goes as follows. For a given flight point $\mathbf{e} = (h, V) \in \mathcal{E}$ pick the optimal controller $K^*(\mathbf{e})$ and apply it not only to $P(\mathbf{e})$ but also to neighboring plants $P(\mathbf{e}')$. As long as $\mathbf{e}'$ is close to $\mathbf{e}$, we expect $K^*(\mathbf{e})$ to work well for $P(\mathbf{e}')$, but eventually, as $\mathbf{e}'$ gets farther away from $\mathbf{e}$, we expect a loss of performance or even stability. We therefore define a neighborhood of $\mathbf{e} \in \mathcal{E}$ as follows:

$$\mathcal{N}(\mathbf{e}) = \Big\{ \mathbf{e}' \in \mathcal{E} : K^*(\mathbf{e}) \text{ stabilizes } P(\mathbf{e}') \text{ internally, and}$$
$$\|T_{w\to z}\left(P(\mathbf{e}'), K^*(\mathbf{e})\right)\|_\infty \le (1+\alpha)\|T_{w\to z}\left(P(\mathbf{e}'), K^*(\mathbf{e}')\right)\|_\infty \Big\}$$

The meaning of $\mathcal{N}(\mathbf{e})$ is simply that $K^*(\mathbf{e})$ works acceptably (in the sense of (2)) on this set. Naturally, we have

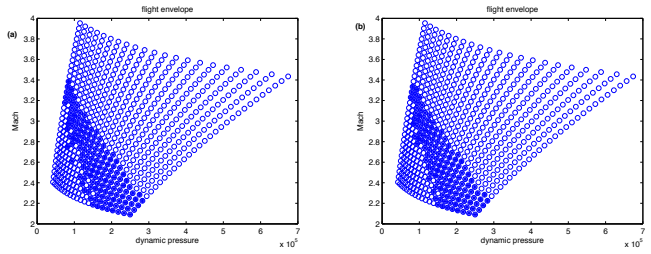

Fig. 4. Example of pre-processing of the regions $\mathcal{N}(\mathbf{e})$. (a) without smoothing , (b) with smoothing.

$\mathbf{e} \in \mathcal{N}(\mathbf{e})$, so that $\{\mathcal{N}(\mathbf{e}) : \mathbf{e} \in \mathcal{E}\}$ is a set-covering of $\mathcal{E}$. Extracting a subcover with a minimum number of elements is now an instance of the so-called *minimum set-covering problem*. To solve it we use a heuristic, called the greedy method, hence the name for the controller so constructed. The greedy method is extremely simple and works as follows. Pick the largest neighborhood $\mathcal{N}_1 := \mathcal{N}(\mathbf{e}_1)$. Now $\{\mathcal{N}(\mathbf{e}) \setminus \mathcal{N}_1 : \mathbf{e} \in \mathcal{E}\}$ is a set cover of $\mathcal{E} \setminus \mathcal{N}_1$. Pick $\mathcal{N}_2 := \mathcal{N}(\mathbf{e}_2)$ such that $\mathcal{N}(\mathbf{e}_2) \setminus \mathcal{N}_1$ is the largest element in this reduced cover. Now $\mathcal{E} \setminus (\mathcal{N}_1 \cup \mathcal{N}_2)$ is covered. Continue in this way until a cover of $\mathcal{E}$ is found. The original cover consists of 651 neighborhoods $\mathcal{N}(\mathbf{e})$, $\mathbf{e} \in \mathcal{E}$. Before applying the greedy algorithm, we have the option to do some pre-processing of these 651 sets $\mathcal{N}(\mathbf{e})$. We eliminate isolated points and use image processing methods to smoothen the boundaries of the $\mathcal{N}(\mathbf{e})$. As we are dealing with 0-1 images, this is simple to perform, for instance, median filtering has an immediate effect, where neighborhoods are slightly reduced to a more pleasant form. Figures 4 show some of the neighborhoods obtained by this procedure. While this greedy procedure is simple to carry out, the obtained controller parametrization is piecewise constant, $K_{\text{greedy}}(\mathbf{e}) = K^*(\mathbf{e}_i)$ when $\mathbf{e} \in \mathcal{N}(\mathbf{e}_i)$, and therefore discontinuous. This is demonstrated by the performance graph shown in Figure 6, right. We mention that it is advisable to use hysteresis to avoid chattering effects along the region boundaries. Formally, a switching controller with hysteresis is not a function of $\mathbf{e} = (h, V)$ alone, but a function of $(\mathbf{e}, \dot{\mathbf{e}})$, where $\dot{\mathbf{e}}$ indicates the direction along which $\mathbf{e}$ is reached. The fact that the performance graph of the greedy controller $K_{\text{greedy}}(\mathbf{e})$ is discontinuous does not mean that its performance is unsatisfactory. The fact that (2) is respected everywhere assures that performance of $K_{\text{greedy}}(\mathbf{e})$ is very similar to the performance of $K^*(\mathbf{e})$ and $K_{\text{int}}(\mathbf{e})$. To conclude, without any specification on the geometrical form of the regions found by greedy (as in our case), for each point $\mathbf{e}$) in the flight envelope, in addition to its h, V informations, we must also store to which of the regions it belongs. Hence, to find the controller $K_{\text{greedy}}(h, V)$ $(651 - 7) \cdot 3 + 7 \cdot 3 = 1953$ data must be stored.

6. FITTING APPROACH

One may consider $K^*(\mathbf{e})$ itself as a valid controller parametrization, with the drawback that it needs storage of $651 \cdot 5$ numbers (5 because of h, V and 3 controller parameters). If this is considered too large, the idea arises to represent the numerically defined optimal gains $k_i^*(\mathbf{e})$, $k_p^*(\mathbf{e})$, $k_g^*(\mathbf{e})$ by approximations $\hat{k}_i(\mathbf{e})$, $\hat{k}_p(\mathbf{e})$, $\hat{k}(\mathbf{e})$, which are simpler to compute. This leads to methods where the

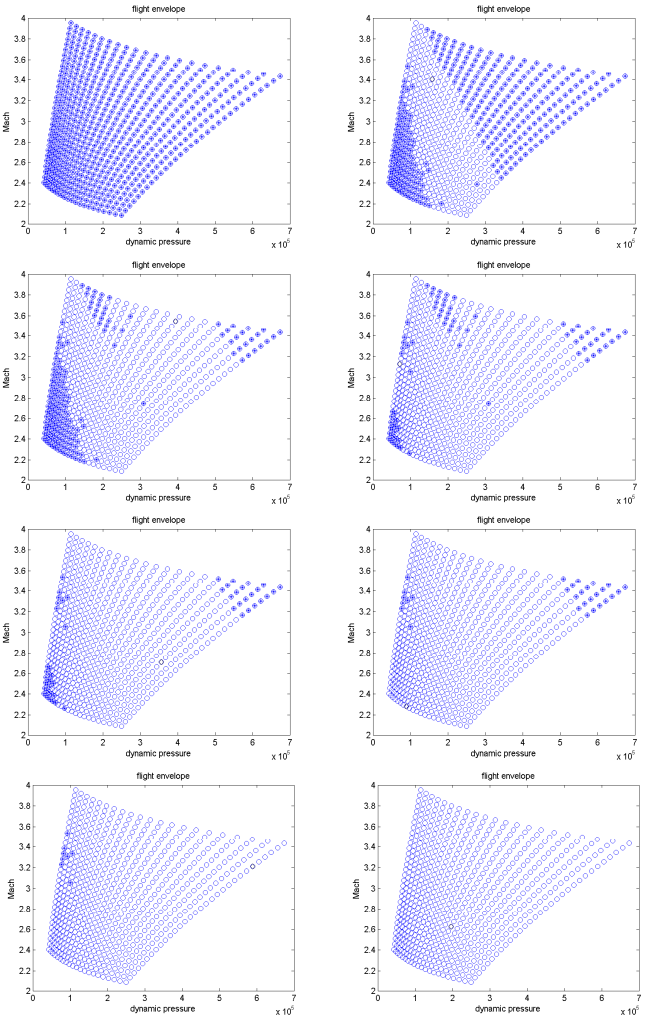

Fig. 5. 7 regions $\mathcal{N}(\mathbf{e}_i)$ which cover $\mathcal{E}$ for which $K_{\text{greedy}}(\mathbf{e})$ respects the 10% performance error margin (2).

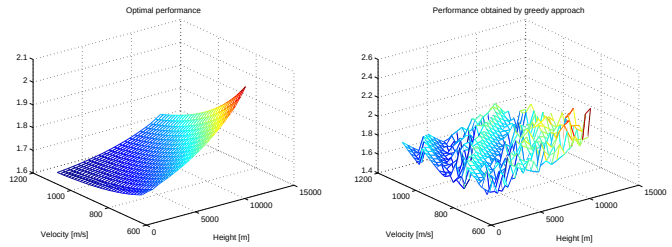

Fig. 6. Performance optimal (left) and its estimation by greedy (right)

gains $k_i^*(\mathbf{e})$ etc. are fitted individually. The resulting controller will be denoted by $K_{\text{int}}(\mathbf{e}) = [\hat{k}_i(\mathbf{e}),\ \hat{k}_p(\mathbf{e}),\ \hat{k}(\mathbf{e})]$. A first idea is to approximate the optimal controller parameters $k_i^*(h, V)$, $k_g^*(h, V)$ $k_p^*(h, V)$ using bilinear expressions:

$$\hat{k}_i(h, V) = a_i + b_i h + c_i V + d_i h V$$

The coefficients a_i, b_i, ... are found using non-linear least squares,

$$\min_{a_i, b_i, c_i, d_i} \sum_{\mathbf{e} \in \mathcal{E}} |k_i^*(\mathbf{e}) - \hat{k}_i(\mathbf{e}; a_i, b_i, c_i, d_i)|^2. \tag{6}$$

The same thing for $\hat{k}_g(h,V)$and $\hat{k}_p(h,V)$. The difficulty here is that approximation needs a tolerance level in each individual gain, while our criterion (2) governs the precision of approximation in closed-loop performance. And indeed, despite a fairly acceptable estimation error in the optimal controller parameters in (6), the approximation $\widehat{K}(\mathbf{e})$ of $K^*(\mathbf{e})$ so obtained performs very badly in the sense that $\|T_{w\to z}(P(\mathbf{e}),\widehat{K}(\mathbf{e}))\|_\infty$ is far from $\|T_{w\to z}(P(\mathbf{e}),K^*(\mathbf{e}))\|_\infty$. Closer inspections shows that the reason for this is the highly nonlinear dependence of the closed-loop performance on the controller parameters. We found that the error in $\hat{k}_p(\mathbf{e})$ was the most important. We therefore decided to approximate $k_p^*(\mathbf{e})$ more accurately, leading to a second approximation $K_{\text{int}}(\mathbf{e})$ where (2) is satisfied. We still use bilinear interpolation for $\hat{k}_i$ and $\hat{k}_g$, but for k_p^* we construct an approximation $\widetilde{k}_p$ with higher accuracy, where the flight envelope $\mathcal{E}$ is divided into $7\cdot 7 = 49$ regions, the grid of variation of altitude and velocity for those regions being

$$h_r = [1500,\ 2025,\ 2550,\ 5175,\ 5700,\ 8325,\ 9375,\ 12000]\ m$$
$$V_r = [700,\ 835,\ 895,\ 910,\ 1030,\ 1075,\ 1135,\ 1150]\ m/s.$$

The $(h_r(i),V_r(i)),\ \ i=1,...,7$ correspond to the coordinates of 7 controllers found by the greedy approach, completed by $(h_r(8),V_r(8))$.
The corresponding parameter values $k_p(h_r(i),V_r(j)),\ i,j = 1,..,8$ form an 8×8 table. For each h and V, 2D interpolation (linear regression) is used to find the corresponding $k_p(h,V)$. Figure 7 shows the improvement in the estimation of closed-loop performance obtained thanks to the regions found by the greedy approach. Condition (3) is satisfied. The gain $\hat{k}_p$ is affine on each of the 49 regions. To conclude, for each point (h,V) in the flight envelope $\mathcal{E}$, the controller $K_{\text{int}}(h,V)$ can be found if $2\cdot 4$ parameters of the bilinear models (for $\hat{k}_i$ and $\hat{k}_g$) and $8\cdot 8 + 2\cdot 8 = 80$ parameters of the table (for $\widetilde{k}_p$). A total of only 88 parameters must be stored.

7. STABILITY

Our control strategy is an extension of [14], where nonlinear plants scheduled at the output are discussed, and from where the concept of frozen system and instantaneous control originates. In that approach the authors obtain sufficient conditions for stability and performance of the nonlinear system, where performance is with regard to the global behavior $w\to z$. Unfortunately, the sufficient conditions in [14] are strong and difficult to check in practice.

In contrast with that classical approach we synthesize the best controller $K^*(\mathbf{e})$ at *every* flight point $\mathbf{e}$, so our design $K^*(\mathbf{e})$ is optimal at every instant. As long as condition (1) holds, this remains approximately true for the three presented practical controllers and has the advantage that the closed-loop system is dissipative in the sense of [6]. This guarantees input-output stability, so that in order to prove internal stability, a property called z-detectability suffices, see [6, Theorem 2.1.3]. The system (P,K) is z-detectable if $w,z\in L^2$ imply $x\in L^2$. While this appears to be just as difficult to verify as the conditions in [14], we believe this condition to be much more intuitive, as it claims some sort of minimality of the model, and therefore augments the plausibility of our approach. In the absence

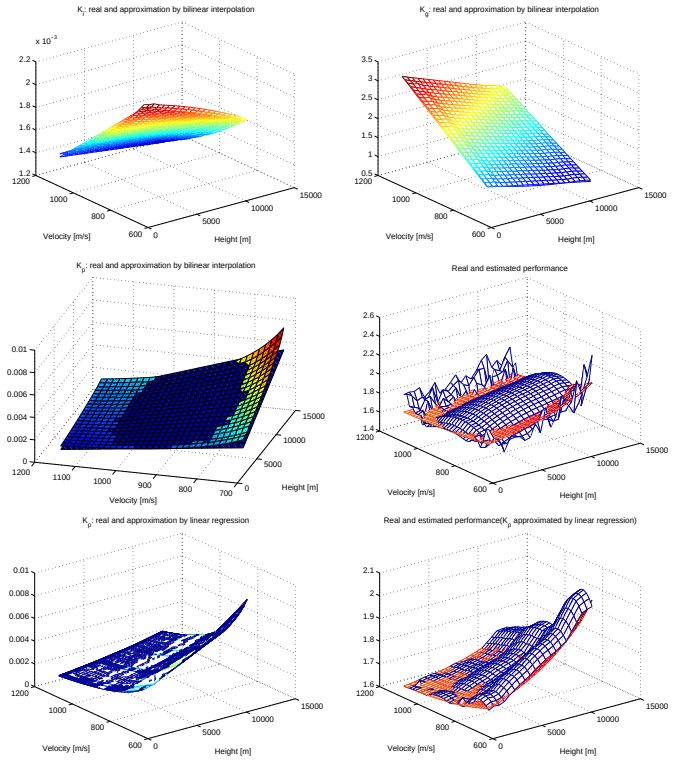

Fig. 7. Approximation of the controller parameters, k_i, k_g and two approximations of k_p. The estimated closed-loop performance is shown in each case.

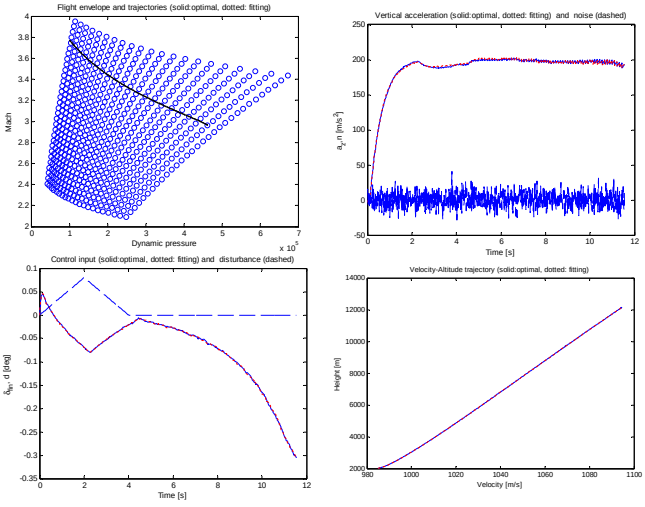

Fig. 8. Vertical acceleration hold for high altitude. K_{int} is used. The parameter trajectories $(M(t),\bar{q}(t))$ are practically identical (upper left). Elevator deflection $\delta_e(t)$ (lower left) and velocity $V(t)$ (lower right) are almost identical. Disturbance in δ_e is shown as dashed line in lower left image, noise in vertical acceleration is shown in upper right image.

of certificates for global stability on $\mathcal{E}$, one has to rely on numerical testing to ensure internal stability. Notice that conservative approaches like parametric stability certificates fail in the present situation. On the other hand, local stability is ensured in each of the three cases by construction.

8. CONCLUSION

We have introduced the pointwise optimal H_∞ PI-I controller $K^*(\mathbf{e})$, $\mathbf{e} \in \mathcal{E}$, which could be understood as the best way to control the system instantaneously in given flight conditions $\mathbf{e} = (h, V)$, if real-time H_∞ control was possible. As this is not the case, we pre-calculated and stored $K^*(\mathbf{e})$ at the 651 points of the flight envelope $\mathcal{E}$. If being too costly to embed, we have proposed three approximations $K_{\text{tri}}(\mathbf{e}), K_{\text{greedy}}(\mathbf{e}), K_{\text{int}}(\mathbf{e})$ of $K^*(\mathbf{e})$, where (a) $K_{\text{tri}}(\mathbf{e})$ is piecewise affine on a triangulation, (b) $K_{\text{greedy}}(\mathbf{e})$ is piecewise constant with or without hysteresis, and (c) $K_{\text{int}}(\mathbf{e})$ uses interpolation of the gain functions. All approximations use the ideal graph $K^*(\mathbf{e})$ as a reference to guarantee an acceptable performance level in closed loop. The resulting controllers have been compared and tested in closed-loop. While little differences occur in performance, the storage requirements vary between (a), (b) and (c).

The greedy controller $K_{\text{greedy}}(\mathbf{e})$ is a good candidate, which needs only 7 exemplars $K^*(\mathbf{e}_i)$, $i = 1, \ldots, 7$ in order to stay within the 10% allowed loss of performance. In exchange, without specification the geometrical form the region, for each one of the 651 points we must store the information concerning h, V and to which of the 7 regions it corresponds, so that we need to store 1953 data. The drawback (if any) of this controller is that the approximation of the performance graph is rather rough.

The controller $K_{\text{tri}}(\mathbf{e})$, obtained by linear interpolation on a triangulation of $\mathcal{E}$, has the advantage of being continuous in $(h, V) \in \mathcal{E}$, which leads to a rather smooth approximation of the performance graph. We need to store 225 data. It may be interesting to elaborate more sophisticated method to construct coarser triangulations requiring less storage.

Finally, the controller $K_{\text{int}}(\mathbf{e})$ obtained by individual approximation or fitting of the gains $k_i^*(\mathbf{e})$, $k_p^*(\mathbf{e})$, $k_g^*(\mathbf{e})$, gives the best reduction in storage. Only 88 informations have to be stored thanks to affine approximation of $k_p^*(\mathbf{e})$ in the flight envelop regions found by the greedy approach. $K_{\text{int}}(\mathbf{e})$ needs 1/22.2 storage as compared to $K_{\text{greedy}}(\mathbf{e})$, 1/2.6 as compared to $K_{\text{tri}}(\mathbf{e})$ and 1/37 compared to $K^*(\mathbf{e})$. This controller is also continuous as a function of (h, V). The closed-loop time based responses in the presence of perturbation and measurement noise show very good accordance with the most accurate controller, $K^*(\mathbf{e})$, the controller which implies storing 37 times plus the data. Thus, in this study, an mixed approach based on the greedy and fitting gives the best results.

ACKNOWLEDGEMENTS

Funding by Fondation de Recherche pour l'Aéronautique et l'Espace under contract *Survol*, and by Fondation EADS under contract *Technicom* is gratefully acknowledged.

REFERENCES

[1] P. Apkarian, P. Gahinet. A convex characterization of gain-scheduled H_∞ controllers. *IEEE Transaction on Automatic Control*, 40:853–864, 1995.

[2] P. Apkarian, P. Gahinet, G. Becker. Self-Scheduled H_∞ control of linear parameter-varying systems: A design example. *Automatica*, 31:1251–1261, 1995.

[3] P. Apkarian, D. Noll. Nonsmooth H_∞-control. *IEEE Transactions on Automatic Control*, 51:71–86, 2006.

[4] G. Becker, R. Mantz. Robust performance of linear parametrically varying systems using parametrically-dependent linear feedback. *Systems and Control Letters* 23 ,205–213, 1994.

[5] Yolanda Bolea, Vicenc Puig and Joaquim Blesa. Gain-scheduled Smith PID controllers for LPV systems with time varying delay: Application to an open-flow canal. *Proc. of IFAC, Korea*, 14564–14569, 2008.

[6] J.W. Helton, M. R. James. Extending H_∞ control to nonlinear systems. *SIAM Advances in Design and Control, Philadelphia*, 1999.

[7] S. D. Jenie, A. Budiyono. Automatic flight control system – classical approach and modern control perspective. *Lecture Notes, Bandung Institute of Technology, Indonesia*, 2006.

[8] T. Keviczky, G.J. Balas. Receding horizon control of an F-16 aircraft: A comparative study. *Control Engineering Practice*, 14,1023–1033, 2006.

[9] B. Lu, F Wu. Switching LPV control design using multiple parameter-dependent Lyapunov functions. *Automatica* 40,1973–1980, 2004.

[10] B. Lu, F Wu, S.W. Kim. Switching LPV control of an F-16 aircraft via controller state reset. *IEEE Trans. On control Systems Technology*, 14(2),267–277,2006.

[11] M. Mattei. Robust multivariable PID control for linear parameter varying systems. *Automatica* 37,1997–2003, 2001.

[12] M. Mattei and V. Scordamaglia. A Full Envelope Small Commercial Aircraft Flight Control Design Using Multivariable Proportional- Integral Control. *IEEE Trans. Cont. Sys. Tech.*, 16 (1), 169–176, 2008.

[13] D. Noll, O. Prot, A. Rondepierre. A proximity control algorithm to minimize nonsmooth and nonconvex functions. *Pacific J. Optim.*, 4(3),569–602, 2008.

[14] J.S. Shamma, M. Athans. Analysis of gain scheduled control for nonlinear plants. *IEEE Transactions on Automatic Control*, 35(4) 898 – 907, 1990.

[15] L. Sonneveldt. Nonlinear F-16 Model Description. *Faculty of Aerospace Engineering, Delft University of Technology, The Netherlands*, *http* : *//www.mathworks.com/matlabcentral /fileexchange/* $11340 - nonlinear - f - 16 - fighter - model$, 2006

[16] B. L.Stevens, F.L.Lewis. Aircraft Control and Simulation. John Wiley & Sons, Inc., 1992.

[MatlabR2010b] Robust toolbox. , MatlabR2010b.

[MatlabR2010b] $airframe\_demo$ of MatlabR2010b. *http* : *//www.mathworks.it/products/demos /shipping/robust/autopilot*$_d$*emo.html*? $product = RC$.