Gain-scheduled two-loop autopilot for an aircraft

Laleh Ravanbod
Dr, Université Paul Sabatier,
Institut de Mathématiques,
Toulouse, France,
Email:LalehRavanbod@yahoo.fr

Dominikus Noll
Professor, Université Paul Sabatier,
Institut de Mathématiques,
Toulouse, France,
Email:d.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.

List of Figures

List of Tables

1 Introduction

We design an attitude 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 an 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 PID 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

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

for every $\mathbf{e} \in \mathcal{E}$, where for instance $\varepsilon = 10\%$. 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 [1] and LMIs. This gives a stability certificate and allows criteria like H_∞ or H_2, see [2–4]. A limitation is that structured controllers like PI-I 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

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

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 [5, 6], reducing conservatism. But even then one has to accept that the LPV approach within PID control has strong limitations. For example: variable parameters are measured precisely, but are not included in the state space [5, 7, 8]. The output matrix is parameter independent and full row rank [9].

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 non-linear 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 3 we discuss the non-linear open-loop model and the simulink file *rct_airframe*1 used in the experiments. In Section 4 we explain how the system is linearized at the flight points $\mathbf{e}$ in the flight envelope $\mathcal{E}$. In Section 5 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 6 – 8. Stability is studied in Section 9. Simulation results follow in Section 10.

Nomenclature

X_e, Y_e, Z_e $[m]$: x,y,z-position w.r.t. earth, $h = -Z_e$ altitude, u, v, w $[m/s]$: longitudinal, lateral and normal velocities, V $[m/s]$: total aircraft velocity, Φ, θ, ψ $[rad]$: roll, pitch, yaw angles, p, q, r $[rad/s]$: angular velocities, α $[rad]$: angle of attack, , β $[rad]$: side slip angle, γ $[rad]$: flight path angle or climb angle, δ_{th}: throttle setting, F_T: thrust, $\delta_{fin} = \delta_e$ $[rad]$ elevator angle, F_x and F_z are the force components along x and z body axes, I_y: moment of inertia about y body axis, M_y : the pitch moment, $\bar{q}$: dynamic pressure, M: Mach number, $\bar{M}$: aerodynamic moment, m: mass, S : wing surface area, $\bar{c}$: wing mean aerodynamic chord, C_{xt}, C_{zt} and C_mt: aerodynamic coefficients which are the function of $\alpha, \beta, p, q, r, \delta_e, \ldots$.

2 Autopilot in longitudinal mode

In longitudinal flight control the term autopilot refers to a flight condition or attitude holding system, sometimes called a displacement autopilot due to its task to restore the state variable to its original desired value. Typical autopilots in longitudinal mode are listed below, see e.g. [10]. The symbols are standard and can for instance be found in [10].

Pitch attitude hold.
In this case $\delta_T = 0$. Via the control of the elevator deflection only the servo gain can still be changed. The vertical gyro sensitivity and servo lag factor are typically constant values.

Speed hold.
Here we have $\delta_e = 0$. The speed difference will be sent to the engine control system (power lever to propulsion control mechanism). The result is a throttle deflection, δ_{th}, applied to the aircraft engine. The aircraft engine in turn changes the thrust of the aircraft by δF_T. This type of flight speed holding system is commonly called an *Auto Throttle System*. The speed hold system is used during the approach and landing in order to reduce the work load of the pilot, who is primarily occupied by the aircraft guidance task. Typically, for a throttle lever system, K_{PC}, the gain can be altered. The only factor that can be varied therefore is the gain associated with engine and propeller, K_e. The gain $K = K_{PC}K_e$ is determined using the root locus.

Altitude hold.
The altitude hold system comprises several feedback loops, including pitch rate hold, forward acceleration and compensator integration. It is a standard for medium and long range transport aircraft and is based on control of the elevator motion. The flight altitude is measured by a pilot static system and the elevator is moved by the basic control mechanism through a servo motor. At first glance this system looks similar to the flight attitude holding system. The difference with the latter is that the flight altitude variable is not part of the aircraft state. From the flight performance analysis, the model of rate of climb can be given as $\frac{dh}{dt} = V_{ss}\sin(\gamma)$, where h is the aircraft altitude with respect to sea level, V_{ss} the aircraft steady state velocity. In the case $\beta = \phi = 0$, $\gamma = \theta - \alpha$, measuring θ and α is sufficient to estimate h.

In this study we will design a two-loop autopilot for controlling the vertical acceleration, where the inner loop includes a pitch rate hold system.

3 Nonlinear aircraft model

We have used a nonlinear aircraft model available in the file *rct_airframe*1 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 [11,12]. For the following see Figure 1.

The nonlinear model consists of the following parts:

Incidence and airspeed
inputs: u, w
outputs: $\alpha = \arctan(\frac{w}{u})$, $V = \sqrt{u^2 + w^2}$

Aerodynamics
inputs: V, α, δ_e, h
outputs: $\bar{q}, \rho, a_x, a_z, \bar{M}, M$.

The following computational steps are performed. h being an input, we obtain temperature as

$$T = T_0 + \Delta T \cdot h, \qquad T_0 = 2.8816\,K, \quad \Delta T = -0.0065\,K/m. \tag{4}$$

Density is then of the form

$$\rho = \rho_0 \exp\left\{-\frac{gh}{RT}\right\}, \qquad \rho_0 = 1.22505, \quad R = 287.04. \tag{5}$$

Next, using (4) and the input V, we obtain the Mach number

$$M = \frac{V}{\sqrt{\gamma \cdot R \cdot T}}, \qquad \gamma = 1.403. \tag{6}$$

The next step is to look up the aerodynamic coefficients $C_{x\alpha} = C_{x\alpha}(\alpha, M)$ in a table organized by α and M. Similarly for $C_{z\alpha} = C_{z\alpha}(\alpha, M)$ and $C_{m\alpha} = C_{m\alpha}(\alpha, M)$. This step is possible because α is an input value.

This done, we use the input δ_e to form the coefficient $C_{xl} = C_{xl0} \cdot \delta_e$, where $C_{xl0} = -1.94806$. Then we obtain the aerodynamic coefficient

$$C_{xt} = C_{x\alpha}(\alpha, M) + C_{xl} = C_{x\alpha}(\alpha, M) + C_{xl0}\delta_e$$

as a function of α, M, δ_e. One proceeds in the same way for $C_{zt} = C_{zt}(\alpha, M, \delta_e)$ and $C_{mt} = C_{mt}(\alpha, M, \delta_e)$.

Now we compute dynamic pressure as

$$\bar{q} = \tfrac{1}{2}\rho \cdot V^2. \tag{7}$$

Finally, we obtain the horizontal acceleration as

$$a_x = \frac{\bar{q} S C_{xt}}{m} = \frac{\bar{q} S}{m}\left(C_{x\alpha} + C_{xl0} \cdot \delta_e\right),$$

the vertical acceleration as

$$a_z = \frac{\bar{q} S C_{zt}}{m} = \frac{\bar{q} S}{m}\left(C_{z\alpha} + C_{zl0} \cdot \delta_e\right),$$

and the aerodynamic moment as

$$\bar{M} = \bar{q} S \bar{c} C_{mt} = \bar{q} S \bar{c}\left(C_{m\alpha} + C_{ml0}\delta_e\right),$$

where the remaining numerical constants are wing surface area S, $\bar{c}$ wing mean aerodynamic chord, I_y the moment of inertia about the y body axis.

Equations of motion
inputs: a_x, thrust F_T, a_z, $\bar{M}$, δ_e
outputs: Solutions of the ODE

$$\begin{aligned}
\dot{u} &= -g\sin(\theta) - qw + a_x + F_T/m \\
\dot{w} &= g\cos(\theta) + qu + a_z \\
\dot{\theta} &= q \\
\dot{q} &= \bar{M}/I_y \\
\dot{X}_e &= u\cos(\theta) + w\sin(\theta) \\
\dot{Z}_e &= -u\sin(\theta) + w\cos(\theta)
\end{aligned} \tag{8}$$

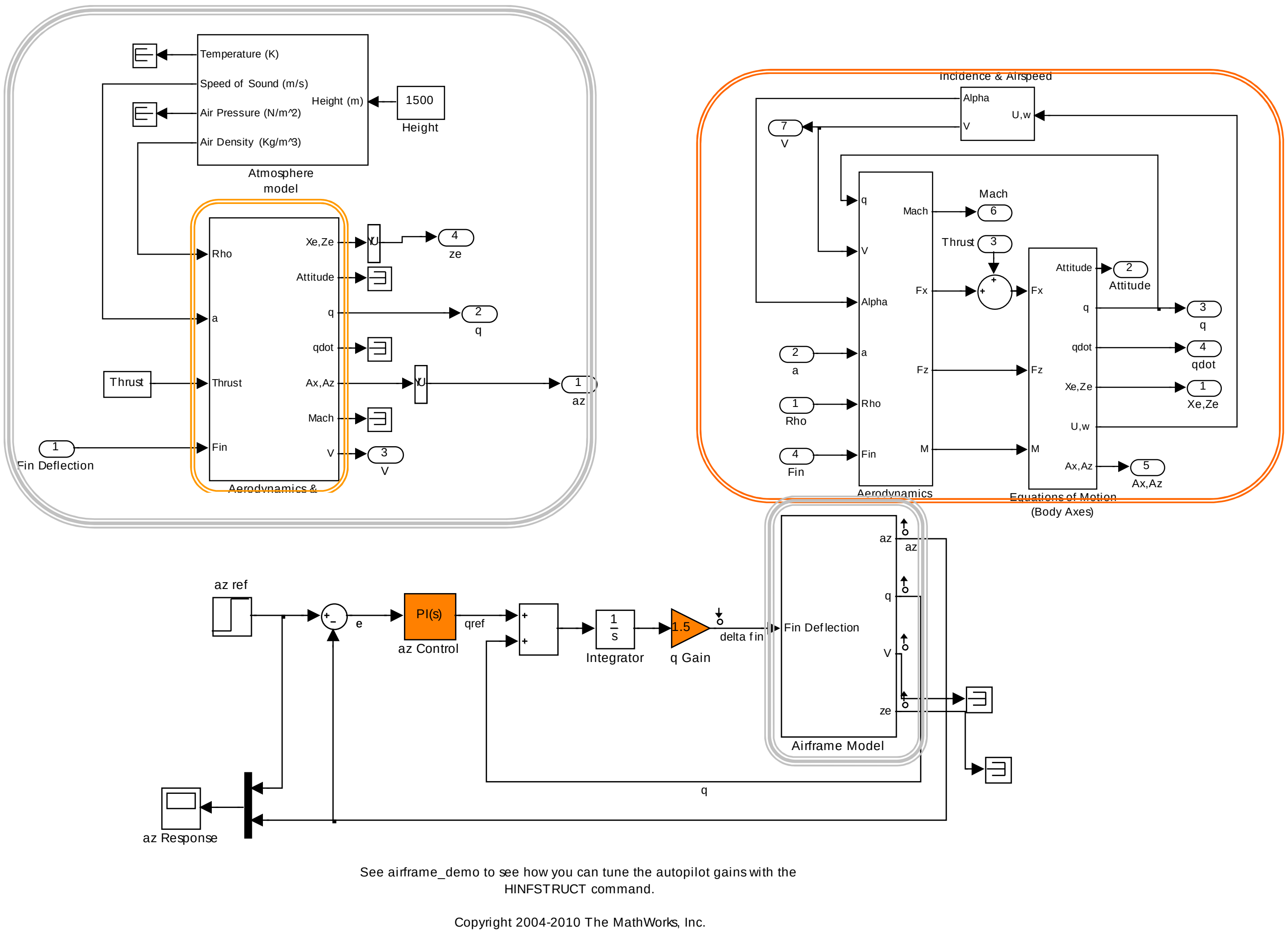


Fig. 1. $rct\_airframe1$ scheme of simulink.

Following the above steps, (8) can be integrated numerically if input data $\delta_e(t)$ and $F_T(t)$ along with initial values are provided.

Remark 1. *It can be shown that the model of rate of climb* $\frac{dh}{dt} = V_{ss}\sin(\gamma)$ *is equivalent to* $\frac{dh}{dt} = u\sin(\theta) - w\cos(\theta)$ *chosen in (8). According to the sign of* Z_e *(positive downward) we have* $Z_e = -h$*, which shows that the model of rate of climb is the same as the model considered in the existing simulink file* $rct\_airframe1$.

Remark 2. *In the present study the thrust* F_T *is held constant at* 1000 *N.*

4 Linearizing the model in closed loop

The longitudinal motion consists of two oscillatory modes: the short period mode, and the long period (phugoid) mode. The short period motion is a well damped, high frequency mode of an aircraft. The variations in velocity are assumed small. Therefore, this mode can be represented by a two degrees of freedom motion instead of the possible five degrees of freedom for longitudinal motion.

The system (8) is linearized at the various flight points $\mathbf{e} = (h,V) \in \mathcal{E}$. The nonlinear 6th order model from $rct\_airframe1$ discussed in section 3 is used to compute a steady-state point, around which the system is then linearized via trimming using the Matlab functions OPERSYSTEM and FINDOP. This leads to a family of second-order models for the short-period longitudinal motion [13, 14]:

$$\begin{bmatrix} d\dot{\alpha} \\ d\dot{q} \end{bmatrix} = \begin{bmatrix} A_{11}(h,V) \ A_{12}(h,V) \\ A_{21}(h,V) \ A_{22}(h,V) \end{bmatrix} \begin{bmatrix} d\alpha \\ dq \end{bmatrix} + \begin{bmatrix} B_{21}(h,V) \\ B_{22}(h,V) \end{bmatrix} d\delta_e. \tag{9}$$

Remark 3. *Regarding small variations of V, if not directly available through sensors, one can synthesize an observer for V. Alternatively, the following relation between aircraft speed V and Mach number can be used to estimate V, see e.g. [14]: $V = M\left[\gamma r \frac{T_{sta}}{1+(\gamma-1)/2M^2}\right]^{0.5}$, where T_{sta} is the static temperature (available on-board) and γ and r are known air data quantities. Therefore, in the following we assume that, in addition to the altitude h, the speed V is also available.*

In our study we have chosen a rectangular grid in the (h,V)-plane

$$h \in [1500, 12000]\, m, \quad \Delta h = 525\, (\cong 21 \text{ steps}),$$

$$V \in [700, 1150]\, m/s^2, \quad \Delta V = 15\, (\cong 31 \text{ steps}),$$

leading to a total of $21 \cdot 31 = 651$ flight points $\mathbf{e} = (h,V)$ forming the flight envelope $\mathcal{E}$. The rectangular (h,V)-geometry is deformed into the curved region $\mathcal{E}$ shown on the left of Figure 3 via the nonlinear mapping $(h,V) \mapsto (M,\bar{q})$ given by (4) – (7).

The trimming procedure for linearization about a flight point $\mathbf{e} = (h,V)$ in the flight envelope $\mathcal{E}$ considers the system in closed loop with a stabilizing controller $K^\sharp$, 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 2) is $k_p + \frac{k_i}{s} = 0.003 + \frac{0.01}{s}$, the static q-gain is $k_g = 1.5$. The resulting linearized system is nonetheless independent of $K^\sharp$.

For synthesis the parameter-varying model (9) has to be completed into a plant $P(\mathbf{e}) = P(h,V)$ by adding disturbances, 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.

5 H_∞ control

The H_∞-control scheme used to synthesize a gain-scheduled controller is shown in Figure 2 (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 $azref$ 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

$$LS(s) = \frac{1 + 0.001\frac{s}{\omega_c}}{0.001 + \frac{s}{\omega_c}}$$

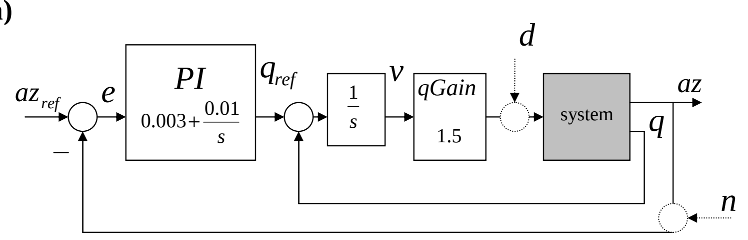
is used. It can be shown that if the peak gain of the closed-loop transfer from w to z is close to 1, then

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 $azref$ step command is small;
The gain from d to az does not exceed $m = 1000$.

To fix the filter k we have used the specific flight point $h = 3050\, m$ and $V = 984\, m/s$, where the peak gain from d to az 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 $w = (az_{\text{ref}}, md, LS(s)n)$ and $z = (LS(s)e, az)$ 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 [15], which is based on the fundamental work [16]. The rationale of HINFSTRUCT relies on non-smooth optimization and can be found in [16] or [17]. In order to increase the precision we have used 18 initializations. For details on the use of HINFSTRUCT see [18].

(a)



(b)

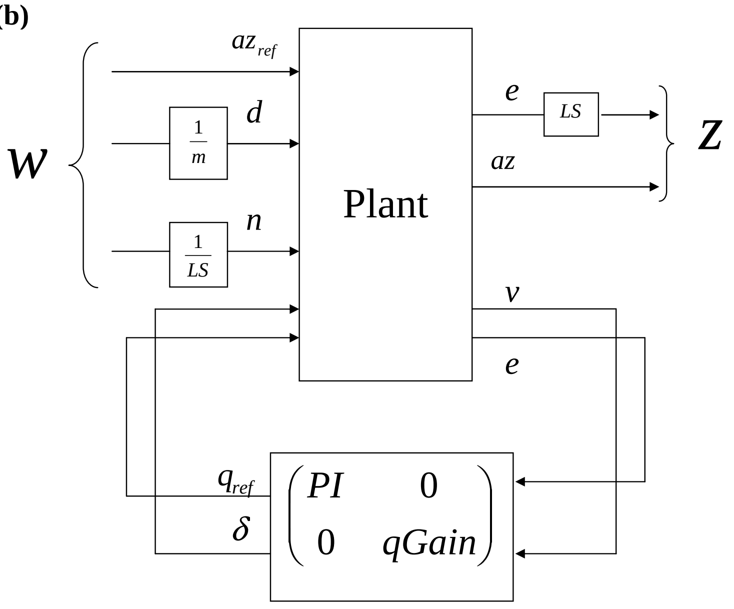


Fig. 2. Schemes used for (a): linearizing the non linear aircraft in closed loop, (b) H_∞ synthesis

The closed-loop performance graph $(h,V) \mapsto \|T_{w\to z}(P(h,V),K^*(h,V))\|_\infty$ is shown on the right of Figure 3. The graphs of $k_i^*(h,V)$, $k_p^*(h,V)$ and $k_g^*(h,V)$ are shown in Figure 9. These functions are only available numerically.

Using the optimal controller $K^*(\mathbf{e})$ would require storing 651×5 numerical values (3 gains for each (h,V) and h,V themselves), plus the rules to look values up in the table. Our goal is therefore to find parametrizations $K(\mathbf{e})$ 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 $\varepsilon = 10\%$. We subsequently describe three approaches along this line, 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 $K_{\text{int}}(\mathbf{e})$.

6 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 $\mathbf{e}_i = (h_i,V_i)$, $\mathbf{e}_j = (h_j,V_j)$, $\mathbf{e}_k =$

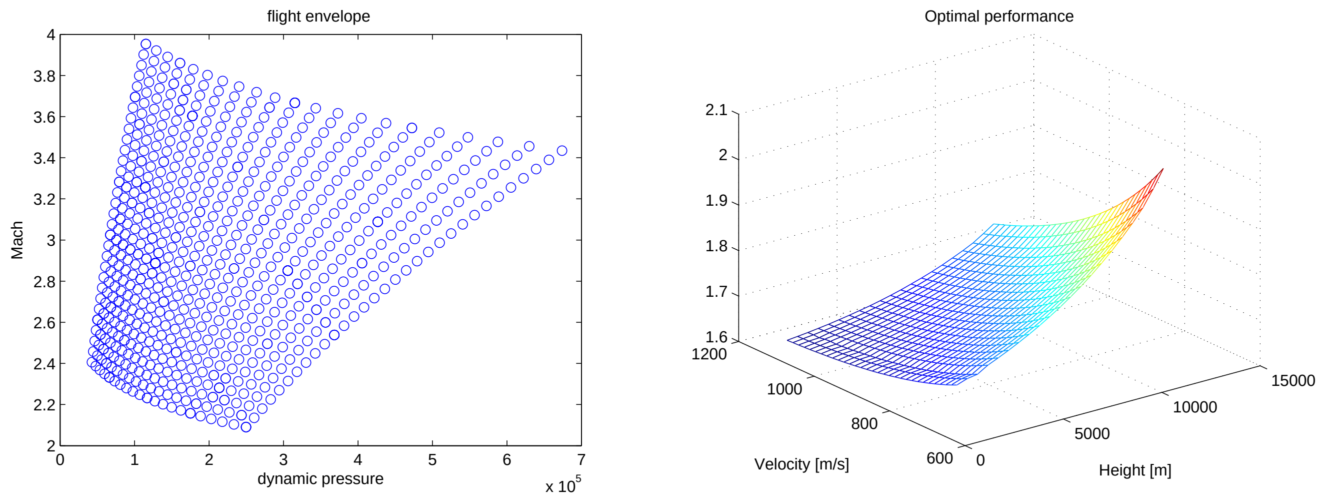


Fig. 3. Left: flight envelope in geometry $(M,\bar{q})$. Right: optimal H_∞ performance over flight envelope geometry $\mathbf{e}=(h,V)$.

(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

$$\begin{bmatrix} a\\ b\\ c\end{bmatrix}=Q^{-1}\begin{bmatrix} k_p(\mathbf{e}_i)\\ k_p(\mathbf{e}_j)\\ k_p(\mathbf{e}_k)\end{bmatrix}\ with\ Q=\begin{bmatrix} h_i & V_i & 1\\ h_j & V_j & 1\\ h_k & V_k & 1\end{bmatrix}. \tag{10}$$

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 4 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. To conclude, for each point (h,V) in the flight envelope, the controller $K_{tri}(h,V)$ can be found if $45\cdot 5=225$ data and an algorithm to find to which triangle h,V belongs are stored. Here 5 corresponds to altitude, velocity and 3 controller parameters.

Remark 4. *Triangulation under constrains is a fundamental problem in the representation of objects. We mention constrained Delaunay triangulation or triangulation guaranteeing a piecewise linear regression approximation of a non-linear projection [19]. Our problem here is more involved, as we search for a triangulation guaranteeing a maximum variation (10%) of a non-linear objective, the system performance. This criterion is well-suited to compute local refinements of a given coarse triangulation in order to assure the desired loss of performance of less than 10%. However, it does not easily lend itself to create such an initial coarse triangulation, so in this phase our method depends on some trial and error steps.*

In [20] a related approach based on a (regular) triangulation of the flight envelope is developed. The authors treat the problem by a receding horizon model-based predictive control method.

7 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:

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

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 $\mathbf{e}\in\mathcal{N}(\mathbf{e})$, 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

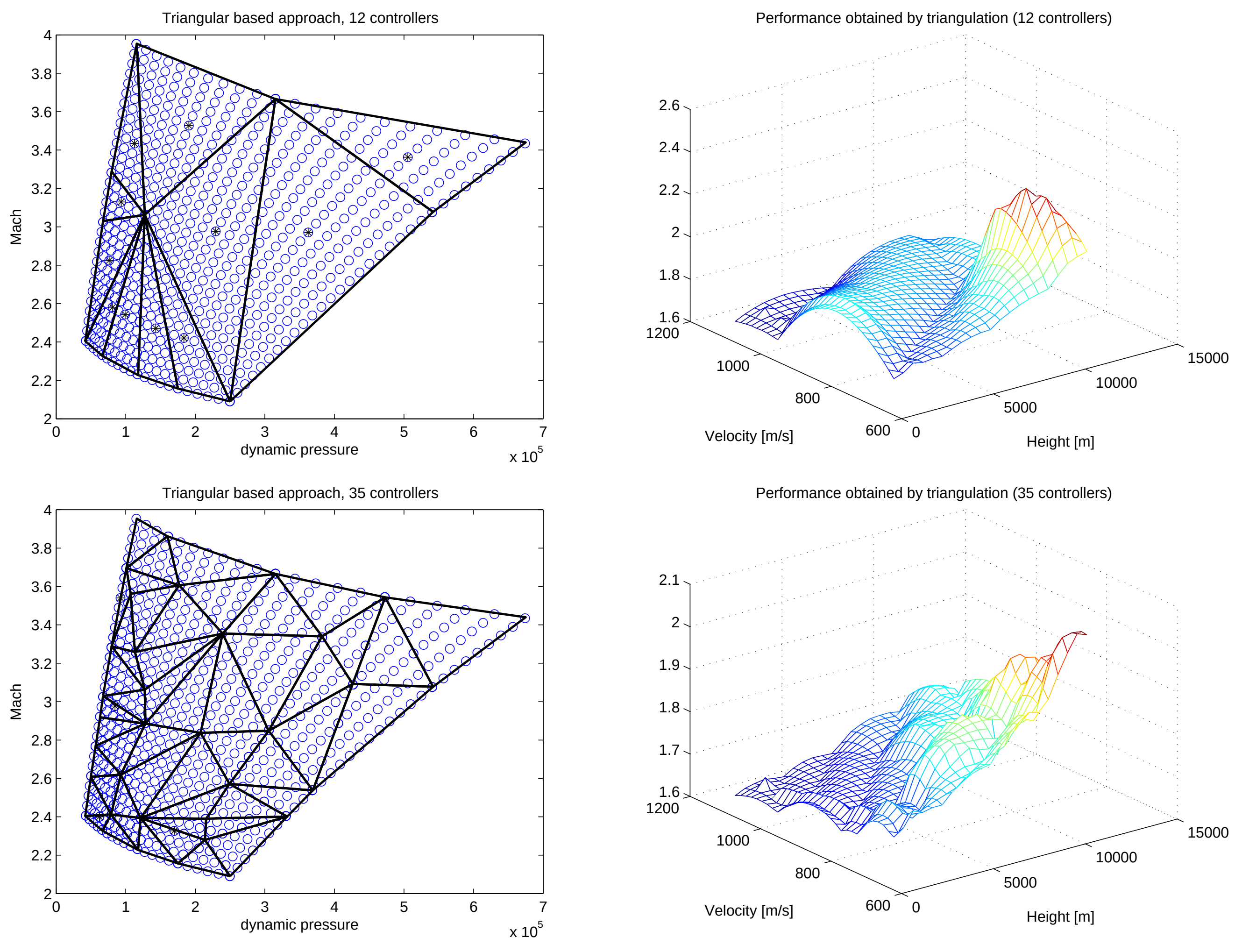


Fig. 4. Two triangulations together with the performance graphs.

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. Figure 5 shows 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 7, 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 searched by greedy (as in our case), for each $\mathbf{e}_i$ point, $i = 1, \dots, 651$ of flight envelope, in addition to its altitude and velocity informations, we must also store to which of the 7 regions it belongs to. Concerning an arbitrary point h, V in flight envelope, it can be surrounded at most by 4 points $\mathbf{e}_i$, $\mathbf{e}_j$, $\mathbf{e}_k$, $\mathbf{e}_l$ where $i, j, k, l \in \{1, \dots, 651\}$ belonging to at most to 4 different regions. Evaluation of $K_{greedy}(h, V)$, hence, involves a 2D interpolation algorithm. Therefore, $(651 - 7).3 + 7.3 = 1953$ data and a 2D interpolation algorithm must be stored.

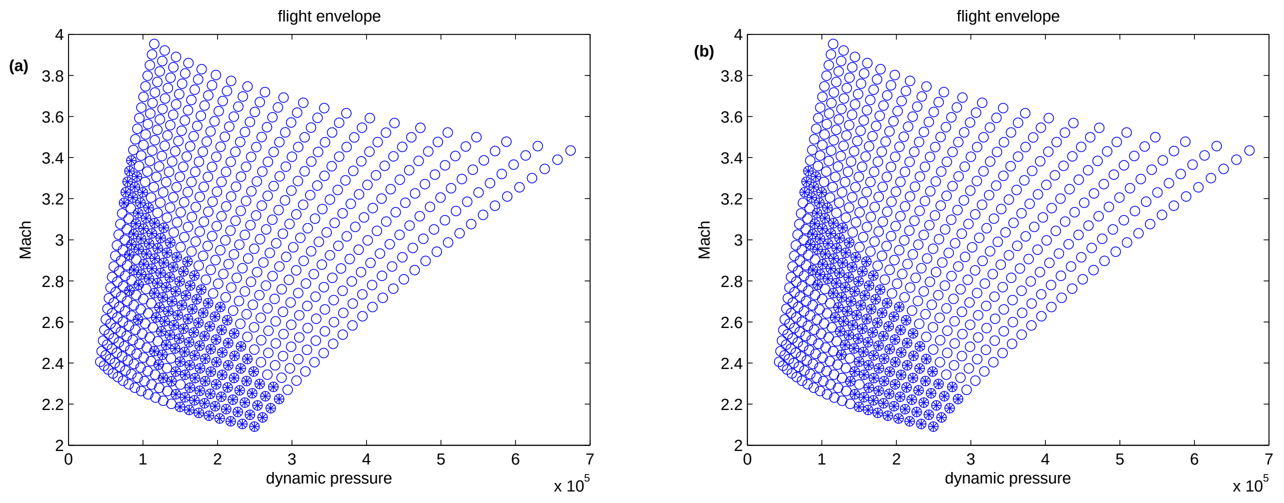


Fig. 5. Example of pre-processing of the regions $\mathcal{N}(\mathbf{e})$. (a) without smoothing , (b) with smoothing.

	a	b	c	d
k_p	4e-3	8.9e-7	-3e-6	-7e-10
k_i	3.4e-3	-6.7e-8	-1.8e-6	4.9e-11
k_g	-1.7	1.14e-5	4.3e-3	-9.6e-8

Table 1. Parameters of bilinear models used for the first approximation

Remark 5. *Notice that the greedy heuristic is sub-optimal and could be replaced by more sophisticated heuristics [21] in order to reduce the number of regions needed to cover $\mathcal{E}$.*

8 The fitting method

One may consider $K^*(\mathbf{e})$ itself as a valid controller parametrization, with the drawback that it needs storage of $651 \cdot 5$ numbers. 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 gains $k_i^*(\mathbf{e})$ etc. are fitted individually. The resulting controller will be denoted $K_{\text{int}}(\mathbf{e})$.

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:

$$\begin{aligned}\hat{k}_i(h,V) &= a_i + b_i h + c_i V + d_i hV \\ \hat{k}_g(h,V) &= a_g + b_g h + c_g V + d_g hV \\ \hat{k}_p(h,V) &= a_p + b_p h + c_p V + d_p hV\end{aligned} \tag{11}$$

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

$$\min_{a_i,b_i,c_i,d_i} \sum_{\mathbf{e}\in\mathcal{E}} |k_i^*(\mathbf{e}) - \hat{k}_i(\mathbf{e};a_i,b_i,c_i,d_i)|^2, \tag{12}$$

solved e.g. by the Matlab function LSQCURVEFIT, and similarly for the other scheduled gain functions. The results are given in the Table 1.

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 (11), (12), the approximation $\widehat{K}(\mathbf{e})$ of $K^*(\mathbf{e})$ so obtained performs very badly in the sense

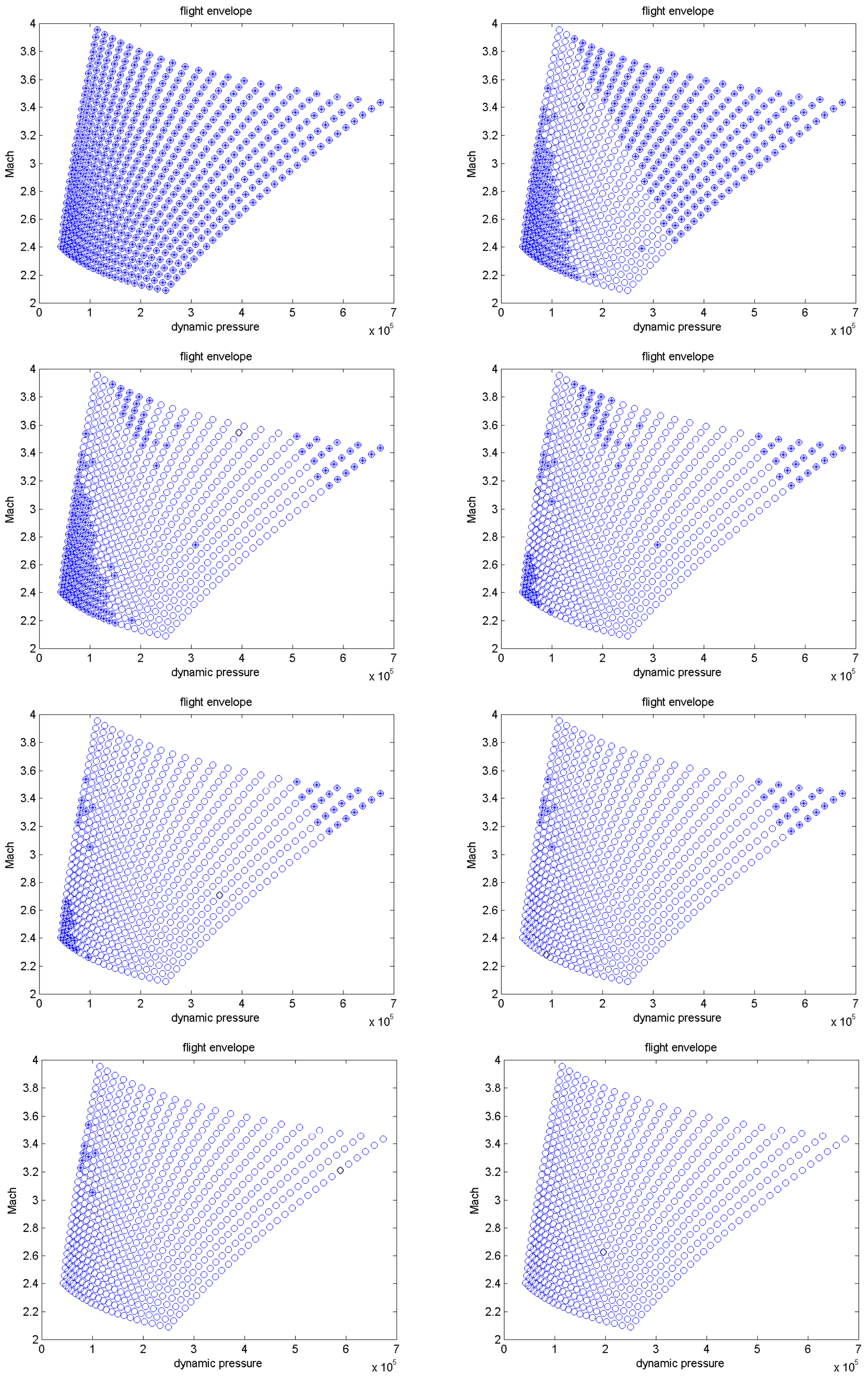


Fig. 6. 7 regions $\mathcal{N}(\mathbf{e}_i)$ which cover $\mathcal{E}$, performance error margin is respected in(2).

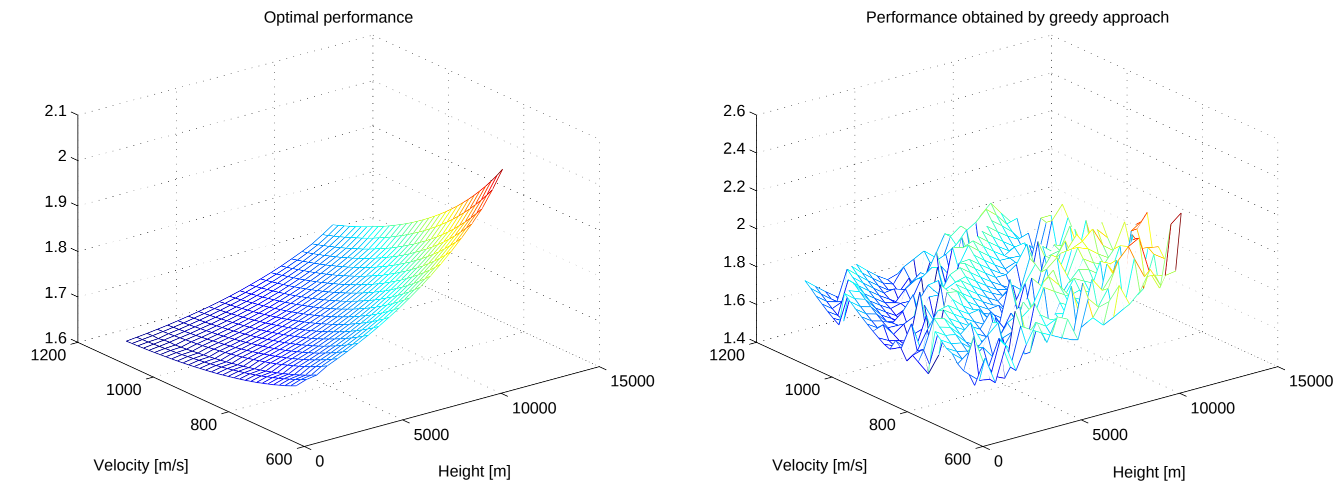


Fig. 7. Performance optimal (left) and its estimation by greedy (right)

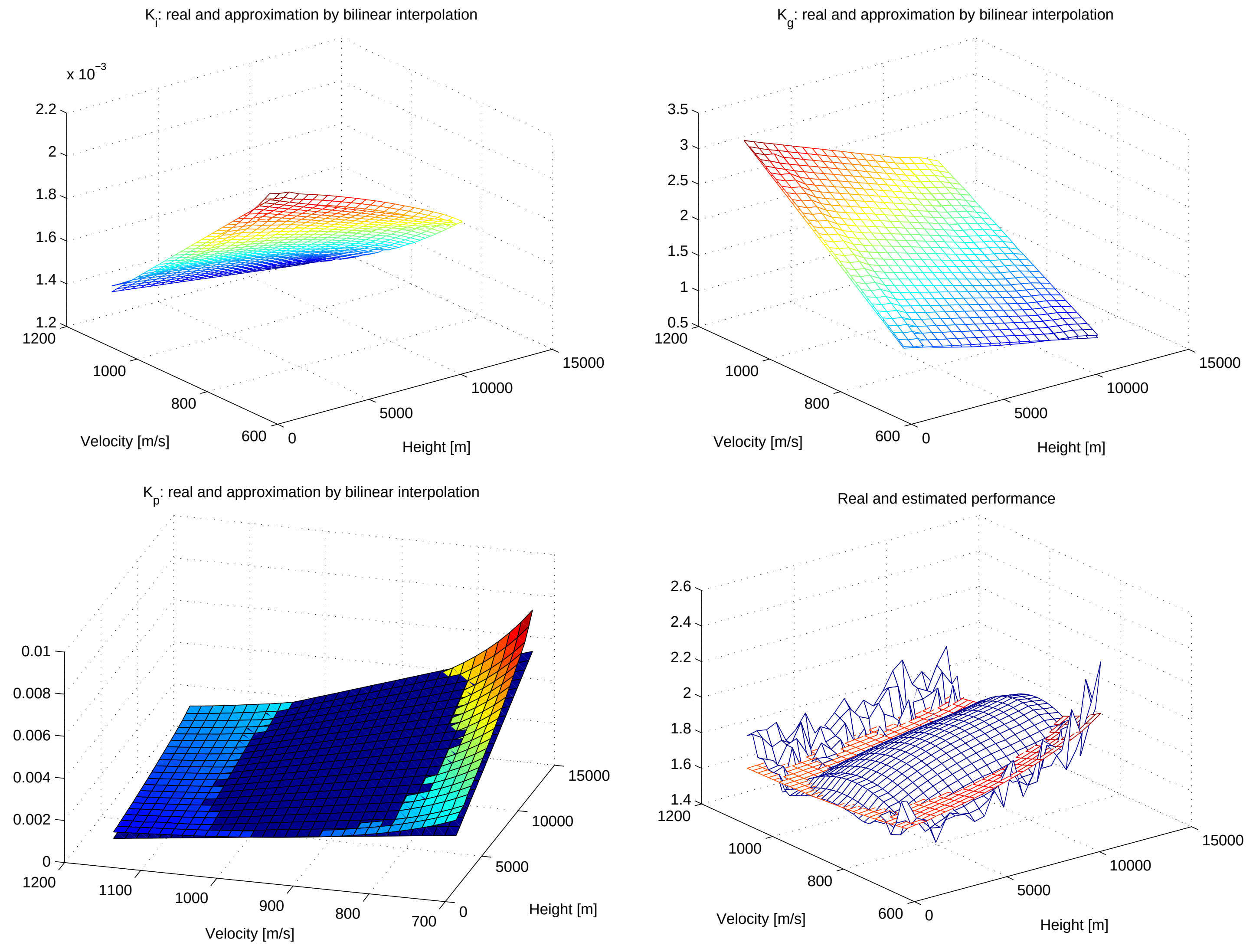


Fig. 8. First approximation of the controller parameters and the performance obtained

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 $\|T_{w\to z}(P(\mathrm{e}),K_{\mathrm{int}}(\mathbf{e}))\|_\infty$ 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 $K_{\mathrm{int}}(\mathbf{e})$ 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 7.7 = 49 regions, the grid of variation of altitude and velocity for those regions being

$$h_r = [1500,\, 2025,\, 2550,\, 5175,\, 5700,\, 8325,\, 9375,\, 12000]\; m,$$

$$V_r = [700,\, 835,\, 895,\, 910,\, 1030,\, 1075,\, 1135,\, 1150]\; m/s.$$

In the $(h_r(i),V_r(i))$, $i = 1,\dots,7$ correspond to the coordinate of 7 controllers found by the greedy approach, completed by $h_r(8) = 12000\,m$ and $V_r(8) = 1150\,m/s$.
The corresponding parameter values $k_p(h_r(i),V_r(j))$, $i = 1,..,8$, $j = 1,...,8$ form an 8×8 table. For each h and V, 2D interpolation (INTERP2 of Matlab with option:linear) is used to find the corresponding $k_p(h,V)$. Figure 10 shows the improvement in the estimation of closed-loop performance so obtained. The gain $\hat{k}_p$ is affine on each of the 49 regions.

To conclude, for each point (h,V) in the flight envelope, the controller $K_{\mathrm{int}}(h,V)$ defined by $\hat{k}_p(h,V),\hat{k}_i(h,V),\widetilde{k}_g(h,V)$ is found using 2.4 parameters of the bilinear models (for $\hat{k}_i$ and $\hat{k}_g$), $8\cdot 8+2\cdot 8 = 80$ parameters of the table (for $\widetilde{k}_p$) and a 2D interpolation algorithm. A total of only 88 parameters must be stored.

Remark 6. *Even though the values for the parameter d in Table 1 are small compared to those of the other parameters like for instance a, we may not conclude that d could be neglected. In fact, the terms $d_{i,g,p}hV$ of the bilinear model, when evaluated for large values of h and V, will have values comparable in magnitude to $a_{i,g,p}$, $b_{i,g,p}h$ or $c_{i,g,p}V$. For example, theses values, in the case of $h = 12000$, $V = 1150$ and k_p model, are respectively:* 0.0097*,* 0.004*,* 0.0107 *and* 0.0034*.*

9 Stability

Our control strategy is an extension of the [22], 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 [22] 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 practical controllers $K(\mathbf{e})$, and has the advantage that the closed-loop system is dissipative in the sense of [23]. This guarantees input-output stability, so that in order to prove weak internal stability, a property called z-detectability suffices, see [23, 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 [22], 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. Nonetheless, in the absence of certificates based on conservative approaches like quadratic stability, which fail in the present situation, one has to rely on numerical testing to ensure weak internal stability of the different controllers $K(\mathbf{e})$.

9.1 Numerical test for weak internal stability

Expanding on an idea in [23], suppose the plant $P(\mathbf{e})$ is available at every $\mathbf{e} \in \mathbb{R}^2$, and that the ideal gain-scheduled instantaneous H_∞ optimal controller $y = K^*(\mathbf{e})u$, parametrized by the linearized system in each $\mathbf{e} \in \mathcal{E}$, is available at every instant $\mathbf{e}(t)$ in closed-loop. Then for a parameter trajectory $\mathbf{e}(t)$, $z(t) = \mathcal{F}_\ell\left(P(\mathbf{e})(t),K^*(\mathbf{e})(t)\right)w(t)$ is in fact the response of the true nonlinear system in closed loop to the input signal $w(t)$.

The ideal performance graph $\mathbf{e} \mapsto \|T_{w\to z}\left(P(\mathbf{e}),K^*(\mathbf{e})\right)\|_\infty$, shown in Figure 7 (left), is now significant, because for every trajectory $\mathbf{e}(t)$, $z(t) = \mathcal{F}_\ell\left(P(\mathbf{e}(t)),K^*(\mathbf{e}(t))\right)w(t)$, we have dissipativity $\|z\|_2 \le \gamma\|w\|_2$ of the nonlinear system, where $\gamma = \max_t \|T_{w\to z}\left(P(\mathbf{e}(t)),K^*(\mathbf{e}(t))\right)\|_\infty$ for the trajectory chosen. In fact, inspecting the performance graph of the ideal K^* or any of its approximations K_{try}, K_{greedy} or K_{int} gives then a visual interpretation of the dissipativity in closed loop. The performance shown in the figures 3, 4, 7 and 10 therefore confirms input-output stability, hence dissipativity, visually.

However, in order to show that the nonlinear closed-loop system is stable, a second property is needed. Following [23], the nonlinear closed-loop system is weakly internally stable as soon as it is also z-detectable, which means $w,z \in L^2$ implies

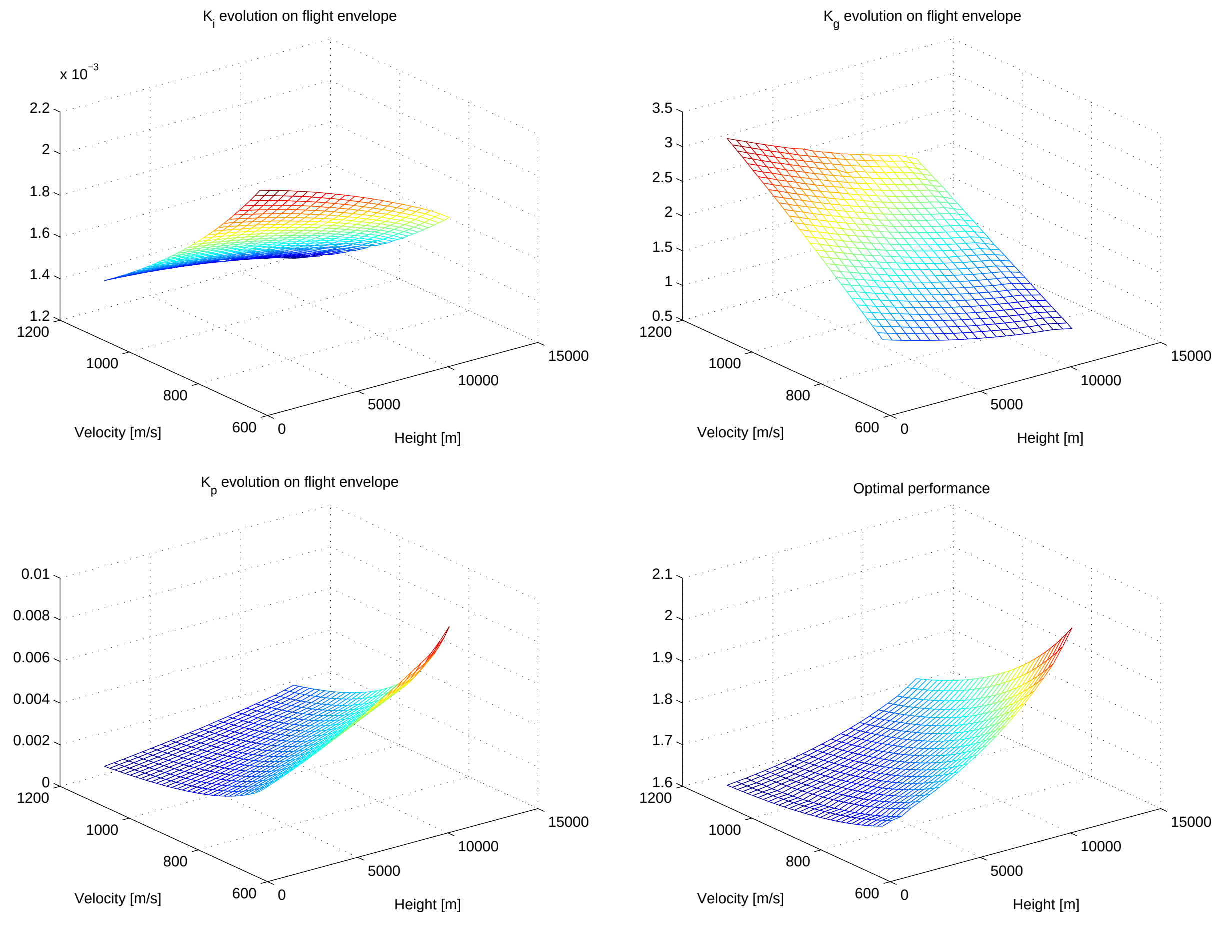


Fig. 9. Graphs of the optimal gains $k_i^*(\mathbf{e})$, $k_p^*(\mathbf{e})$ and $k_g^*(\mathbf{e})$ and the optimal closed-loop performance (lower right) displayed over $\mathcal{E}$.

$x \in L^2$ for any input w, output z and the corresponding state x. This property is in fact slight weaker than input-to-state stability.

While z-detectablity is difficult to check and still requires computation of a Lyapunov function, a difficult task in practice, it helps to gain a more intuitive understanding. In fact, z-detectability implies that each of the frozen systems is detectable. Computing the detectability constant $C(\mathbf{e})$ at every flight point $\mathbf{e} \in \mathcal{E}$, that is, the smallest constant $C(\mathbf{e})$ satisfying $\|x\|_2 \leq C(\mathbf{e})\,(\|w\|_2 + \|z\|_2)$, a necessary condition for z-detectablity is boundedness of $C(\mathbf{e}(t))$ along the trajectories. We can now visualize the instantaneous detectability of the system by inspecting the graph of $C(\mathbf{e})$ over the flight envelope. Note that $x = (sI - A_{cl})^{-1} b_{\text{cl}} w$ at instant $\mathbf{e}$, while the norm $\|z\|_2$ is already controlled by $\|w\|$ via dissipativity. That means, it suffices to plot the H_∞-norm graph $\mathbf{e} \mapsto \|(sI - A_{cl}(\mathbf{e})^{-1} B_{cl}(\mathbf{e})\|_\infty$ over $\mathcal{E}$ to see whether there are regions where the amplification factor is high, indicating the loss of detectability and possibly the risk of instability.

Figure 11 shows indeed that $\mathbf{e} \mapsto \|(sI - A_{cl}(\mathbf{e})^{-1} B_{cl}(\mathbf{e})\|_\infty$ remains bounded over $\mathcal{E}$, confirming weak internal stablity, even though a tendency for higher cost is clearly observed for flight with low velocity at large height, indication that detectablitity should be harder to realize in this region.

10 Simulation of controllers in closed-loop

The final step is to evaluate and compare the scheduled PI-I controllers in closed-loop. As $K_{\text{int}}(\mathbf{e})$ needs the least storage, we have compared it to $K^*(\mathbf{e})$. Here the full model (8) is used. We compare time domain responses to a step input in az_{ref} for $K^*(\mathbf{e})$ and $K_{\text{int}}(\mathbf{e})$. Application of any control law creates a trajectory $(h(t), V(t))$ respectively $(M(t), \bar{q}(t))$ within the flight envelope $\mathcal{E}$. See Figures 12 and 13 upper left. The simulation uses a time step $\Delta t = 0.1$ seconds for the plant $P(\mathbf{e}(t))$, while the time step for $K(\mathbf{e}(t))$ was chosen as 0.5 seconds. For flight points not in $\mathcal{E}$ linear interpolation is used. The simulation considers noise in the vertical acceleration (shown in the upper right images), and a disturbance of the elevator deflection $\delta_e(t)$ shown in the lower right image.

Figures 12 and 13 also illustrate the resulting differences in the time domain responses, where $K^*(\mathbf{e})$ and $K_{\text{int}}(\mathbf{e})$ are compared. The difference is the most sizable in high altitude and low speed. The latter is not entirely surprising, because as can be seen in Figure 10, the approximation error of the closed-loop performance is not homogeneous over the flight

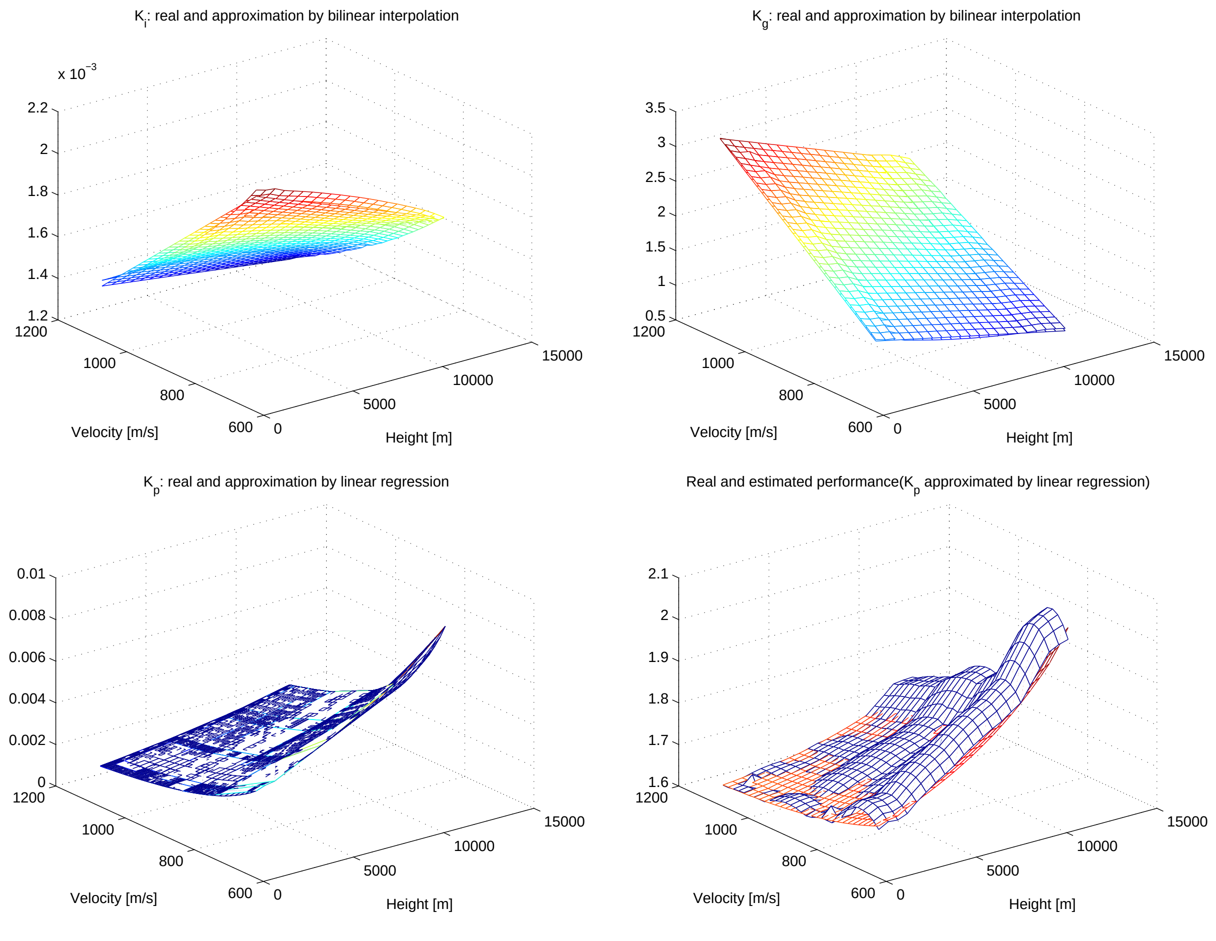


Fig. 10. Second approximation of the controller parameters and its performance.

envelope $\mathcal{E}$. It is largest at high altitude and low speed. Neither is the cost function itself constant over $\mathcal{E}$. The highest cost occurs for flight with high altitude and low velocity, which in some sense corroborates the fact that flying in these conditions is difficult. For instance, a performance graph like in Figure 15, obtained in a different situation, would certainly be a strong incentive for a loss of stability, to be expected when the trajectory $\mathbf{e}(t)$ entered the region of flight at low altitude and high speed.

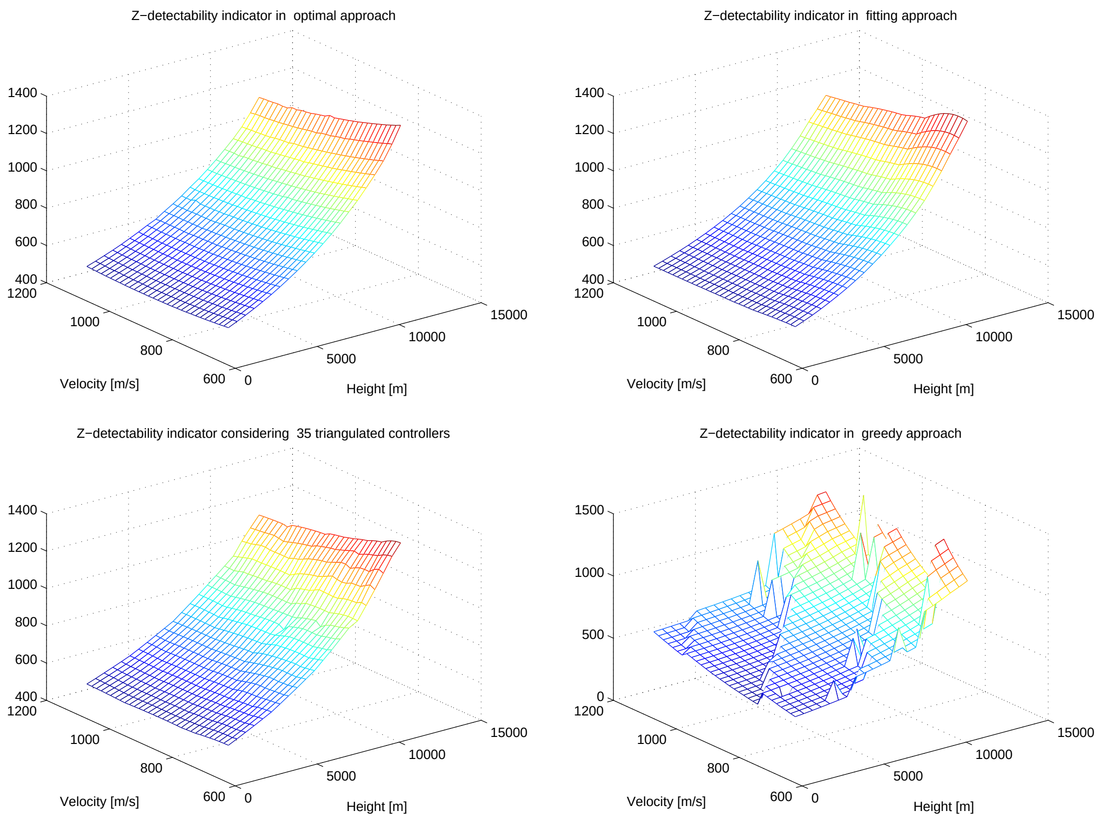


Fig. 11. Detectability rate, respectively $\mathbf{e} \mapsto \|(sI - A_{cl}(\mathbf{e}))^{-1}B_{cl}(\mathbf{e})\|_\infty$, visualized over the flight envelope $\mathcal{E}$. Ideal K^* upper left, K_{int} upper right, K_{tri} (35 controllers) lower left, and K_{greedy} lower right.

11 Conclusion and discussion

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 $\mathbf{e} = (h, V)$. This controller would be used if real-time H_∞ control was possible.

As this is not the case, we pre-calculated and stored $K^*(\mathbf{e})$ at the 651 flight points of the flight envelope $\mathcal{E}$. As this is still 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 of the geometrical form of the region, we must store for each of the 651 points the coordinates h, V and information to which one of the 7 regions it corresponds, so that we need in total 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 pieces of information have to be stored thanks to the affine approximations of $k_p^*(\mathbf{e})$ in the flight envelope regions found by the greedy approach. $K_{\text{int}}(\mathbf{e})$ needs only $1/22.2$ of the storage required for $K_{\text{greedy}}(\mathbf{e})$, $1/2.6$ of that for $K_{\text{tri}}(\mathbf{e})$, and $1/37$ of the storage needed for $K^*(\mathbf{e})$. The controller $K_{\text{int}}(\mathbf{e})$ has the further advantage of being continuous in (h, V). The closed-loop time responses in the presence of perturbation and measurement noise show very good accordance with the most accurate controller, $K^*(\mathbf{e})$, the controller which requires 37 times more data to store. In consequence, the mixed approach based on the greedy and fitting gives the best results in the present study.

We mention that performance and robustness filters could at any moment be chosen to be parameter dependent. While

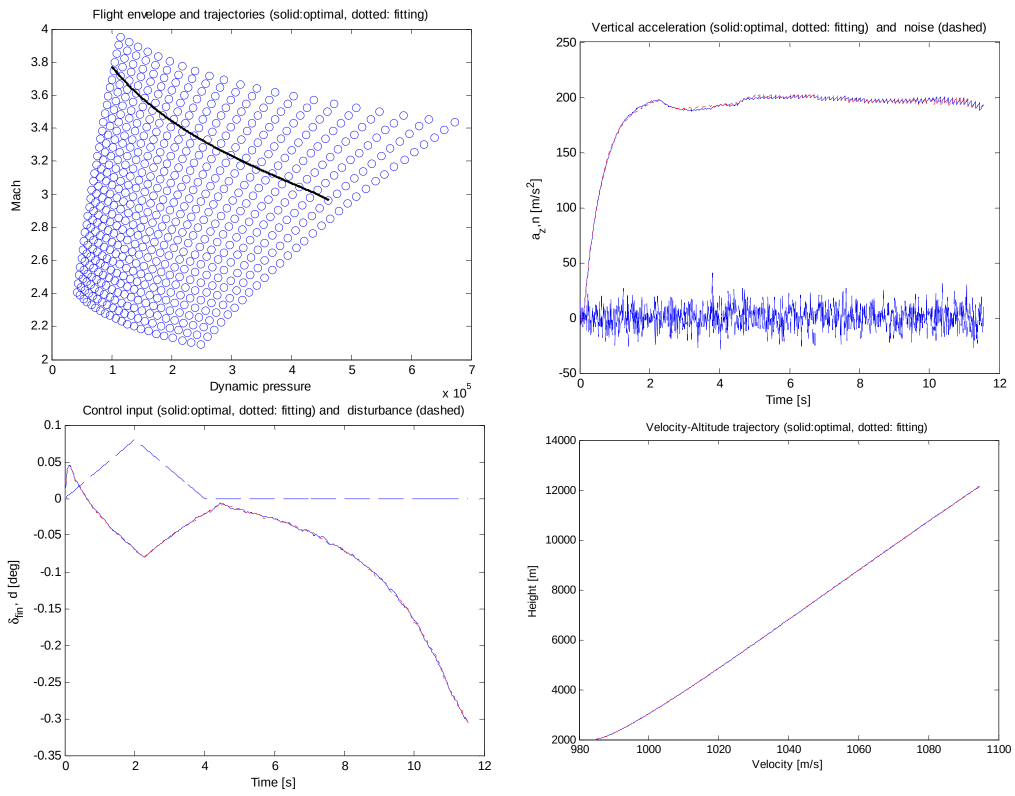


Fig. 12. Vertical acceleration hold for high altitude.

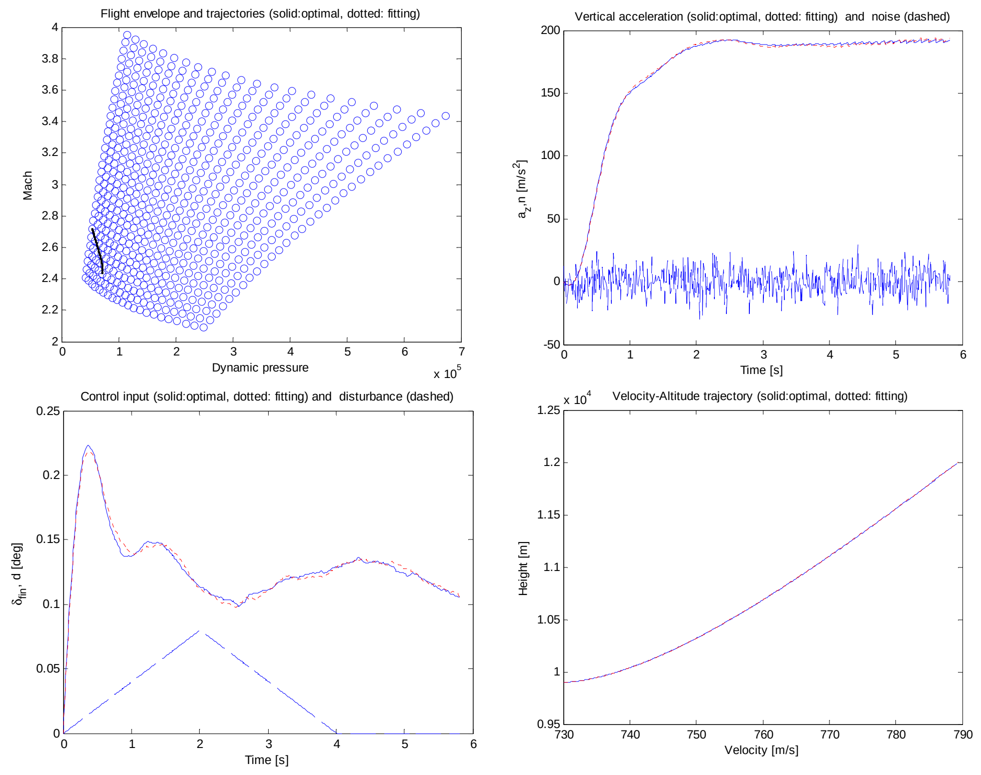
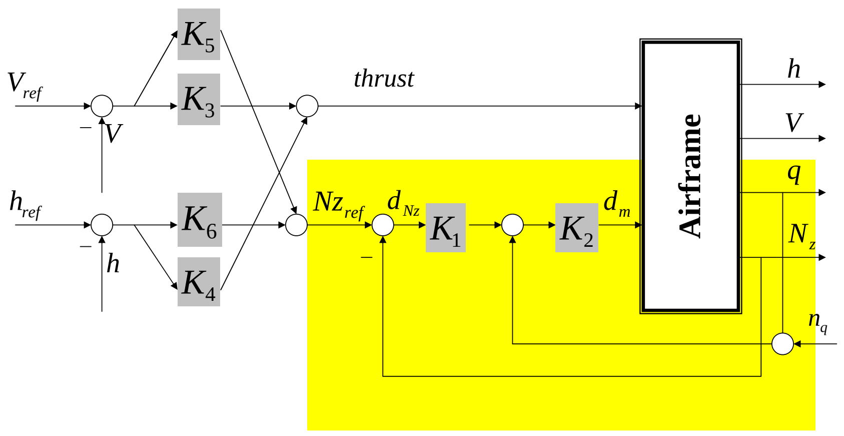


Fig. 13. Vertical acceleration hold for low altitude.



$$K_1 = Kp_1 + Ki_1 / s \qquad K_2 = g_1 /(s^2 + a_1 s + b_1) \qquad K_3 = Kp_2 + Ki_2 / s$$
$$K_4 = g_2 /(s + b_2) \qquad K_5 = g_3 \qquad K_6 = Kp_3 + Ki_3 / s + Kd_3.s /(T_f s + 1)$$

Fig. 14. Control scheme for flight control.

the present study choses n and LS in Figure 2 part b) independent of $\mathbf{e} = (h, V)$, a similar study where parameter dependent filters were successfully used was presented in [24]. We point out, however, that we have found cases where parameter dependent performance filters may lead to challenging situations for the methods discussed here. Consider for instance the performance graph in Figure 15, where in a related study in flight control design the gains of 6 controller blocks $K_1, \dots, K_6$ had to be tuned to match a parameter dependent performance specification over the flight envelope. Inspection shows that the performance graph has a very high cost for flight at high velocity and low altitude. It is clear that the size of the triangles in that region needs to be adapted to the steep slopes, leading to a large number of small triangles in that region, and the other techniques show similar behavior. The theoretical question is then whether the performance index may be changed in order to reduce the storage load, or whether we believe our criterion to represent cost accurately. In the latter case we would be stipulating that flight with high speed and low altitude simple *is* difficult, that our criterion represents this fact accurately, and that we should accept the higher cost to control as inevitable.

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

[2] Apkarian, P., and Gahinet, P., 1995. "A convex characterization of gain-scheduled h_∞ controllers". *IEEE Transaction on Automatic Control,* **40**, pp. 853–864.

[3] Apkarian, P., Gahinet, P., and Becker, G., 1995. "Self-scheduled H_∞ control of linear parameter-varying systems: A design example". *Automatica,* **31**(9), pp. 1251–1261.

[4] Zhang, Y. S., and Mehra, A. S., 2012. "Hinfty pid control for multivariable networked control systems with disturbance noise attenuation". *International Journal of Robust and Nonlinear Control,* **22**(2), pp. 183–204.

[5] Lu, B., and Wu, F., 2004. "Switching lpv control design using multiple parameter-dependent lyapunov functions". *Automatica*(40), pp. 1973–1980.

[6] Lu, B., Wu, F., and Kim, S., 2006. "Switching lpv control of an f-16 aircraft via controller state reset". *IEEE Trans. On control Systems Technology*, pp. 267–277.

[7] Mattei, M., 2001. "Robust multivariable pid control for linear parameter varying systems". *Automatica*(37), pp. 1997–2003.

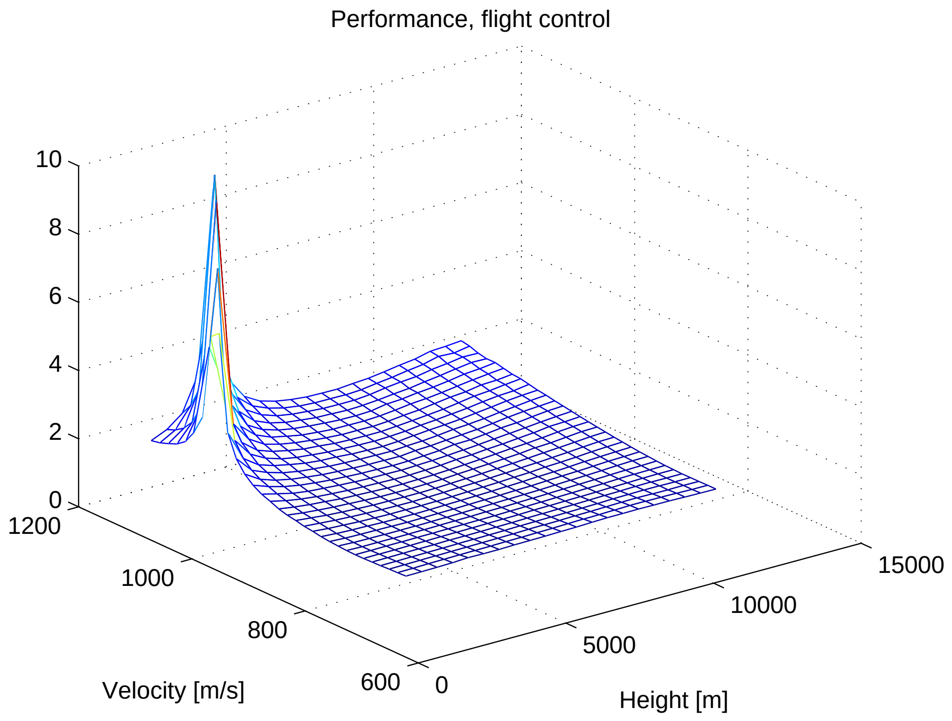


Fig. 15. Optimal performance for flight control case.

[8] Bolea, Y., Puig, V., and Blesa, J., 2008. "Gain-scheduled smith pid controllers for lpv systems with time varying delay: Application to an open-flow canal". *Proceedings of IFAC*, pp. 14564–14569.

[9] Mattei, M., and Scordamaglia, V., 2008. "A full envelope small commercial aircraft flight control design using multivariable proportional- integral control". *IEEE Transaction on Control System Technology,* **16**(1), pp. 169–176.

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

[11] L.Stevens, B., and F.L.Lewis, eds., 1992. *Aircraft Control and Simulation*. John Wiley and Sons Inc.

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

[13] Magni, J., Bennani, S., and Terlouw, J., eds., 1997. *Robust flight control:A design challenge*, Vol. 224 of *Lecture notes in control and information sciences*. Springer, Berlin.

[14] Zolghadri, A., 2000. "A redundancy-based strategy for safety management in a modern civil aircraft". *Control Engineering Practice,* **8**, pp. 545–554.

[15] "Robust toolbox, M.".

[16] Apkarian, P., and Noll, D., 2006. "Nonsmooth h_∞-control". *IEEE Transactions on Automatic Control,* **51**(1), pp. 71–8.

[17] Noll, D., Prot, O., and Rondepierre, A., 2008. "A proximity control algorithm to minimize nonsmooth and nonconvex functions". *Pacific Journal of Optimization,* **4**(3), pp. 569–602.

[18] airframe-demo of MatlabR2010b
http://www.mathworks.it/products/demos/shipping/robust/autopilot_demo.html?product=RC.

[19] Pottmann, H., Krasauskas, R., Hamann, B., Joy, K., and Seibold, W., 2000. "On piecewise linear approximation of quadratic functions". *Journal for Geometry and Graphics,* **4**(1), pp. 31–53.

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

[21] Hassin, R., and Levin, A. "A better-than-greedy approximation algorithm for the minimum set cover problem". *SIAM J. Computing,* **35**, pp. 189–200.

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

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

[24] Gabarrou, M., Alazard, D., and Noll, D., 2010. "Structured flight control law design using non-smooth optimization".

In 18th IFAC Symposium on Automatic Control in Aerospace.