TECHNICAL RESEARCH REPORT

Techniques for Designing Rotorcraft Control Systems

Final Report

by G. Yudilevitch, W.S. Levine

T.R. 94-54

Sponsored by
the National Science Foundation
Engineering Research Center Program,
the University of Maryland,
Harvard University,
and Industry

FINAL REPORT NASA AMES NAG 2-794

April 1, 1992 - August 31, 1994

Techniques for Designing Rotorcraft Control Systems

Gil Yudilevitch William S. Levine*

Department of Electrical Engineering and Institute for Systems Research
University of Maryland at College Park, MD 20742
* Principal Investigator

Techniques for Designing Rotorcraft Control Systems

Final Report

Gil Yudilevitch William S. Levine

Department of Electrical Engineering and Institute for Systems Research

University of Maryland at College Park, MD 20742.

Abstract

Over the last two and a half years we have been demonstrating a new methodology for the design of rotorcraft flight control systems (FCS) to meet handling qualities requirements. This method is based on multicriterion optimization as implemented in the optimization package CONSOL-OPTCAD (C-O). This package has been developed at the Institute for Systems Research (ISR) at the University of Maryland at College Park. This design methodology has been applied to the design of a FCS for the UH-60A helicopter in hover having the ADOCS control structure. The controller parameters have been optimized to meet the ADS-33C specifications. Furthermore, using this approach, an optimal (minimum control energy) controller has been obtained and trade-off studies have been performed.

Contents

1 Introduction 1

1.1 Design of rotorcraft flight control systems to meet handling qualities requirements . . . 1

1.2 Research objectives . . . 2

1.3 Organization . . . 2

2 Multicriterion-Optimization-Based Design Methodology 3

2.1 Introduction . . . 3

2.2 Preparing the design setup . . . 7

2.2.1 Modeling the system . . . 8

2.2.2 Translating the design requirements into "nice" mathematical functions . . . 8

2.2.3 Defining the optimization criteria . . . 9

2.3 The computerized tools used in the design process . . . 9

2.3.1 Optimization . . . 9

2.3.2 Simulation . . . 10

2.3.3 Design Evaluation . . . 10

2.4 The human designer . . . 11

3 The Optimization Package: CONSOL-OPTCAD 11

3.1 Introduction to CONSOL-OPTCAD . . . 11

3.2 Using C-O to meet the handling qualities requirements (ADS-33C) . . . 14

3.3 Using C-O to find an optimal controller . . . 14

3.4 The C-O/designer interface . . . 16

4 Design Setup 18

4.1 Introduction . . . 18

4.2 Modeling the UH-60A in hover . . . 19

4.3 The design specifications and their mathematical translations 22
4.3.1 Spec 1: Small Amplitude Changes, Short Term Response to Control Inputs 22
4.3.2 Spec 2: Small Amplitude Changes, Mid-Term Response to Control Inputs 25
4.3.3 Spec 3: Moderate-Amplitude Attitude Changes (Attitude Quickness) 27
4.3.4 Spec 4: Interaxis coupling 29
4.3.5 Spec 5: Wind-gust rejection 30
4.4 The design parameters 32

5 The nominal design **33**
5.1 Introduction 33
5.2 Nominal design with the ADOCS control structure 34
5.3 A feasible controller for the nominal design using an improved ADOCS control structure 37
5.4 The optimal design 38

6 Trade-off and robustness studies **45**
6.1 Introduction 45
6.2 Performance to specification 45
6.3 Performance to actuator limits 52
6.4 Performance to actuator time-constants 57
6.5 Robustness tests 58

7 Conclusions **62**
7.1 Design methodology 62
7.2 UH-60A in hover - results 62
7.3 Future work 63

A List of all the required computer codes **67**

A.1 C-O problem description files (PDF) 67

A.2 MATLAB simulation files 75

A.3 Some M-files for design evaluation 87

B A tutorial example **96**

1 Introduction

1.1 Design of rotorcraft flight control systems to meet handling qualities requirements

In a classical SISO control system design process, the practical design specifications are often replaced by standard control design specifications such as bandwidth, overshoot, etc. The main reason for using these standard specifications is that some (or all) of them can be satisfied using classical techniques such as root locus, Bode, etc. Of course, using standard specifications, instead of the original ones, leads to an approximate design. Moreover, these design techniques are limited to one or two (competing) specifications. It is difficult, if not impossible to use them for the design of MIMO control systems such as a modern rotorcraft FCS. Note that for MIMO systems some of the standard SISO specifications are not defined or have only "conservative" meaning (e.g., gain and phase margins).

In modern rotorcraft FCS design for handling qualities some MIMO control design methods have recently been used. These methods include optimal techniques such as LQR [1], $\mathcal{H}_2/\mathcal{H}_\infty$ [2,3], parametric optimization [4,5], etc., and other MIMO design techniques such as eigenstructure assignment [6], QFT [7], Nyquist array [8], etc. (see [9] for partial review). The main disadvantage of these MIMO techniques is that using precisely the theoretical design procedure it is either impossible to meet all the design requirements or many design iterations are required. Therefore many ad hoc design methods, based on theoretical MIMO techniques, are also used. Using those techniques (e.g., [10]) the design process is usually made in two phases. First we solve the theoretical problem (e.g., LQR), then we tune the design to meet the handling qualities requirements using simulation and/or test flight results (see for example [11] and [3]).

In fact there are no direct design methods for meeting handling qualities requirements. In this research we propose to use multicriterion parametric optimization as

the basis for a new direct design methodology.

1.2 Research objectives

The main objectives of this research are:

- Development of a rotorcraft FCS design technique based on multicriterion optimization. This method allows the designer to design "directly from the specs" to meet any (nonstandard) design specifications, such as handling qualities.
- Demonstration of this method by finding a set of FCS parameters such that the UH-60A in hover meets the LEVEL 1 performance requirements of the ADS-33C [12] (a feasible solution).
- Further demonstration of this method by meeting the ADS-33C level 1 specifications with minimum actuator "energy" (an optimal solution).
- Demonstration of the use of this method and the above criterion to perform trade-off studies.

1.3 Organization

The report is organized into seven sections. In Section 2 the general multicriterion-optimization-based design process is presented followed by a brief description of each of its components. The optimization package CONSOL-OPTCAD is introduced in Section 3. The design setup for the UH-60A in hover is given in Section 4. In Section 5 the nominal (LEVEL 1) design is presented including design considerations and results. In Section 6 two trade-off cases, performance/specification and performance/hardware, are studied. In addition the robustness of the optimal design is examined. Finally, some concluding remarks and some suggestions for future work are given in Section 7

This report has two appendices. Appendix A is a listing of all the computer code used in the design process. A tutorial example is given in Appendix B.

2 Multicriterion-Optimization-Based Design Methodology

2.1 Introduction

Real design problems are usually multifaceted. There is usually a variety of constraints on the solution as well as a group of, often conflicting, objectives. It is extremely difficult, if not impossible, to translate such a collection of specifications into a single objective functional as is required by most optimal control methods. As systems become more complex and their required performance levels increase it becomes difficult, if not impossible, to use classical methods to design satisfactory controllers.

The multicriterion-optimization-based design methodology is based on a combination of computer-based parametric optimization and human control designers. The idea is to use the computer to compute performance measures and find controller parameters that optimize them. The human designer decides whether the computer-generated design is adequate, and if it is not, changes the problem posed to the computer so as to drive the computer-generated solution in a better direction.

The role of the computer is a multicriterion parametric optimization. The human designer's first step is to translate the constraints and objectives of the design into a collection of smooth scalar functions of the (vector of) design parameters, say $f_i(\underline{x})$ where $i = 1, 2, \ldots, n$ and $\underline{x}$ is an m-vector of design parameters. The computer then tries to optimize (maximize)

$$\max_{\underline{x} \in C} f(\underline{x}) = \max_{\underline{x} \in C} \left\{ \min_{i=1,2,\ldots,n} \alpha_i f_i(\underline{x}) \right\} \tag{2.1}$$

where C is the set of allowable $\underline{x}$'s (defined by some of the constraints) and α_i, $i = 1, 2, \ldots, n$ is a set of real weights chosen by the designer.

To understand why the design problem is posed this way, consider the following example.

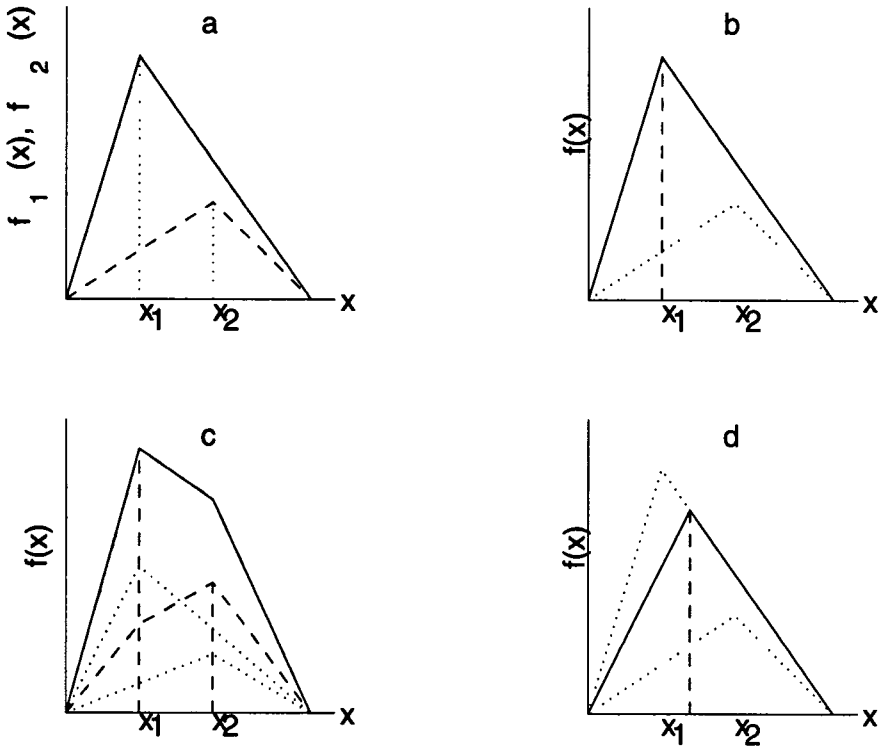

Figure 2.1: a: solid - $f_1(x)$, dashed - $f_2(x)$. b: solid - $f(x) = f_1(x)$. c: solid - $f(x)$ for $\alpha_1 = 1$, $\alpha_2 = 1.5$, dashed - $f(x)$ for $\alpha_1 = 0.2$, $\alpha_2 = 2$. d: solid - $f(x)$ for $\alpha_1 = 1$, $\alpha_2 = 3$. b-d: dotted - $f_1(x), f_2(x)$.

Let $f_1(x)$ and $f_2(x)$ of Figure 2.1-a be two performance measures of a control system, where x is the scalar design (controller) parameter. Let $f(x) = F(f_1(x), f_2(x))$ be the optimization (maximization) objective (i.e., $\max_x f(x)$). Suppose we define $f(x) = \max_{i=1,2} f_i(x)$ (Figure 2.1-b). Then $f_2(x)$ has no effect on the solution to the optimization problem. In fact then, x_1 is the only possible "optimal" solution. One may try also taking $f_2(x)$ into account by defining a weighted objective function $f(x) = \sum_{i=1}^{2} \alpha_i f_i(x)$, $\alpha_i \in \mathbb{R}$ (Figure 2.1-c). Then only x_1 or x_2 can be "selected". In order to have more design degrees-of-freedom, let $f(x) = \min_{i=1,2} \alpha_i f_i(x)$. Then by choosing α_i's the designer can change the "optimal" solution to be any $x \in [x_1, x_2]$, see Figure 2.1-d.

This simple example emphasizes the importance of the proper choice of $F(\cdot,\cdot)$ (the "design specifications"). Such a "max/min" ($\max_x \min_i \alpha_i f_i(x)$) or "min/max" ($\min_x \max_i \alpha_i f_i(x)$) optimization problem is the basis for the design methodology presented in this report. In fact, given n performance measures $f_i(x)$, $i = 1, 2, \ldots, n$, where $x \in \mathbb{R}^m$ is the vector of the design parameters (d.p.'s), the design can be accomplished by performing the following three steps

Step 1 - Choose n α_i's.

Step 2 - Optimize $f(x)$ over x. (2.2)

Step 3 - Check the design. If it is good, stop. Else, go to Step 1.

Note that we do not know yet how to implement the above design procedure. In fact this is only a conceptual procedure, a more practical procedure is given below. Note also that we do not even know how to define and compute the performance measures, the $f_i(x)$'s. Usually obtaining "good" $f_i(x)$'s is a very complicated problem. This task is a part of the preliminary stage which has to be completed prior to the implementation of the above procedure.

The overall design process is schematically presented in Figure 2.2. It is assumed that the physical system and the design specs are given, as is typically true in rotorcraft

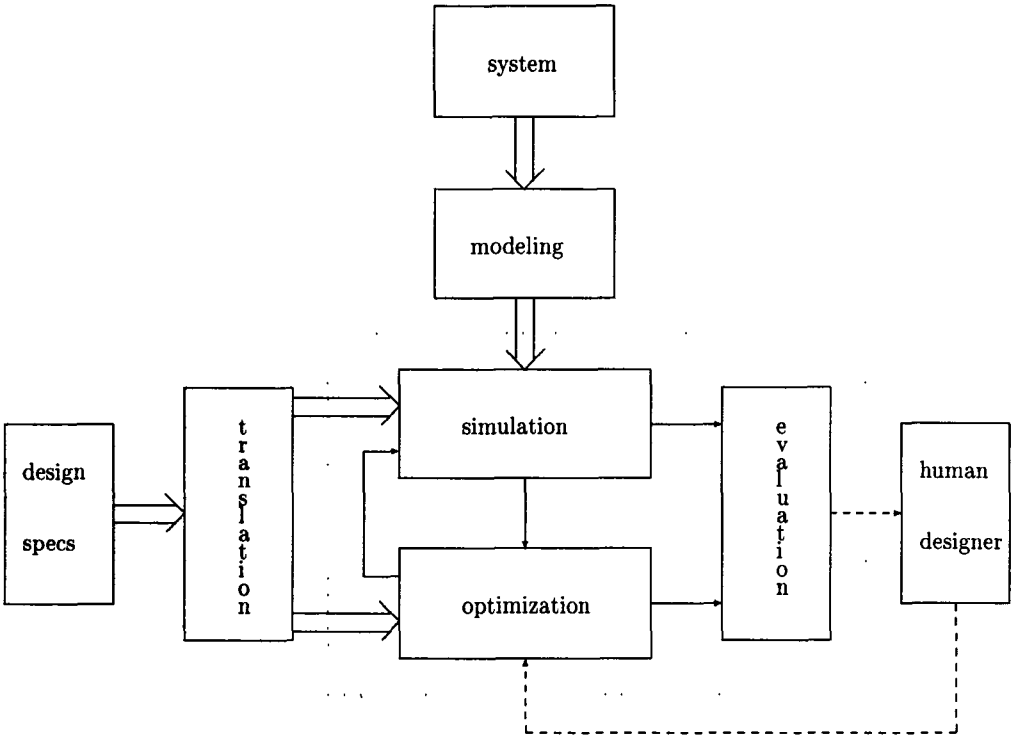

Figure 2.2: Schematic description of a multicriterion-optimization-based design process

control design. The preliminary part of the design process (modeling, translation of the design requirements into mathematical functions, and development of a simulation program) is marked by thick (double line) arrows. The iterative parts of the process are the computerized optimization (thin solid arrows) and the human designer interrupts (thin dashed arrows).

The multicriterion-optimization-based design process has two main parts. First, the designer has to prepare the design setup. This noninteractive part of the process includes: modeling (mathematical representation of the system) and translation of the design specifications into nice mathematical functions.

In the interactive part of the process, the optimization is done automatically by the optimization/simulation package (see [13] and Section 3 below). The designer has to continuously monitor the results by using the design evaluation tools, then if he finds it necessary he can interrupt the process. The specific action that the designer may take depends on several factors. Because the designer usually interrupts the process through the designer/optimization interface, a major factor is the properties of the specific optimization package. Some suggestions are given in Section 3.4 below.

A detailed description of the design process and its components is given in the following sections.

2.2 Preparing the design setup

In preparing the design setup, much care should be taken because this stage affects the whole design process. We especially have to consider the practical trade-off between "accuracy" and "time". For higher accuracy we would like to choose the most comprehensive model, to write a sophisticated simulation code, to use a small step size for optimization, etc. However this choice leads to a very "time consuming" design process. This problem becomes critical because of the presence of a human designer in the design process.

Given the system and the design specifications, the following preparations have to

be made prior to the implementation of the iterative procedure.

2.2.1 Modeling the system

Very complicated mathematical models are often required in order to obtain accurate simulation of real control systems. The first design task is to find the simplest model which still captures the main dynamic characteristics of the system. Choosing simpler models may spoil the design. On the other hand increasing the size and complexity, may lead to a long and annoying design process. In most cases finding the "optimal" model is not a simple task.

In this stage issues such as linear model vs. nonlinear model, continuous-time model vs. discrete-time model, small model vs. large model, etc., have to be studied carefully.

Remark 2.1 *Usually the model also has to represent implementation limitations such as actuator saturation, controller size, sampling rate, etc. Therefore sometimes (e.g., actuator saturation) a linearized model can not be used.*

2.2.2 Translating the design requirements into "nice" mathematical functions

Modern design requirements such as "handling qualities" are very difficult to translate into smooth mathematical functions. In some cases these functions are defined in terms of numerical evaluation of the solution to some differential equations ("simulation"). In other cases even more mathematical manipulations (on the simulation results) are required.

Using optimization special care should be taken with these translations. For most optimization techniques, these mathematical functions have to be "smooth" with respect to the design parameters (at least continuously differentiable). Even standard performance measures, which are used in conventional design techniques, are often not

useful for optimization. For example, spectral characteristics such as eigenvalues, are usually not differentiable with respect to the control law parameters.

Because most of these functions have no analytical expression, it is difficult and in most cases even impossible to check their smoothness. Therefore in some cases we may use nonsmooth functions for optimization. Usually these functions are not smooth only at a finite number of points in the parameter space. Thus practically they may work. However using such functions for optimization may cause the design process to get stuck.

2.2.3 Defining the optimization criteria

This is the second part of the translation of the design specifications. The optimization criteria are based on both the performance translations and some properties of the optimization package. For example using CONSOL-OPTCAD [13] with a vector of time (or frequency) dependent performance measure, the optimization criterion may be given as a functional constraint or objective.

2.3 The computerized tools used in the design process

Three computerized tools are used in the design process: optimization, simulation, and design evaluation (see the dotted box in Figure 2.2). These tools interact with each other, thus it is essential to choose or develop the proper software packages and interfaces.

2.3.1 Optimization

The role of the optimization in the design process is defined in Step 2 of the conceptual procedure (2.2). Namely, finding a set of design parameters x^* such that the $f_i(x^*)$'s satisfy the criteria.

The particular implementation of the optimization "block" in the design procedure depends on the choice of the optimization package. In this research the

CONSOL-OPTCAD optimization package is used. This package has a built-in interface with some common simulation languages including MATLAB. For more details on CONSOL-OPTCAD see Section 3 below.

2.3.2 Simulation

Given the system model, the definition for $f_i(\cdot)$, and the design parameters x, the simulation generates $f_i(x)$'s. Moreover the simulation also provides most of the information required for the design evaluation.

In order to obtain more information required for the optimization, such as gradients, the simulation is often used iteratively. Therefore it is important to choose the proper simulation language and to write an efficient simulation code. Other properties of the simulation language (package) should also be taken into account such as: standard interface (to be used with the optimization package), "user friendly", technically supported, etc. In fact, for these reasons we chose the MATLAB package as the simulation package for this research.

Remark 2.2 *Using other simulation packages, or even writing a special simulation program using a standard computer language (e.g., C) may lead to a "faster" simulation. However, because these solutions do not have the nice advantages of MATLAB, the total time which may be spent using these alternatives may be larger.*

2.3.3 Design Evaluation

"Design evaluation" stands for all the computerized tools which help the human designer to obtain the right decisions during the design process. In some cases no additional tools are required, e.g., when the standard outputs of the simulation and optimization packages provide all the necessary information. However, in most real-system-design cases some special tools are required. In particular this is true when the design is made to meet requirements such as the handling qualities.

In this research we have developed such a specific evaluation "toolbox". This toolbox performs an on-line graphical analysis of the system performance which allows the human designer to make on-line decisions. For more details see Appendix A.

2.4 The human designer

The human designer is definitely the most important component of the design procedure. His or her role in the design process is given by Step 1 and Step 3 of the conceptual algorithm (2.2). However contrary to the computerized tools, it is generally not clear what the designer has to do at any design iteration. In fact the decisions that the designer takes are in many cases based on his intuition and experience. Therefore, at least for the time being, the human designer can not be replaced by a computer.

The experience gained in using this methodology in this research and in other related projects may be used as the basis of new rules and guidelines for future designers.

3 The Optimization Package: CONSOL-OPTCAD

3.1 Introduction to CONSOL-OPTCAD

In an attempt to better represent real world design problems, CONSOL-OPTCAD (C-O) allows for three qualitatively different types of design specifications. An *objective* is a specification of a quantity that should be optimized (minimized or maximized). Typically, multiple competing objectives are present. A *hard constraint* is a specification of a quantity that must achieve a specified threshold. A *soft constraint* is a specification of a quantity that should achieve, or at least approach, a specified threshold, i.e., should be optimized as long as this threshold is not achieved. Soft constraints can be thought of as intermediate between objectives and hard constraints.

Choosing the proper scale factor for each design specification may be a difficult task. Therefore instead of using a single scale factor (weight), each objective and soft constraint (*value*) is scaled by C-O using "*good*" and "*bad*" values according to the

formula

$$\text{scaled } value \ = value^s = \frac{value - good}{bad - good} \tag{3.1}$$

where having any *value* achieve its corresponding *good* (*bad*) value should provide the same level of satisfaction (dissatisfaction) to the designer. This uniform scaling rule helps the designer in choosing the proper weights (if $good = 0$ then α_i of (2.2) is $\frac{1}{bad}$). Note that using practical values for the good and bad scaling parameters may cause too large differences between the sensitivity of the various d.p.'s. Therefore, in order to make the parameter space more "uniform", C-O also allows the designer to scale each d.p. separately. In addition to the above design specifications, it is also possible to put hard bounds on the d.p.'s.

In order to have a better understanding of these quantities, consider the following simple example. Suppose we have to design a cheap audio amplifier, with $\frac{\text{noise}}{\text{signal}} \leq a$ and maximum input power $\approx b$. The design (amplifier) parameters have some bounds (e.g., negative feedback gain for stability). The corresponding optimization problem has the following setup

$$\begin{array}{lcll} \textit{objective} & = & \text{dollar cost} & \min \\ \textit{hard const.} & = & \frac{\text{noise}}{\text{signal}} & \leq a \\ \textit{soft const.} & = & \text{input power} & \leq b \\ \textit{hard bound} & = & \text{f.b. gain} & < 0. \end{array} \tag{3.2}$$

C-O divides the optimization process into 3 phases. In Phase 1, if $\frac{\text{noise}}{\text{signal}} > a$ for every choice of parameters, then it is impossible to satisfy the hard constraint unless we allow C-O to increase the input power and the cost. Then C-O minimizes the amplifier input power (Phase 2) and its cost (Phase 3).

C-O uses FSQP (Feasible Sequential Quadratic Programing) to solve the following

general optimization problem

$$\begin{aligned} &\min_x obj_i^s(x)\ \forall i \\ \text{subject to:}\quad &soft_j^s(x) \quad \leq 0\ \forall j \\ &hard_k^s(x) \quad \leq 0\ \forall k \\ &bound_l^s(x) \quad \leq 0\ \forall l \end{aligned} \tag{3.3}$$

where obj_i^s, $soft_j^s$, $hard_k^s$, and $bound_l^s$ are the scaled values of objectives, soft constraints, hard constraints and hard bounds, respectively. Problem (3.3) is then assigned three different meanings, corresponding to three different phases, according to feasibility or infeasibility of x with respect to hard and soft constraints.

Phase 1 (hard feasibility problem): Not all hard constraints are satisfied. (3.3) takes the form

$$\begin{aligned} &\min_x \max_k\ hard_k^s(x) \\ \text{subject to:}\quad &bound_l^s(x) \quad \leq 0\ \forall l. \end{aligned} \tag{3.4}$$

Phase 2 (soft feasibility problem): All hard constraints are satisfied. Not all (scaled) objectives and soft constraints are nonpositive. (3.3) takes the form

$$\begin{aligned} &\min_x \max_{i,j}\ \left\{obj_i^s(x), soft_j^s(x)\right\} \\ \text{subject to:}\quad &hard_k^s(x) \quad \leq 0\ \forall k \\ &bound_l^s(x) \quad \leq 0\ \forall l. \end{aligned} \tag{3.5}$$

Phase 3 (optimization problem): All hard constraints are satisfied and all (scaled) objectives and soft constraints are nonpositive. (3.3) takes the form

$$\begin{aligned} &\min_x \max_i\ obj_i^s(x) \\ \text{subject to:}\quad &soft_j^s(x) \quad \leq 0\ \forall j \\ &hard_k^s(x) \quad \leq 0\ \forall k \\ &bound_l^s(x) \quad \leq 0\ \forall l. \end{aligned} \tag{3.6}$$

C-O has a PCOMB (performance "comb") pseudo-graphical output display which gives very useful information. An example of a PCOMB display is shown in figure 3.1 below.

3.2 Using C-O to meet the handling qualities requirements (ADS-33C)

The design of a flight control system to meet the ADS-33C (Aeronautical Design Standard) [12] requirements can be obtained as a solution to the soft feasibility problem (3.5). That is, all the ADS-33C requirements are translated into C-O hard and soft constraints (no objective). The design goal is achieved when C-O successfully completes Phase 2.

Theoretically, if the feasible set is *nonempty*, and all the performance functions are globally *smooth* and *convex* with respect to the d.p.'s, then starting with any (infeasible) point, the C-O solution will converge to a feasible solution. However practically, none of the above three conditions (existence, smoothness, and convexity) is guaranteed. We can not change the feasibility of the problem (and also its convexity), but we can improve the smoothness of the performance functions by using proper translations and/or smooth approximations. In fact, part of the art is finding ways to describe the desired system performance mathematically so that the C-O solution will converge to a "good" controller.

3.3 Using C-O to find an optimal controller

If there is more than one feasible solution, it is natural to search for the optimal solution. First we have to choose the optimization objective (if it is not given a priori). A natural choice for the objective function can be any scalar function of the design parameters which has no constraints, that it would be nice to have it small (or large), such as the control energy. Multiple objective functions may be taken. The choice of objective functions has to be done with the same care (smoothness!) as for

```
<0>
Pcomb (Iter=  0) (Phase 1) (MAX_COST_HARD= 400.00000)

SPECIFICATION    PRESENT   GOOD              G          B             BAD
O1   objective1 -9.66e+02  0.00e+00 <==      |          |             1.00e+00
O2   objective2  1.20e+00  2.00e+00          |       *============  1.00e+00
C1   hard_cons1  4.00e-01  0.00e+00 ------------------------------> 1.00e-03
C2   soft_cons1  3.50e-01  0.00e+00 ==============*      |          1.00e+00
C3   hard_cons2  1.05e+00  1.00e+00 <------------------------------ 1.00e+04
FC1  soft_fcons  0.06e+00  0.00e+00 <================================ -5.00e-02

<6>
Pcomb (Iter=  6) (Phase 2) (MAX_COST_SOFT= 0.3500000)

SPECIFICATION    PRESENT   GOOD              G          B             BAD
O1   objective1 -9.66e+02  0.00e+00 <==      |          |             1.00e+00
O2   objective2  2.20e+00  2.00e+00         *========================  1.00e+00
C1   hard_cons1 -4.06e-02  0.00e+00 <--      |          |             1.00e-03
C2   soft_cons1  3.50e-01  0.00e+00 ==============*      |          1.00e+00
C3   hard_cons2  1.16e+00  1.00e+00 <------------------------------ 1.00e+04
FC1  soft_fcons  1.36e+00  0.00e+00 <================================ -5.00e-02

<10>
Pcomb (Iter= 10) (Phase 3) (MAX_COST= -1000.0000)

SPECIFICATION    PRESENT   GOOD              G          B             BAD
O1   objective1 -9.99e+02  0.00e+00 <==      |          |             1.00e+00
O2   objective2  2.20e+00  2.00e+00         *========================  1.00e+00
C1   hard_cons1 -4.06e-02  0.00e+00 <--      |          |             1.00e-03
C2   soft_cons1 -0.10e-00  0.00e+00 =========*|         |             1.00e+00
C3   hard_cons2  1.16e+00  1.00e+00 <------------------------------ 1.00e+04
FC1  soft_fcons  1.36e+00  0.00e+00 <================================ -5.00e-02
```

Figure 3.1: An example of the C-O PCOMB output display. In Phase 1, C1 (hard cons.) is minimized. In Phase 2, C2 (soft cons.) is minimized. In Phase 3, O2 (objective) is maximized. Double dashed lines - soft constraints and objectives. Single dashed lines - hard constraints. $*$ - scaled value $\in (-1, 2)$. $>$ - scaled value > 2 (too bad). $<$ - scaled value < -1 (very good). For more details on the PCOMB display see [13].

the problem constraints. Note that the natural objective function may not always be smooth (see for example Figure 5.7).

Theoretically, if the objective functions are *smooth* and *convex* with respect to the d.p.'s, then starting with any (feasible) point, the C-O solution will converge to the unique optimal solution. Practically, starting with any arbitrary feasible point, if the C-O solution converges, then it converges to a local minimum.

3.4 The C-O/designer interface

As we already mentioned, this is usually the only interface between the human designer and the optimization process. The ability of the designer to guide and control the whole process depends on the design degrees-of-freedom that C-O provides. The human designer's actions also depend on the simulation and optimization results. It is hard to give a recipe which covers all the possible situations. The following guidelines are given as general advice for the designer:

- Make the constraint and objective functions as smooth as possible (a smooth approximation is better then a more accurate nonsmooth approximation).
- Freeze all the unnecessary d.p.'s. It saves time.
- Use all possible C-O information ("pcomb","print", "trace","active", etc.). Also use the other design evaluation tools (e.g., time and frequency response, performance map, etc.).
- If C-O gets stuck, "shake" the optimization.

Again there is no general solution for this situation. However the following list may be used if the process gets stuck (termination of the computer run, the computer starts to work very slowly, etc.).

If C-O gets stuck during Phase 2:

- Look for competing specs.

- Change constraint weights ("good", "bad" values).
- Keep the scaled objective functions below the scaled constraint functions.
- Change the nominal variations of some d.p.'s.
- Perfom one or two dimensional analysis (see Section 6 for example).
- Change manually the value of some d.p.'s.
- Improve the smoothness of constraint functions (use alternative definitions, or approximations).
- Change controller structure (e.g., add dynamics).

If C-O has trouble switching from Phase 2 to Phase 3:

Near the boundaries C-O may work slowly. Change the variation step ("scale"), or if it is possible (very close to the boundaries), "right-shift" or "left-shift" the "good" value (i.e., spoil the good value so that C-O "jumps" to Phase 3).

If C-O gets stuck during Phase 3:

- Make all objective functions negative for all feasible d.p.'s (e.g., for minimization less than their good values). This is required by C-O [13].
- Put small weights on the relevant constraints which might oppose the minimization process (e.g., min. actuator "energy" vs. quickness).
- Look for competing objectives (no competing objectives in our design problem).
- Change objective weights ("good", "bad" values).
- Change the nominal variations of some d.p.'s.
- Perfom one or two dimensional analysis.
- Change manually the values of some d.p.'s.

- Improve the smoothness of objective functions (use alternative definitions, or approximations).

Remark 3.1 *When C-O finds a local minimum it gives a CONGRATULATIONS message. Using objective functions such as control energy, "push" the solution (some performance measures) to the boundaries of the feasible set (LEVEL 1). Then C-O may start to move slowly. Usually you do not have to wait for this message, just use the design evaluation tools to decide when to stop the optimization process.*

4 Design Setup

4.1 Introduction

The UH-60A (Black-Hawk) helicopter in hover was chosen as a benchmark example for this research. Its flight control system has an ACAH (Attitude Command Attitude Hold) response type. The design specifications are given in the ADS-33C [12], Paragraph 3.3 - hover and low speed. The design specification affects both parts of the design setup (i.e., system modeling and specification translation).

In addition to the optimization package (C-O) and the simulation package (MATLAB), the design setup contains:

- A main C-O PDF *adocs* and some "include" *spec* files (see [13] and Appendix A). These files contain all the ADS-33C information required by C-O.
- Initialization (*init.m*) and simulation (*simu.m*) M-files, containing the MATLAB simulation code required to evaluate all the ADS-33C performance measures (see Appendix A).
- The design evaluation M-files (see Appendix A).

The background and other considerations used to develop the above setup are summarized in the following.

4.2 Modeling the UH-60A in hover

The comprehensive rotorcraft aerodynamic model "UMGenhel" [14] requires too much computer time for one simulation run to be useful for the purposes of this research. Thus we wrote a simplified model which can be used by the optimization package (C-O). The simplified model represents the helicopter linear dynamics and aerodynamics, as well as the most important system nonlinearities (i.e., actuator saturation). The specific representation of the helicopter in hover has been modified several times in order to obtain the system performance measures as well as to satisfy some computational limitations.

The final configuration, shown in Figure 4.1, includes the following parts:

(*i*) **Linearized and reduced (UMGenhel) dynamic model** $P(s)$ - The bare airframe model has a total of 11 states, 9 states for the 6 DOF fuselage dynamics and 2 states for the main rotor flapping motion (using model reduction techniques, the effect of higher dynamics is also included in the model). The dynamic model is given by the following state equation

$$\begin{aligned} \dot{x} &= Ax + Bu_a + Wd_g \\ y &= Cx \end{aligned} \tag{4.1}$$

where $x = (u, v, w, p, q, r, \phi, \theta, \psi)^\top$, which stands for the longitudinal velocity (u), lateral velocity (v), vertical velocity (w), roll rate (p), pitch rate (q), yaw rate (r), roll angle (ϕ), pitch angle (θ), and yaw angle (ψ). The state equation (4.1) has two inputs, $u_a = (u_\theta, u_\phi, u_\psi, u_c)^\top$ representing respectively the longitudinal, lateral, tail rotor collective, and main rotor collective actuator displacements (control). The second input $d_g = (d_\theta, d_\phi, d_\psi)^\top$ represents respectively the pitch, roll, and yaw (wind gust) disturbances. The output y, required to evaluate all the desired specifications, is in fact all of the state vector excluding the longitudinal (u) and lateral velocities (v).

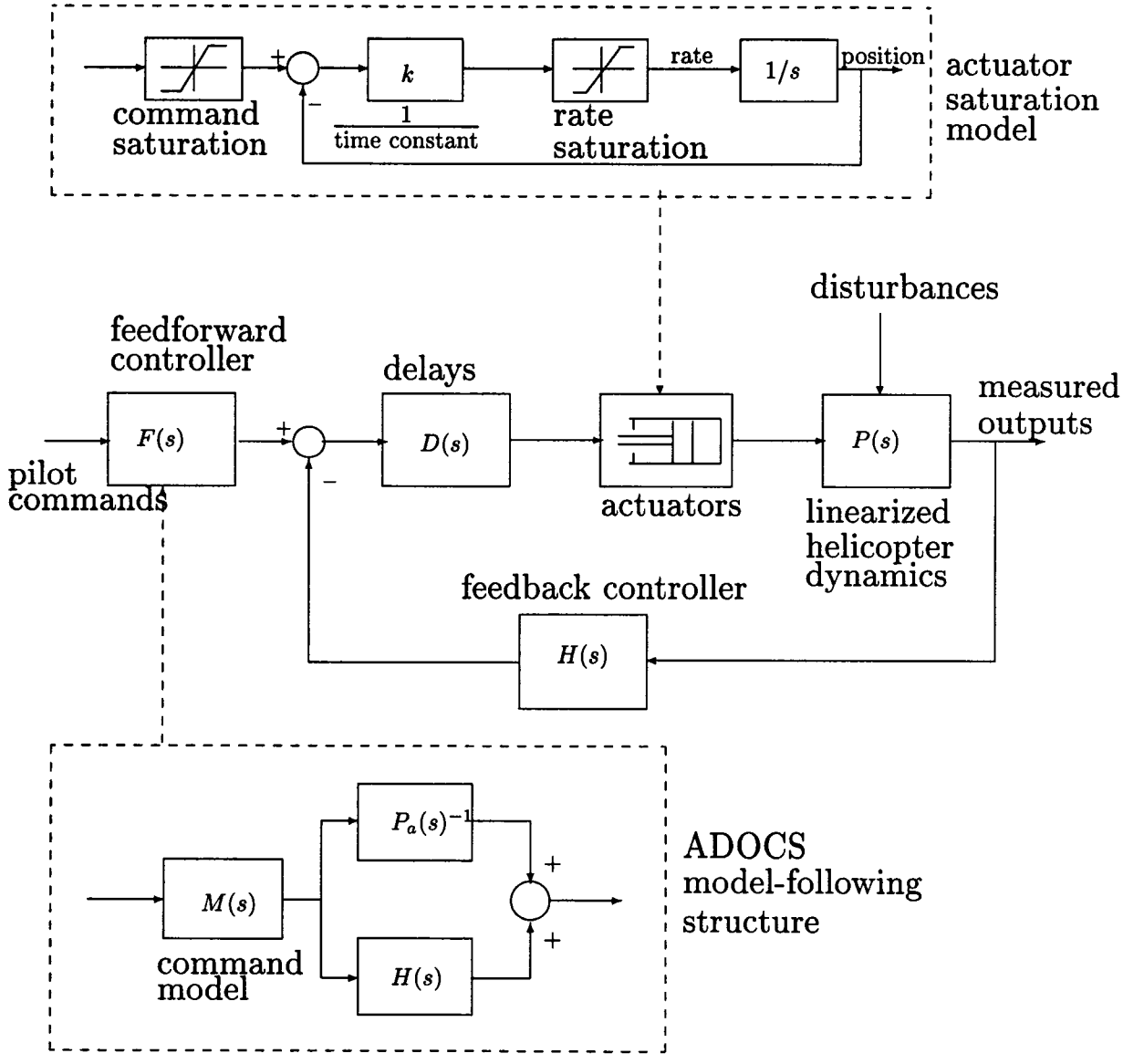

Figure 4.1: Model of the UH-60A.

An important component of the initial task of this project was to verify the (linearized) helicopter model. First this model was compared with the full UM-Genhel model (39 states) [14], to verify the use of only 11 states. Then it was compared with a simplified 6 DOF model (9 states) (used in [15]), to verify its main characteristics. In addition, using "real" ADOCS parameters, its closed-loop response was qualitatively checked.

(*ii*) **Actuator Model** - The swashplate actuators are modeled using standard saturation functions for displacement and rate limits and a 1^{st} order approximation for the actuator dynamics (see Figure 4.1). Note that the displacement saturation is implemented on the actuator command as is standard [16].

(*iii*) **Delay** $D(s)$ - Pure delay ($e^{-\tau s}$) represents an overall (τ sec.) channel delay including: computation delays, A/D and D/A delays, unmodeled dynamics etc.

(*iv*) **ADOCS control law** - H is a constant gain, output feedback matrix. $F(s)$ is a decentralized feedforward dynamic controller obtained from the model following concept [17], based on a command model dynamics $M(s)$ and on a 1^{st} or 2^{nd} order approximation for the helicopter dynamics $P_a(s)$, see Figure 4.1. Note that if $P_a(s) = P(s)$ then the resultant closed-loop transfer function is $M(s)$. The overall closed-loop system is controlled by the pilot using four input commands $\delta = (\delta_\theta, \delta_\phi, \delta_\psi, \delta_c)^\top$ representing respectively the longitudinal, lateral, tail rotor collective, and main rotor collective cockpit commands.

In order to calculate efficiently all the desired performance measures, this model has two different versions. There is a continuous-time linear model, for small amplitude performance, where the actuator model is a simple 1^{st} order model (no saturation) and the delay $D(s)$ is a 2^{nd} order Padé approximation. There is also a discrete-time nonlinear model, for large amplitude performance, where all the continuous time parts (i.e., $P(s)$ and $F(s)$) are replaced by suitable ZOH equivalents, and the saturation functions are implemented directly. The nonlinear simulation is obtained by solving

the difference equations recursively. For more details see the M-files, *init.m*, *simu.m*, and *drsc.m* in Appendix A.

4.3 The design specifications and their mathematical translations

The following five specifications were identified as those that are essential to meet the ADS-33C [12]. These five requirements are naturally divided into two groups. Spec's 1, 2, and 5 relate to small-amplitude responses and can be checked using linear models. The remainder are related to moderate-amplitude responses and must be checked by nonlinear simulation.

4.3.1 Spec 1: Small Amplitude Changes, Short Term Response to Control Inputs

This modern frequency based criterion (bandwidth/phase-delay) replaces the traditional specifications which used limits based on time-delay and rise-time. The bandwidth/phase-delay criterion emphasizes features directly related to closure of the piloted loop, and it is a better metric than rise-time for the prediction of handling qualities for small-amplitude precision tracking tasks. It is clear that pilots are also sensitive to the shape of the phase curve at frequencies beyond the bandwidth frequency. This shape is characterized by the phase-delay parameter [16]. Thus, the level regions for the short-term response requirement are defined in the bandwidth/phase-delay plane as shown in Figure 4.2-d. Actually, for small phase-delay systems this is a "pure bandwidth" criterion. Above a certain value of phase-delay (about 0.2 sec) it becomes a trade-off between bandwidth and phase-delay (i.e., the pilot can tolerate higher phase-delay but then, in order to achieve the same performance level, he needs higher bandwidth).

Bandwidth and phase-delay are measured from a frequency response (Bode) plot of angular attitude response to cockpit controller input. Bandwidth and phase-delay, as defined in the specification [12], Paragraphs 3.3.2.1 (Pitch, Roll) 3.3.5.1 (Yaw), are

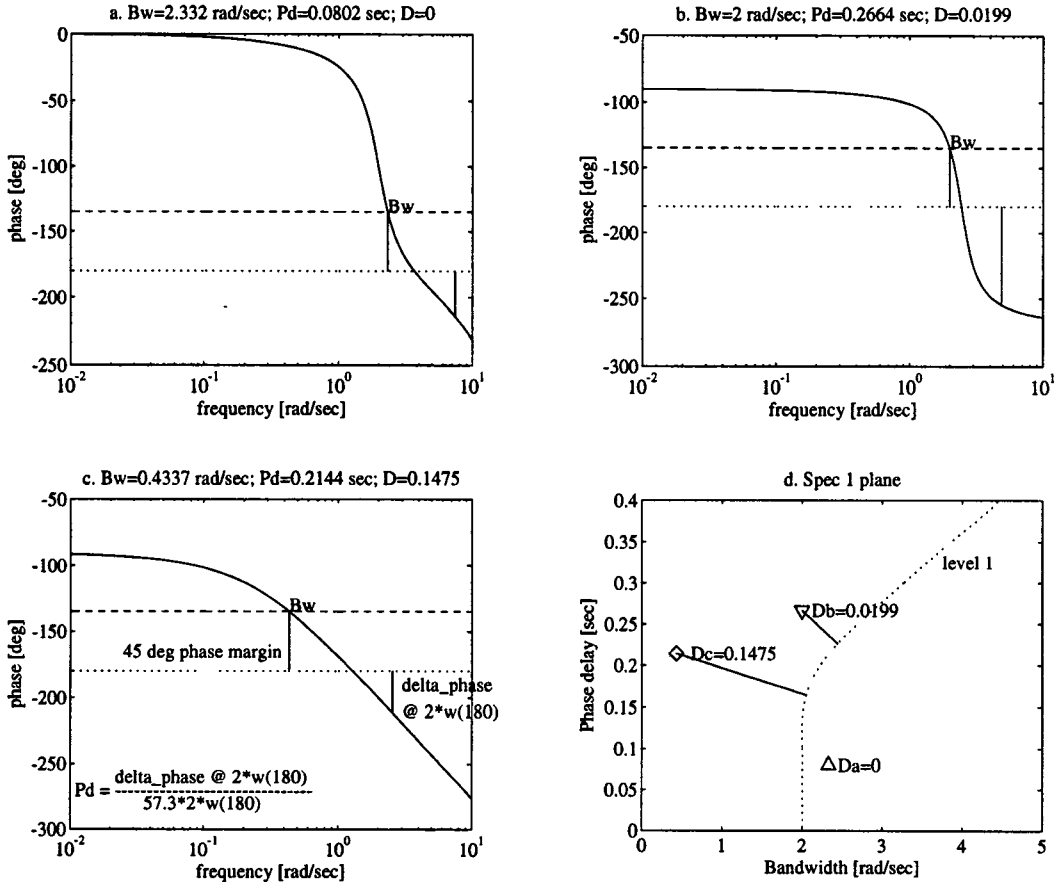

Figure 4.2: Spec 1: Bandwidth vs. Phase-delay (a-c) Bode phase plot, bandwidth and phase-delay of three examples. (d) their D meassures in the Spec 1 plane.

referenced to the helicopter with all augmentation loops closed. Thus, they can not be simply calculated from the closed-loop design parameters. Usually, two bandwidth frequencies are measured: the frequency for 6 db gain margin ($\omega_{BW_{\text{gain}}}$), and the frequency for 45° of phase margin ($\omega_{BW_{\text{phase}}}$). For ACAH response types $\omega_{BW_{\text{phase}}}$ is taken, since the nature of ACAH is such that the pilot does not have to close the attitude loop for stabilization purposes, so the gain margin problems are less apparent. Phase-delay is defined so that it represents all of the contributions to phase less than $-180°$, see Figure 4.2-c, and is based on the observation that the phase curve tends to be linear in the neighborhood of the crossover frequency.

In order to meet the LEVEL 1 (also true for other levels) requirement a two dimensional geometrical measure D (normalized quadratic distance) is defined, such that minimizing this measure implies better performance. The computation of this measure is done in three steps. First, the graphical level curve is converted into an analytic smooth function using a polynomial curve fitting algorithm. In order to achieve a univalent function the level curve is represented as $\omega_{BW} = f(\tau_d)$, i.e, the bandwidth frequency is a function of the phase-delay, which is a nondecreasing C^∞ function, Figure 4.2-d (the dotted curve). This calculation is done only once in the initialization routine (for more details see *init.m* in Appendix A). The other two steps are executed in each iteration. First, the bandwidth and phase-delay are computed, based on the definitions of Figure 4.2-c, using an efficient search algorithm. Second, the D measure is calculated as the minimum normalized quadratic distance from the current (ω_{BW}, τ_d) point to the level curve. Moreover, measure D is calculated only for points which are not in the LEVEL 1 region (i.e., they are to the left of the dotted curve in Figure 4.2-d), and it is set to zero for any (ω_{BW}, τ_d) point within the LEVEL 1 region. Because $D(\omega_{BW}, \tau_d)$ is quadratic, using the above definition keeps it differentiable even on the boundary of the LEVEL 1 set (i.e., $\omega_{BW} = f(\tau_d)$), and gives an identical weight for any point in the desired set (LEVEL 1 region). The computation of D uses the fact that the level curve is a nondecreasing function, so it

eliminates the need to evaluate the function $\omega_{BW} = f(\tau_d)$ at each $\omega_{BW} = f(\tau_d)$ (for more details see *d_bw_pd.m* in Appendix A).

Remark 4.1 *The use of polynomial curve fitting is limited for the given data points. That is, the approximation $\omega_{BW} = f(\tau_d)$ holds only for $\tau_d \in [0, 0.4]$ [12]. Therefore hard constraints are used to guarantee that $\tau_d \leq 0.4$.*

4.3.2 Spec 2: Small Amplitude Changes, Mid-Term Response to Control Inputs

This requirement is, in general, the complementary part of the short term response requirements of Spec 1. The short term criterion emphasizes features related to the high frequency modes, whereas the mid-term influences mainly the low frequency modes. Although, because of the particular definitions of both criteria, it is unavoidable that this requirement overlaps the short-term one. The mid-term requirement is specified for two different pilot operation modes: "Fully Attended Operations" - where all of the helicopter tasks can be accomplished with full pilot attention to aircraft control (e.g., other crew members handle the non-control tasks), and "Divided Attention Operations" - where the pilot should be able to relinquish control of the helicopter for short periods of time without encountering significant excursions. Consideration of divided attention (as done in this research) ensures that, at least practically, any flight control system which meets this requirement is stable. In fact a helicopter which meets the fully attended requirement can have unstable mid-term response (as with many present-day helicopters). For more information see [12], Paragraphs 3.3.2.2 (Pitch, Roll) 3.3.5.2 (Yaw).

Remark 4.2 *Although it is not an ADS-33C requirement, an asymptotic stability hard constraint is used (i.e., $\Re\{\lambda_m(A - BH)\} < 0$). This constraint is required for numerical reasons.*

At the beginning we tried to use the damping ratio parameter ζ for this requirement. This parameter, which is well defined for second-order systems, can be interpreted in several different ways for higher order systems. Unfortunately, most of these interpretations lead to numerical algorithms which are either significant time consumers, or are not smooth enough, or both. For example, computing ζ as the logarithmic decrement of the two first peaks of the system step response is very time consuming. On the other hand, using eigenvalues, in order to find ζ as the ratio between the imaginary and the real part of the 2^{nd}order approximated model, is generally, not a smooth calculation. The computer time problem becomes critical because this algorithm has to be executed in a multi-iteration optimization process. The smoothness problem is sometimes even more critical because it may cause failure of the optimization process.

During this work we have also examined several approximations for ζ based on model reduction techniques, system identification, 2^{nd} and 3^{rd} order approximations, etc. Unfortunately, these methods have been found to be either not accurate enough, or too complicated, or both. Furthermore the original ADS-33C specification has a few disadvantages. First, theoretically it also contains the undesired "long term" response (phugoid modes). Second, using only the original definition does not always prevent PIO (Pilot Induced Oscillations).

Therefore instead of using the original "damping ratio" (or "pole location") criterion we have used a classical stability margin criterion, gain margin $(GM) \geq 6$ db and phase margin $(PM) \geq 45°$. Note that these classical stability margins are actually SISO specifications. Therefore, for each channel, we have used the "broken-loop" of figure 4.3 as the channel SISO approximation.

Remark 4.3 *The Spec 2 results presented in the sequel have been obtained using an incorrect broken-loop scheme, where only the position loop was broken. This sometimes has led to an oscillatory closed loop characteristic (see for example the roll channel step response in Figure 5.9). Using this incorrect scheme does not affect the main results*

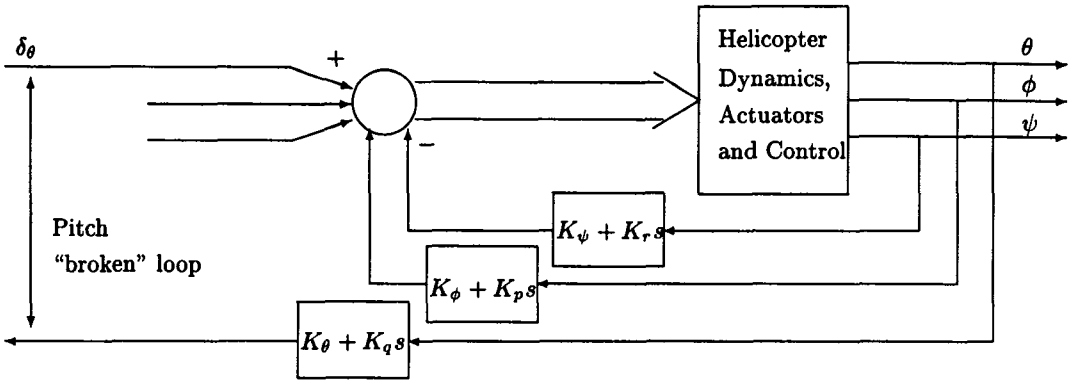

Figure 4.3: "Broken loop" scheme for stability margins.

and the conclusions presented in this report. Furthermore the updated computer code contains the correct scheme (see simu.m in Appendix A and the tutorial example of Appendix B).

4.3.3 Spec 3: Moderate-Amplitude Attitude Changes (Attitude Quickness)

Frequency domain-based criteria (e.g., bandwidth) fail in the presence of strong nonlinearities such as saturation. Therefore, in order to check the helicopter performance during large maneuvers we have to define alternative criteria. Recall that for a 2^{nd} order linear system with a step input the following "quickness" ratio

$$\text{quickness} = \frac{\text{peak rate}}{\text{steady-state displacement}} \tag{4.2}$$

is directly related to the system bandwidth. This ratio is also a good approximation for the system bandwidth for high order systems [16]. Using this criterion instead of one of the classical linear measures of bandwidth allows the specification to make the required bandwidth a decreasing function of the size of the maneuver, as shown in Figure 4.4. For nonlinear systems, especially with saturation nonlinearities, this concession is essential (i.e, it is unreasonable to require the same "bandwidth" for all input heights). In these cases it is not correct to interpret this ratio as a bandwidth,

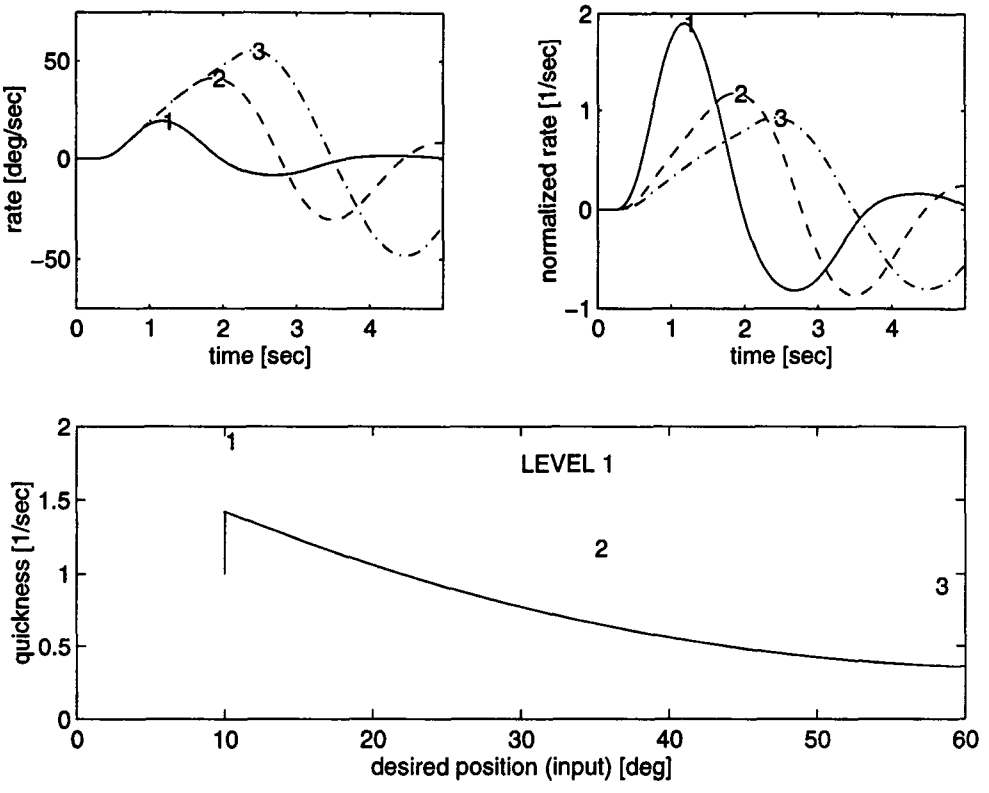

Figure 4.4: Spec 3: Attitude quickness.

but it is better interpreted as a measure of agility (i.e., "quickness" ratio). For more information see [16], Paragraphs 3.3.3 (Pitch, Roll) 3.3.6 (Yaw).

The calculation of the quickness ratio (4.2) is very simple. The max(·) operator is used over the sampled rate vector. It is then normalized by the input height. Note that using the max(·) operator in this case does not destroy smoothness because we try to maximize the quickness ratio and the combination max/max is smooth. This test theoretically has to be checked for an infinite number of inputs. Practically we check it for each channel for only three different angles (1,2,3) as shown in Figure 4.4 (for more details see *simu.m* in Appendix A).

By definition, for nonlinear systems the quickness specification depends on the input signal. In [12] the input signal is not explicitly specified. It is required that the pilot perform agressive (displacement) manuevers. This may be interpreted as step

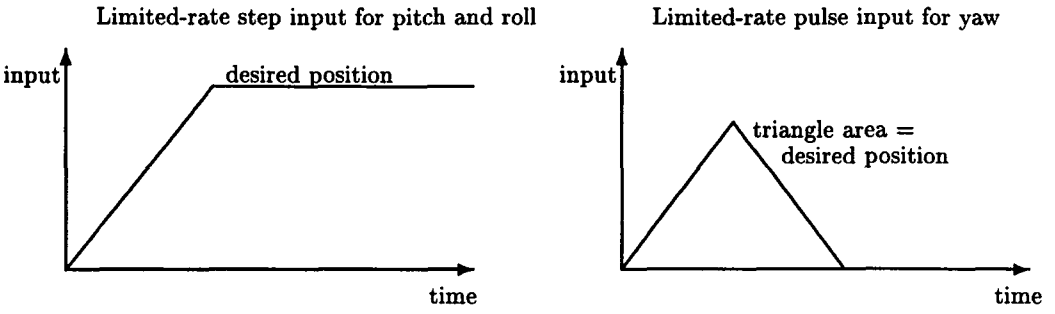

Figure 4.5: Input signals for Spec 3.

inputs for the pitch and roll channels, and a pulse input for the yaw channel (the yaw channel has an integrator in its desired model $M(s)$ [18]). Therefore, taking into account the rate limitations for the cockpit joystick and pedal [3], we have used the rate-limited inputs of Figure 4.5.

4.3.4 Spec 4: Interaxis coupling

Helicopter dynamics are naturally interaxis coupled. The following requirements relate to the two common helicopter interaxis couplings. These are the cross-coupling between pitch and roll (i.e., pitch/roll and roll/pitch), and yaw (rate) due to collective. These couplings are caused mostly by aerodynamic rotor moments and by the nonsymmetric tail moments. Both couplings would adversely affect the pilot's ability to complete some high maneuver tasks. One of the most important design objectives is to minimize these couplings.

The decoupling requirement is mostly significant for high maneuver responses. Thus, it is checked only for large amplitude "step" responses (The coupling measures are relative measures. Hence, in the case of linear systems small input amplitudes can be used as well). The natural measure is used for the cross coupling between pitch and roll (i.e., the ratio of peak off-axis response to desired response, θ_{pk}/ϕ_{des}, and ϕ_{pk}/θ_{des}). To avoid use of nondifferentable functions such as max(·) and abs(·), suitable upper and lower limits are defined to bound the response over the relevant time interval (i.e., C-O functional constraints [13]).

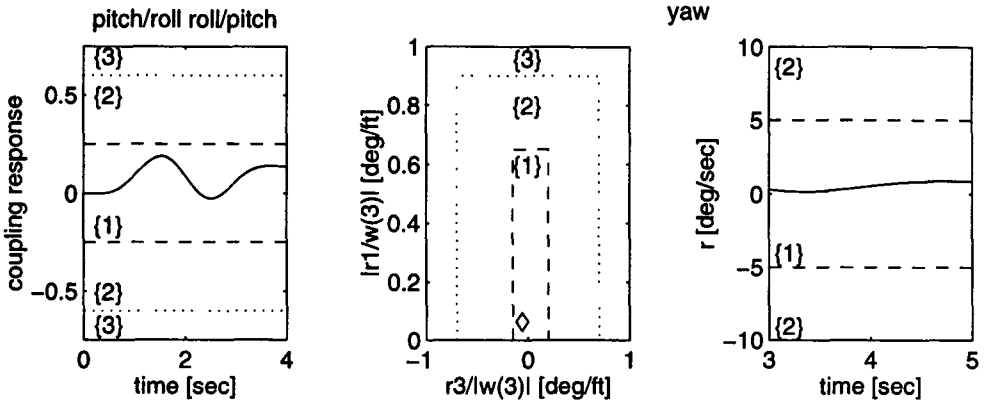

Figure 4.6: Spec 4: Interaxis coupling.

For the yaw-to-collective decoupling requirement a much more involved criterion is required. This criterion is designed to meet the pilot's needs during aggressive tasks. This criterion is a measure of not only the magnitude, but also the shape of the yaw rate response to a step collective stick input. The shape of the yaw rate is taken into account by measuring the peak yaw rate $r1$ and the value of yaw rate after 3 seconds. In case there is no peak in the time interval [0,3] the yaw rate at 1 second $r(1)$ is taken instead ([12], Paragraph 3.3.9). In addition to the above requirement, it is also required that the maximum oscillation amplitude, following a step collective change be below a certain limit. From accumulated experience this limit has to have units [12], (i.e., it is not relative as in the pitch-roll case).

All the information required for the coupling specs is taken from the largest input simulation of the quickness test (for more details see *simu.m* in Appendix A). An example for Spec 4 performance measures is given in Figure 4.6.

4.3.5 Spec 5: Wind-gust rejection

The model-following concept allows the designer to increase the I/O bandwidth (theoretically unlimited) by changing the parameters of the desired model $M(s)$ only. Actually, this is the major advantage of using a model-following control, because in most cases the designer can not achieve the desired bandwidth due to closed-loop

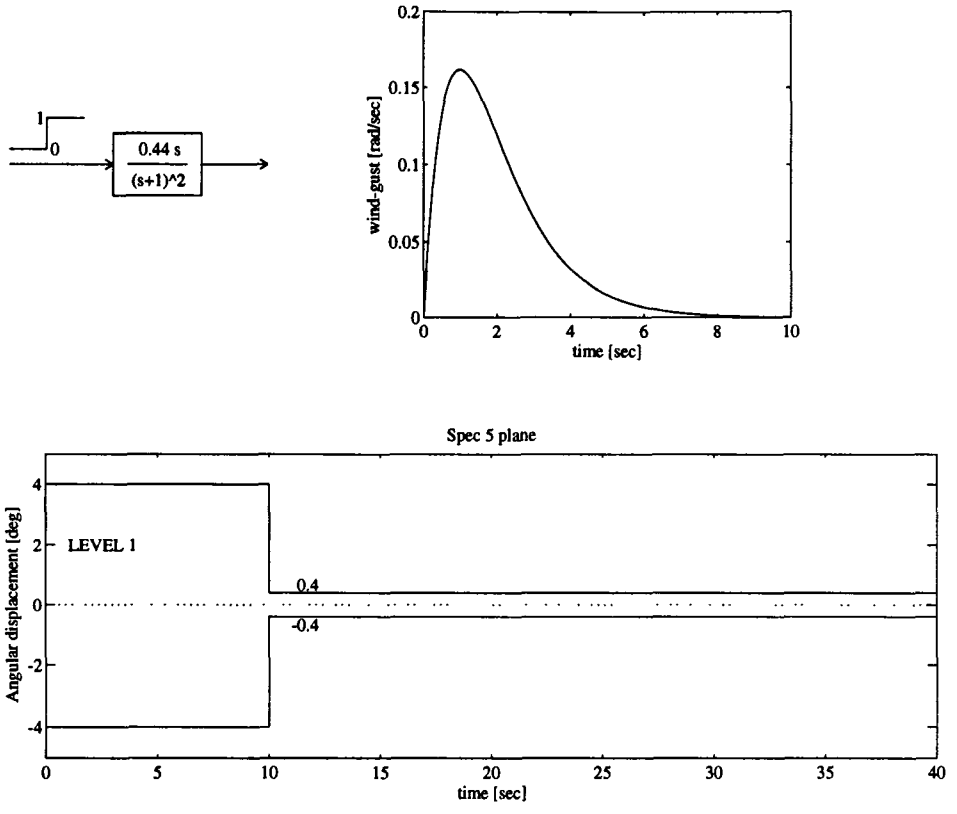

Figure 4.8: Spec 5: Wind-gust rejection - required envelope

stability limitations (e.g., limited state feedback gains). However, this approach has some disadvantages. One of them is that the disturbance rejection requirement (system "stiffness") is no longer directly related to the overall system bandwidth (i.e., we can have a high-bandwidth, low-stiffness system). Therefore, the original wind-gust rejection criterion ([12], Paragraph 3.2.6) is not suitable for the ADOCS configuration. A new requirement was defined by using an approximate gust model, where the gust peak value is chosen to fit the disturbance input point (recall from (4.1) that the wind-gust is applied directly to the helicopter state equations). The wind-gust input wave form is shown in Figure 4.7, the detailed definition of the new requirement is presented in [19].

The wind-gust rejection criterion is a two parameter criterion. Following precisely the requirements for the ACAH response type [12] leads to a pulse-input-settling-time

criterion. The first parameter, settling-time, $t_s(a)$ is defined by the condition that the absolute value of the response $|y(t)| \leq a \ \forall \ t \geq t_s(a)$, where a is 10% of the response peak value. The second parameter is the peak value itself. Practically, settling-time is obtained using a discrete-time search algorithm, which may cause $t_s(a)$ not to be smooth with respect to the design parameters. Moreover, in order to obtain the peak value we have to use the max$(\cdot)$ operator, and since the optimization algorithm tries to minimize this peak it may cause some smoothness difficulties. To avoid this possibility it is highly recommended that one uses functional constraints. Therefore, the ACAH response type requirement was slightly changed such that the angular position following a wind-gust input should lie between the two curves of Figure 4.8.

Remark 4.4 *Theoretically the wind-gust rejection requirement should be checked using the time history data from the nonlinear simulation. Because this test has to be simulated for 40 seconds it becomes the biggest time consumer of the simulation. In order to save time, this test is implemented using a linear simulation. The linear simulation holds as far as the actuators rates and displacements do not hit the saturation level. This (practically) holds when the system is stable (no counter example has been found throughout this research). Proper "flags" have been added to the simulation to ensure that the linearity assumption holds.*

4.4 The design parameters

An important component of the design setup is the choice of the proper design parameters (d.p.'s). The considerations for this choice are a mixture of classical control and optimization considerations. From the optimization point of view we would like to use as few d.p.'s as possible in order to save "running time". The control considerations, based on the classical ADOCS design of [15], are summarized in the following.

- For stability and closed-loop properties a position/rate ("PD") feedback is used. Therefore the rate (K_q, K_p, K_r) and the position $(K_\theta, K_\phi, K_\psi)$ feedback gains

are used as the optimization d.p.'s.

- Recall from Section 2.2.1 that the feedforward controller $F(s)$ consists of two parts. There is the "open-loop inverse" part which "cancels" part of the helicopter dynamics. This part consists of the following 1^{st} order approximation of $P(s)$, $P_a(s) = \text{diag}\,(\tilde{p}_\theta(s), \tilde{p}_\phi(s), \tilde{p}_\psi(s))$, with

$$\begin{aligned} \tilde{p}_\theta(s) &= \frac{M_{\delta_\theta}}{s(s+\frac{1}{\tau_\theta})} \\ \tilde{p}_\phi(s) &= \frac{M_{\delta_\phi}}{s(s+\frac{1}{\tau_\phi})} \\ \tilde{p}_\psi(s) &= \frac{M_{\delta_\psi}}{s(s+\frac{1}{\tau_\psi})}. \end{aligned} \tag{4.3}$$

  We want to keep the "dynamic cancellation" properties of the feedforwaed controller. Therefore the above approximate inverse parameters are fixed.

- The second part of the (model following) feedforward controller is the desired model $M(s) = \text{diag}\,(M_\theta(s), M_\phi(s), M_\psi(s))$, with

$$\begin{aligned} M_\theta(s) &= \frac{\alpha_\theta^2}{(s+\alpha_\theta)^2} \\ M_\phi(s) &= \frac{\alpha_\phi^2}{(s+\alpha_\phi)^2} \\ M_\psi(s) &= \frac{\alpha_\psi}{s(s+\alpha_\psi)}. \end{aligned} \tag{4.4}$$

  Note that the desired model is not an ADS-33C requirement. Therefore we can free $\alpha_\theta, \alpha_\phi$ and α_ψ as the optimization d.p.'s.

5 The nominal design

5.1 Introduction

For the nominal design we consider performance LEVEL 1 for all MTEs (Mission Task Elements) excluding target acquisition and tracking for UCE (Usable Cue Environment) 2 and 3. For detailed definitions see [12], Paragraph 3.3 - hover and low speed.

The ADOCS controllers [17], [15] were chosen as the initial controllers.

The design was completed in two steps. First we solved the feasibility problem (with no objective). Then we added an objective function and we found the optimal controller.

The starting design controller (initial guess) was chosen as the final ADOCS design of [18]. The performance map of this design is shown in Figure 5.1. Although this design has been tuned by both simulations and flight tests, 8 out of the 25 ADS-33C requirements are not satisfied.

5.2 Nominal design with the ADOCS control structure

The ADOCS flight control law is based on a decentralized SISO design. Using this concept, the only way to reduce the system coupling level is by using high feedback gains [15], [20]. Usually, high gains imply poor robustness (low stability margins). Therefore using the ADOCS structure with the multicriterion-optimization-based design process leads to "competing specs". That is, trying to satisfy one design specification (Spec 4) cause other(s) (Spec 2) to fail. Competing specs slow the design process and may cause the process to get stuck (even for smooth constraint functions).

In the first attempt, we tried to solve the feasibility problem or, at least, to find the "best" (infeasible) solution without changing the ADOCS structure. After many design iterations (C-O iterations and human designer interrupts) we concluded that it is impossible to simultaneously satisfy all the ADS-33C requirements using the ADOCS control structure. In the "best" design, as shown in Figure 5.2 only 3 performance measures are not satisfied.

Remark 5.1 *There is no definition for the "best" infeasible design (the best feasible design is the optimal one). However using engineering intuition and experience, it is possible to choose such a design. Moreover one may use C-O information and/or other design measures to quantitatively order the infeasible solutions.*

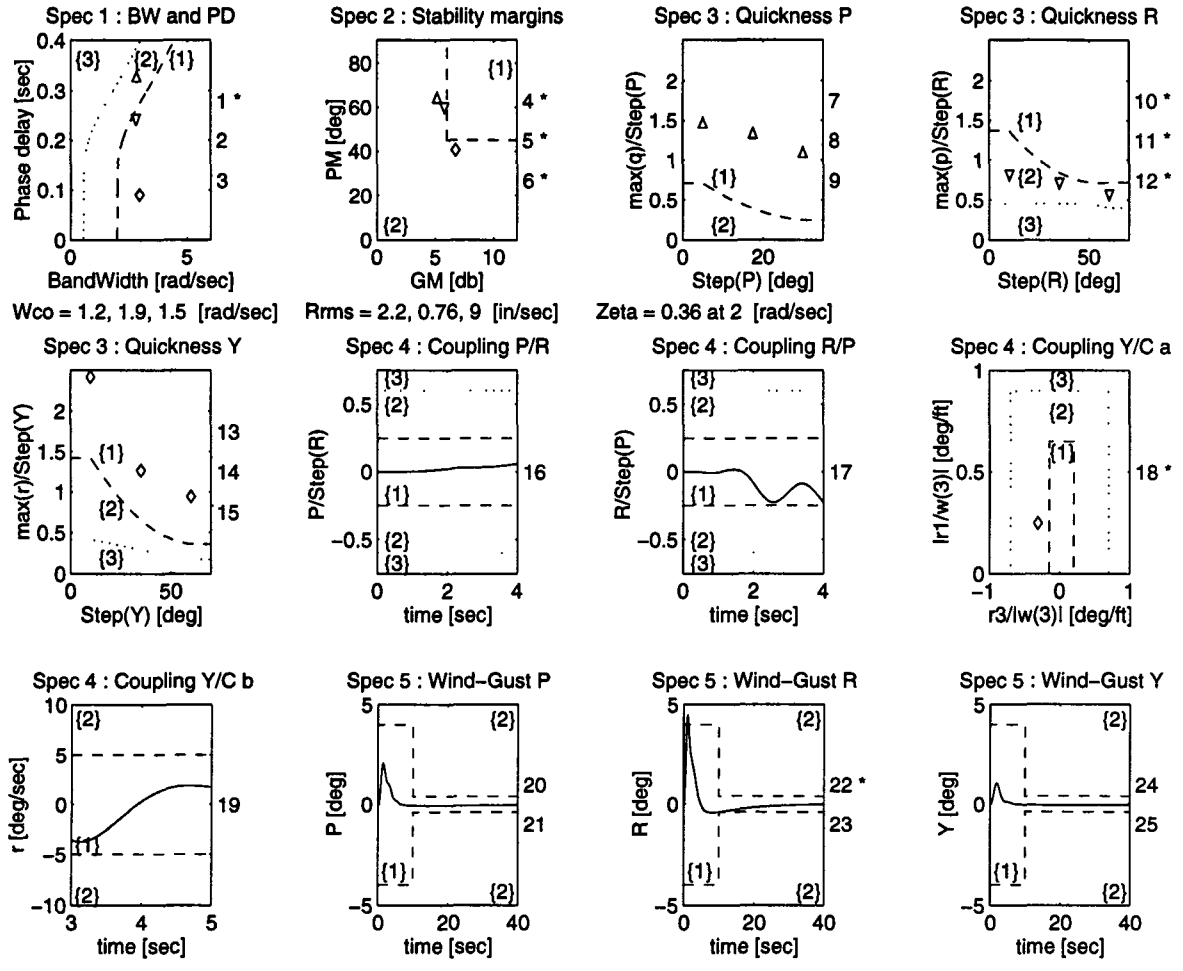

Figure 5.1: Performance map for the final ADOCS design; All 5 specs are shown in 12 sub-figures. Dashed line - LEVEL 1 / LEVEL 2 boundary. Dotted line - LEVEL 2 / LEVEL 3 boundary. Specs are numbered 1, 2,..., 25. $\triangle$ - pitch, $\triangledown$ - roll, $\diamond$ - yaw, $*$ - not in the desired level. For more details see *specs.m* in Appendix A .

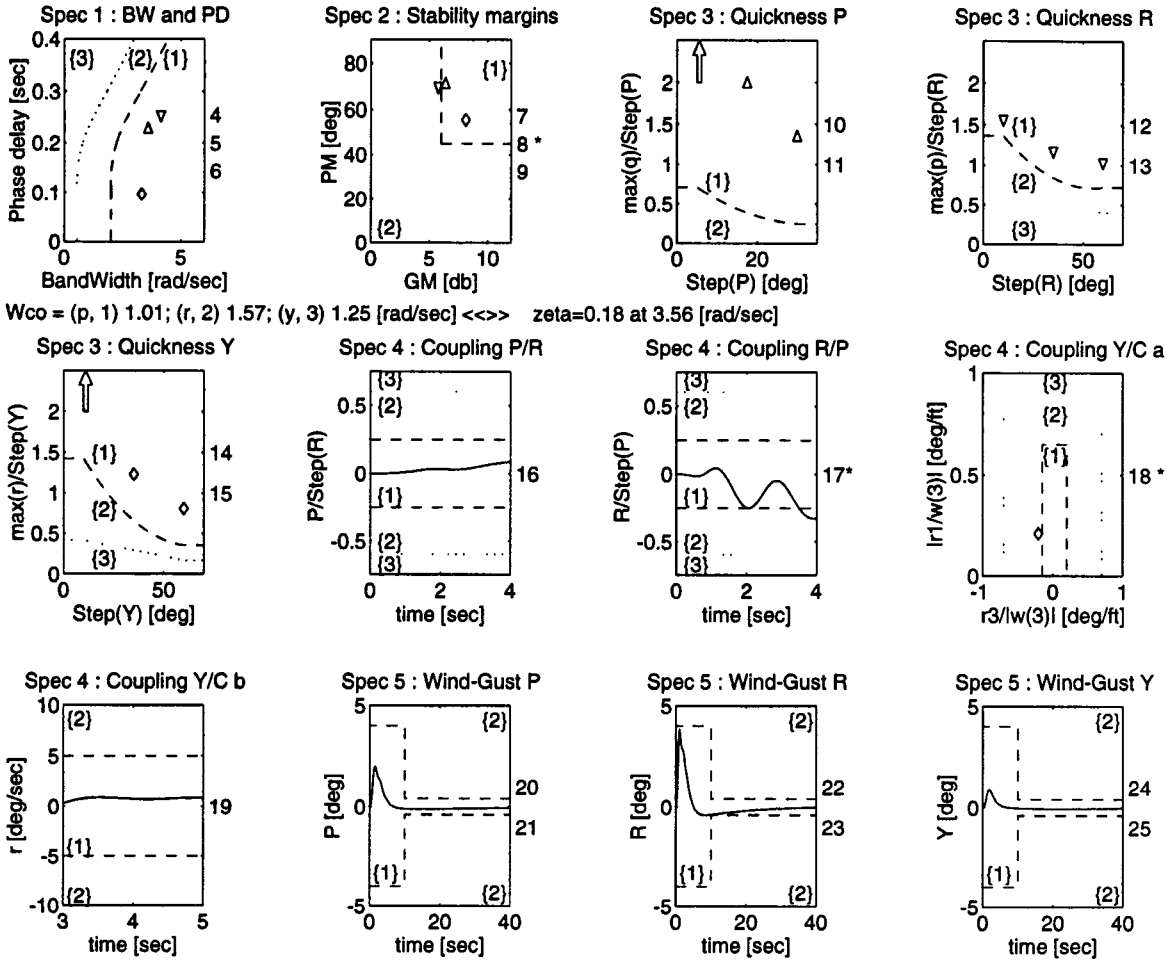

Figure 5.2: Best design using the ADOCS structure; $\triangle$ - pitch, $\triangledown$ - roll, $\diamond$ - yaw, $*$ - not in the desired level.

Remark 5.2 *It is difficult to prove that there is no feasible ADOCS controller. That is, there may be such a feasible controller which can be found after many design iterations. However then we can not consider it to be a practical solution. Furthermore, the same conclusion ("LEVEL 1 performance can not be achieved using ADOCS") was also obtained in [21].*

5.3 A feasible controller for the nominal design using an improved ADOCS control structure

Two different techniques were studied in order to improve the results of Figure 5.2. Recall that not any classical structure can be used as a good structure for multicriterion-optimization-based design. We were especially looking for a structure which required only a few additional d.p.'s.

In the first (SISO) technique we tried to keep the decentralized control structure of ADOCS and to add dynamic compensation to each channel. For example, we applied

$$C(s) = \frac{s+z}{s} \cdot \frac{\alpha s + w_c}{s + \alpha w_c} \tag{5.1}$$

to the roll and pitch channels, where z, α and w_c are additional design parameters. Using all six additional d.p.'s as free, C-O drives, $z \to 0$, $\alpha \to 1$, i.e., the compensator (5.1) is canceled. On the other hand freezing all/some of these d.p.'s, leads to new competing specs (e.g., Spec 4 vs. Spec 5).

In the second (MIMO) technique we tried to improve the decoupling performance of the controller by adding crossfeed compensation. In fact, in order to save d.p.'s we used only crossfeed gains. The implementation of the improved ADOCS control law is shown in Figure 5.3, where $K_{cf} = \begin{bmatrix} 1 & K_{\theta/\phi} & 0 & 0 \\ K_{\phi/\theta} & 1 & 0 & 0 \\ 0 & 0 & 1 & K_{\psi/c} \\ 0 & 0 & 0 & 1 \end{bmatrix}$ consists of static gains.

This very simple and cheap solution, has only 3 additional d.p.'s. Using this controller the helicopter performance is improved substantially as shown in Figure 5.4.

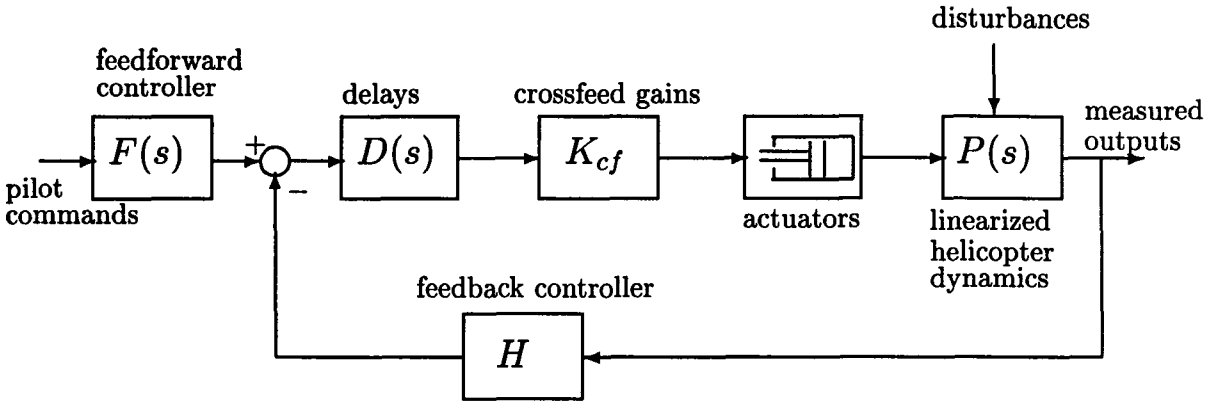

Figure 5.3: An improved (ADOCS + crossfeed) control structure.

Furthermore, although the additional d.p.'s make the C-O iteration a little longer, the overall design process converges faster (no competing specs).

Parameter changes for a "background run" of $\approx$ 100 C-O iterations (i.e., C-O ran 100 iterations using fixed weights and step size without designer interrupts) are shown in Figure 5.5.

5.4 The optimal design

The feasible design shown in Figure 5.4 is not unique. Changing some optimization parameters such as: initial guess, step size, etc., leads to other feasible solutions. In order to find the best (optimal) solution, we first have to define the design objective.

In this research the design objective is to minimize the helicopter actuators energy. However, using the simple actuator model of Figure 4.1 allows us to compute only rates and displacements of the control actuators. In order to compute the actuator energy a more detailed model is required ($E = \int_0^T (rate \times load)\, dt$). Therefore, in order to save running time, we used the RMS actuator rates as an approximation for the RMS actuator power ($E \propto$ RMS power).

Using such an objective function, we expected the C-O solution to converge to some boundaries of the feasible set (LEVEL 1 curves). Choosing the feasible controller of Figure 5.4 as the starting point for this design, C-O chose this solution as the optimal

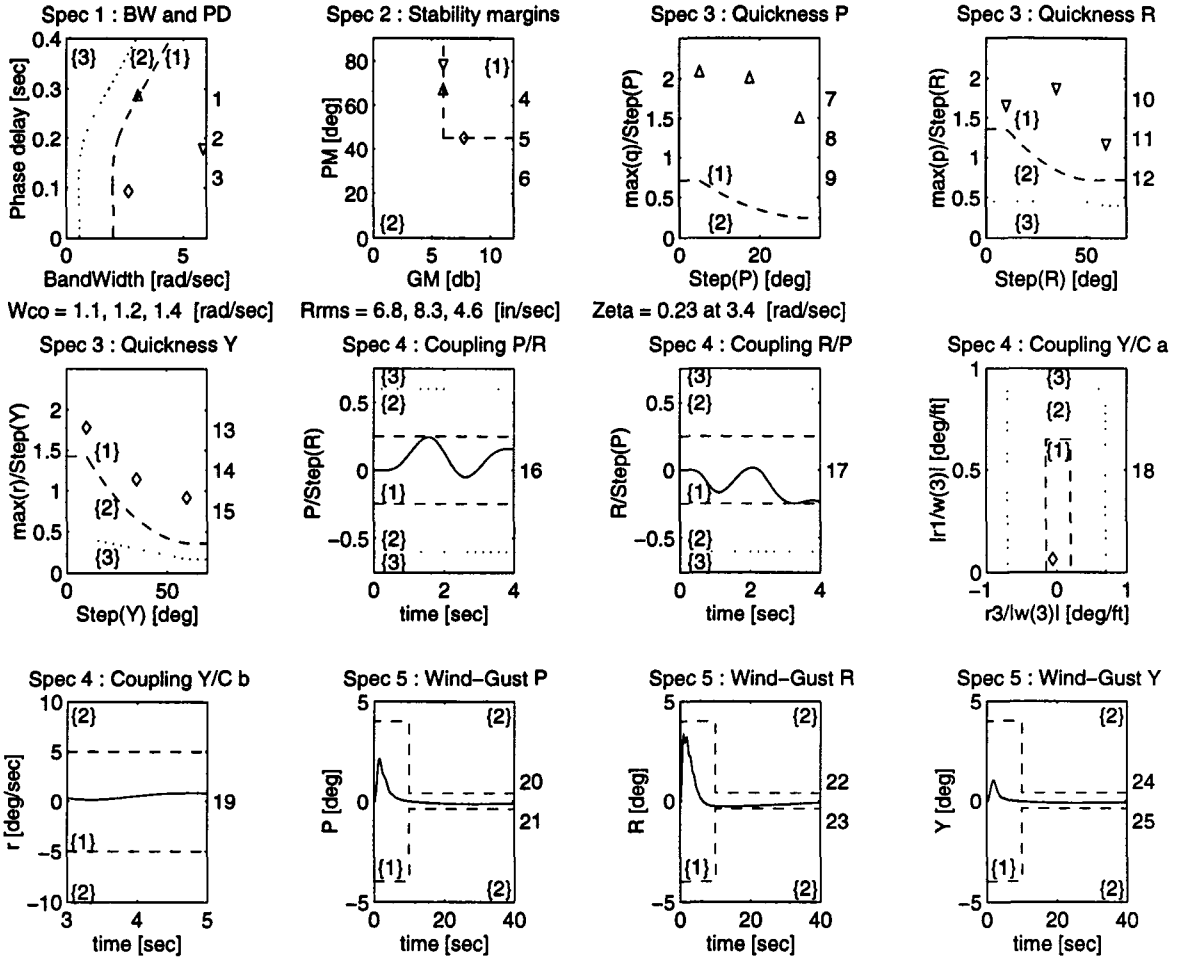

Figure 5.4: A feasible controller using the ADOCS structure with crossfeed gains; $\triangle$ - pitch, $\bigtriangledown$ - roll, $\diamond$ - yaw, $*$ - not in the desired level.

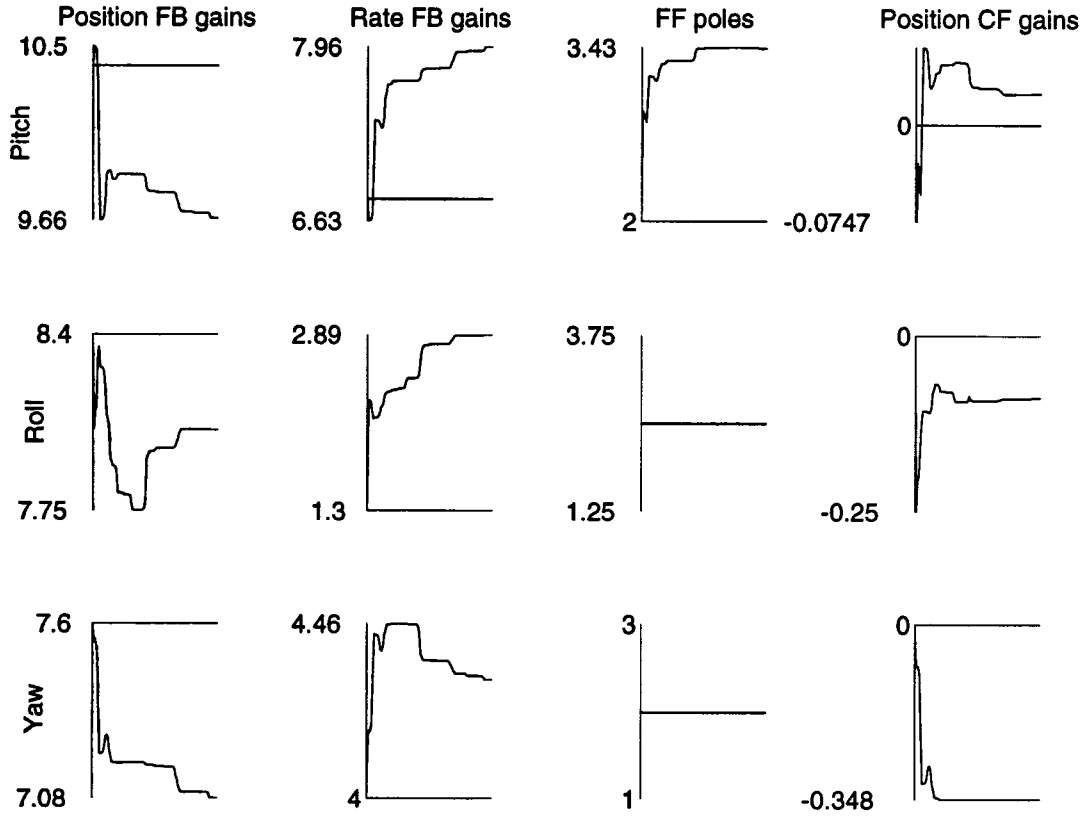

Figure 5.5: Parameter changes for a background run of $\approx$ 100 C-O iteration. Vertical axes - parameter values. Horizontal axes - iteration number.

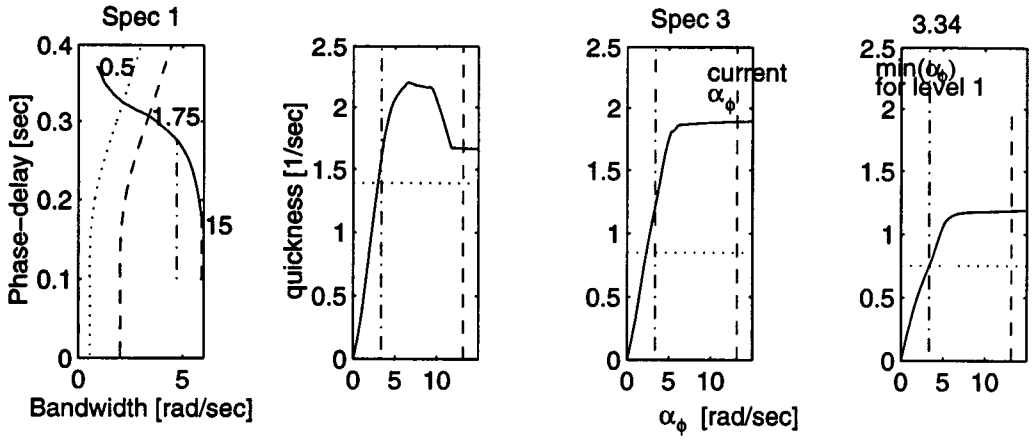

Figure 5.6: Defining the feasible set for α_ϕ. dashed line is the current value α_ϕ and the dash-dotted line is the minimum α_ϕ, which is 3.34 in this example, for achieving level 1 performance

one after only one iteration. In order to understand why this happened, we performed a local, one-dimensional analysis (see Section 2) as described in the following example:

- We identified α_ϕ as the only parameter which (locally) affects the quickness ratio of the roll channel (see "Spec 3: Quickness R" in Figure 5.4).
- We defined the feasible set (line segment) for α_ϕ as shown in Figure 5.6.
- We plotted the roll quickness ratio as a function of α_ϕ (Figure 5.7-a) for α_ϕ values including the current value (dashed line) and the boundary line (dash-dotted) of the feasible set.
- The other three plots (b-d) of Figure 5.7 are for alternative objective functions.

From Figure 5.7-a, we concluded that at the starting point in the d.p.'s space the objective function is "flat" (at least in one direction). Therefore C-O did not move from this point (the same is true for the yaw channel). Note that for the pitch channel the solution is already on the LEVEL 1 boundary of Spec 1 (Figure 5.7). Thus it is not possible to reduce the pitch channel quickness ratio.

Of course we can solve this problem (at least for the roll channel) using one of the alternative objective functions (c or d of Figure 5.7). However these measures

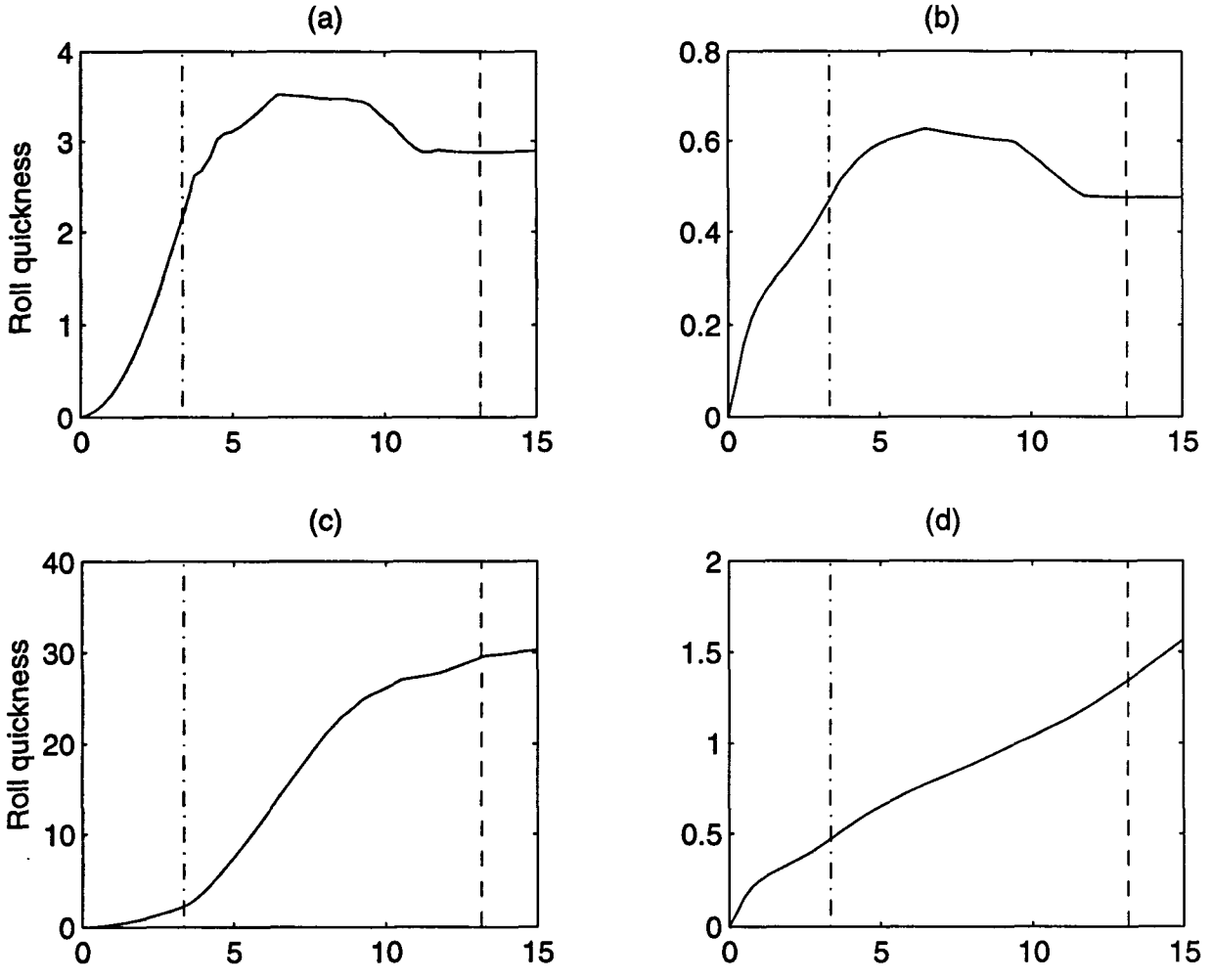

Figure 5.7: In all subplots: Horizontal axes - α_ϕ [rad/sec], vertical axes - Roll quickness [1/sec], dashed lines - current α_ϕ, dash-dotted lines - minimum feasible α_ϕ. (a) and (c) use the RMS actuator rate as the objective function. (b) and (d) use the RMS actuator stroke as the objective function. (a) and (b) consider the saturation of the actuator, while (c) and (d) do not.

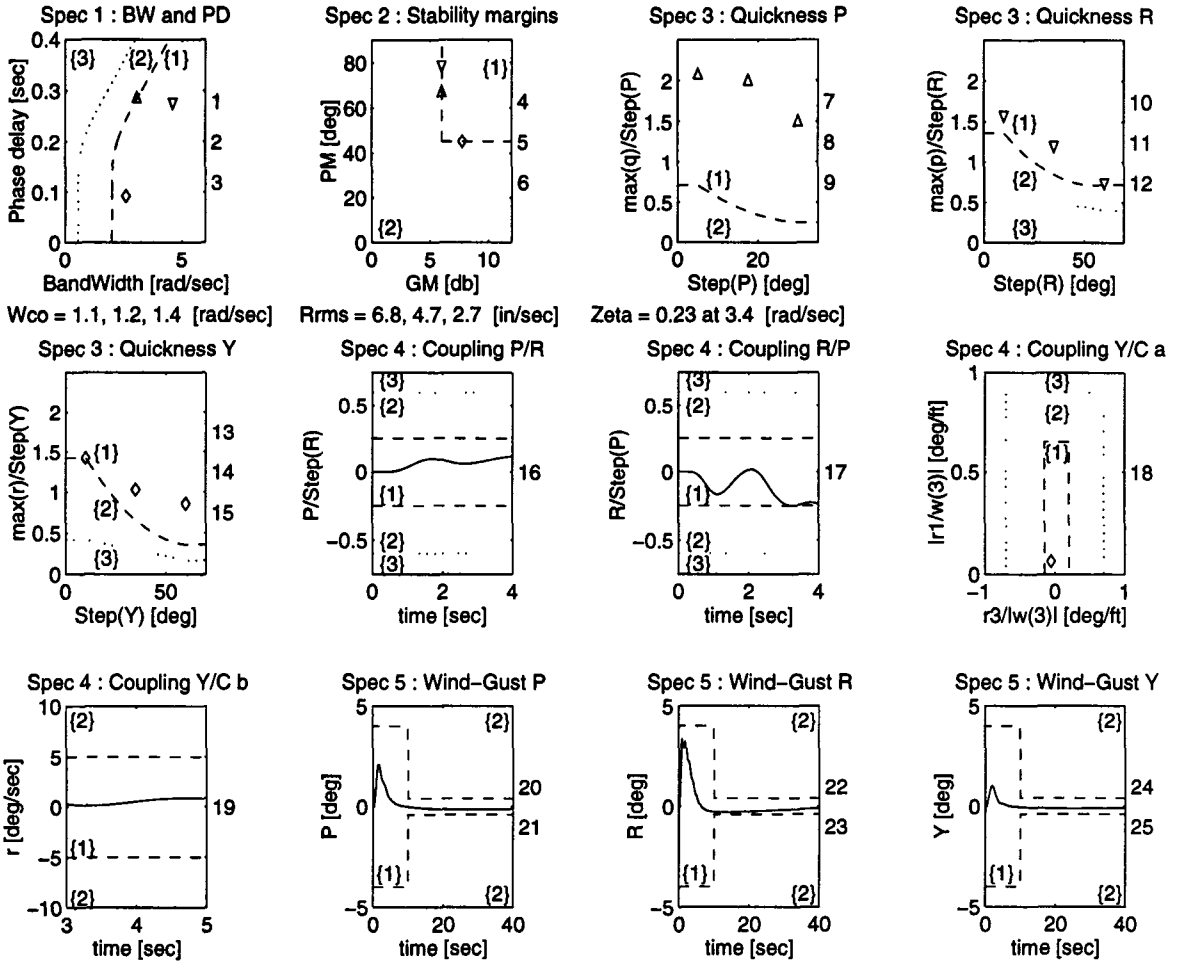

Figure 5.8: The optimal (nominal) controller; $\triangle$ - pitch, $\triangledown$ - roll, $\diamond$ - yaw, $*$ - not in the desired level.

may not be good a approximation for the actuator energy. A better solution is to "shake" the optimization using the ideas suggested in Section 3.4. In fact in this case we have manually set $\alpha_\phi = 5$. This made C-O converge to the LEVEL 1 boundaries as expected. The optimal performance map is shown in Figure 5.8.

Remark 5.3 *Using linear models, the cross-over frequency can be used as a good measure for the control energy. Then closed-loop parameters which affect the cross-over frequency (i.e., feedback gains), also affect the control energy. However, using the nonlinear model of Figure 4.1 the saturation effect is dominant over the linear effects.*

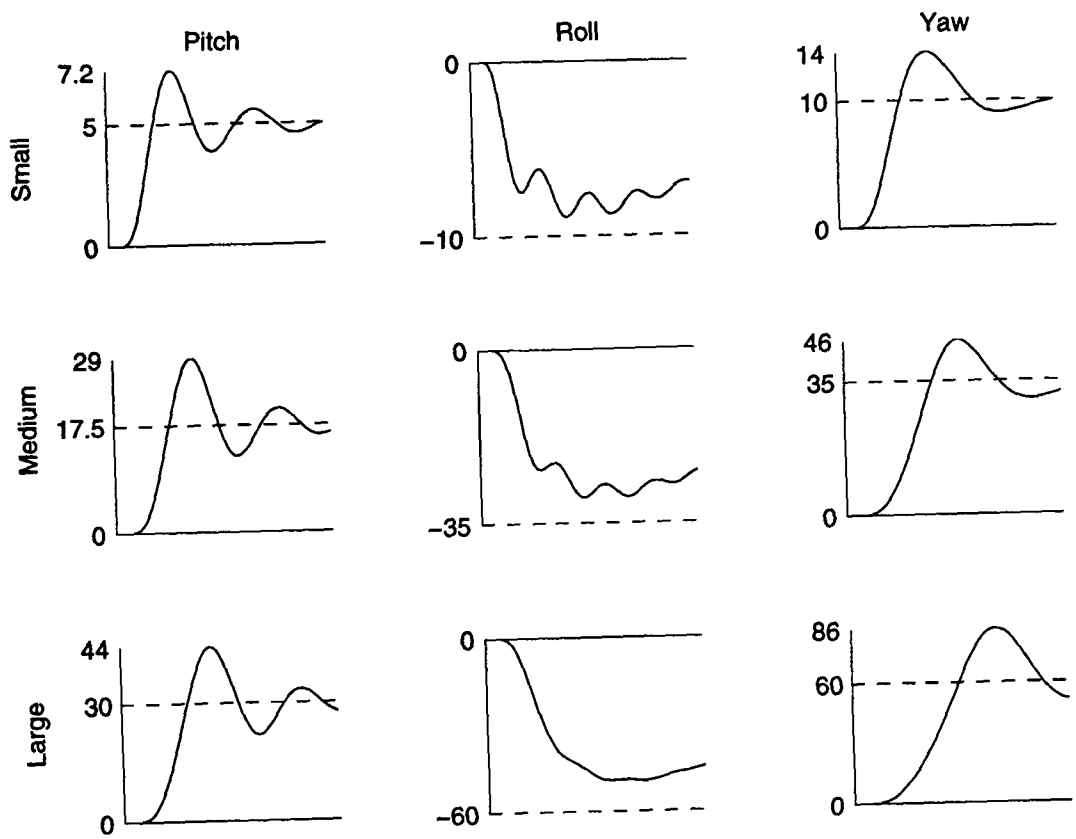

Figure 5.9: Attitude change for the nominal optimal design. Horizontal axes - time [0-5 sec]. Vertical axes - attitude [deg]. Dashed1+0.75 lines - desired attitude [deg].

Therefore minimizing the cross-over frequency does not necessary lead to minimum actuator energy. Moreover from Figure 5.8 we conclude that the feedback gains are already on the LEVEL 1 boundaries (pitch ⇔ Spec 1, roll and yaw ⇔ Spec 4). Thus they can not be further reduced.

The attitude change for the nominal optimal design is shown in Figure 5.9. Recall that using an incorrect broken-loop scheme led to oscillations in the roll channel response (see Remark 4.3).

6 Trade-off and robustness studies

6.1 Introduction

In this section we used the proposed design methodology to perform some trade-off studies. We checked the changes in the performance of the helicopter as some of the parameters change, while maintaining optimality.

Two main trade-off studies are presented:

- **Performance/specifications** - Going from the nominal optimal design (LEVEL 1) to higher performance levels.
- **Performance/hardware** - Changing some of the helicopter (actuators) parameters.

In order to make these trade-off studies quantitatively, a trade-off criterion is required. A natural choice for this criterion is the optimization objective (actuator RMS rate). In fact we used the following normalized measure

$$\text{actuator effort} \stackrel{\text{def}}{=} \frac{\text{actuator RMS rate}}{\text{actuator rate limit}} \in [0,1]. \tag{6.1}$$

In addition to the trade-off studies we tested the robustness of the FCS design to perturbations in hover flight conditions.

6.2 Performance to specification

In this set of trade-off studies we tried to "push" the helicopter optimal performance as high as possible. That is, we tried to drive the nominal optimal design (LEVEL 1) to higher performance levels. As the design target (the highest level) we have chosen LEVEL 1 for "target acquisition and tracking" ([12], Paragraph 3.3). This level is marked as "LEVEL 1+1" in Table 6.1 (note that LEVEL 1+1 $\neq$ LEVEL 2). Using this notation allows us to quantify the intermediate levels between LEVEL 1 and

MTE	UCE	Attention	Design level
target acquisition and tracking	–	–	"Level 1+1"
all others	> 1	–	"Level 1"
	–	divided	
	1	full	not considered

Table 6.1: ADS-33C design levels > 1 for hover.

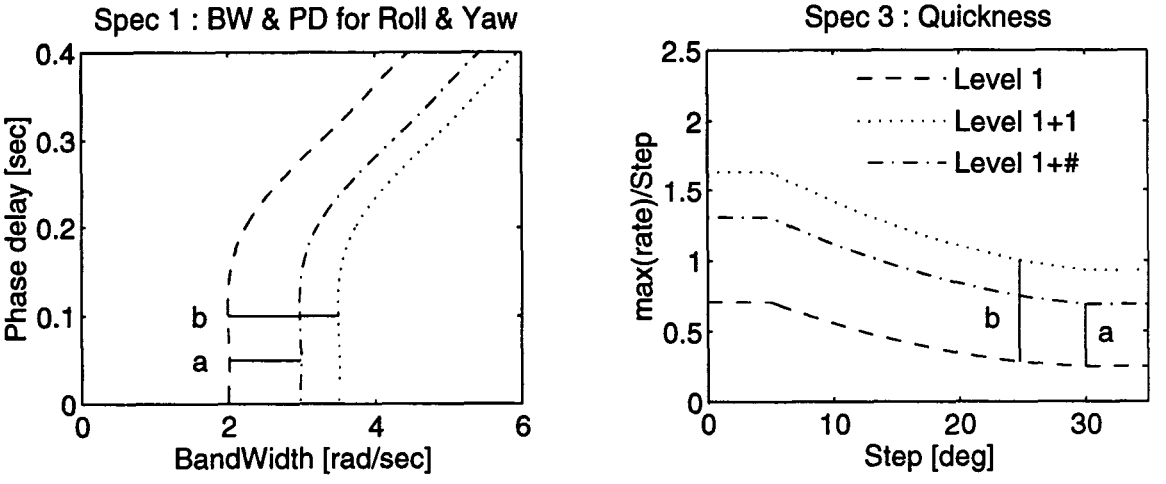

Figure 6.1: Definition of levels > 1. LEVEL 1+# where $\# = a/b$.

LEVEL 1+1 as shown in Figure 6.1. Recall from [12] that only Spec 1 (for roll and yaw channels) and Spec 3 (for all channels) are changed. In the first step, using only a few C-O iterations, we achieved the design LEVEL 1+0.5 of Figure 6.2. Note that the new dashed-dotted line indicates the boundaries of LEVEL 1+0.5 (for Spec 1 and Spec 3). However in the second step, when we tried to achieve the LEVEL 1+0.75 C-O got stuck in Phase 2 (see Figure 6.3).

Using the list of Section 2 we performed a one-dimensional analysis along the α_ψ axis of the d.p.'s space (all other d.p.'s are frozen). The quickness ratios of the three points of Spec 3 for the yaw channel, as a function of α_ψ, are shown in Figure 6.4 where the dotted line indicates the minimum required quickness ratio (the spec line). The feasible set for α_ψ is the intersection of the three line segments for which the quickness ratios of Figure 6.4 are above the spec line. Therefore the feasible set for LEVEL 1+0.75 is empty (the quickness ratio of the second point in Figure 6.4 is below

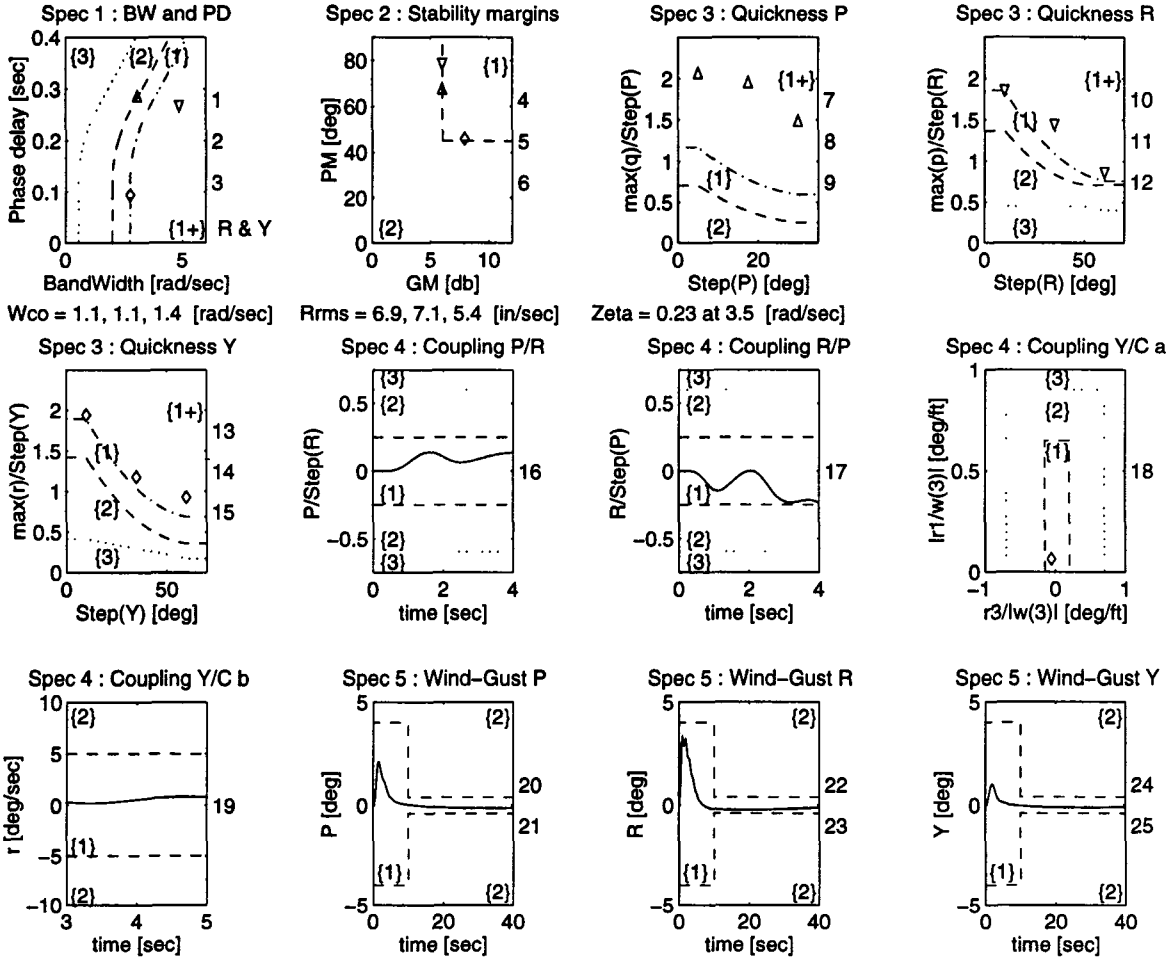

Figure 6.2: Optimal design for LEVEL 1+0.5; $\triangle$ - pitch, $\triangledown$ - roll, $\diamond$ - yaw, $*$ - not in the desired level.

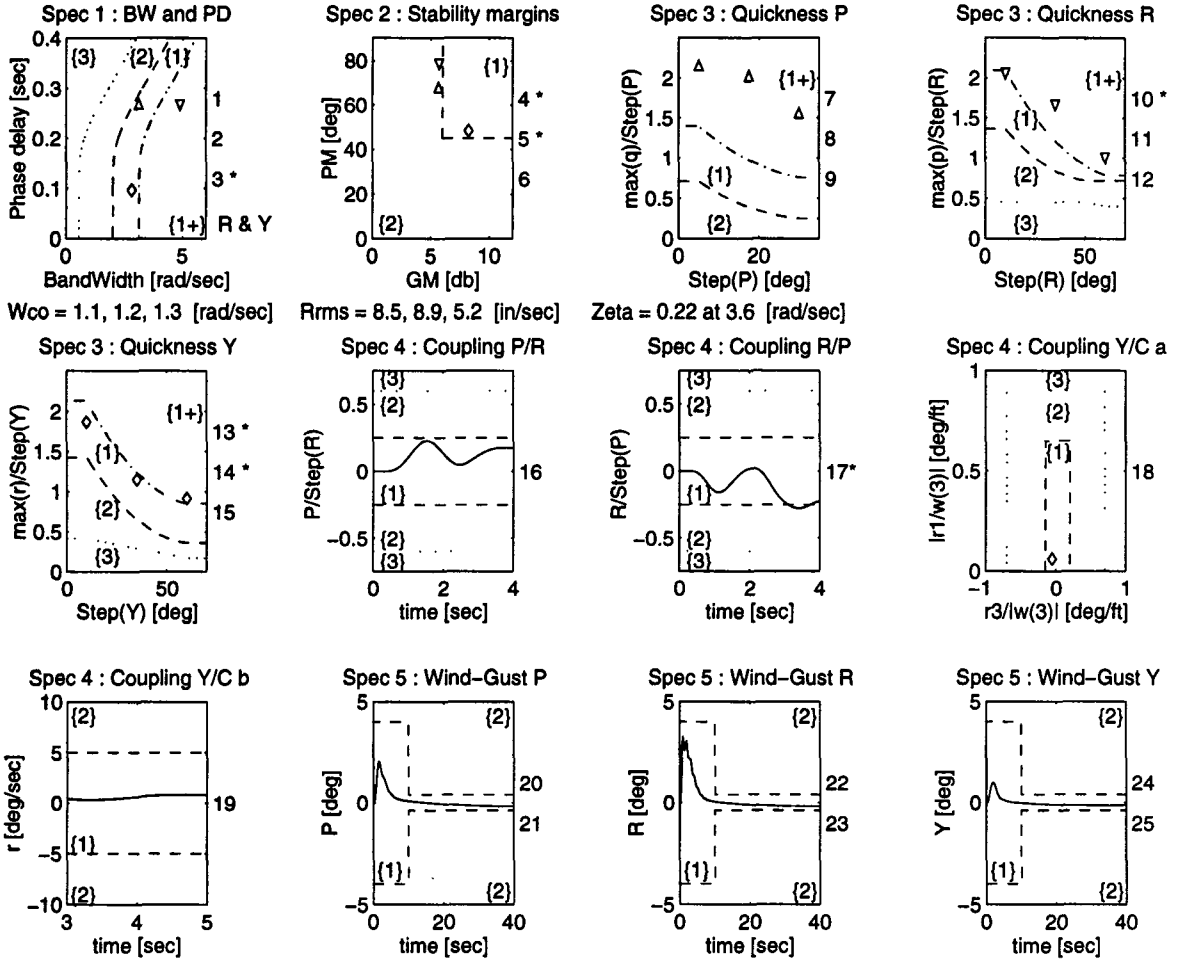

Figure 6.3: Final design for LEVEL 1+0.75; $\triangle$ - pitch, ∇ - roll, $\diamond$ - yaw, $*$ - not in the desired level.

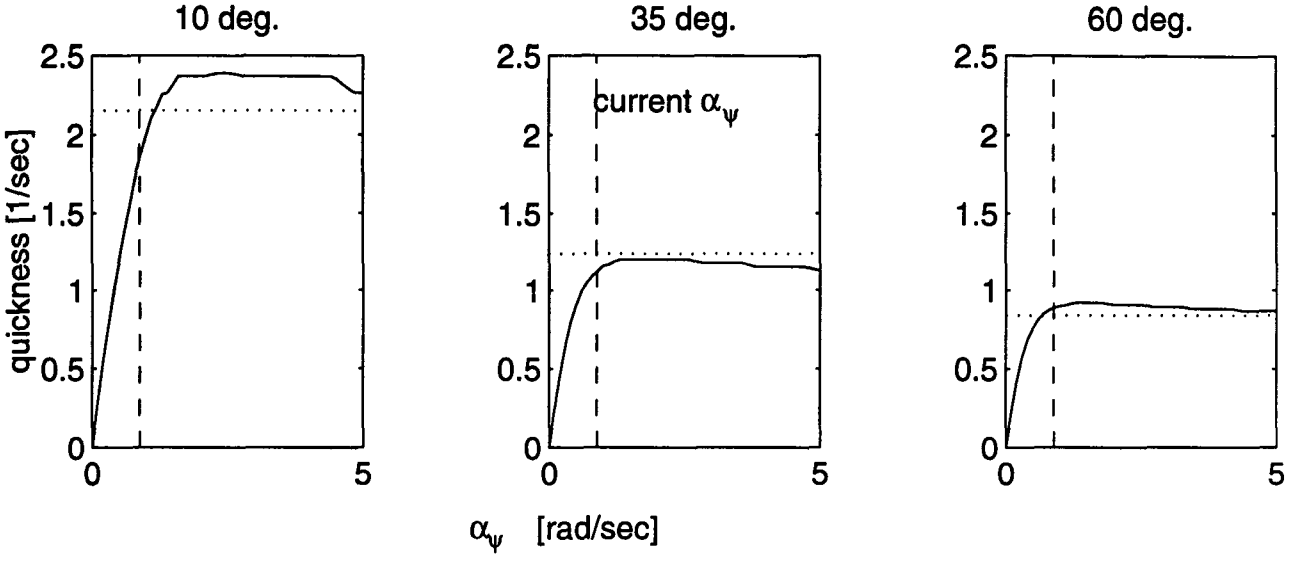

Figure 6.4: Yaw quickness ratios as function of α_ψ. Dashed lines - Spec 3 boundaries of Level 1+0.75.

the spec line for all α_ψ).

Reducing the specification level (spec line), we found that the maximum # is 0.65. In Figure 6.6 we can see the actuator effort as a function of # for all three channels. For the nominal design (# = 0) all actuators have about the same effort (20 - 30%). However increasing # (the required performance level) causes the effort of the tail rotor actuator (yaw) to increase while that of the main rotor actuators remain (almost) the same. That is, high level helicopter performance in hover is limited by the performance (size ?) of the tail rotor actuator.

The change of the d.p.'s as a function of # is shown in Figure 6.5. Note that C-O increases the feedforward poles α_ϕ and α_ψ to achieve the quickness requirement. The crossfeed gain $K_{\theta/\phi}$ is increased to compensate for the corresponding increase in pitch-to-roll coupling.

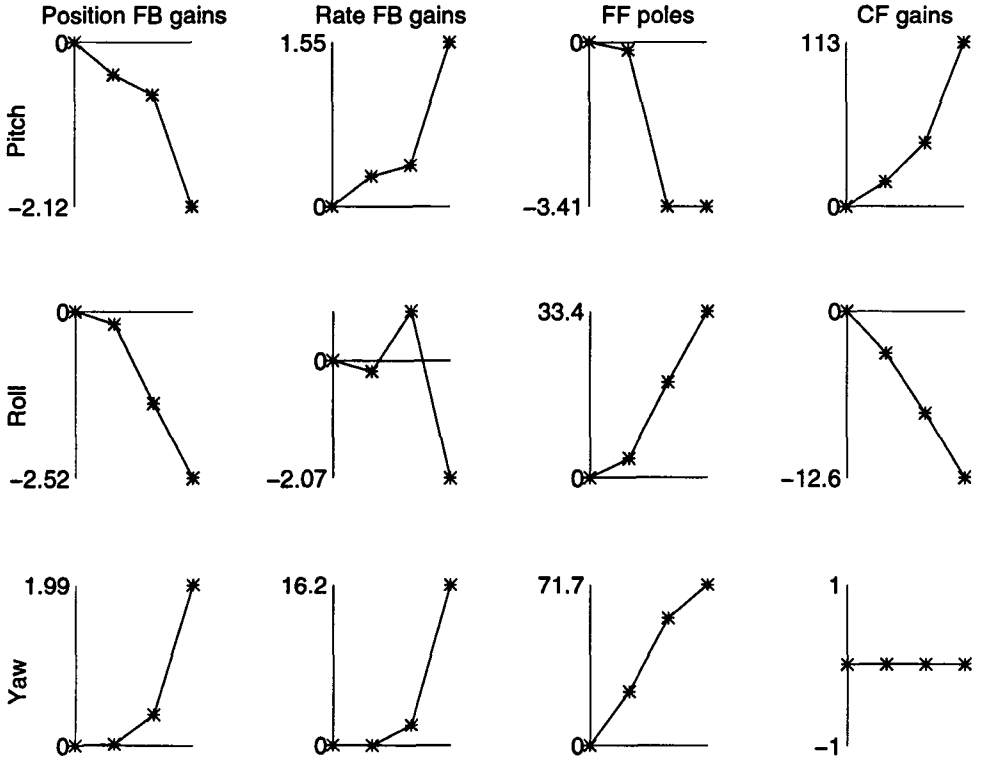

Figure 6.5: The percent change of the d.p.'s as a function of #.

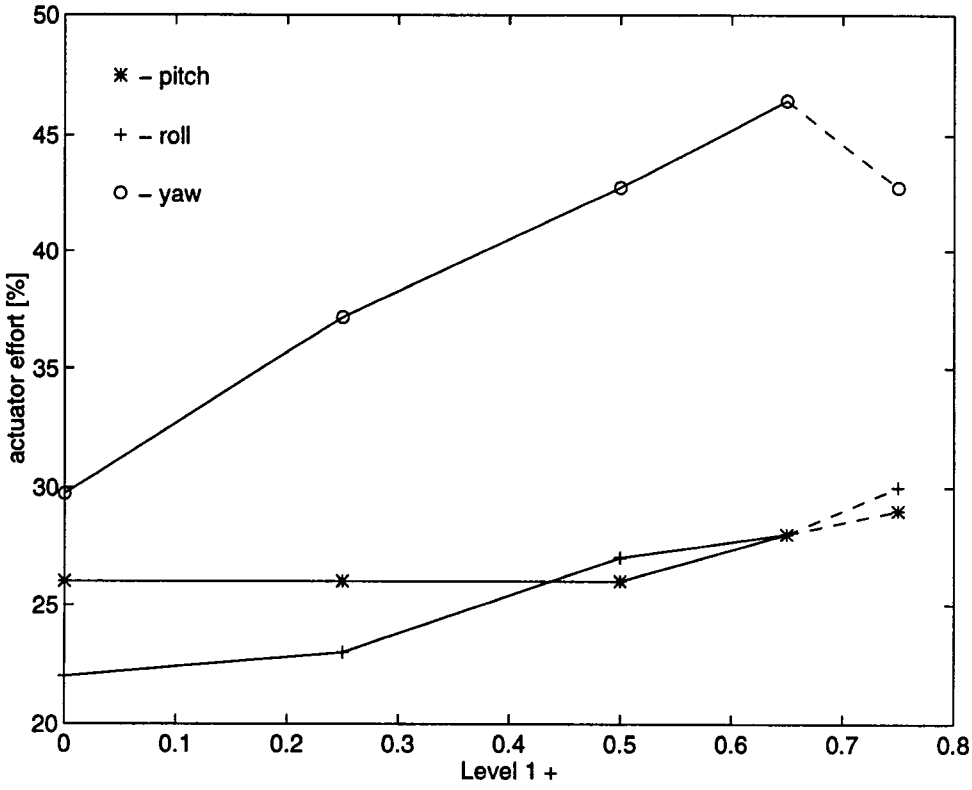

Figure 6.6: Actuator effort as a function of #.

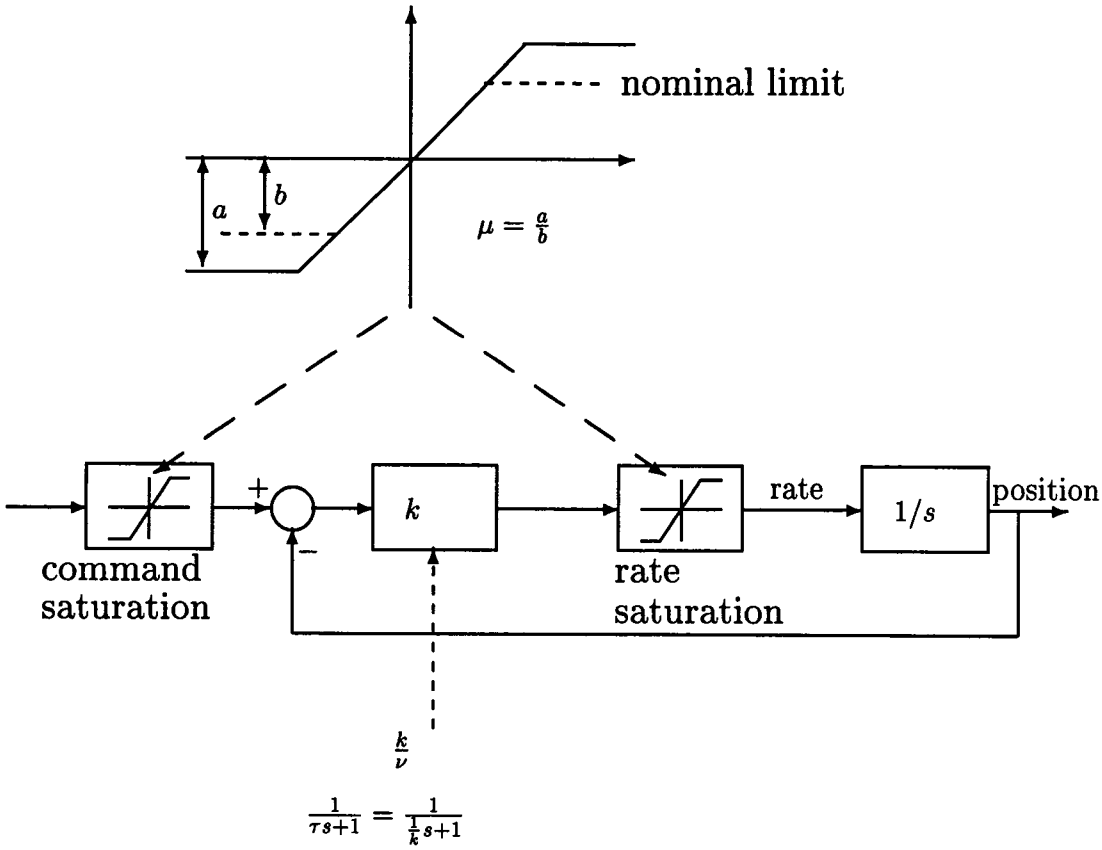

Figure 6.7: Definitions of the limits factor μ and the time-constant factor ν.

6.3 Performance to actuator limits

The actuator trade-off parameters are the rate and stroke limits and the time-constant. The corresponding factors $\mu \in [0.5, 2]$ (for limits) and $\nu \in [0.5, 2]$ (for time-constant) are defined in Figure 6.7. Note that each one of these parameters was changed simultaneously for all three channels.

The design for $\mu \in [0.75, 2]$ is (almost) the same as for the nominal design (see for example Figure 6.8 for $\mu = 2$). However trying to achieve LEVEL 1 performance with $\mu = 0.5$ C-O got stuck. Again we performed a one-dimensional analysis along the α_ϕ axis for three values of μ, 0.5, 0.625, and 0.75 as shown in Figure 6.9. We saw that, for $\mu = 0.5$ and $\mu = 0.625$, the feasible set of α_ϕ is empty (the quickness ratios for the third point are below the spec line). We also found that $\mu = 0.65$ is the minimum

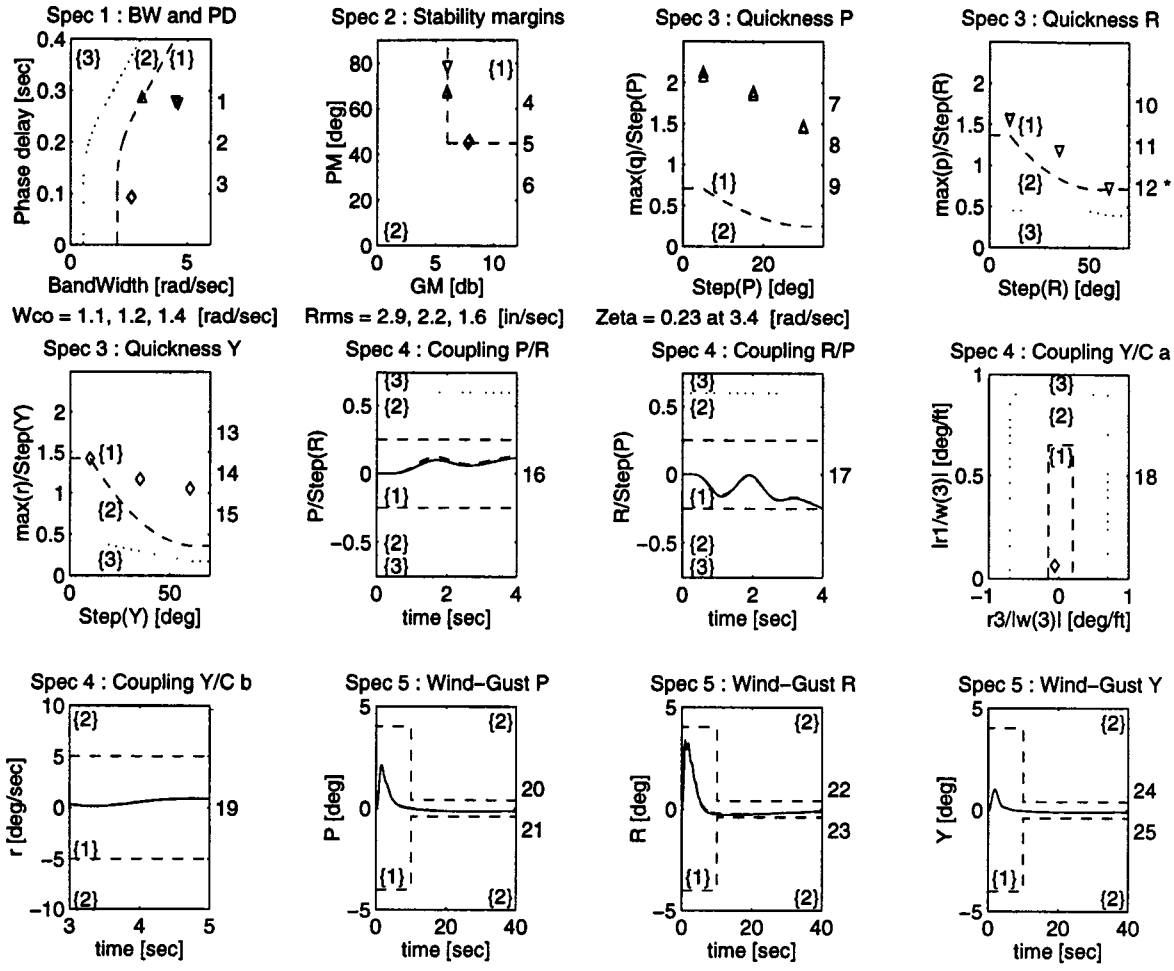

Figure 6.8: Nominal design for $\mu = 2$; $\triangle$ - pitch, $\triangledown$ - roll, $\diamond$ - yaw, $*$ - not in the desired level.

admissible limit factor. The performance map for this case is shown in Figure 6.10.

From Figure 6.11 it can be seen that the actuator RMS rates change very little as μ increases. This makes the actuator effort decrease approximately as $1/\mu$. Parameter changes as function of μ are shown in Figure 6.12. Note that generally rate saturation improves the effective "damping ratio" of the system. That is, increasing μ reduces the effective damping ratio. In order to compensate for this loss of damping, C-O increases the rate gains (especially K_p) as μ increases.

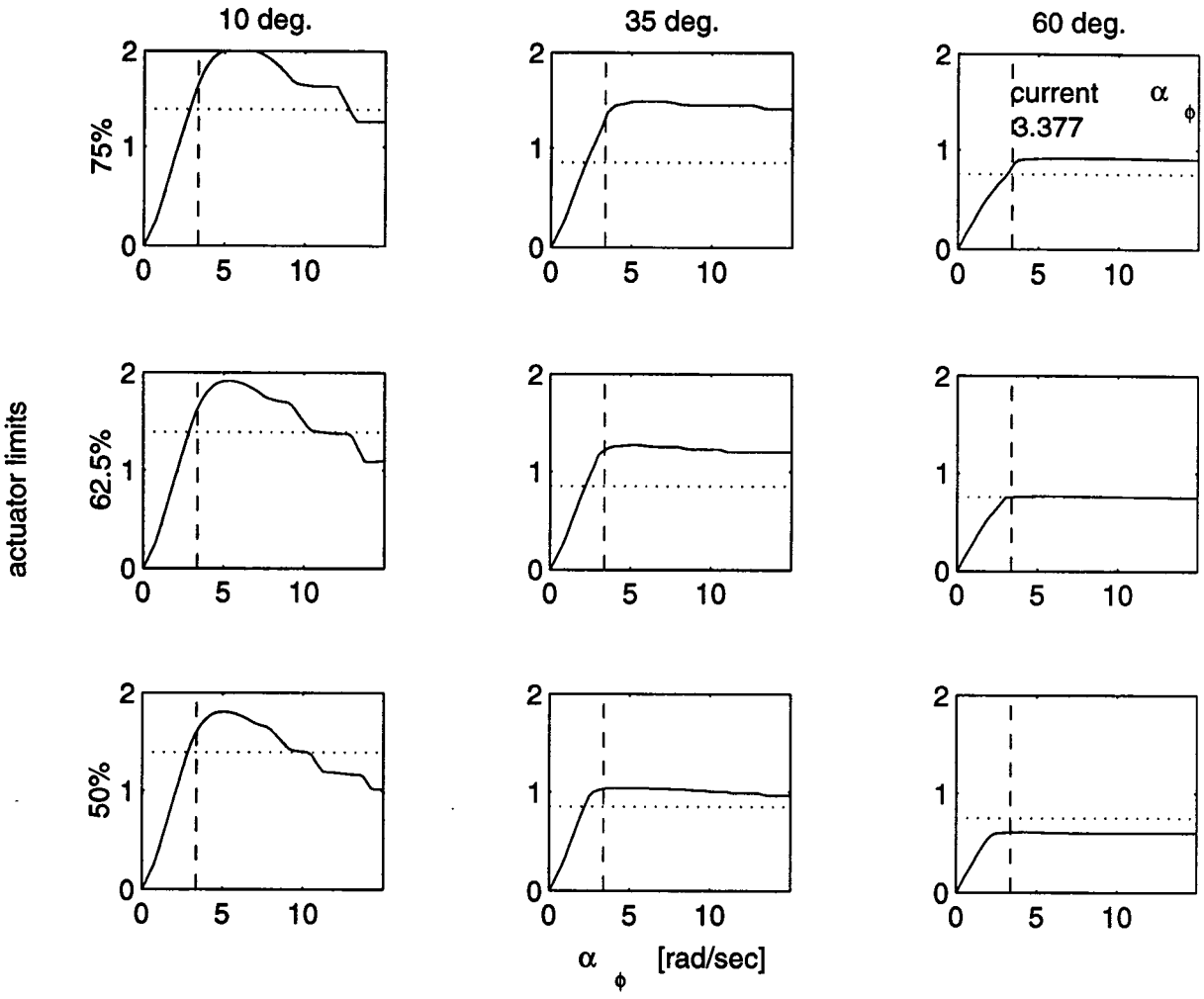

Figure 6.9: Plot of the roll quickness as a function of α_ϕ for different actuator limits. Horizontal axes - α_ϕ [rad/sec]. Vertical axes - quickness [1/sec]. Dashed lines - α_ϕ of the last iteration. Dotted lines - spec.

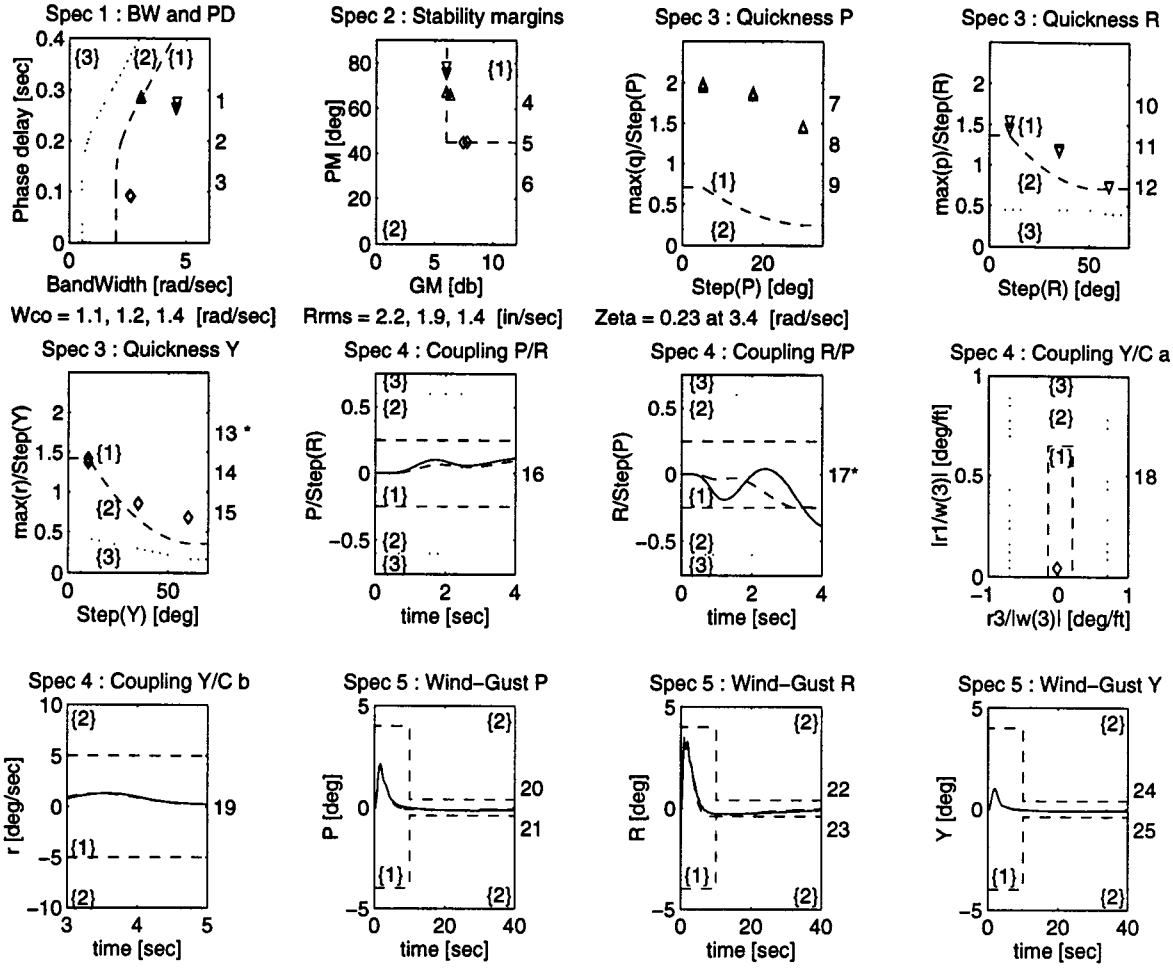

Figure 6.10: Nominal design for $\mu = 0.65$; $\triangle$ - pitch, $\triangledown$ - roll, $\diamond$ - yaw, $*$ - not in the desired level.

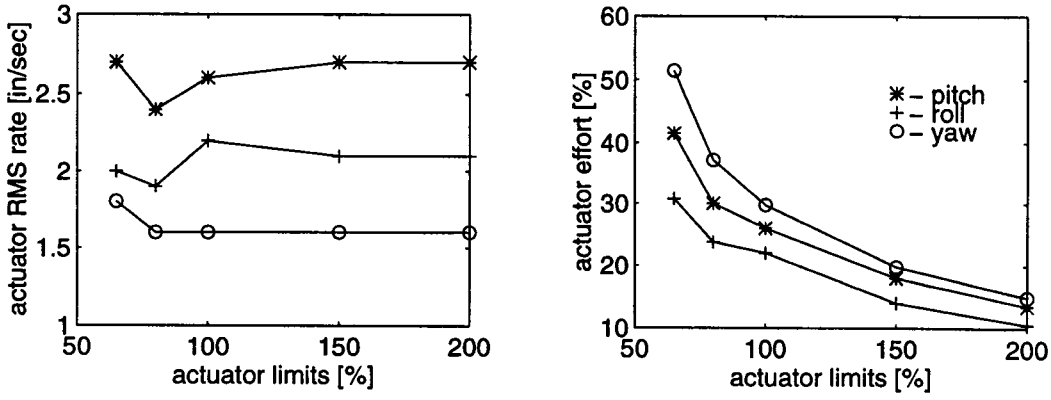

Figure 6.11: Actuator effort as a function of μ.

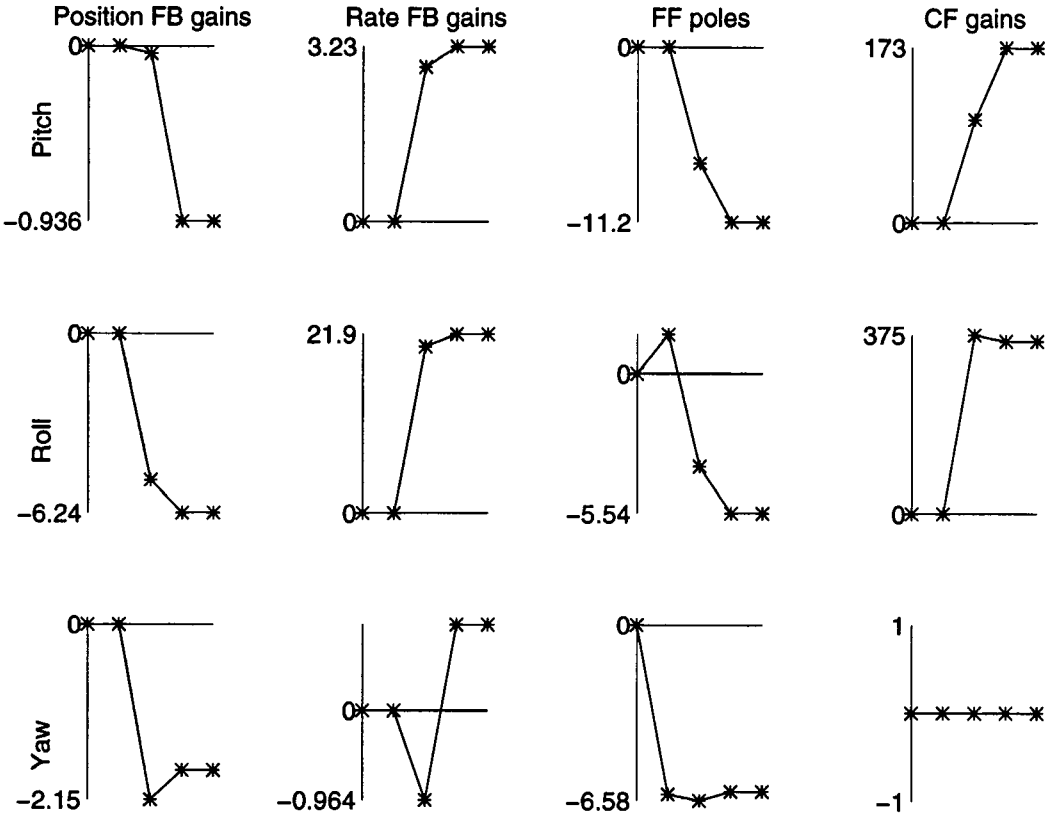

Figure 6.12: The percent change of d.p.'s as a function of μ

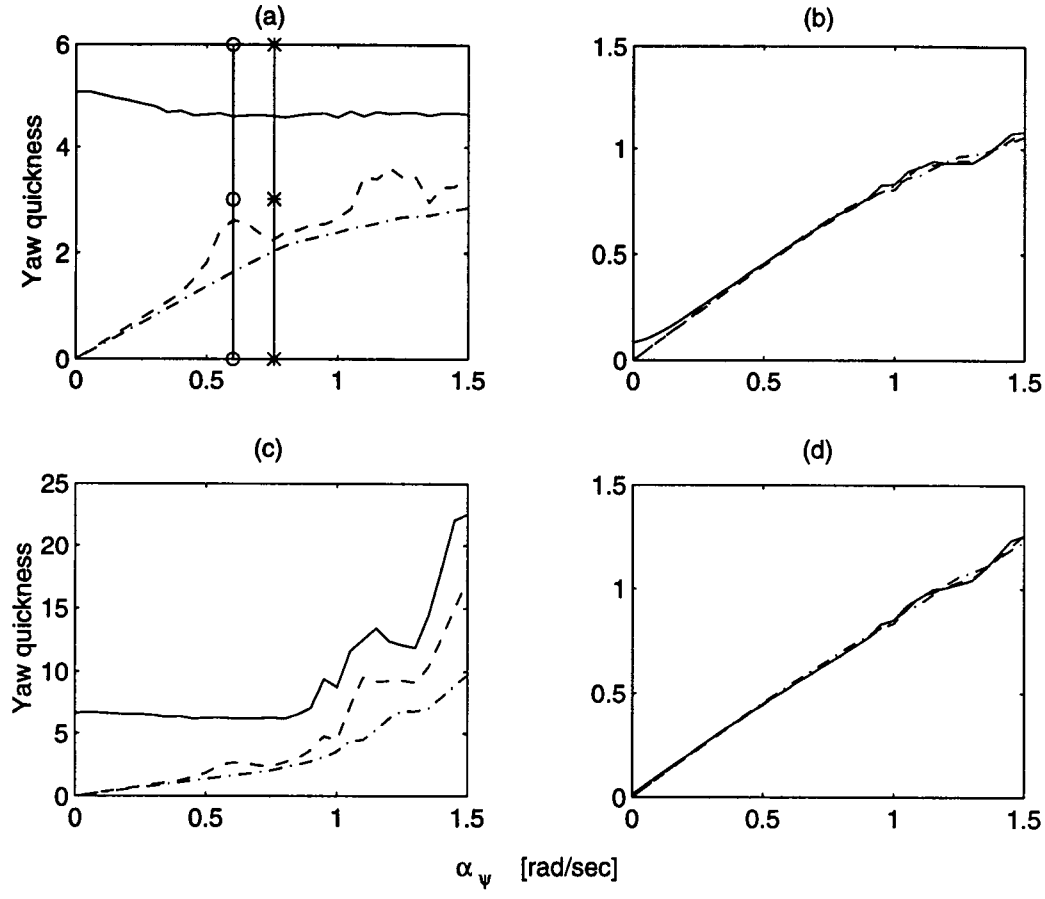

Figure 6.13: (a) and (b) consider the saturation, while (c) and (d) do not. (a) and (c) use the RMS autuator rate as the objective function. (b) and (d) use the RMS actuator stroke as the objective function. o - nominal design. * - optimal design. Solid line - 50% time constant. Dashed line - 65%. Dash-dotted line - 100%.

6.4 Performance to actuator time-constants

In the second performance/hardware trade-off study, we first checked "slower" actuators (i.e., $\nu \in (1,2]$). The helicopter optimal performance for $\nu \in (1,1.75]$ does not change much. However LEVEL 1 performance for $\nu = 2$ is difficult to achieve (design requires many human/C-O design iterations).

Apparently "faster" actuators should give better results. However checking $\nu =$ 0.5, C-O moved the nominal yaw quickness solution (performance) from (above) the boundaries of the feasible set (LEVEL 1). Performing a one-dimensional analysis along

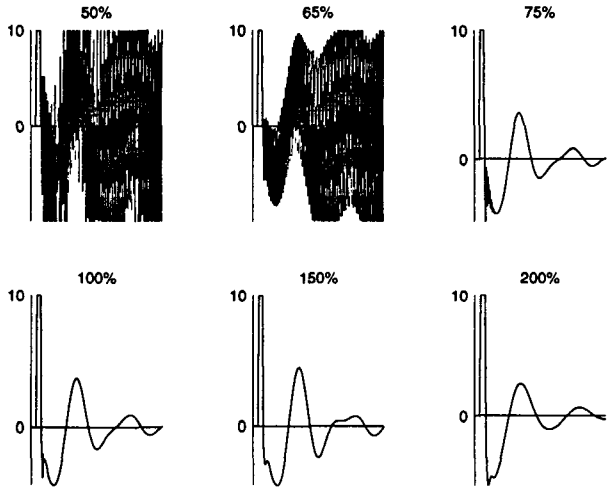

Figure 6.14: Pitch actuator rate as a function of ν

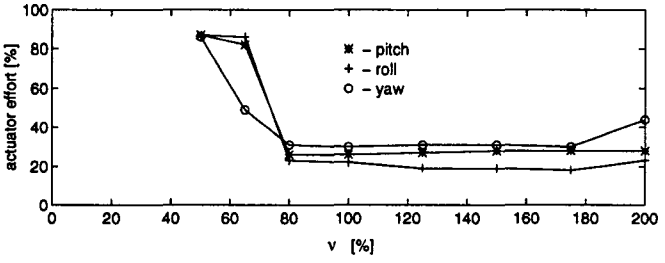

Figure 6.15: Actuator effort as a function of ν

the α_ψ axis (Figure 6.13), we can see that the objective function (Figure 6.13-a) is not smooth for all α_ψ. Note that $\alpha_{\psi_{\text{optimal}}} > \alpha_{\psi_{\text{nominal}}}$, thus the optimal performance is above the LEVEL 1 boundary (yaw quickness $\propto \alpha_\psi$). From Figure 6.13 we also conclude that the other objective function candidates are not smooth everywhere (although RMS stroke looked better). Decreasing ν for actuators with fixed rate and stroke limits causes the actuators to act (almost) like relays. Indeed from Figure 6.14 it can be seen that actuators with $\nu \leq 0.65$ have "limit-cycle" response type. That is, they may achieve the same performance level but with much higher effort as shown in Figure 6.15. Parameter changes for this trade-off study are shown in Figure 6.16.

6.5 Robustness tests

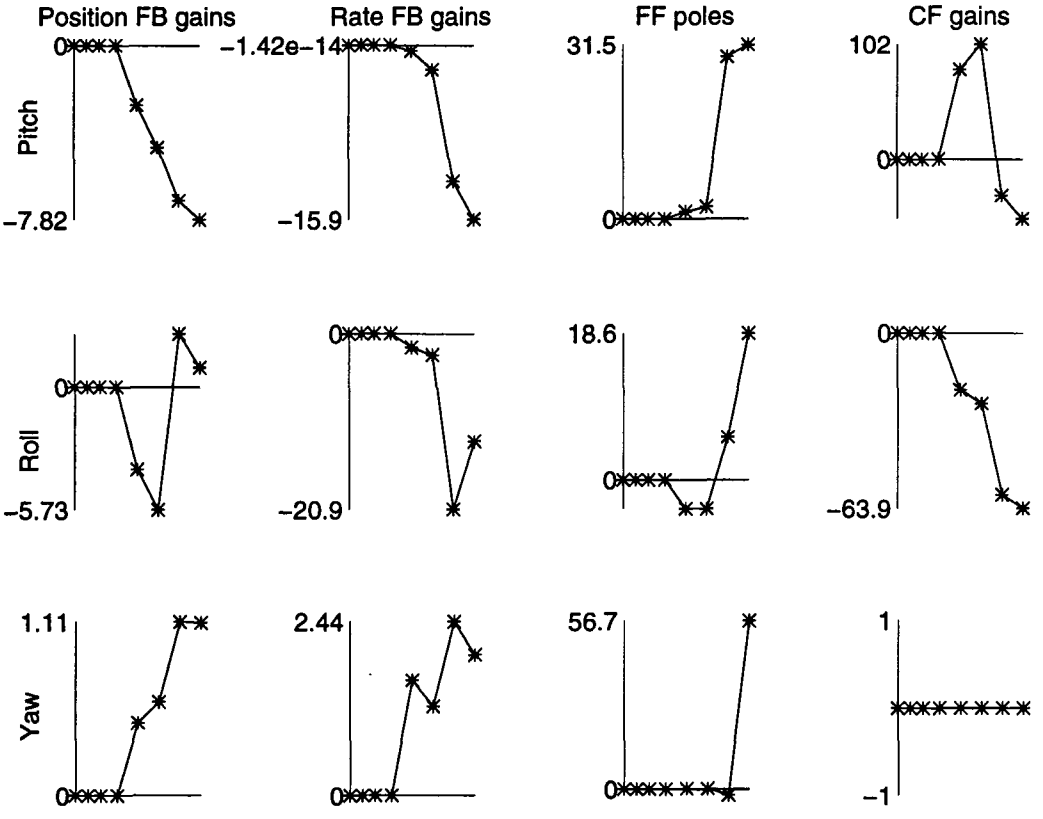

Figure 6.16: The percent change of the d.p.'s as a function of ν

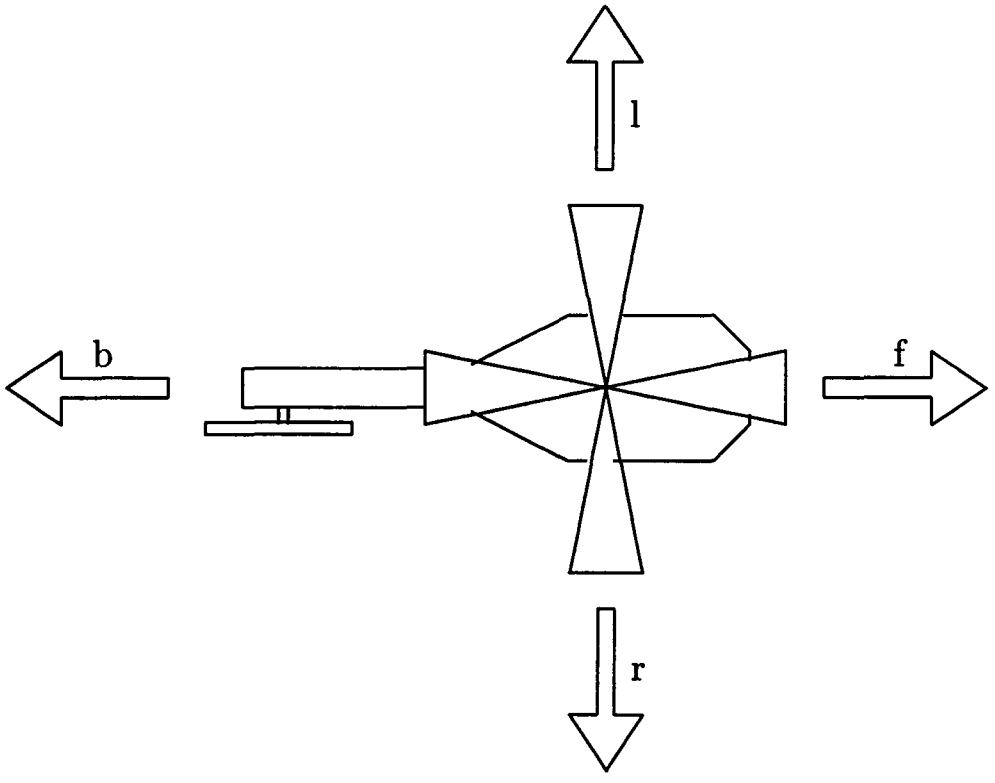

Figure 6.17: 10 kts perturbed hover level-flights.

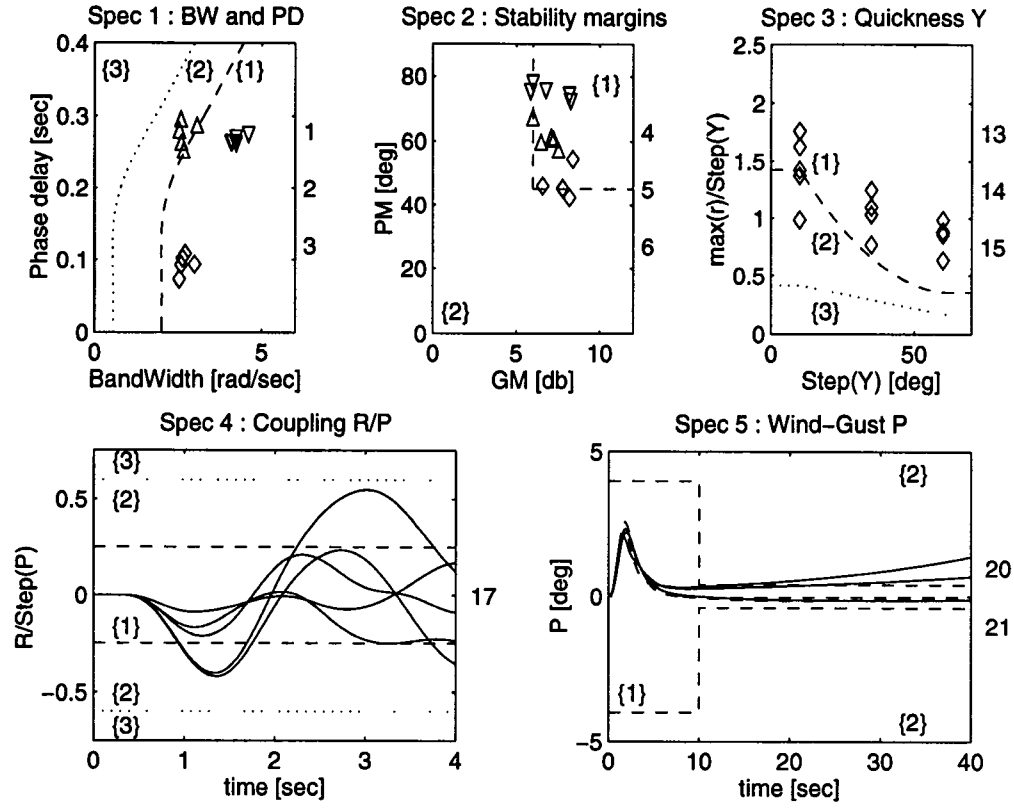

Figure 6.18: Performance for perturbed-hover; △ - pitch, ▽ - roll, ◇ - yaw.

The helicopter performance has been checked for four 10 kts perturbed level-flights about hover as shown in Figure 6.17. The results of the nominal optimal design applied to the perturbed-hover flight conditions are shown in Figure 6.18. Recall that using actuator "energy" as the design objective makes the optimal design converge to some (LEVEL 1) boundaries. Therefore using only this design methodology may lead to poor robustness. Robustness considerations may be included in the design process as it is implemented in the new NCD MATLAB toolbox [22].

7 Conclusions

7.1 Design methodology

- A new rotorcraft FCS design methodology, based on multicriterion optimization, was presented.
- Using this approach, the FCS designer uses optimization (C-O) to tune the parameters of a fixed-structure controller. The designer has to monitor the process continuously and interrupt if necessary.
- In addition to his "engineering" skills (knowledge, experience, and intuition), the designer has to develop "optimization" skills.
- Today's computers are powerful enough to make this method a real design tool.

7.2 UH-60A in hover - results

- Level 1 feasible and optimal solutions for the UH-60A in hover were found using this multicriterion-optimization-based design method.
- Higher performance levels may be achieved without changing the helicopter parameters, but higher actuator effort is required. The "Level 1 + 1" can not be achieved with the current actuators.
- For level 1 performance, increasing actuator limits does not improve the rotorcraft performance or the optimization criterion (except for the obvious reduction of actuator effort).
- The actuator time-constant has naturally an upper bound (200%). Practically it also has a lower bound ($\sim$ 70%). Below this bound the actuator may oscillate as can be seen in Figure 6.14.
- Using an objective function which "pushes" the optimal solutions to the boundaries of the feasible set leads to poor robustness.

7.3 Future work

Many theoretical and practical questions remain open for further research. First, in order to demonstrate the usefulness of this design methodology, it should be applied to other design tasks (e.g., other flight conditions, other helicopters, etc.). The human/process interface has to be simplified so this method can be used as a real design tool. In addition some special issues in modeling, translating handling qualities requirements, improving robustness, etc., have to be further studied.

Second, the specific design problem (UH-60A in hover) has to be completed, including: rerunning the trade-off studies for the correct "broken-loop" scheme (see Section 4), updated parameters (e.g., smaller time-delays), and updated specifications (e.g., Spec 3 for yaw channel).

References

[1] J. J. Gribble & D. J. Murray-Smith, "Command Following Control Law Design by Linear Quadratic Optimisation," *The 16-th European Rotorcraft Forum, Paper III.5.3*, Glasgow (Sept., 1990).

[2] M. D. Takahashi, "Design of Flight-Control Laws for a UH-60 Helicopter in Hover Using an $\mathcal{H}_2$ Loop-Shaping Synthesis Method," NASA, US Army ASC, NASA Tech. Memo. 103834, USAAVSCOM TR 91-A-006, Dec., 1991.

[3] M. D. Takahashi, "Design and Comparison of Pitch-Roll $\mathcal{H}_\infty$ Control Laws With and Without Rotor-State Feedback for a Hovering Helicopter," NASA, US Army ATCOM, NASA Tech. Memo. 108793, USAATCOM TR 93-A-013, Jan., 1994.

[4] M. K. H. Fan, A. L. Tits, J. Barlow, N-K. Tsing, M. Tischler & M. Takahashi, "On the Design of Decoupling Controllers for Advanced Rotorcraft in the Hover Case," *AIAA 29th Aerospace Sciences Meeting* , Reno, Nevada (Jan., 1991).

[5] N-K. Tsing, "Computer-Based Techniques for Control System Design, with applications to Rotorcraft Control," University of Maryland , Ph.D. Thesis, Dept. of Electrical Engineering, College Park, MD, 1992.

[6] G. Hughes, M. A. Manness & D. J. Murray-Smith, "Eigenstructure Assignment for Handling Qualities in Helicopter Flight Control Law Design," *The 16-th European Rotorcraft Forum, Paper III.10.2*, Glasgow (Sept., 1990).

[7] R. A. Hess & P. J. Gorder, "Quantitative Feedback Theory Applied to the Design of a Rotorcraft Flight Control System," *J. Guidance, Control and Dynamics* Vol. 16, No. 4 (July, 1993), 748–753.

[8] S. J. Williams, "The Application of Frequency Response Multivariable Design Techniques to VSTOL Aircraft," Cambridge Control Ltd., Report No. 5, May, 1988.

[9] M. A. Manness, J. J. Gribble & D. J. Murray-Smith, "Multivariable Methods for Helicopter Flight Control Law Design : A Review," *The 16-th European Rotorcraft Forum, Paper III.5.2*, Glasgow (Sept., 1990).

[10] W. Von Grünhagen, G. Bouwer, H-J. Pausder, F. Henscher & J. Kaletka, "A High Bandwidth Control System for a Helicopter In-Flight Simulator," *Int. J. Control* Vol. 59, No. 1 (1994), 239–261.

[11] E. Low & W. L. Garrard, "Design of Flight Control Systems to Meet Rotorcraft Handling Qualities Specifications," *J. Guidance, Control and Dynamics* Vol. 16, No. 1 (Jan., 1993), 69–78.

[12] R. H. Hoh, D. G. Mitchell & B. L. Aponso, "Handling Qualities Requirements for Military Rotorcraft," US Army ASC, Aeronautical Design Standard, ADS-33C, Aug., 1989.

[13] M. K. H. Fan, A. L. Tits, J. L. Zhou, L. S. Wang & J. Koninckx, "CONSOL - OPTCAD, User's Manual," Institute for System Research, Univ. of Maryland at College Park , Tech. Report TR 87-212r2a, Aug., 1991.

[14] F. D. Kim, S. R. Turnour & R. Celi, *UMGenhel - Version 1.2*, Center for Rotorcraft Education and Research, Department of Aerospace Engineering, University of Maryland, College Park, MD , Sept., 1991.

[15] M. B. Tischler, "Digital Control of Highly Augmented Combat Rotorcraft," NASA, US Army ASC, NASA Tech. Memo. 88346, USAAVSCOM TR 87-A-5, May, 1987.

[16] R. H. Hoh, D. G. Mitchell, B. L. Aponso, D. L Key & C. L. Blanken, "Background Information and User's Guide for Handling Qualities Requirements for Military Rotorcraft," US Army ASC, USAAVSCOM TR 89-A-8, Dec., 1989.

[17] K. H. Landis & S. I. Glusman, "Development of ADOCS Controllers and Control Laws Volume 2 - Literature Review and Preliminary Analysis," NASA Ames Research Center, US Army ASC, NASA Contractor Report 177339, USAAVSCOM TR 84-A-7, 1984.

[18] M. B. Tischler, J. W. Fletcher, P. M. Patrick, M. Morris & G. E. Tucker, "Flying Analysis and Flight Evaluation of a Highly Augmented Combat Rotorcraft," *J. of Guidance, Control, and Dynamics* Vol. 14, No. 5 (Oct., 1991), 954–963.

[19] M. B. Tischler, "Sample Disturbance Response Specification," Sept., 1990.

[20] G. Yudilevitch, "Optimal Decoupling Control," Institute for Systems Research, University of Maryland at College Park , Ph.D. Thesis, Dept. of Electrical Engineering, ISR PHD 94-10, 1994.

[21] W. F. Jewell & W. F. Clement, "Crossfeed Compensation Techniques for Decoupling Rotorcraft Response to Control Inputs," Systems Technology Inc., TR-1229-1, Sept., 1985.

[22] A. Potvin, *Nonlinear Control Design Toolbox for Use with MATLAB - User's Guide*, The MathWorks Inc., 1993.

A List of all the required computer codes

A.1 C-O problem description files (PDF)

ADOCS - **Main C-O PDF**

```
/*=================================================*/
/*   ADOCS - PDF File                              */
/*   Gil Yudilevitch        ISR      11-11-93      */
/*=================================================*/

include "dp_adocs" /* Defines design parameters    */

include "adocs.dp" /* Updates design parameters    */

dt  = 1/30
tf  = 5
dtg = 0.08
tfg = 40

global double getout();

global double geto(name, t, dt)
global char *name;
global double t, dt;
global {
global    double r;
global    int i;
global    i = t/dt + 1.5;
global    r = getout(name, i);
global    return r;
global }

include "object.sat" /* objective - actuator RMS rate */

include "stable.all" /* stability                     */

include "spec1.pit"  /* bandwidth vs. phase-delay     */
include "spec1.rol"
include "spec1.yaw"

include "spec2.pit"  /* stability margins             */
include "spec2.rol"
include "spec2.yaw"

include "spec3.pit"  /* attitude quickness            */
include "spec3.rol"
include "spec3.yaw"

include "spec4.pit"  /* interaxis coupling            */
include "spec4.rol"
include "spec4.yaw"

include "spec5.pit"  /* wind gust rejection           */
include "spec5.rol"
include "spec5.yaw"

/*====================  adocs   ======================*/
```

spec.chn* - **objective and constraint files**

* - Spec number (1,2,3,4,5); chn - channel (pit,rol,yaw)

```
/*=========================================================*/
/*  OBJECTIVE - Min. Actuator RMS Rates                    */
/*  Ref. briefing 04-94                                    */
/*  Included file of ADOCS                                 */
/*  Gil Yudilevitch        ISR          04-05-94           */
/*=========================================================*/


objective "p_act_rate"
minimize {
         return getout("pitch_AE",1)-1000;
         }
         good_value = 0.000
         bad_value  = 1

objective "r_act_rate"
minimize {
         return getout("roll_AE",1)-1000;
         }
         good_value = 0.000
         bad_value  = 1

objective "y_act_rate"
minimize {
         return getout("yaw_AE",1)-1000;
         }
         good_value = 0.000
         bad_value  = 1

/*====================  object.sat  ====================*/

/*=========================================================*/
/*  Included file of ADOCS.PDF                             */
/*  Gil Yudilevitch        ISR          05-14-93           */
/*=========================================================*/

constraint "stable all" hard
      {
      return getout("eigen_A", 1);
      }
   <=
   good_value =  0.000
   bad_value =   0.001

/*====================    END OF stable.all   ====================*/

*=============================================================*/
/*  SPECIFICATION 1 - Small Amplitude, Short Term Response     */
/*  Bandwidth & Phasedelay                                     */
/*  Pitch (ref. ADS-33C  3.3.2.1)                              */
/*  Included file of ADOCS.PDF                                 */
/*  Gil Yudilevitch        ISR          09-09-92               */
/*=============================================================*/

constraint  "pit bw pd" soft
      {
      return getout("pitch_dist", 1);
      }
   <=
   good_value = 0.000
   bad_value  = 0.002

constraint "pit Td" hard
      {
```

```
      return getout("pitch_pd", 1)-0.4;
      }
   <=
   good_value = 0.0000
   bad_value =  0.0001

/*==================    END OF spec1.pit   =====================*/

/*================================================================*/
/*  SPECIFICATION 1 - Small Amplitude, Short Term Response      */
/*  Bandwidth & Phasedelay                                      */
/*  Roll  (ref. ADS-33C  3.3.2.1)                               */
/*  Included file of ADOCS.PDF                                  */
/*  Gil Yudilevitch        ISR           09-09-92               */
/*================================================================*/

constraint "rol bw pd" soft
      {
      return getout("roll_dist", 1);
      }
   <=
   good_value = 0.000
   bad_value  = 0.002

constraint "rol Td" hard
      {
      return getout("roll_pd", 1)-0.4;
      }
   <=
   good_value = 0.0000
   bad_value =  0.0001

/*================    END OF spec1.rol   ==================*/

/*================================================================*/
/*  SPECIFICATION 1 - Small Amplitude, Short Term Response      */
/*  Bandwidth & Phasedelay                                      */
/*  Yaw   (ref. ADS-33C  3.3.5.1)                               */
/*  Included file of ADOCS.PDF                                  */
/*  Gil Yudilevitch        ISR           09-09-92               */
/*================================================================*/

constraint "yaw bw pd" soft
      {
      return getout("yaw_dist", 1);
      }
   <=
   good_value = 0.000
   bad_value  = 0.002

constraint "yaw Td" hard
      {
      return getout("yaw_pd", 1)-0.4;
      }
   <=
   good_value = 0.0000
   bad_value =  0.0001

/*================    END OF spec1.yaw   ==================*/

/*=====================================================*/
/*  SPEC 2 - Small Amplitude, Mid Term Response        */
/*  Relative Stability (replaces: Damping Ratio)       */
/*  Pitch (ref. ADS-33C  3.3.2.2, briefing 10-93)      */
/*  Included file of ADOCS                             */
```

```
/*  Gil Yudilevitch        ISR           11-11-93      */
/*=========================================================*/

constraint "pit damp" soft
      {
      return getout("pitch_damp", 1);
      }
   <=
   good_value = 0.000
   bad_value =  0.001

/*====================   spec2.pit   ====================*/

/*=========================================================*/
/*  SPEC 2 - Small Amplitude, Mid Term Response          */
/*  Relative Stability (replaces: Damping Ratio)         */
/*  Pitch (ref. ADS-33C  3.3.2.2, briefing 10-93)        */
/*  Included file of ADOCS                               */
/*  Gil Yudilevitch        ISR           11-11-93      */
/*=========================================================*/

constraint "rol damp" soft
      {
      return getout("roll_damp", 1);
      }
   <=
   good_value = 0.000
   bad_value =  0.001

/*====================   spec2.rol   ====================*/

/*=========================================================*/
/*  SPEC 2 - Small Amplitude, Mid Term Response          */
/*  Relative Stability (replaces: Damping Ratio)         */
/*  Pitch (ref. ADS-33C  3.3.5.2, briefing 10-93)        */
/*  Included file of ADOCS                               */
/*  Gil Yudilevitch        ISR           11-11-93      */
/*=========================================================*/

constraint "yaw damp" soft
      {
      return getout("yaw_damp", 1);
      }
   <=
   good_value = 0.000
   bad_value =  0.001

/*====================   spec2.yaw   ====================*/

/*=========================================================*/
/*  SPEC3 - Moderate Amplitude, Attitude Quickness     */
/*  q_pk/theta_pk                                        */
/*  Pitch (ref. ADS-33C  3.3.3)                          */
/*  Included file of ADOCS                               */
/*  Gil Yudilevitch        ISR         11-11-93        */
/*=========================================================*/

constraint  "pit quick1" soft
      {
      return getout("pitch_r1s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.06
```

```
constraint  "pit quick2" soft
      {
      return getout("pitch_r2s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.04

constraint  "pit quick3" soft
      {
      return getout("pitch_r3s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.03

/*==================   spec3.pit   ==================*/

/*======================================================*/
/*  SPEC3 - Moderate Amplitude, Attitude Quickness     */
/*  p_pk/phi_pk                                        */
/*  Roll (ref. ADS-33C  3.3.3)                         */
/*  Included file of ADOCS                             */
/*  Gil Yudilevitch        ISR         11-11-93        */
/*======================================================*/

constraint  "rol quick1" soft
      {
      return getout("roll_r1s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.014

constraint  "rol quick2" soft
      {
      return getout("roll_r2s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.08

constraint  "rol quick3" soft
      {
      return getout("roll_r3s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.07

/*================  spec3.rol   ====================*/

/*======================================================*/
/*  SPEC3 - Moderate Amplitude, Attitude Quickness     */
/*  r_pk/psi_pk                                        */
/*  Yaw (ref. ADS-33C  3.3.6)                          */
/*  Included file of ADOCS                             */
/*  Gil Yudilevitch        ISR         11-11-93        */
/*======================================================*/

constraint "yaw quick1" soft
      {
      return getout("yaw_r1s", 1);
      }
   >=
```

```
   good_value =  0.00
   bad_value  = -0.1

constraint "yaw quick2" soft
      {
      return getout("yaw_r2s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.07

constraint "yaw quick3" soft
      {
      return getout("yaw_r3s", 1);
      }
   >=
   good_value =  0.00
   bad_value  = -0.04

/*=================   spec3.yaw   ====================*/

/*================================================================*/
/*  SPECIFICATION 4 - Decoupling                                  */
/*  theta/delta_phi                                               */
/*  Pitch (ref. ADS-33C  3.3.9.2)                                 */
/*  Included file of ADOCS.PDF                                    */
/*  Gil Yudilevitch        ISR         12-23-92                   */
/*================================================================*/

functional_constraint "pit dec up" soft
   for t from 0 to 4 by dt
   {  import dt;
      return geto("pitch_d", t, dt)-0.25; }
 <= good_curve = { return  0.00; }
    bad_curve =  { return  0.05; }

functional_constraint "pit dec lo" soft
   for t from 0 to 4 by dt
   {  import dt;
      return geto("pitch_d", t, dt)+0.25; }
 >= good_curve = { return  0.00; }
    bad_curve =  { return -0.05; }

/*==================    END OF spec4.pit   ====================*/

/*================================================================*/
/*  SPECIFICATION 4 - Decoupling                                  */
/*  phi/delta-theta                                               */
/*  Roll (ref. ADS-33C  3.3.9.2)                                  */
/*  Included file of ADOCS.PDF                                    */
/*  Gil Yudilevitch        ISR         12-23-92                   */
/*================================================================*/

functional_constraint "rol dec up" soft
   for t from 0 to 4 by dt
   {  import dt;
      return geto("roll_d", t, dt)-0.25; }
 <= good_curve = { return  0.00; }
    bad_curve =  { return  0.05; }

functional_constraint "rol dec lo" soft
   for t from 0 to 4 by dt
   {  import dt;
      return geto("roll_d", t, dt)+0.25; }
```

```
 >= good_curve = { return  0.00; }
    bad_curve =  { return -0.05; }

/*================    END OF spec4.rol   ==================*/

/*=============================================================*/
/*  SPECIFICATION 4 - Decoupling                               */
/*  yaw/collective                                             */
/*  Yaw (ref. ADS-33C  3.3.9.1)                                */
/*  Included file of ADOCS.PDF                                 */
/*  Gil Yudilevitch        ISR         04-29-93                */
/*=============================================================*/

functional_constraint "yaw dec1 u" soft
   for t from 3 to tf by dt
   {  import dt;
      return geto("yaw_d1", t-3, dt)-5; }
 <= good_curve = { return  0.0; }
    bad_curve =  { return  0.5; }

functional_constraint "yaw dec1 l" soft
   for t from 3 to tf by dt
   {  import dt;
      return geto("yaw_d1", t-3, dt)+5; }
 >= good_curve = { return  0.0; }
    bad_curve =  { return -0.5; }

constraint "yaw dec2 d" soft
      {
      return getout("yaw_d2", 1);
      }
   <=
   good_value = 0.000
   bad_value  = 0.002

/*================    END OF spec4.yaw   ==================*/

/*=============================================================*/
/*  SPECIFICATION 5 - Wind-Gust Rejection                      */
/*  Pitch (ref. M. Tischler 9/26/90)                           */
/*  Included file of ADOCS.PDF                                 */
/*  Gil Yudilevitch        ISR         09-09-92                */
/*=============================================================*/

functional_constraint "p gust p u" soft
   for t from 0 to 10 by dtg
   {  import dtg;
      return geto("pitch_g", t, dtg)-4; }
 <= good_curve = { return  0.0; }
    bad_curve =  { return  0.3; }

functional_constraint "p gust p l" soft
   for t from 0 to 10 by dtg
   {  import dtg;
      return geto("pitch_g", t, dtg)+4; }
 >= good_curve = { return  0.0; }
    bad_curve =  { return -0.3; }

functional_constraint "p gust t u" soft
   for t from 10+dtg to tfg by dtg
   {  import dtg;
      return geto("pitch_g", t, dtg)-0.4; }
 <= good_curve = { return  0.00; }
```

```
    bad_curve =  { return  0.03; }

functional_constraint "p gust t l" soft
   for t from 10+dtg to tfg by dtg
   {  import dtg;
      return geto("pitch_g", t, dtg)+0.4; }
 >= good_curve = { return  0.00; }
    bad_curve =  { return -0.03; }

/*====================    END OF spec5.pit   =====================*/

/*============================================================*/
/*  SPECIFICATION 5 - Wind-Gust Rejection                     */
/*  Roll  (ref. M. Tischler 9/26/90)                          */
/*  Included file of ADOCS.PDF                                */
/*  Gil Yudilevitch        ISR         09-09-92               */
/*============================================================*/

functional_constraint "r gust p u" soft
   for t from 0 to 10 by dtg
   {  import dtg;
      return geto("roll_g", t, dtg)-4; }
 <= good_curve = { return  0.0; }
    bad_curve =  { return  0.3; }

functional_constraint "r gust p l" soft
   for t from 0 to 10 by dtg
   {  import dtg;
      return geto("roll_g", t, dtg)+4; }
 >= good_curve = { return  0.0; }
    bad_curve =  { return -0.3; }

functional_constraint "r gust t u" soft
   for t from 10+dtg to tfg by dtg
   {  import dtg;
      return geto("roll_g", t, dtg)-0.4; }
 <= good_curve = { return  0.00; }
    bad_curve =  { return  0.03; }

functional_constraint "r gust t l" soft
   for t from 10+dtg to tfg by dtg
   {  import dtg;
      return geto("roll_g", t, dtg)+0.4; }
 >= good_curve = { return  0.00; }
    bad_curve =  { return -0.03; }

/*================    END OF spec5.rol   ==================*/

/*============================================================*/
/*  SPECIFICATION 5 - Wind-Gust Rejection                     */
/*  Yaw   (ref. M. Tischler 9/26/90)                          */
/*  Included file of ADOCS.PDF                                */
/*  Gil Yudilevitch        ISR         09-09-92               */
/*============================================================*/

functional_constraint "y gust p u" soft
   for t from 0 to 10 by dtg
   {  import dtg;
      return geto("yaw_g", t, dtg)-4; }
 <= good_curve = { return  0.0; }
    bad_curve =  { return  0.3; }

functional_constraint "y gust p l" soft
```

```
   for t from 0 to 10 by dtg
   {  import dtg;
      return geto("yaw_g", t, dtg)+4; }
 >= good_curve = { return  0.0; }
    bad_curve =  { return -0.3; }

functional_constraint "y gust t u" soft
   for t from 10+dtg to tfg by dtg
   {  import dtg;
      return geto("yaw_g", t, dtg)-0.4; }
 <= good_curve = { return  0.00; }
    bad_curve =  { return  0.03; }

functional_constraint "y gust t l" soft
   for t from 10+dtg to tfg by dtg
   {  import dtg;
      return geto("yaw_g", t, dtg)+0.4; }
 >= good_curve = { return  0.00; }
    bad_curve =  { return -0.03; }

/*================    END OF spec5.yaw   ==================*/
```

A.2 MATLAB simulation files

init.m - **Initialization file** (run only once for each C-O run)

```
% ****************************************************************************
%   System Initialization File for ADOCS
%   init.m CONSOL-MATLAB file
%   Continous-Linear Model for specs: 1 (Bandwidth-Phase delay) & 2 (stability
%   margins) & 5 (Wind-gust).
%   Discrete-Nonlinear Model for specs: 3 (Quickness) & 4 (coupling).
%   Gil Yudilevitch          ISR - UMD        04-09-94
% ****************************************************************************

% Optimization Set-up
% -------------------

OPSYS  = 1; % Operating system: 0 - DOS (PC); 1 - UNIX (Working station).
MATLAB = 1; % Matlab run type: 0 - C-O/MATLAB run; 1 - autonomous MATLAB run.
CONSOL = 0; % C-O run type: 0 - Background; 1 - interactive.
LEVEL  = 0; % Contineuos performance level (0,1): 0 - Level 1, 1 - Level 1+.
LIMIT  = 1; % Actuator limits: actual_limit = LIMIT*nominal_limit.
TIME   = 1; % Actuator time const. (TC): actual_TC = TIME*nominal_TC.
SPEC1  = 1; % Specs: 0 - Specs are not included; 1 - Specs are included.
SPEC2  = 1;
SPEC3  = 1;
SPEC4  = 1;
SPEC5  = 1;   %                                    f
FLIGHT = 'h'; % Flight conditions (about hover)   l h r  10 kts each direction
              %                                    b

% System Block Diagram
% --------------------
%                                                          Gd
%                                                          |
%                                                          v
%  R   ,---,Rm  E   ,---,Ed ,----,Rs      ,----,Us ,-----,Xs  ,---,    Y
% --->| F |-->O-->| D |-->| Sc |-->O-->| Sr |-->| K/s |-+->| P |---+-->
%     `---' + ^-  `---'   `----' + ^-   `----'   `-----'   |  `---'   |
%             |                   |                       |          |
```

```
%            |      ,---.          '--------------------'          |
%            '------| H |<-----------------------------------------'
%                   '---'
%
% General Information
% -------------------

dt = 1/30;                                  % Sampling rate 30 Hz
tf = 5; t =0:dt:tf; nt=length(t);           % Command Type Time Axis
t1= min(find(t==1));
t3= min(find(t==3));                        % Index for 1 & 3 sec.
dtg=0.08; tfg=40;
tg=0:dtg:tfg;                               % Time vector for wind-gust inputs
n10=max(find(tg<=10)); nfg=length(tg);
wo=logspace(-1,1,50);                       % Frequency vector

%  Model Components
%  ----------------
%  1. Helicopter 6 DOF Dynamics + Rotor Aerodynamics
%     Linear Continuous Time Model

%  # of states    = 11 ( 9 for 6 dof dynamics + 2 for rotor aerodynamics )
%  # of inputs    = 4
%  # of outputs   = 7  ( 4 for the linear case --> C 4 x 11 )

%                                          State     Output   Input
%                                          -----     ------   -----
%  u     - linear velocity along X axis      1
%  v     - linear velocity along Y axis      2
%  w     - linear velocity along Z axis      3       * 5        4  delta_c
%  p     - angular velocity about X axis     4       * 6
%  q     - angular velocity about Y axis     5       * 7
%  r     - angular velocity about Z axis     6         4
%  phi   - angle about X axis                7         2        2  delta_phi
%  theta - angle about Y axis                8         1        1  delta_theta
%  psi   - angle about Z axis                9         3        3  delta_psi
%  b1c   - longitudinal flap  (-a1s)        10
%  b1s   - lateral flap       (-b1s)        11
%                                                  * - used for NL only
%  Units:  linear velocity  [ft /sec]                          delta_# [in]
%  -----   angular velocity [rad/sec]
%          helicopter angle [rad    ]
%          flap angle       [rad    ]

B10to4 = zeros(10,4);          % Trans. 10 (UMGENHEL) --> 4 inputs
B10to4(1,2) = 1; B10to4(2,1) = 1; B10to4(4,3) = 1; B10to4(3,4) = 1;

% Continueous Time
% ----------------

if OPSYS == 1,
  eval(['load A11',FLIGHT,' -ascii;']); % Load A,B UMGENHEL matrices for Unix
  eval(['load B11',FLIGHT,' -ascii;']); % for the FLIGHT conditions
  eval(['Ap = A11',FLIGHT,';']);
  eval(['Bp = B11',FLIGHT,'*B10to4;']);
else,
  eval(['load a11',FLIGHT,'.;']);       % Load A,B UMGENHEL matrices for PC
  eval(['load b11',FLIGHT,'.;']);       % for the FLIGHT conditions
  eval(['Ap = a11',FLIGHT,';']);
  eval(['Bp = b11',FLIGHT,'*B10to4;']);
end;

Cp = [ 0 0 0 0 0 0 0 1 0 0 0
```

```
       0 0 0 0 0 0 1 0 0 0 0
       0 0 0 0 0 0 0 0 0 1 0 0
       0 0 0 0 0 1 0 0 0 0 0 0];

Dp = zeros(4,4);

[Apm,Bpm,Cpm,Dpm]=modred(Ap,Bp,Cp,Dp,[10,11]);  % Reduced to 9 states for
                                                % "inverse plant" parameters

% Discrete Time
% -------------

[Apd,Bpd] = c2d(Ap,Bp ,dt);  % digital ZOH equivalence for the plant

% Pure Delay 2_nd Order Pade Approximation (for the continuous linear part !!)
% ----------------------------------------
%
% Total Delay was taken equal for all channels. Total delay for Pitch
% from Mark's AIAA paper is 223 ms ==> 200 ms "pure" and 25 ms Actuator
% Time-Constant (see later in SP-Actuator Model)

Td = [0.2;0.2;0.2;0.2];
[Ad1,Bd1,Cd1,Dd1] = pade(Td(1),2);
[Ad2,Bd2,Cd2,Dd2] = pade(Td(2),2);
[Ad3,Bd3,Cd3,Dd3] = pade(Td(3),2);
[Ad4,Bd4,Cd4,Dd4] = pade(Td(4),2);
[Ad,Bd,Cd,Dd]=append(Ad1,Bd1,Cd1,Dd1,Ad2,Bd2,Cd2,Dd2);
[Ad,Bd,Cd,Dd]=append(Ad,Bd,Cd,Dd,Ad3,Bd3,Cd3,Dd3);
[Ad,Bd,Cd,Dd]=append(Ad,Bd,Cd,Dd,Ad4,Bd4,Cd4,Dd4);
Ndel = round(Td/dt);  % # of delayed sampling intervals (for NL computation)

% Swashplate-Actuators Model
% --------------------------

% LINEAR PART :-

K   =  1/(TIME*0.025)*ones(1,4);    % 1/(Time Constant) equal actuators
[as1,bs1,cs1,ds1] = tf2ss([0,K(1)],[1,K(1)]);
[as2,bs2,cs2,ds2] = tf2ss([0,K(2)],[1,K(2)]);
[as3,bs3,cs3,ds3] = tf2ss([0,K(3)],[1,K(3)]);
[as4,bs4,cs4,ds4] = tf2ss([0,K(4)],[1,K(4)]);
[As,Bs,Cs,Ds] = append(as1,bs1,cs1,ds1,as2,bs2,cs2,ds2);
[As,Bs,Cs,Ds] = append(As,Bs,Cs,Ds,as3,bs3,cs3,ds3);
[As,Bs,Cs,Ds] = append(As,Bs,Cs,Ds,as4,bs4,cs4,ds4);

% NONLINEAR SATURATIONS :-

% *** All data is given at cockpit units (i.e., in's of stick or pedal)
% *** The trim (hover) conditions set to be zero (see table below)
% *** The swashplate mechanizem ("limits coupling") is NOT taken in acount
% *** The rate limits are taken as: full strok per 1 sec

% Command Limits Table, Ref. Sikorsky SER 70452 p 6.16 Fig. 6.3.1
% ..........................................................................
% channel   |     Pitch     |     Roll      |       Yaw       |   Collective  |
% actual    |-5.0 +0.2 +5.0|-5.0 -0.5 +5.0|-2.69 +0.85 +2.69| 0.0 +5.0 +10.0|
% simulation|-5.2  0.0 +4.8|-4.5  0.0 +5.5|-3.54  0.00 +1.84|-5.0  0.0 + 5.0|

Csl = LIMIT*[ -5.2 -4.5 -3.54  -5];  % Displacement (Commands) Lower Limits
Csu = LIMIT*[  4.8  5.5  1.84   5];  % Displacement (Commands) Upper Limits
Rsl = LIMIT*[-10  -10   -5.38 -10];  % Rate Lower Limits
Rsu = LIMIT*[ 10   10    5.38  10];  % Rate Upper Limits

% Linear Continuous-Time Open-Loop
```

```
% ------------------------------

[Ao,Bo,Co,Do] = series(Ad,Bd,Cd,Dd,As,Bs,Cs,Ds);
[Ao,Bo,Co,Do] = series(Ao,Bo,Co,Do,Ap,Bp,Cp,Dp);

% Closed-Loop (Design) Parameters, Initial Values
% ------------------------------------------------

% Design parameters
% -----------------

if MATLAB == 1,
   dp_NOM
   %Kr_p=0; Kp_r=0; Kc_y=0; % Crossfeed gains (for decentralized control)
end;

% Inverse plant (ff) constants
% ----------------------------

FGtet =  Bpm(5,1); FGphi =  Bpm(4,2); FGpsi =  Bpm(6,3);
FTtet = -Apm(5,5); FTphi = -Apm(4,4); FTpsi = -Apm(6,6);

% Time Response Inputs
% --------------------

DtR  = pi/180;
Stet = DtR*[ 5,17.5,30];
Sphi =-DtR*[10,35  ,60]; % the worst direction (nonsymmetric saturation)
Spsi = DtR*[10,35  ,60];
Scol = 5;

% Linear Wind-Gust Model
% ----------------------

[ag,bg,cg,dg] = tf2ss([0 0.44 0],[1 2 1]);
[Ag,Bg,Cg,Dg]=append(ag,bg,cg,dg,ag,bg,cg,dg);
[Ag,Bg,Cg,Dg]=append(Ag,Bg,Cg,Dg,ag,bg,cg,dg);
[Ag,Bg,Cg,Dg]=append(Ag,Bg,Cg,Dg,[],[],[], 0);
Bgg=zeros(23,4);
Bgg(17,1)=-Apm(5,5); Bgg(16,2)=-Apm(4,4); Bgg(18,3)=-Apm(6,6);

% Level 1 curve for spec 1
% ------------------------

Ts = [0.000 0.025 0.050 0.075 0.100 0.125 0.150 0.175 0.200 ...
      0.225 0.250 0.275 0.300 0.325 0.350 0.375 0.400];
Ws = [2.005 2.005 2.005 2.005 2.005 2.005 2.005 2.105 2.235 ...
      2.405 2.685 3.005 3.275 3.505 3.905 4.155 4.455];
PL1=polyfit(Ts,Ws,6);
Ws = Ws + 1.5*LEVEL;
PL =polyfit(Ts,Ws,6);

% LEVEL factors for spec 3
% ------------------------

d3p1 = 0.93; d3p2 = 0.78; d3p3 = 0.68;
d3r1 = 0.96; d3r2 = 0.50; d3r3 = 0.10;
d3y1 = 0.98; d3y2 = 0.76; d3y3 = 0.64;

% Optimization Parameters (optional for interactive C-0/MATLAB run)
% -----------------------

if MATLAB == 0 & CONSOL == 1,
  CONS = [1:3,2,4,6,8:16,1,3,17,5:2:17];
  BAD  = [3;3;3;0.002;0.002;0.002;0.01;0.01;0.01;0.04;0.03;0.09;0.08;...
```

```
        0.07;0.04;0.05;0.05;0.002;0.5;0.3;0.03;0.3;0.03;0.3;0.03];
  Iter=-1; iter=[]; witer=[]; www=[]; kkk=[];
  perf=[]; dps=[]; grad=[]; bad=BAD;
end;

% SIMU.MAT saved variables, default values
% ---------------------------------------

pitch_dist=0; roll_dist=0; yaw_dist=0; pitch_bw  =0; roll_bw  =0; yaw_bw  =0;
pitch_pd  =0; roll_pd  =0; yaw_pd  =0; pitch_damp=0; roll_damp=0; yaw_damp=0;
pitch_r1s =0; roll_r1s =0; yaw_r1s =0; pitch_r2s =0; roll_r2s =0; yaw_r2s =0;
pitch_r3s =0; roll_r3s =0; yaw_r3s =0; pitch_d   =0; roll_d   =0; yaw_d1  =0;
yaw_d2    =0; pitch_g  =0; roll_g  =0; yaw_g     =0; eigen_A  =0; OBJ     =0;
pitch_Wco =0; roll_Wco =0; yaw_Wco =0; pitch_AE  =0; roll_AE  =0; yaw_AE  =0;

Name='Gil Yudilevitch';
Date=amdate;

% **************   end of init.m   ****************************************
```

simu.m - Main simulation file

```
% ***************************************************************************
%   System Simulation File for ADOCS
%   simu.m CONSOL-MATLAB file
%   Continous-Linear Model for specs: 1 (Bandwidth-Phase delay) & 2 (stability
%   margins) & 5 (Wind-gust).
%   Discrete-Nonlinear Model for specs: 3 (Quickness) & 4 (coupling).
%   Gil Yudilevitch         ISR - UMD         04-09-94
% ***************************************************************************
%===========================================================================

% Feedforward (including Model Following & "Inverse Closed-Loop")
% ---------------------------------------------------------------

if MATLAB==1, disp('Feedforward control'); end;

% Continuous Time
% ---------------

[Af_tet,Bf_tet,Cf_tet,Df_tet] = ...
tf2ss(Mtet^2*[1/FGtet Kq+1/FGtet/FTtet Ktet],[1 2*Mtet Mtet^2]);
[Af_phi,Bf_phi,Cf_phi,Df_phi] = ...
tf2ss(Mphi^2*[1/FGphi Kp+1/FGphi/FTphi Kphi],[1 2*Mphi Mphi^2]);
[Af_psi,Bf_psi,Cf_psi,Df_psi] = ...
tf2ss(  Mpsi*[1/FGpsi Kr+1/FGpsi/FTpsi Kpsi],[1   Mpsi      0]);
[Af,Bf,Cf,Df]=append(Af_tet,Bf_tet,Cf_tet,Df_tet,Af_phi,Bf_phi,Cf_phi,Df_phi);
[Af,Bf,Cf,Df]=append(Af,Bf,Cf,Df,Af_psi,Bf_psi,Cf_psi,Df_psi);
[Af,Bf,Cf,Df]=append(Af,Bf,Cf,Df,[],[],[],1);

% Discrete Time
% -------------

[Afd,Bfd] = c2d(Af,Bf,dt);

% Crossfeed design
% ----------------

if MATLAB==1, disp('Crossfeed control'); end;

Kcf = [1    Kr_p 0 0
       Kp_r 1    0 0
```

```
       0    0    1 Kc_y
       0    0    0 1];

Bocf = Bo*Kcf; Docf = Do*Kcf;

% Stabilization Feedback Gains Matrix
% -----------------------------------

if MATLAB==1, disp('Feedback control'); end;

%   u v w p  q  r  phi theta psi b1c b1s
H = [ ...
    0 0 0 0  Kq 0  0    Ktet 0    0   0     % delta_theta
    0 0 0 Kp 0  0  Kphi 0    0    0   0     % delta_phi
    0 0 0 0  0  Kr 0    0    Kpsi 0   0     % delta_psi
    0 0 0 0  0  0  0    0    0    0   0];   % delta-collective

% Linear Closed-Loop
% ------------------
% order and # of states:
%  6  - Feedforward
%  8  - Delay (4 x 2nd order Pade apprx.)
%  4  - Swashplate actuators
% 11  - 6 DOF + rotor (outputs: theta, phi, psi, r)

HH=[zeros(4,12),H];
Ac=Ao-Bocf*HH;
eigen_A=max(real(eig(Ac)));
[A,B,C,D]=series(Af,Bf,Cf,Df,Ac,Bocf,Co,Docf);

% Linear "Broken-Loop" (open-loop)
% --------------------------------
HH(1,17) = 0; HH(1,20) = 0;
[Ao1,Bo1,Co1,Do1] = ...
series(Ao-Bocf*HH,Bocf(:,1),Cor(1,:),Docf(1,1),0,1,Ktet,Kq);
HH(1,17) = Kq; HH(1,20) = Ktet;
HH(2,16) = 0; HH(2,19) = 0;
[Ao2,Bo2,Co2,Do2] = ...
series(Ao-Bocf*HH,Bocf(:,2),Cor(2,:),Docf(2,2),0,1,Kphi,Kp);
HH(2,16) = Kp; HH(2,19) = Kphi;
HH(3,18) = 0; HH(3,21) = 0;
[Ao3,Bo3,Co3,Do3] = ...
series(Ao-Bocf*HH,Bocf(:,3),Cor(3,:),Docf(3,3),0,1,Kpsi,Kr);
HH(3,18) = Kr; HH(3,21) = Kpsi;

% Nonlinear Closed-Loop (Time Response)
% -------------------------------------
% Command Response | oredr and # of states:
%  6  - Feedforward
%  4  - Swashplate actuators
% 11  - 6 DOF + rotor (outputs: theta, phi, psi, p, q, r, w)

if MATLAB==1, disp('Nonlinear step response:'); end;

% Pitch input
% -----------

nR=1;
if SPEC3 == 1,
  aR=pilot(t,Stet(3),Stet(1),'p'); drsc; X11 = X; Ur11 = Us;
  if MATLAB == 1,
    disp('                              Pitch Small');
    Us11 = Xs(1:nt,:);
  end;
```

```
  aR=pilot(t,Stet(3),Stet(2),'p'); drsc; X12 = X;
  if MATLAB==1,
    disp('                          Pitch Medium');
    Ur12 = Us; Us12 = Xs(1:nt,:);
  end;
end;
if (SPEC3+SPEC4) >= 1,
  aR=pilot(t,Stet(3),Stet(3),'p'); drsc; X13 = X;
  if MATLAB==1,
    disp('                          Pitch Large');
    Ur13 = Us; Us13 = Xs(1:nt,:);
  end;
end;

% Roll input
% ----------

nR=2;
if SPEC3 == 1,
  aR=pilot(t,Sphi(3),Sphi(1),'p'); drsc; X21 = X; Ur21 = Us;
  if MATLAB == 1,
    disp('                          Roll Small');
    Us21 = Xs(1:nt,:);
  end;
  aR=pilot(t,Sphi(3),Sphi(2),'p'); drsc; X22 = X;
  if MATLAB==1,
    disp('                          Roll Medium');
    Ur22 = Us; Us22 = Xs(1:nt,:);
  end;
end;
if (SPEC3+SPEC4) >= 1,
  aR=pilot(t,Sphi(3),Sphi(3),'p'); drsc; X23 = X;
  if MATLAB==1,
    disp('                          Roll Large');
    Ur23 = Us; Us23 = Xs(1:nt,:);
  end;
end;

% Yaw input
% ---------

nR=3;
if SPEC3 == 1,
  aR=pilot(t,Spsi(3),Spsi(1),'r'); drsc; X31 = X; Ur31 = Us;
  if MATLAB == 1,
    disp('                          Yaw Small');
    Us31 = Xs(1:nt,:);
  end;
  aR=pilot(t,Spsi(3),Spsi(2),'r'); drsc; X32 = X;
  if MATLAB==1,
    disp('                          Yaw Medium');
    Ur32 = Us; Us32 = Xs(1:nt,:);
  end;
  aR=pilot(t,Spsi(3),Spsi(3),'r'); drsc; X33 = X;
  if MATLAB==1,
    disp('                          Yaw Large');
    Ur33 = Us; Us33 = Xs(1:nt,:);
  end;
end;

% Collective input
% ----------------

nR=4;
if SPEC4 == 1,
```

```
  aR=pilot(t,Scol,Scol,'p');    drsc; X41 = X;
  if MATLAB==1,
    disp('                          Collective');
    Ur41 = Us; Us41 = Xs(1:nt,:);
  end;
end;

if SPEC5 == 1,
    % Linear Wind Gust response
    % ------------------------
    % Gust Response | States order:
    %  6  - Gust model
    %  8  - Pure delay
    %  4  - Swashplate actuators
    % 11  - 6 DOF + rotor (outputs: theta, phi, psi)

    [Agc,Bgc,Cgc,Dgc]=series(Ag,Bg,Cg,Dg,Ac,Bgg,Co,Docf);
    [yg1,x] = step(Agc,Bgc(:,1),Cgc(1,:),Dgc(1,1),1,tg);
    pitch_up=x(:,08); pitch_ur=K(1)*(x(:,08)-x(:,15));
    [yg2,x] = step(Agc,Bgc(:,2),Cgc(2,:),Dgc(2,2),1,tg);
    roll_up =x(:,10); roll_ur =K(2)*(x(:,10)-x(:,16));
    [yg3,x] = step(Agc,Bgc(:,3),Cgc(3,:),Dgc(3,3),1,tg);
    yaw_up  =x(:,12); yaw_ur  =K(3)*(x(:,12)-x(:,17));
    if MATLAB==1, disp('Linear wind-gust response');
     if max(pitch_up) > Csu(1) | ...
        min(pitch_up) < Csl(1) | max(abs(pitch_ur)) > abs(Rsl(1)),
        disp('                          Warning: Pitch actuator is saturated !');
     end;
     if max(roll_up) > Csu(2) | ...
        min(roll_up) < Csl(2) | max(abs(roll_ur)) > abs(Rsl(2)),
        disp('                          Warning: Roll actuator is saturated !');
     end;
     if max(yaw_up) > Csu(3) | ...
        min(yaw_up) < Csl(3) | max(abs(yaw_ur)) > abs(Rsl(3)),
        disp('                          Warning: Yaw actuator is saturated !');
     end;
    end;
end;

% ============================================================================

% New objectives (4/94)
% ---------------------

% Quadratic objectives (actuator "energy")
% --------------------

pitch_AE = qcost(dt,Ur11(:,1),1);
 roll_AE = qcost(dt,Ur21(:,2),1);
  yaw_AE = qcost(dt,Ur31(:,3),1);

% Actuator efforts
% ----------------

eff1 = sqrt(qcost(dt,Ur11(:,1),1)/5)/10;
eff2 = sqrt(qcost(dt,Ur21(:,2),1)/5)/10;
eff3 = sqrt(qcost(dt,Ur31(:,3),1)/5)/5.38;

if MATLAB==1, disp('Computing specs performance:'); end;

% Spec 1
% ------
```

```
if SPEC1 == 1,
  if MATLAB==1, disp('                                Spec 1'); end;
  [pitch_dist,pitch_bw,pitch_pd] = d_bw_pd(A,B,C,D,1,1,PL1); % [ ,rad/sec,sec]
  [roll_dist ,roll_bw ,roll_pd ] = d_bw_pd(A,B,C,D,2,2,PL);
  [yaw_dist  ,yaw_bw  ,yaw_pd  ] = d_bw_pd(A,B,C,D,3,4,PL);
end;

% Spec 2
% ------

if SPEC2 == 1,

 if MATLAB == 1,
  % zeta is not used for optimization anymore
  [Zeta,max_wn] = zeta(A,max([pitch_bw,roll_bw,yaw_bw]));  % [ ], [ ]
  disp('                              Spec 2');
 end;

% we are using GM & PM criterion instead of zeta
GM0=6; PM0=45; % standard gain and phase margin
[mo1,po1]=bode(Ao1,Bo1,Co1,Do1,1,wo);
[Gm,pitch_PM,Wg,pitch_Wco] = imargin(mo1,po1,wo); pitch_GM = 20*log10(Gm);
d_g_p = (pitch_GM/GM0-1)^2; d_p_p = (pitch_PM/PM0-1)^2;
    if pitch_GM >= GM0 & pitch_PM >= PM0, pitch_damp = 0;
elseif pitch_GM >= GM0 & pitch_PM <  PM0, pitch_damp =       d_p_p;
elseif pitch_GM <  GM0 & pitch_PM >= PM0, pitch_damp = d_g_p;
else,                                     pitch_damp = d_g_p+d_p_p; end;
[mo2,po2]=bode(Ao2,Bo2,Co2,Do2,1,wo);
[Gm,roll_PM ,Wg,roll_Wco ] = imargin(mo2,po2,wo); roll_GM  = 20*log10(Gm);
d_g_r = (roll_GM/GM0-1)^2; d_p_r = (roll_PM/PM0-1)^2;
    if roll_GM >= GM0 & roll_PM >= PM0, roll_damp = 0;
elseif roll_GM >= GM0 & roll_PM <  PM0, roll_damp =       d_p_r;
elseif roll_GM <  GM0 & roll_PM >= PM0, roll_damp = d_g_r;
else,                                   roll_damp = d_g_r+d_p_r; end;
[mo3,po3]=bode(Ao3,Bo3,Co3,Do3,1,wo);
[Gm,yaw_PM  ,Wg,yaw_Wco  ] = imargin(mo3,po3,wo); yaw_GM   = 20*log10(Gm);
d_g_y = (yaw_GM/GM0-1)^2; d_p_y = (yaw_PM/PM0-1)^2;
    if yaw_GM >= GM0 & yaw_PM >= PM0, yaw_damp = 0;
elseif yaw_GM >= GM0 & yaw_PM <  PM0, yaw_damp =       d_p_y;
elseif yaw_GM <  GM0 & yaw_PM >= PM0, yaw_damp = d_g_y;
else,                                 yaw_damp = d_g_y+d_p_y; end;

end;

% Spec 3
% ------

if SPEC3 == 1,
  if MATLAB == 1, disp('                                Spec 3'); end;
  pitch_r1 = max(X11(:,15))/Stet(1);  % [1/sec]
  pitch_r1s= pitch_r1-(0.70+d3p1*LEVEL);
  pitch_r2 = max(X12(:,15))/Stet(2);
  pitch_r2s= pitch_r2-(0.40+d3p2*LEVEL);
  pitch_r3 = max(X13(:,15))/Stet(3);
  pitch_r3s= pitch_r3-(0.25+d3p3*LEVEL);
  roll_r1  = min(X21(:,14))/Sphi(1);
  roll_r1s = roll_r1-(1.39+d3r1*LEVEL);
  roll_r2  = min(X22(:,14))/Sphi(2);
  roll_r2s = roll_r2-(0.85+d3r2*LEVEL);
  roll_r3  = min(X23(:,14))/Sphi(3);
  roll_r3s = roll_r3-(0.75+d3r3*LEVEL);
  yaw_r1   = max(X31(:,16))/Spsi(1);
```

```
  yaw_r1s  = yaw_r1-(1.42+d3y1*LEVEL);
  yaw_r2   = max(X32(:,16))/Spsi(2);
  yaw_r2s  = yaw_r2-(0.67+d3y2*LEVEL);
  yaw_r3   = max(X33(:,16))/Spsi(3);
  yaw_r3s  = yaw_r3-(0.36+d3y3*LEVEL);
end;

% Spec 4
% ------

if SPEC4 == 1,
  if MATLAB==1, disp('                                Spec 4'); end;

  pitch_d = X23(1:121,18)/Sphi(3);           % [ ]
  roll_d  = X13(1:121,17)/Stet(3);
  yaw_d1 = 180/pi*X41(t3:nt,16);          % [deg/sec]
   Mr1 = max(X41(:,16)); Jr1 = find(X41(:,16) == Mr1);     % for w(t) < 0
   if t(Jr1) >= t3, r1 =  X41(t1,16); else, r1 = Mr1; end; % and r(t) > 0
   r3 = X41(t3,16)-r1;
   y_y = -180/pi*r1/X41(t3,13);
   x_y =  180/pi*r3/X41(t3,13);
   a_y = (x_y-0.15)^2; b_y = (y_y-0.65)^2; c_y = (x_y+0.2)^2;
  if     (x_y >= 0.15) & (y_y < 0.65),
     yaw_d2 = a_y;
  elseif (x_y <  -0.2) & (y_y < 0.65),
     yaw_d2 = c_y;
  elseif (x_y >= -0.2) & (x_y < 0.15) & (y_y >= 0.65),
     yaw_d2 = b_y;
  elseif (x_y >= 0.15) & (y_y >= 0.65),
     yaw_d2 = a_y + b_y;
  elseif (x_y <  -0.2) & (y_y >= 0.65),
     yaw_d2 = b_y + c_y;
  else,
     yaw_d2 = 0;
  end;
end;

% Spec 5
% ------

if SPEC5 == 1,

  if MATLAB==1, disp('                                Spec 5'); end;
  pitch_g = 180/pi*yg1;  % [deg]
  roll_g  = 180/pi*yg2;
  yaw_g   = 180/pi*yg3;

end;

% 1 objective
% -----------

% OBJ = 60*(pitch_Wco+roll_Wco+yaw_Wco)+pitch_AE+6*roll_AE+30*yaw_AE - 1000;

if MATLAB==0,
 save simu pitch_dist roll_dist yaw_dist ...
           pitch_bw   roll_bw   yaw_bw   ...
           pitch_pd   roll_pd   yaw_pd   ...
           pitch_damp roll_damp yaw_damp ...
           pitch_Wco  roll_Wco  yaw_Wco  ...
           pitch_AE   roll_AE   yaw_AE   ...
           pitch_r1s  roll_r1s  yaw_r1s   ...
           pitch_r2s  roll_r2s  yaw_r2s   ...
           pitch_r3s  roll_r3s  yaw_r3s   ...
```

```
          pitch_d    roll_d    yaw_d1 yaw_d2 ...
          pitch_g    roll_g    yaw_g  eigen_A %OBJ
end;

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%  end of simu.m  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

*.m - MATLAB functions (used in *simu.m*)

```
function [d,Bw,Pd] = d_bw_pd(A,B,C,D,Iu,Iy,PL);

% Calculate bandwidth [rad/sec] and phase-delay [sec] for
% the SISO system (input Iu output Iy) of:
%    dx/dt = Ax + Bu
%      y   = Cx + Du
% Then calculate the distance (d) of (Bw,Pd) for a given boundary
% curve (PL).
%
%    [d,Bw,Pd] = d_bw_pd(A,B,C,D,Iu,Iy)

% MATLAB function for ADOCS
% Gil Yudilevitch       ISR        11-19-92

% Calculate Bw
% ------------

w=0.5; p=0;
for i = 1:3,
  epsilon = 0.1^(i-1);
  while p > -135,
    q=p;
    w = w+epsilon;
    p=angle(C(Iy,:)*inv(j*w*eye(size(A))-A)*B(:,Iu)+D(Iy,Iu))*180/pi;
    if p > 0, p=p-360; end;
  end;
  p=q;
  w = w-epsilon;
end;
Bw = w+epsilon/2;

% Calculate Pd
% ------------

p=0;
for i = 1:5,
  epsilon = 0.1^(i-1);
  while p > -180,
    q=p;
    w = w+epsilon;
    p=angle(C(Iy,:)*inv((j*w*eye(size(A))-A))*B(:,Iu)+D(Iy,Iu))*180/pi;
    if p > 0, p=p-360; end;
  end;
  p=q;
  w = w-epsilon;
end;
w = w+epsilon/2;
p=angle(C(Iy,:)*inv((j*2*w*eye(size(A))-A))*B(:,Iu)+D(Iy,Iu))*180/pi-360;

% Assumption, at 2*w180 : -180 deg  >=  phase >= -540 deg !!!

Pd = -(180+p)*pi/360/w;

% Calculate d
```

```
% ----------

d=Bw-polyval(PL,Pd);
if d > 0, d=0;
else,
  dmin=100;
  d=(d/4.45)^2;
  Pd0=Pd;
  while d < dmin,
    dmin=d;
    Pd0=Pd0-0.005;
    d=((Pd-Pd0)/0.4)^2+((Bw-polyval(PL,Pd0))/4.45)^2;
  end;
  d=dmin;
end;

function Xd = delay(XX,D,k)

% Delays the time vector XX(k) k=1,2,... by D time units

% MATLAB function for ADOCS
% Gil Yudilevitch       ISR        14-03-93

[m,n] = size(XX);
for i=1:n,
  if D(i) >= k,
     Xd(i) = XX(1,i);
  else,
     Xd(i) = XX(k-D(i),i);
  end;
end;

% ************************************************************
%   Digital Recursive Simulation Command (inputs)
%   Solve NON-LINEAR equations for simu.m (CONSOL-MATLAB file)
%   CAUTION: This is a MATLAB SCRIPT file !!!
%   Gil Yudilevitch           ISR        02-04-93
% ************************************************************

Xf = zeros(nt,6 );
Xs = zeros(nt,4 );
Xp = zeros(nt,11);
R=zeros(nt,4); R(:,nR)=aR';

for k=1:nt,

  Rm(k,:)  = Xf(k,:)*Cf' + R(k,:)*Df';
  E(k,:)   = Rm(k,:)     - Xp(k,:)*H';
  Ed(k,:)  = delay(E,Ndel,k);
  Ec(k,:)  = Ed(k,:)*Kcf';
  Rs(k,:)  = limit(Csl,Csu,Ec(k,:));
  Uu(k,:)  = (Rs(k,:)-Xs(k,:)).*K;
  Us(k,:)  = limit(Rsl,Rsu,Uu(k,:));

  Xf(k+1,:) = Xf(k,:)*Afd' + R(k,:)*Bfd';
  Xs(k+1,:) = Xs(k,:)      + Us(k,:)*dt;
  Xp(k+1,:) = Xp(k,:)*Apd' + Xs(k,:)*Bpd';

end;

  X=[Xf(1:nt,:) Xs(1:nt,:) Xp(1:nt,:)];

function y=limit(l,u,x)

% Saturation function:
%
```

```
%            ^ y (output component)
%          u | /----
%            |/
%      ------/------> x (input component)
%           /|
%     ---- / | l
%
% x,y,l,u are all vectors of the same length
%
%     y = limit(l,u,x)

% MATLAB function for ADOCS
% Gil Yudilevitch       ISR       09-22-92

n=length(l);
for i=1:n,
   if x(i) >= u(i), y(i)=u(i);
  else,
   if x(i) <= l(i), y(i)=l(i);
  else,         y(i)=x(i);
  end; end;

function u=pilot(t,r,a,type)

%    type     u ^
%               |  ------- a        for pitch and roll channels
%    'p'osition | /
%               |/r
%               /         -> t
%
%             u ^
%               |   a               for yaw channel
%    'r'ate     | /\
%               |/r \
%               /    \----> t

% MATLAB function for ADOCS
% Gil Yudilevitch       ISR       06-02-94

if type == 'p',
  T=max(find(t<=a/r));
  u=[r*t(1:T),a*ones(1,length(t)-T)];
else,
  T=max(find(t<=sqrt(a/r)));
  u=[r*t(1:T),r*(t(T)-t(2:T)),zeros(1,length(t)-2*T+1)];
end;
```

A.3 Some M-files for design evaluation

specs.m - **Performance map** (color)

```
% MATLAB M-file (SCRIPT !!!) for PERFORMANCE MAP (COLOR) DISPLAY

%
% Gil Yudilevitch      ISR    12-23-92
%

clg; mp=4; delta=1e-5;

% Spec 1
% ------
```

```
if SPEC1 == 0,
  [pitch_dist,pitch_bw,pitch_pd] = d_bw_pd(A,B,C,D,1,1,PL1);
  [roll_dist ,roll_bw ,roll_pd ] = d_bw_pd(A,B,C,D,2,2,PL);
  [yaw_dist  ,yaw_bw  ,yaw_pd  ] = d_bw_pd(A,B,C,D,3,4,PL);
end;
wl=0:0.001:0.4;
xl=polyval(PL1,wl);
xlp=polyval(PL,wl);
subplot(341);
if LEVEL == 0,
fill([0.55,xl,xl(401:-1:1)-1.45],[0,wl,wl(401:-1:1)],'r',...
     [xl,6,6,2],[wl,0.4,0,0],'b',...
     [0,xl-1.45,0],[0,wl,0.4],'m');
else,
fill([xlp,6,6,3.5],[wl,0.4,0,0],'c',...
     [0.55,xl,xl(401:-1:1)-1.45],[0,wl,wl(401:-1:1)],'r',...
     [xlp,6,xl(401:-1:1),3.5],[wl,0.4,wl(401:-1:1),0],'b',...
     [0,xl-1.45,0],[0,wl,0.4],'m');
end;
axis([0,6,0,0.4]); hold on;
putfill(yaw_bw  ,yaw_pd  ,0,6,0,0.4,3);
putfill(pitch_bw,pitch_pd,0,6,0,0.4,1);
putfill(roll_bw ,roll_pd ,0,6,0,0.4,2);
if exist('COLOR') == 0, title('Spec 1 : Bawdwidth & Phase-delay');
else,                   title('1 : BW vs. PD'); end;
ind=''; if pitch_dist > delta, ind='*'; end;
xtext([0,6],0.28,['1 ',ind,']',mp);
ind=''; if roll_dist  > delta, ind='*'; end;
xtext([0,6],0.20,['2 ',ind,']',mp);
ind=''; if yaw_dist   > delta, ind='*'; end;
xtext([0,6],0.12,['3 ',ind,']',mp);
ylabel('Phase delay [sec]'); xlabel('BandWidth [rad/sec]'); axis;
if exist('Title'), Tit=Title; else, Tit='UH-60 Hover performance'; end;
if exist('Iter'), if Iter >= 0,
  Tit=[Tit,';   Completed ',num2str(Iter),' C-0 iterations'];
end; end;
if exist('DeGr'),
  Tit=[Tit,';   Design grade: ',num2str(DeGr)];
end;
if exist('Name'),
  Tit=[Tit,';   ',Name];
end;
if exist('Date'),
  Tit=[Tit,';   ',Date];
end;
if exist('COLOR') == 0, text(-2.5,.5,Tit); end;
hold off;

% Spec 2
% ------

[Zeta,max_wn] = zeta(A,max([pitch_bw,roll_bw,yaw_bw]));  % [ ], [ ]

if SPEC2 == 0,
  [mo1,po1]=bode(Ao1,Bo1,Co1,Do1,1,wo);
  [Gm,pitch_PM,Wg,pitch_Wco] = imargin(mo1,po1,wo); pitch_GM = 20*log10(Gm);
  [mo2,po2]=bode(Ao2,Bo2,Co2,Do2,1,wo);
  [Gm,roll_PM ,Wg,roll_Wco ] = imargin(mo2,po2,wo); roll_GM  = 20*log10(Gm);
  [mo3,po3]=bode(Ao3,Bo3,Co3,Do3,1,wo);
  [Gm,yaw_PM  ,Wg,yaw_Wco  ] = imargin(mo3,po3,wo); yaw_GM   = 20*log10(Gm);
end;

subplot(342);
fill([6,12,12,6]    ,[45,45,90,90]    ,'b',...
```

```
    [0,12,12,6,6,0],[0,0,45,45,90,90],'r');
axis([0,12,0,90]);
if exist('COLOR') == 0, title('Spec 2 : Stability margins');
else,                   title('2 : Stability margins'); end;
xlabel('GM [db]'); ylabel('PM [deg]'); hold on;
putfill(pitch_GM,pitch_PM,0,12,0,90,1);
putfill(roll_GM ,roll_PM ,0,12,0,90,2);
putfill(yaw_GM  ,yaw_PM  ,0,12,0,90,3);
ind=''; if pitch_damp > delta, ind='*'; end;
xtext([0,12],63,['4 ',ind],']',mp);
ind=''; if roll_damp  > delta, ind='*'; end;
xtext([0,12],45,['5 ',ind],']',mp);
ind=''; if yaw_damp   > delta, ind='*'; end;
xtext([0,12],27,['6 ',ind],']',mp);

if exist('COLOR') == 0,
  text(-30,-30,['Wco = ',num2str(n2s(pitch_Wco,2)),...
                    ', ',num2str(n2s(roll_Wco ,2)),...
                    ', ',num2str(n2s(yaw_Wco  ,2)),'  [rad/sec]']);
  text(  5,-30,['Rrms = ',num2str(n2s(sqrt(pitch_AE/5),2)),...
                     ', ',num2str(n2s(sqrt(roll_AE/5) ,2)),...
                     ', ',num2str(n2s(sqrt(yaw_AE/5)  ,2)),'  [in/sec]']);
  text( 40,-30,['Zeta = ',num2str(n2s(Zeta,2)),' at ',...
                          num2str(n2s(max_wn,2)),'  [rad/sec]']);
end;
hold off;

% Spec 3
% ------

if SPEC3 == 0,
 nR=1;
  aR=Stet(1); drscd; X11 = X;
  aR=Stet(2); drscd; X12 = X;
  aR=Stet(3); drscd; X13 = X;
 nR=2;
  aR=Sphi(1); drscd; X21 = X;
  aR=Sphi(2); drscd; X22 = X;
  aR=Sphi(3); drscd; X23 = X;
 nR=3;
  aR=Spsi(1); drscmd; X31 = X;
  aR=Spsi(2); drscmd; X32 = X;
  aR=Spsi(3); drscmd; X33 = X;
  pitch_r1 = max(X11(:,15))/Stet(1);  % [1/sec]
  pitch_r1s= pitch_r1-(0.70+d3p1*LEVEL);
  pitch_r2 = max(X12(:,15))/Stet(2);
  pitch_r2s= pitch_r2-(0.40+d3p2*LEVEL);
  pitch_r3 = max(X13(:,15))/Stet(3);
  pitch_r3s= pitch_r3-(0.25+d3p3*LEVEL);
  roll_r1  = min(X21(:,14))/Sphi(1);
  roll_r1s = roll_r1-(1.39+d3r1*LEVEL);
  roll_r2  = min(X22(:,14))/Sphi(2);
  roll_r2s = roll_r2-(0.85+d3r2*LEVEL);
  roll_r3  = min(X23(:,14))/Sphi(3);
  roll_r3s = roll_r3-(0.75+d3r3*LEVEL);
  yaw_r1   = max(X31(:,16))/X31(length(t),6)/Kpsi;
  yaw_r1s  = yaw_r1-(1.42+d3y1*LEVEL);                    %\
  yaw_r2   = max(X32(:,16))/X32(length(t),6)/Kpsi;         %\
  yaw_r2s  = yaw_r2-(0.67+d3y2*LEVEL);              % impulse !
  yaw_r3   = max(X33(:,16))/X33(length(t),6)/Kpsi;         %/
  yaw_r3s  = yaw_r3-(0.36+d3y3*LEVEL);                    %/
end;
```

```
Xps=[  5    10   15   20   25   30];
Yps=[0.70 0.58 0.42 0.35 0.28 0.25];
Pp=polyfit(Xps,Yps,2); Xpi=5:.25:30; Yp=polyval(Pp,Xpi);
Xrs=[  10   20   30   40   50   60];
Yrs=[1.39 1.07 0.90 0.80 0.75 0.70];
Pr=polyfit(Xrs,Yrs,2); Xri=10:.5:60; Yr=polyval(Pr,Xri);
Xys=[  10   20   30   40   50   60];
Yys=[1.42 1.06 0.79 0.54 0.43 0.36];
Py=polyfit(Xys,Yys,2); Xyi=10:.5:60; Yy=polyval(Py,Xyi);
Ypsp=Yps+LEVEL*([1.63 1.42 1.25 1.10 1.00 0.93]-Yps);
Ppp=polyfit(Xps,Ypsp,2); Ypp=polyval(Ppp,Xpi);
Yrsp=Yrs+LEVEL*([2.35 1.85 1.50 1.21 0.98 0.80]-Yrs);
Prp=polyfit(Xrs,Yrsp,2); Yrp=polyval(Prp,Xri);
Yysp=Yys+LEVEL*([2.40 1.87 1.58 1.33 1.13 1.00]-Yys);
Pyp=polyfit(Xys,Yysp,2); Yyp=polyval(Pyp,Xyi);
ly=length(Yp);

% P
subplot(343);
if LEVEL == 0,
fill([0,Xpi,35,35,0],[Yp(1),Yp,Yp(ly),2.5,2.5],'b',...
     [0,Xpi,35,35,0],[Yp(1),Yp,Yp(ly),0,0],'r');
else,
fill([0,Xpi,35,35,0],[Ypp(1),Ypp,Ypp(ly),2.5,2.5],'c',...
     [0,Xpi,35,35,Xpi(ly:-1:1),0],...
     [Yp(1),Yp,Yp(ly),Ypp(ly),Ypp(ly:-1:1),Ypp(1)],'b',...
     [0,Xpi,35,35,0],[Yp(1),Yp,Yp(ly),0,0],'r');
end;
if exist('COLOR') == 0, title('Spec 3 : Quickness P');
else,                   title('3 : Quickness P'); end;
axis([0,35,0,2.5]); hold on;
xlabel('Step(P) [deg]'); ylabel('max(q)/Step(P) [1/sec]');
putfill(   5,max(pitch_r1),0,35,0,2.5,1);
putfill(17.5,max(pitch_r2),0,35,0,2.5,1);
putfill(  30,max(pitch_r3),0,35,0,2.5,1);
ind=''; if pitch_r1s < -delta, ind='*'; end;
xtext([0,35],1.75,['7 ',ind],']',mp);
ind=''; if pitch_r2s < -delta, ind='*'; end;
xtext([0,35],1.25,['8 ',ind],']',mp);
ind=''; if pitch_r3s < -delta, ind='*'; end;
xtext([0,35],0.75,['9 ',ind],']',mp);
hold off;

% R
subplot(344);
if LEVEL == 0,
fill([0,Xri,70,70,0],[Yr(1),Yr,Yr(ly),2.5,2.5],'b',...
     [0,Xri,70,70,60,50,0],[Yr(1),Yr,Yr(ly),0.4,0.4,0.45,0.45],'r',...
     [0,50,60,70,70,0],[0.45,0.45,0.4,0.4,0,0],'m');
else,
fill([0,Xri,70,70,0],[Yrp(1),Yrp,Yrp(ly),2.5,2.5],'c',...
     [0,Xri,70,70,Xri(ly:-1:1),0],...
     [Yr(1),Yr,Yr(ly),Yrp(ly),Yrp(ly:-1:1),Yrp(1)],'b',...
     [0,Xri,70,70,60,50,0],[Yr(1),Yr,Yr(ly),0.4,0.4,0.45,0.45],'r',...
     [0,50,60,70,70,0],[0.45,0.45,0.4,0.4,0,0],'m');
end;
if exist('COLOR') == 0, title('Spec 3 : Quickness R');
else,                   title('3 : Quickness R'); end;
axis([0,70,0,2.5]); hold on;
xlabel('Step(R) [deg]'); ylabel('max(p)/Step(R) [1/sec]');
putfill(10,max(roll_r1),0,70,0,2.5,2);
putfill(35,max(roll_r2),0,70,0,2.5,2);
```

```
putfill(60,max(roll_r3),0,70,0,2.5,2);
ind=''; if roll_r1s < -delta, ind='*'; end;
xtext([0,70],1.75,['10 ',ind],']',mp);
ind=''; if roll_r2s < -delta, ind='*'; end;
xtext([0,70],1.25,['11 ',ind],']',mp);
ind=''; if roll_r3s < -delta, ind='*'; end;
xtext([0,70],0.75,['12 ',ind],']',mp);
hold off;

% Y
subplot(345);
if LEVEL == 0,
fill([0,Xyi,70,70,0],[Yy(1),Yy,Yy(ly),2.5,2.5],'b',...
     [0,Xyi,70,70,60,10,0],[Yy(1),Yy,Yy(ly),0.17,0.17,0.417,0.417],'r',...
     [0,10,60,70,70,0],[0.417,0.417,0.17,0.17,0,0],'m');
else,
fill([0,Xyi,70,70,0],[Yyp(1),Yyp,Yyp(ly),2.5,2.5],'c',...
     [0,Xyi,70,70,Xyi(ly:-1:1),0],...
     [Yy(1),Yy,Yy(ly),Yyp(ly),Yyp(ly:-1:1),Yyp(1)],'b',...
     [0,Xyi,70,70,60,10,0],[Yy(1),Yy,Yy(ly),0.17,0.17,0.417,0.417],'r',...
     [0,10,60,70,70,0],[0.417,0.417,0.17,0.17,0,0],'m');
end;
if exist('COLOR') == 0, title('Spec 3 : Quickness Y');
else,                   title('3 : Quickness Y'); end;
axis([0,70,0,2.5]); hold on;
xlabel('Step(Y) [deg]'); ylabel('max(r)/Step(Y) [1/sec]');
putfill(10,max(yaw_r1),0,70,0,2.5,3);
putfill(35,max(yaw_r2),0,70,0,2.5,3);
putfill(60,max(yaw_r3),0,70,0,2.5,3);
ind=''; if yaw_r1s < -delta, ind='*'; end;
xtext([0,70],1.75,['13 ',ind],']',mp);
ind=''; if yaw_r2s < -delta, ind='*'; end;
xtext([0,70],1.25,['14 ',ind],']',mp);
ind=''; if yaw_r3s < -delta, ind='*'; end;
xtext([0,70],0.75,['15 ',ind],']',mp);
hold off;

% Spec 4
% ------

if SPEC4 == 0,
  nR=4; aR=Scol;    drscd; X41 = X;
  pitch_d = X23(1:121,18)/Sphi(3);           % [ ]
  roll_d  = X13(1:121,17)/Stet(3);
  yaw_d1 = 180/pi*X41(t3:nt,16);          % [deg/sec]
   Mr1 = max(X41(:,16)); Jr1 = find(X41(:,16) == Mr1);     % for w(t) < 0
   if t(Jr1) >= t3, r1 =  X41(t1,16); else, r1 = Mr1; end; % and r(t) > 0
   r3 = X41(t3,16)-r1;
   y_y = -180/pi*r1/X41(t3,13);
   x_y =  180/pi*r3/X41(t3,13);
   a_y = (x_y-0.15)^2; b_y = (y_y-0.65)^2; c_y = (x_y+0.2)^2;
  if     (x_y >= 0.15) & (y_y < 0.65),
     yaw_d2 = a_y;
  elseif (x_y <  -0.2) & (y_y < 0.65),
     yaw_d2 = c_y;
  elseif (x_y >= -0.2) & (x_y < 0.15) & (y_y >= 0.65),
     yaw_d2 = b_y;
  elseif (x_y >= 0.15) & (y_y >= 0.65),
     yaw_d2 = a_y + b_y;
  elseif (x_y <  -0.2) & (y_y >= 0.65),
     yaw_d2 = b_y + c_y;
  else,
     yaw_d2 = 0;
  end;
```

```
end;

% P/R
subplot(346);
fill([0,4,4,0],[-.25,-.25,.25 ,.25 ],'b',...
     [0,4,4,0],[ .25, .25,.60 ,.60 ],'r',...
     [0,4,4,0],[ .60, .60,.75 ,.75 ],'m',...
     [0,4,4,0],[-.60,-.60,-.25,-.25],'r',...
     [0,4,4,0],[-.75,-.75,-.60,-.60],'m');
hold on;
plot(t(1:121),pitch_d,'y'); axis([0,4,-.75,.75]); hold off;
if exist('COLOR') == 0, title('Spec 4 : Coupling P/R');
else,                   title('4 : Coupling P/R'); end;
xlabel('time [sec]'); ylabel('P/Step(R) ');
ind=''; if max(abs(pitch_d)) > 0.25+delta, ind='*'; end;
xtext([0,4],0,['16 ',ind],']',mp);

% R/P
subplot(347);
fill([0,4,4,0],[-.25,-.25,.25 ,.25 ],'b',...
     [0,4,4,0],[ .25, .25,.60 ,.60 ],'r',...
     [0,4,4,0],[ .60, .60,.75 ,.75 ],'m',...
     [0,4,4,0],[-.60,-.60,-.25,-.25],'r',...
     [0,4,4,0],[-.75,-.75,-.60,-.60],'m');
hold on;
plot(t(1:121),roll_d,'g'); axis([0,4,-.75,.75]); hold off;
if exist('COLOR') == 0, title('Spec 4 : Coupling R/P');
else,                   title('4 : Coupling R/P'); end;
xlabel('time [sec]'); ylabel('R/Step(P) ');
ind=''; if max(abs(roll_d)) > 0.25+delta, ind='*'; end;
xtext([0,4],0,['17',ind],']',mp);

% Y/C
subplot(348);
fill([-1.,-.70,-.70,.7,.7,1.,1.,-1.],[0,0,.90,.90,0,0,1.,1.],'m',...
     [-.7,-.15,-.15,.2,.2,.7,.7,-.7],[0,0,.65,.65,0,0,.9,.9],'r',...
     [-.15,.2,.2,-.15],[0,0,.65,.65],'b'); axis([-1,1,0,1]); hold on;
if exist('COLOR') == 0, title('Spec 4 : Coupling Y/C a');
else,                   title('4 : Coupling Y/C a'); end;
putfill(-x_y,y_y,-1,1,0,1,3);
xlabel('r3/|w(3)| [deg/ft]'); ylabel('|r1/w(3)| [deg/ft]'); hold off;
ind=''; if yaw_d2 > delta, ind='*'; end; xtext([-1,1],0.5,['18 ',ind],']',mp);
subplot(349); t3_5=3:dt:5;
fill([3,5,5,3],[-5 ,-5 ,5 ,5 ],'b',[3,5,5,3],[5,5,10,10],'r',...
     [3,5,5,3],[-10,-10,-5,-5],'r'); hold on;
plot(t3_5,yaw_d1,'w'); axis([3,5,-10,10]); hold off;
if exist('COLOR') == 0, title('Spec 4 : Coupling Y/C b');
else,                   title('4 : Coupling Y/C b'); end;
xlabel('time [sec]'); ylabel('r [deg/sec]');
ind=''; if max(abs(yaw_d1)) > 5+delta, ind='*'; end;
xtext([3,5],0,['19 ',ind],']',mp);
if exist('COLOR') == 0,
  text(2.5,-15,'level 1 - blue');
  if LEVEL > 0, text(5,-15,'level 1+ - cyan '); end;
  text(10,-15,'pitch - yellow');
  text(2.5,-17,'level 2 - red');
  text(10,-17,'roll    - green');
  text(2.5,-19,'level 3 - magenta');
  text(10,-19,'yaw   - white');
end;
% Spec 5
% ------
```

```
if SPEC5 == 0,
  [Agc,Bgc,Cgc,Dgc]=series(Ag,Bg,Cg,Dg,Ac,Bgg,Co,Do);
  [yg1,x] = step(Agc,Bgc(:,1),Cgc(1,:),Dgc(1,1),1,tg);
  pitch_up=x(:,8 ); pitch_ur=K(1)*(x(:,8 )-x(:,15));
  [yg2,x] = step(Agc,Bgc(:,2),Cgc(2,:),Dgc(2,2),1,tg);
  roll_up =x(:,10); roll_ur =K(2)*(x(:,10)-x(:,16));
  [yg3,x] = step(Agc,Bgc(:,3),Cgc(3,:),Dgc(3,3),1,tg);
  yaw_up  =x(:,12); yaw_ur  =K(3)*(x(:,12)-x(:,17));
  pitch_g = 180/pi*yg1;  % [deg]
  roll_g  = 180/pi*yg2;
  yaw_g   = 180/pi*yg3;
end;

Xr=[0,10,10,40,40,0]; Xb=[0,10,10,40,40,10,10,0];

% P
subplot(3,4,10);
fill(Xr,[4,4,0.4,0.4,5,5],'r',Xr,[-4,-4,-0.4,-0.4,-5,-5],'r',...
     Xb,[-4,-4,-0.4,-0.4,0.4,0.4,4,4],'b'); hold on;
plot(tg,pitch_g,'y'); axis([0,40,-5,5]);
if exist('COLOR') == 0, title('Spec 5 : Wind-Gust P');
else,                   title('5 : Wind-Gust P'); end;
xlabel('time [sec]'); ylabel('P [deg]'); hold off;
ind=''; if max(abs(pitch_g(1:n10))) > 4+delta, ind='*'; end;
xtext([0,40],1,['20 ',ind,']',mp);
ind=''; if max(abs(pitch_g(n10+1:nfg))) > 0.4+delta, ind='*'; end;
xtext([0,40],-1,['21 ',ind,']',mp);
if max(pitch_up) > Csu(1) | ...
   min(pitch_up) < Csl(1) | max(abs(pitch_ur)) > abs(Rsl(1)),
  satp='Saturated';
else, satp=''; end;
text(11,1.5,satp);

% R
subplot(3,4,11); roll_g;
fill(Xr,[4,4,0.4,0.4,5,5],'r',Xr,[-4,-4,-0.4,-0.4,-5,-5],'r',...
     Xb,[-4,-4,-0.4,-0.4,0.4,0.4,4,4],'b'); hold on;
plot(tg,roll_g,'g'); axis([0,40,-5,5]);
if exist('COLOR') == 0, title('Spec 5 : Wind-Gust R');
else,                   title('5 : Wind-Gust R'); end;
xlabel('time [sec]'); ylabel('R [deg]'); hold off;
ind=''; if max(abs(roll_g(1:n10))) > 4+delta, ind='*'; end;
xtext([0,40],1,['22 ',ind,']',mp);
ind=''; if max(abs(roll_g(n10+1:nfg))) > 0.4+delta, ind='*'; end;
xtext([0,40],-1,['23 ',ind,']',mp);
if max(roll_up) > Csu(2) | ...
   min(roll_up) < Csl(2) | max(abs(roll_ur)) > abs(Rsl(2)),
  satr='Saturated';
else, satr=''; end;
text(11,1.5,satr);

% Y
subplot(3,4,12);
fill(Xr,[4,4,0.4,0.4,5,5],'r',Xr,[-4,-4,-0.4,-0.4,-5,-5],'r',...
     Xb,[-4,-4,-0.4,-0.4,0.4,0.4,4,4],'b'); hold on;
plot(tg,yaw_g,'w'); axis([0,40,-5,5]);
if exist('COLOR') == 0, title('Spec 5 : Wind-Gust Y');
else,                   title('5 : Wind-Gust Y'); end;
xlabel('time [sec]'); ylabel('Y [deg]'); hold off;
ind=''; if max(abs(yaw_g(1:n10))) > 4+delta, ind='*'; end;
xtext([0,40],1,['24 ',ind,']',mp);
ind=''; if max(abs(yaw_g(n10+1:nfg))) > 0.4+delta, ind='*'; end;
```

```
xtext([0,40],-1,['25 ',ind,']',mp);
if max(yaw_up) > Csu(3) | ...
   min(yaw_up) < Csl(3) | max(abs(yaw_ur)) > abs(Rsl(3)),
  saty='Saturated';
else, saty=''; end;
text(11,1.5,saty);

axis('normal');
```

attitude.m - Attitude response

```
% MATLAB M-file (SCRIPT !!!) for ATTITUDE CHANGE DISPLAY

%
% Gil Yudilevitch      ISR    02-20-94
%

l=length(t);
O=ones(1,1);

figure(1); clf;

subplot(3,3,1); nplot(t,[X11(:,18)/DtR,5*O   ],2);
ylabel('Small'); title('Pitch'); rtext(-0.2,5,'5'); ftext(-2,2,'Small');
subplot(3,3,4); nplot(t,[X12(:,18)/DtR,17.5*O],2);
ylabel('Medium'); rtext(-0.2,17.5,'17.5'); ftext(-2,10,'Medium');
subplot(3,3,7); nplot(t,[X13(:,18)/DtR,30*O  ],2);
ylabel('Large'); rtext(-0.2,30,'30'); ftext(-2,15,'Large');

Mn=min(min(Ur13(:,1))); Mx=max(max(Ur13(:,1)));

subplot(3,3,2); nplot(t,[X21(:,17)/DtR,-10*O],2); title('Roll');
subplot(3,3,5); nplot(t,[X22(:,17)/DtR,-35*O],2);
subplot(3,3,8); nplot(t,[X23(:,17)/DtR,-60*O],2);

subplot(3,3,3); nplot(t,[X31(:,19)/DtR,10*O],2); title('Yaw');
rtext(-0.2,10,'10');
subplot(3,3,6); nplot(t,[X32(:,19)/DtR,35*O],2);
rtext(-0.2,35,'35');
subplot(3,3,9); nplot(t,[X33(:,19)/DtR,60*O],2);
rtext(-0.2,60,'60');
```

actuator.m - Actuator rates and strokes

```
% MATLAB M-file (SCRIPT !!!) for ACTUATOR RATES & STROKES DISPLAY

%
% Gil Yudilevitch      ISR    02-20-94
%

figure(1); clf;

subplot(3,4,1 ); nplot(t,Ur11(:,1),2); title('Pitch'); ftext(-2,0,'Small');
subplot(3,4,5 ); nplot(t,Ur12(:,1),2); ftext(-2,-5,'Medium');
subplot(3,4,9 ); nplot(t,Ur13(:,1),2); ftext(-2,-5,'Large');

Mn=min(min(Ur13(:,1))); Mx=max(max(Ur13(:,1)));
```

```
subplot(3,4,2 ); nplot(t,Ur21(:,2),2); title('Roll');
subplot(3,4,6 ); nplot(t,Ur22(:,2),2);
subplot(3,4,10); nplot(t,Ur23(:,2),2);

subplot(3,4,3 ); nplot(t,Ur31(:,3),2); title('Yaw');
subplot(3,4,7 ); nplot(t,Ur32(:,3),2);
subplot(3,4,11); nplot(t,Ur33(:,3),2);

subplot(3,4,4 ); nplot(t,Ur41(:,4),2); title('Collective');

Title = 'Actuators Strokes';

figure(2); clf;

subplot(3,4,1 ); nplot(t,Us11(:,1),2); ftext(-2,-0.5,'Small'); title('Pitch');
subplot(3,4,5 ); nplot(t,Us12(:,1),2); ftext(-2,-1.8,'Medium');
subplot(3,4,9 ); nplot(t,Us13(:,1),2); ftext(-2,-2.5,'Large');

Mn=min(min(Us13(:,1))); Mx=max(max(Us13(:,1)));

subplot(3,4,2 ); nplot(t,Us21(:,2),2); title('Roll');
subplot(3,4,6 ); nplot(t,Us22(:,2),2);
subplot(3,4,10); nplot(t,Us23(:,2),2);

subplot(3,4,3 ); nplot(t,Us31(:,3),2); title('Yaw');
subplot(3,4,7 ); nplot(t,Us32(:,3),2);
```

B A tutorial example

In the following design example the initial controller is the final ADOCS design of [18], with the crossfeed gains K_{rp}, K_{pr} and K_{cy} initially set to zero. The final design is the nominal optimal design (see Section 5). Two changes are implemented in this example. First, a correct "broken-loop" scheme of Figure 4.3 is used (see Remark 4.3). Second, updated (75 ms) time delays for all channels are used. This final optimal design has better "damping" characteristics than the design of Section 5.

The tutorial example is a complete "record" of the screen (except some unnecessary messages). In addition some comments are given following the % sign. They are not generated by the computer during the design process.

```
% The following tap command only works in the GLUE (UMCP) system

archimedes:/software/control/adocs/pjpotter/adocs/convert: tap console

% Convert the adocs file (which includes dp.adocs.fin) to adocs.o and adocs.d
% files for use by Solve

archimedes:/software/control/adocs/pjpotter/adocs/convert: convert adocs

                    Welcome to CONSOL-OPTCAD (TM)
                  CONVERT Version 1.2 (Released 5/92)

      Copyright (c) 1991, University of Maryland at College Park.
                          All Rights Reserved.
           (developed by  Michael K.H. Fan,  Andre L. Tits,
            Jian L. Zhou,  Li-Sheng Wang and  Jan Koninckx)

[processing adocs]
   [processing dp_adocs]
   [processing adocs.dp.fin]
design parameter Ktet is set to 10.40000
design parameter Kphi is set to 8.40000
design parameter Kpsi is set to 7.60000
design parameter Kq is set to 6.80000
design parameter Kp is set to 1.30000
design parameter Kr is set to 4.00000
design parameter Mtet is set to 2.00000
design parameter Mphi is set to 2.50000
design parameter Mpsi is set to 2.00000
design parameter Kr_p is set to 0.00000
design parameter Kp_r is set to 0.00000
design parameter Kc_y is set to 0.00000
   [processing stable.all]
   [processing object.sat]
   [processing spec1.pit]
   [processing spec1.rol]
   [processing spec1.yaw]
   [processing spec2.pit]
   [processing spec2.rol]
```

```
   [processing spec2.yaw]
   [processing spec3.pit]
   [processing spec3.rol]
   [processing spec3.yaw]
   [processing spec4.pit]
   [processing spec4.rol]
   [processing spec4.yaw]
   [processing spec5.pit]
    [processing spec5.rol]
   [processing spec5.yaw]
warning: design parameter Kpsi never used
warning: design parameter Ktet never used
warning: design parameter Mpsi never used
warning: design parameter Mtet never used
warning: design parameter Kp never used
warning: design parameter Kq never used
warning: design parameter Kr never used
warning: design parameter Kc_y never used
warning: design parameter Kp_r never used
warning: design parameter Kr_p never used
warning: design parameter Kphi never used
warning: design parameter Mphi never used
[compiling adocs.c]
[writing adocs.d]

% After copying adocs.d and adocs.o files to working solve directory,
% change path to that directory and initiate the Solve command

archimedes:/software/control/adocs/pjpotter/adocs/convert: cd ../solve/1
archimedes:/software/control/adocs/pjpotter/adocs/solve/1: solve -matlab adocs

                  Welcome to CONSOL-OPTCAD (TM)
                SOLVE Version 1.7 (Released 8/92)

      Copyright (c) 1991, University of Maryland at College Park.
                          All Rights Reserved.
           (developed by  Michael K.H. Fan,  Andre L. Tits,
            Jian L. Zhou,  Li-Sheng Wang and  Jan Koninckx)

[loading/reading adocs.o and ...]
[reading adocs.d]
[calling simulator initialization (if any)]
[connecting to MATLAB engine ...]
[including file init.m ...]
[calling problem initialization (if any)]

type "help" for help
type "help info" for information

% Identify command to identify the optimization problem

<0> identify
PROBLEM: adocs
 12  Design Parameter(s)
  3  Objective(s)
 20  Constraint(s)
 18  Functional Constraint(s)

% Print command to ensure correct initial design parameters
% (dp's) were included

<0> print

Name        Value        Variation  wrt 0   Prev   Iter=0
```

```
Ktet        1.04000e+01  1.0e+00
Kphi        8.40000e+00  1.0e+00
Kpsi        7.60000e+00  1.0e+00
Kq          6.80000e+00  1.0e+00
Kp          1.30000e+00  1.0e+00
Kr          4.00000e+00  1.0e+00
Mtet        2.00000e+00  1.0e+00
Mphi        2.50000e+00  1.0e+00
Mpsi        2.00000e+00  1.0e+00
Kr_p        0.00000e+00  1.0e+00
Kp_r        0.00000e+00  1.0e+00
Kc_y        0.00000e+00  1.0e+00

% Pcomb command to get an initial look at the
% feasibility problem (note Phase 2)

<0> pcomb

Pcomb (Iter=  0) (Phase 2) (MAX_COST_SOFT= 48.0187)

SPECIFICATION    PRESENT   GOOD               G          B             BAD
O1   p_act_rate -9.19e+01  0.00e+00 <==       |          |             1.00e+00
O2   r_act_rate -9.60e+01  0.00e+00 <==       |          |             1.00e+00
O3   y_act_rate -6.53e+01  0.00e+00 <==       |          |             1.00e+00
C1   stable all -6.04e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.36e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.70e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    2.72e-03  0.00e+00 ================================>  1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    6.68e-03  0.00e+00 ================================>  1.00e-03
C11  pit quick1  4.24e-01  0.00e+00 <================================= -6.00e-02
C12  pit quick2  6.26e-01  0.00e+00 <================================= -4.00e-02
C13  pit quick3  6.07e-01  0.00e+00 <================================= -3.00e-02
C14  rol quick1 -6.72e-01  0.00e+00           |          |        ==> -1.40e-02
C15  rol quick2 -2.54e-01  0.00e+00           |          |        ==> -8.00e-02
C16  rol quick3 -2.53e-01  0.00e+00           |          |        ==> -7.00e-02
C17  yaw quick1  8.31e-01  0.00e+00 <================================= -1.00e-01
C18  yaw quick2  5.42e-01  0.00e+00 <================================= -7.00e-02
C19  yaw quick3  5.60e-01  0.00e+00 <================================= -4.00e-02
C20  yaw dec2 d  3.31e-03  0.00e+00 ============================*      2.00e-03
FC1  pit dec up -1.94e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.40e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  2.64e-02  0.00e+00     *============================= -5.00e-02
FC5  yaw dec1 u -4.01e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  3.82e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.22e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  3.92e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -4.47e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  3.10e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -2.92e-01  0.00e+00 *         |          |             3.00e-01
FC12 r gust p l  3.57e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.25e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l -1.61e-03  0.00e+00           |*=====================  -3.00e-02
FC15 y gust p u -3.06e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.96e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.36e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  3.31e-01  0.00e+00 <================================= -3.00e-02

% Sim command, connects to MATLAB to allow graphical interpretation of pcomb
```

```
<0> sim
Enter MATLAB, type 'back' to leave
>> specs
>> prspecs
>> print
>> back
back to SOLVE

% Run command, run 2 iterations of the feasibility problem

<0> run 2

<2> print

Name        Value        Variation  wrt 0   Prev   Iter=2
Ktet        1.02668e+01  1.0e+00      -1%     0%
Kphi        8.85963e+00  1.0e+00       5%     1%
Kpsi        7.48843e+00  1.0e+00      -1%     0%
Kq          6.86021e+00  1.0e+00       0%     0%
Kp          1.78719e+00  1.0e+00      37%     4%
Kr          4.25991e+00  1.0e+00       6%     0%
Mtet        1.99901e+00  1.0e+00       0%     0%
Mphi        4.75462e+00  1.0e+00      90%     6%
Mpsi        2.00000e+00  1.0e+00       0%     0%
Kr_p        9.58741e-02  1.0e+00    ****% -1442%
Kp_r       -3.38780e-01  1.0e+00    ****%    50%
Kc_y       -5.40903e-03  1.0e+00    ****%  ****%
<2> pcomb

Pcomb (Iter=  2) (Phase 2) (MAX_COST_SOFT= 1.93252)

SPECIFICATION    PRESENT   GOOD                G          B             BAD
O1   p_act_rate -9.19e+01  0.00e+00 <==         |          |            1.00e+00
O2   r_act_rate -7.35e+01  0.00e+00 <==         |          |            1.00e+00
O3   y_act_rate -6.53e+01  0.00e+00 <==         |          |            1.00e+00
C1   stable all -4.43e-02  0.00e+00 <--         |          |            1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*           |            2.00e-03
C3   pit Td     -2.30e-01  0.00e+00 <--         |          |            1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*           |            2.00e-03
C5   rol Td     -2.85e-01  0.00e+00 <--         |          |            1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*           |            2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--         |          |            1.00e-04
C8   pit damp    1.93e-03  0.00e+00 ================================*   1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*           |            1.00e-03
C10  yaw damp    1.34e-03  0.00e+00 ========================*           1.00e-03
C11  pit quick1  4.34e-01  0.00e+00 <================================== -6.00e-02
C12  pit quick2  6.33e-01  0.00e+00 <================================== -4.00e-02
C13  pit quick3  6.11e-01  0.00e+00 <================================== -3.00e-02
C14  rol quick1  4.20e-02  0.00e+00 <================================== -1.40e-02
C15  rol quick2  3.00e-01  0.00e+00 <================================== -8.00e-02
C16  rol quick3 -5.72e-02  0.00e+00           |        *============== -7.00e-02
C17  yaw quick1  8.23e-01  0.00e+00 <================================== -1.00e-01
C18  yaw quick2  5.39e-01  0.00e+00 <================================== -7.00e-02
C19  yaw quick3  5.59e-01  0.00e+00 <================================== -4.00e-02
C20  yaw dec2 d  1.78e-03  0.00e+00 ====================*|             2.00e-03
FC1  pit dec up -1.68e-01  0.00e+00 <==         |          |            5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================== -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==         |          |            5.00e-02
FC4  rol dec lo  3.48e-02  0.00e+00   *============================== -5.00e-02
FC5  yaw dec1 u -4.14e+00  0.00e+00 <==         |          |            5.00e-01
FC6  yaw dec1 l  4.20e+00  0.00e+00 <================================== -5.00e-01
FC7  p gust p u -2.22e+00  0.00e+00 <==         |          |            3.00e-01
FC8  p gust p l  3.97e+00  0.00e+00 <================================== -3.00e-01
```

```
FC9  p gust t u -4.28e-01  0.00e+00 <==       |          |            3.00e-02
FC10 p gust t l  2.98e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -6.91e-01  0.00e+00 <==       |          |            3.00e-01
FC12 r gust p l  3.74e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.57e-01  0.00e+00 <==       |          |            3.00e-02
FC14 r gust t l  1.44e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.07e+00  0.00e+00 <==       |          |            3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.30e-01  0.00e+00 <==       |          |            3.00e-02
FC18 y gust t l  3.17e-01  0.00e+00 <================================ -3.00e-02

<2> sim
Enter MATLAB, type 'back' to leave
>> specs
>> prspecs
>> print
>> back
back to SOLVE

% Still in Phase 2, run 2 more iterations

<2> run 2

<4> print

Name        Value        Variation  wrt 0   Prev   Iter=4
Ktet         9.84782e+00  1.0e+00      -5%    -2%
Kphi         9.18396e+00  1.0e+00       9%     2%
Kpsi         7.38528e+00  1.0e+00      -2%     0%
Kq           7.05046e+00  1.0e+00       3%     1%
Kp           2.17128e+00  1.0e+00      67%    11%
Kr           4.48474e+00  1.0e+00      12%     2%
Mtet         1.99720e+00  1.0e+00       0%     0%
Mphi         5.87986e+00  1.0e+00     135%    12%
Mpsi         2.00000e+00  1.0e+00       0%     0%
Kr_p         6.68756e-02  1.0e+00    ****%    77%
Kp_r        -3.21602e-01  1.0e+00    ****%    10%
Kc_y        -6.54780e-02  1.0e+00    ****%    71%
<4> pcomb

Pcomb (Iter=  4) (Phase 2) (MAX_COST_SOFT= 0.0666746)

SPECIFICATION    PRESENT   GOOD                G          B             BAD
01   p_act_rate -9.19e+01  0.00e+00 <==        |          |             1.00e+00
02   r_act_rate -6.26e+01  0.00e+00 <==        |          |             1.00e+00
03   y_act_rate -6.49e+01  0.00e+00 <==        |          |             1.00e+00
C1   stable all -4.52e-02  0.00e+00 <--        |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.30e-01  0.00e+00 <--        |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.90e-01  0.00e+00 <--        |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--        |          |             1.00e-04
C8   pit damp    6.67e-05  0.00e+00 ===========*         |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  4.27e-01  0.00e+00 <================================ -6.00e-02
C12  pit quick2  6.25e-01  0.00e+00 <================================ -4.00e-02
C13  pit quick3  6.04e-01  0.00e+00 <================================ -3.00e-02
C14  rol quick1  2.46e-01  0.00e+00 <================================ -1.40e-02
C15  rol quick2  5.71e-01  0.00e+00 <================================ -8.00e-02
C16  rol quick3  9.74e-02  0.00e+00 <================================ -7.00e-02
C17  yaw quick1  8.26e-01  0.00e+00 <================================ -1.00e-01
C18  yaw quick2  5.39e-01  0.00e+00 <================================ -7.00e-02
```

```
C19  yaw quick3  5.60e-01  0.00e+00 <================================= -4.00e-02
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |              2.00e-03
FC1  pit dec up -1.66e-01  0.00e+00 <==       |          |              5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |              5.00e-02
FC4  rol dec lo  4.21e-02  0.00e+00  *============================== -5.00e-02
FC5  yaw dec1 u -4.25e+00  0.00e+00 <==       |          |              5.00e-01
FC6  yaw dec1 l  4.79e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.20e+00  0.00e+00 <==       |          |              3.00e-01
FC8  p gust p l  3.96e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -4.43e-01  0.00e+00 <==       |          |              3.00e-02
FC10 p gust t l  2.89e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -8.98e-01  0.00e+00 <==       |          |              3.00e-01
FC12 r gust p l  3.74e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.53e-01  0.00e+00 <==       |          |              3.00e-02
FC14 r gust t l  1.42e-01  0.00e+00 <================================= -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |              3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.31e-01  0.00e+00 <==       |          |              3.00e-02
FC18 y gust t l  3.18e-01  0.00e+00 <================================= -3.00e-02

<4> sim
Enter MATLAB, type 'back' to leave
>> specs
>> prspecs
>> print
>> back
back to SOLVE

% Still in phase 2, run 1 iteration

<4> run 1

<5> print

Name        Value        Variation  wrt 0   Prev   Iter=5
Ktet        9.78278e+00  1.0e+00      -5%     0%
Kphi        9.23057e+00  1.0e+00       9%     0%
Kpsi        7.37243e+00  1.0e+00      -2%     0%
Kq          7.07734e+00  1.0e+00       4%     0%
Kp          2.22992e+00  1.0e+00      71%     2%
Kr          4.51307e+00  1.0e+00      12%     0%
Mtet        1.99695e+00  1.0e+00       0%     0%
Mphi        6.04718e+00  1.0e+00     141%     2%
Mpsi        2.00000e+00  1.0e+00       0%     0%
Kr_p        6.21835e-02  1.0e+00    ****%    -7%
Kp_r       -3.24220e-01  1.0e+00    ****%     0%
Kc_y       -7.20799e-02  1.0e+00    ****%    10%
<5> pcomb

Pcomb (Iter=  5) (Phase 2) (MAX_COST_SOFT= 0.00946629)

SPECIFICATION    PRESENT   GOOD              G          B          BAD
01   p_act_rate -9.19e+01  0.00e+00 <==       |          |          1.00e+00
02   r_act_rate -6.25e+01  0.00e+00 <==       |          |          1.00e+00
03   y_act_rate -6.49e+01  0.00e+00 <==       |          |          1.00e+00
C1   stable all -4.53e-02  0.00e+00 <--       |          |          1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |          2.00e-03
C3   pit Td     -2.30e-01  0.00e+00 <--       |          |          1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |          2.00e-03
C5   rol Td     -2.91e-01  0.00e+00 <--       |          |          1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |          2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--       |          |          1.00e-04
C8   pit damp    9.47e-06  0.00e+00 ==========*          |          1.00e-03
```

```
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  4.26e-01  0.00e+00 <================================ -6.00e-02
C12  pit quick2  6.24e-01  0.00e+00 <================================ -4.00e-02
C13  pit quick3  6.03e-01  0.00e+00 <================================ -3.00e-02
C14  rol quick1  2.45e-01  0.00e+00 <================================ -1.40e-02
C15  rol quick2  6.07e-01  0.00e+00 <================================ -8.00e-02
C16  rol quick3  1.19e-01  0.00e+00 <================================ -7.00e-02
C17  yaw quick1  8.27e-01  0.00e+00 <================================ -1.00e-01
C18  yaw quick2  5.39e-01  0.00e+00 <================================ -7.00e-02
C19  yaw quick3  5.60e-01  0.00e+00 <================================ -4.00e-02
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.66e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================ -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  4.31e-02  0.00e+00  *============================== -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.85e+00  0.00e+00 <================================ -5.00e-01
FC7  p gust p u -2.19e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  3.96e+00  0.00e+00 <================================ -3.00e-01
FC9  p gust t u -4.44e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.88e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -9.25e-01  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.74e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.53e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  1.42e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.31e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  3.18e-01  0.00e+00 <================================ -3.00e-02

<5> run 1

<6> print

Name         Value         Variation  wrt 0   Prev   Iter=6
Ktet         9.75853e+00   1.0e+00      -6%     0%
Kphi         9.24782e+00   1.0e+00      10%     0%
Kpsi         7.36767e+00   1.0e+00      -3%     0%
Kq           7.08728e+00   1.0e+00       4%     0%
Kp           2.25177e+00   1.0e+00      73%     0%
Kr           4.52359e+00   1.0e+00      13%     0%
Mtet         1.99686e+00   1.0e+00       0%     0%
Mphi         6.10938e+00   1.0e+00     144%     1%
Mpsi         2.00000e+00   1.0e+00       0%     0%
Kr_p         6.02392e-02   1.0e+00    ****%    -3%
Kp_r        -3.25297e-01   1.0e+00    ****%     0%
Kc_y        -7.45238e-02   1.0e+00    ****%     3%
<6> pcomb

Pcomb (Iter=  6) (Phase 2) (MAX_COST_SOFT= 0.00134145)

SPECIFICATION    PRESENT   GOOD              G          B          BAD
O1   p_act_rate -9.19e+01  0.00e+00 <==      |          |          1.00e+00
O2   r_act_rate -6.24e+01  0.00e+00 <==      |          |          1.00e+00
O3   y_act_rate -6.49e+01  0.00e+00 <==      |          |          1.00e+00
C1   stable all -4.53e-02  0.00e+00 <--      |          |          1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |          2.00e-03
C3   pit Td     -2.30e-01  0.00e+00 <--      |          |          1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |          2.00e-03
C5   rol Td     -2.91e-01  0.00e+00 <--      |          |          1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |          2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--      |          |          1.00e-04
```

```
C8   pit damp    1.34e-06  0.00e+00 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  4.25e-01  0.00e+00 <================================ -6.00e-02
C12  pit quick2  6.24e-01  0.00e+00 <================================ -4.00e-02
C13  pit quick3  6.02e-01  0.00e+00 <================================ -3.00e-02
C14  rol quick1  2.45e-01  0.00e+00 <================================ -1.40e-02
C15  rol quick2  6.20e-01  0.00e+00 <================================ -8.00e-02
C16  rol quick3  1.27e-01  0.00e+00 <================================ -7.00e-02
C17  yaw quick1  8.27e-01  0.00e+00 <================================ -1.00e-01
C18  yaw quick2  5.39e-01  0.00e+00 <================================ -7.00e-02
C19  yaw quick3  5.60e-01  0.00e+00 <================================ -4.00e-02
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |              2.00e-03
FC1  pit dec up -1.66e-01  0.00e+00 <==       |          |              5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================ -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |              5.00e-02
FC4  rol dec lo  4.35e-02  0.00e+00 *================================ -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |              5.00e-01
FC6  yaw dec1 l  4.87e+00  0.00e+00 <================================ -5.00e-01
FC7  p gust p u -2.19e+00  0.00e+00 <==       |          |              3.00e-01
FC8  p gust p l  3.96e+00  0.00e+00 <================================ -3.00e-01
FC9  p gust t u -4.45e-01  0.00e+00 <==       |          |              3.00e-02
FC10 p gust t l  2.87e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -9.35e-01  0.00e+00 <==       |          |              3.00e-01
FC12 r gust p l  3.74e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.53e-01  0.00e+00 <==       |          |              3.00e-02
FC14 r gust t l  1.42e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |              3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.31e-01  0.00e+00 <==       |          |              3.00e-02
FC18 y gust t l  3.18e-01  0.00e+00 <================================ -3.00e-02

% Still in phase 2, run 2 more iterations

<6> run 2

<8> pcomb

Pcomb (Iter=  8) (Phase 2) (MAX_COST_SOFT= 2.66865e-05)

SPECIFICATION    PRESENT   GOOD               G          B             BAD
O1   p_act_rate -9.19e+01  0.00e+00 <==       |          |             1.00e+00
O2   r_act_rate -6.24e+01  0.00e+00 <==       |          |             1.00e+00
O3   y_act_rate -6.49e+01  0.00e+00 <==       |          |             1.00e+00
C1   stable all -4.54e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.30e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.91e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    2.67e-08  0.00e+00 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  4.25e-01  0.00e+00 <================================ -6.00e-02
C12  pit quick2  6.23e-01  0.00e+00 <================================ -4.00e-02
C13  pit quick3  6.02e-01  0.00e+00 <================================ -3.00e-02
C14  rol quick1  2.45e-01  0.00e+00 <================================ -1.40e-02
C15  rol quick2  6.26e-01  0.00e+00 <================================ -8.00e-02
C16  rol quick3  1.31e-01  0.00e+00 <================================ -7.00e-02
C17  yaw quick1  8.27e-01  0.00e+00 <================================ -1.00e-01
C18  yaw quick2  5.39e-01  0.00e+00 <================================ -7.00e-02
C19  yaw quick3  5.60e-01  0.00e+00 <================================ -4.00e-02
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |              2.00e-03
```

```
FC1  pit dec up -1.66e-01  0.00e+00 <==       |          |            5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================ -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |            5.00e-02
FC4  rol dec lo  4.36e-02  0.00e+00 *================================ -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |            5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================ -5.00e-01
FC7  p gust p u -2.19e+00  0.00e+00 <==       |          |            3.00e-01
FC8  p gust p l  3.96e+00  0.00e+00 <================================ -3.00e-01
FC9  p gust t u -4.45e-01  0.00e+00 <==       |          |            3.00e-02
FC10 p gust t l  2.87e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -9.41e-01  0.00e+00 <==       |          |            3.00e-01
FC12 r gust p l  3.74e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.53e-01  0.00e+00 <==       |          |            3.00e-02
FC14 r gust t l  1.42e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |            3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.31e-01  0.00e+00 <==       |          |            3.00e-02
FC18 y gust t l  3.18e-01  0.00e+00 <================================ -3.00e-02

% Note numerous constraints on the "good" boundary, especially C8, use
% 'sim' to graphically display pcomb results

<8> sim
Enter MATLAB, type 'back' to leave
>> specs
>> prspecs
>> print
>> back
back to SOLVE

% Push good value of C8 just above present value, using setgb command,
% to shake process into phase 3

<8> setgb C 8 = 1.00e-07,1.00e-03
<8> run 0
<8> pcomb

% Note Phase 3

Pcomb (Iter=  8) (Phase 3) (MAX_COST= -62.3947)

SPECIFICATION    PRESENT   GOOD               G          B             BAD
O1   p_act_rate -9.19e+01  0.00e+00 <==       |          |            1.00e+00
O2   r_act_rate -6.24e+01  0.00e+00 <==       |          |            1.00e+00
O3   y_act_rate -6.49e+01  0.00e+00 <==       |          |            1.00e+00
C1   stable all -4.54e-02  0.00e+00 <--       |          |            1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C3   pit Td     -2.30e-01  0.00e+00 <--       |          |            1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C5   rol Td     -2.91e-01  0.00e+00 <--       |          |            1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--       |          |            1.00e-04
C8   pit damp    2.67e-08  1.00e-07 ==========*          |            1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |            1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |            1.00e-03
C11  pit quick1  4.25e-01  0.00e+00 <================================ -6.00e-02
C12  pit quick2  6.23e-01  0.00e+00 <================================ -4.00e-02
C13  pit quick3  6.02e-01  0.00e+00 <================================ -3.00e-02
C14  rol quick1  2.45e-01  0.00e+00 <================================ -1.40e-02
C15  rol quick2  6.26e-01  0.00e+00 <================================ -8.00e-02
C16  rol quick3  1.31e-01  0.00e+00 <================================ -7.00e-02
C17  yaw quick1  8.27e-01  0.00e+00 <================================ -1.00e-01
C18  yaw quick2  5.39e-01  0.00e+00 <================================ -7.00e-02
C19  yaw quick3  5.60e-01  0.00e+00 <================================ -4.00e-02
```

```
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.66e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  4.36e-02  0.00e+00 *================================= -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.19e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  3.96e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -4.45e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.87e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -9.41e-01  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.74e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.53e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  1.42e-01  0.00e+00 <================================= -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.31e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  3.18e-01  0.00e+00 <================================= -3.00e-02

% Now in phase 3, concerned only with the objectives, so to speed up process
% reduce priority of following constraints

<8> setgb C 11 =0,-10e4
<8> setgb C 12 =0,-10e4
<8> setgb C 13 =0,-10e4
<8> setgb C 14 =0,-10e4
<8> setgb C 15 =0,-10e4
<8> setgb C 16 =0,-10e4
<8> setgb C 17 =0,-10e4
<8> setgb C 18 =0,-10e4
<8> setgb C 19 =0,-10e4

% Weight the first objective in order to push it toward the boundary

<8> setgb O 1 =0,10
<8> run 0
<8> pcomb

Pcomb (Iter=  8) (Phase 3) (MAX_COST= -9.18756)

SPECIFICATION    PRESENT   GOOD              G          B             BAD
O1   p_act_rate -9.19e+01  0.00e+00 <==       |          |             1.00e+01
O2   r_act_rate -6.24e+01  0.00e+00 <==       |          |             1.00e+00
O3   y_act_rate -6.49e+01  0.00e+00 <==       |          |             1.00e+00
C1   stable all -4.54e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.30e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.91e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    2.67e-08  1.00e-07 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  4.25e-01  0.00e+00           *========================= -1.00e+05
C12  pit quick2  6.23e-01  0.00e+00           *========================= -1.00e+05
C13  pit quick3  6.02e-01  0.00e+00           *========================= -1.00e+05
C14  rol quick1  2.45e-01  0.00e+00           *========================= -1.00e+05
C15  rol quick2  6.26e-01  0.00e+00           *========================= -1.00e+05
C16  rol quick3  1.31e-01  0.00e+00           *========================= -1.00e+05
C17  yaw quick1  8.27e-01  0.00e+00           *========================= -1.00e+05
C18  yaw quick2  5.39e-01  0.00e+00           *========================= -1.00e+05
C19  yaw quick3  5.60e-01  0.00e+00           *========================= -1.00e+05
```

```
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.66e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================ -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  4.36e-02  0.00e+00 *================================ -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================ -5.00e-01
FC7  p gust p u -2.19e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  3.96e+00  0.00e+00 <================================ -3.00e-01
FC9  p gust t u -4.45e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.87e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -9.41e-01  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.74e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.53e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  1.42e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.31e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  3.18e-01  0.00e+00 <================================ -3.00e-02
<8> run 2

<10> print

Name        Value        Variation  wrt 0   Prev   Iter=10
Ktet        9.68140e+00  1.0e+00      -6%     0%
Kphi        9.25585e+00  1.0e+00      10%     0%
Kpsi        7.36426e+00  1.0e+00      -3%     0%
Kq          7.06584e+00  1.0e+00       3%     0%
Kp          2.26597e+00  1.0e+00      74%     0%
Kr          4.52861e+00  1.0e+00      13%     0%
Mtet        1.32703e+00  1.0e+00     -33%    -6%
Mphi        6.14147e+00  1.0e+00     145%     0%
Mpsi        2.00000e+00  1.0e+00       0%     0%
Kr_p        8.29262e-02  1.0e+00    ****%    10%
Kp_r       -3.17911e-01  1.0e+00    ****%    -2%
Kc_y       -7.57832e-02  1.0e+00    ****%     0%
<10> sim
Enter MATLAB, type 'back' to leave
>> specs
>> back
back to SOLVE

% First objective has been pushed to boundary, weight second objective
% to try and push it

<10> setgb 0 1 =0,1
<10> setgb 0 2 =0,10
<10> run 0
<10> pcomb

Pcomb (Iter=  10) (Phase 3) (MAX_COST= -6.2533)

SPECIFICATION    PRESENT   GOOD              G          B            BAD
O1   p_act_rate -9.83e+01  0.00e+00 <==       |          |            1.00e+00
O2   r_act_rate -6.25e+01  0.00e+00 <==       |          |            1.00e+01
O3   y_act_rate -6.49e+01  0.00e+00 <==       |          |            1.00e+00
C1   stable all -4.39e-02  0.00e+00 <--       |          |            1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C3   pit Td     -2.10e-01  0.00e+00 <--       |          |            1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C5   rol Td     -2.91e-01  0.00e+00 <--       |          |            1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--       |          |            1.00e-04
C8   pit damp    6.29e-08  1.00e-07 ==========*          |            1.00e-03
```

```
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  1.40e-02  0.00e+00           *======================= -1.00e+05
C12  pit quick2  2.72e-01  0.00e+00           *======================= -1.00e+05
C13  pit quick3  3.48e-01  0.00e+00           *======================= -1.00e+05
C14  rol quick1  2.49e-01  0.00e+00           *======================= -1.00e+05
C15  rol quick2  6.34e-01  0.00e+00           *======================= -1.00e+05
C16  rol quick3  1.36e-01  0.00e+00           *======================= -1.00e+05
C17  yaw quick1  8.27e-01  0.00e+00           *======================= -1.00e+05
C18  yaw quick2  5.39e-01  0.00e+00           *======================= -1.00e+05
C19  yaw quick3  5.60e-01  0.00e+00           *======================= -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.62e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================ -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  8.11e-02  0.00e+00 <================================ -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================ -5.00e-01
FC7  p gust p u -2.19e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  3.96e+00  0.00e+00 <================================ -3.00e-01
FC9  p gust t u -4.41e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.85e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -9.53e-01  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.76e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.57e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  1.57e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.32e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  3.16e-01  0.00e+00 <================================ -3.00e-02
<10> run 2

<12> print

Name        Value        Variation  wrt 0   Prev   Iter=12
Ktet        9.08983e+00  1.0e+00      -6%    -5%
Kphi        9.55674e+00  1.0e+00       3%     3%
Kpsi        7.35924e+00  1.0e+00       0%     0%
Kq          7.24229e+00  1.0e+00       2%     2%
Kp          2.12648e+00  1.0e+00      -6%    -5%
Kr          4.52575e+00  1.0e+00       0%     0%
Mtet        1.32703e+00  1.0e+00       0%     0%
Mphi        5.97543e+00  1.0e+00      -2%    -2%
Mpsi        2.00000e+00  1.0e+00       0%     0%
Kr_p        2.54135e-01  1.0e+00     206%   153%
Kp_r       -3.08781e-01  1.0e+00      -2%    -3%
Kc_y       -7.57832e-02  1.0e+00       0%     0%
<12> pcomb

Pcomb (Iter=  12) (Phase 3) (MAX_COST= -6.40774)

SPECIFICATION    PRESENT   GOOD              G          B            BAD
O1   p_act_rate -9.82e+01  0.00e+00 <==      |          |            1.00e+00
O2   r_act_rate -6.41e+01  0.00e+00 <==      |          |            1.00e+01
O3   y_act_rate -6.49e+01  0.00e+00 <==      |          |            1.00e+00
C1   stable all -3.19e-02  0.00e+00 <--      |          |            1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--      |          |            1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C5   rol Td     -2.89e-01  0.00e+00 <--      |          |            1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--      |          |            1.00e-04
C8   pit damp    0.00e+00  1.00e-07 ==========*          |            1.00e-03
```

```
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  3.06e-02  0.00e+00           *====================== -1.00e+05
C12  pit quick2  2.85e-01  0.00e+00           *====================== -1.00e+05
C13  pit quick3  3.56e-01  0.00e+00           *====================== -1.00e+05
C14  rol quick1  2.91e-01  0.00e+00           *====================== -1.00e+05
C15  rol quick2  6.61e-01  0.00e+00           *====================== -1.00e+05
C16  rol quick3  1.56e-01  0.00e+00           *====================== -1.00e+05
C17  yaw quick1  8.28e-01  0.00e+00           *====================== -1.00e+05
C18  yaw quick2  5.38e-01  0.00e+00           *====================== -1.00e+05
C19  yaw quick3  5.59e-01  0.00e+00           *====================== -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.34e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================ -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  8.66e-02  0.00e+00 <================================ -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================ -5.00e-01
FC7  p gust p u -2.20e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <================================ -3.00e-01
FC9  p gust t u -3.95e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.64e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -1.02e+00  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.98e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  2.52e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.32e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  2.97e-01  0.00e+00 <================================ -3.00e-02
<12> sim
Enter MATLAB, type 'back' to leave
>> specs
>> back
back to SOLVE

% Note in the last 2 iterations the second objective changed very
% little, rather than try and continue to push it, freeze some dp's (to
% speed up the optimization) and try to push the third objective, then
% return to the second objective

<12> freeze Ktet Kphi Kpsi Kq Kp Kr
<12> setgb 0 2 =0,1e00
<12> setgb 0 3 =0,1e01
<12> run 0
<12> pcomb

Pcomb (Iter=  12) (Phase 3) (MAX_COST= -6.4906)

SPECIFICATION    PRESENT   GOOD               G          B             BAD
01   p_act_rate -9.82e+01  0.00e+00 <==       |          |             1.00e+00
02   r_act_rate -6.41e+01  0.00e+00 <==       |          |             1.00e+00
03   y_act_rate -6.49e+01  0.00e+00 <==       |          |             1.00e+01
C1   stable all -3.19e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.89e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.56e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    0.00e+00  1.00e-07 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  3.06e-02  0.00e+00           *====================== -1.00e+05
```

```
C12  pit quick2  2.85e-01  0.00e+00          *====================  -1.00e+05
C13  pit quick3  3.56e-01  0.00e+00          *====================  -1.00e+05
C14  rol quick1  2.91e-01  0.00e+00          *====================  -1.00e+05
C15  rol quick2  6.61e-01  0.00e+00          *====================  -1.00e+05
C16  rol quick3  1.56e-01  0.00e+00          *====================  -1.00e+05
C17  yaw quick1  8.28e-01  0.00e+00          *====================  -1.00e+05
C18  yaw quick2  5.38e-01  0.00e+00          *====================  -1.00e+05
C19  yaw quick3  5.59e-01  0.00e+00          *====================  -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |            2.00e-03
FC1  pit dec up -1.34e-01  0.00e+00 <==       |          |            5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================ -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |            5.00e-02
FC4  rol dec lo  8.66e-02  0.00e+00 <================================ -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |            5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================ -5.00e-01
FC7  p gust p u -2.20e+00  0.00e+00 <==       |          |            3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <================================ -3.00e-01
FC9  p gust t u -3.95e-01  0.00e+00 <==       |          |            3.00e-02
FC10 p gust t l  2.64e-01  0.00e+00 <================================ -3.00e-02
FC11 r gust p u -1.02e+00  0.00e+00 <==       |          |            3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <================================ -3.00e-01
FC13 r gust t u -4.98e-01  0.00e+00 <==       |          |            3.00e-02
FC14 r gust t l  2.52e-01  0.00e+00 <================================ -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |            3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================ -3.00e-01
FC17 y gust t u -4.32e-01  0.00e+00 <==       |          |            3.00e-02
FC18 y gust t l  2.97e-01  0.00e+00 <================================ -3.00e-02
<12> run 2

<14> sim
Enter MATLAB, type 'back' to leave
>> specs
>> prspecs
>> print
>> back
back to SOLVE
<14> run 2

<16> sim
Enter MATLAB, type 'back' to leave
>> back
back to SOLVE
<16> print

Name        Value        Variation  wrt 0   Prev   Iter=16
Ktet        9.08983e+00  1.0e+00      -6%     0%   frozen
Kphi        9.55674e+00  1.0e+00       3%     0%   frozen
Kpsi        7.35924e+00  1.0e+00       0%     0%   frozen
Kq          7.24229e+00  1.0e+00       2%     0%   frozen
Kp          2.12648e+00  1.0e+00      -6%     0%   frozen
Kr          4.52575e+00  1.0e+00       0%     0%   frozen
Mtet        1.32703e+00  1.0e+00       0%     0%
Mphi        5.98800e+00  1.0e+00      -2%     0%
Mpsi        8.13820e-01  1.0e+00     -59%   -10%
Kr_p        2.19954e-01  1.0e+00     165%     4%
Kp_r       -3.82576e-01  1.0e+00      20%    -4%
Kc_y       -7.57832e-02  1.0e+00       0%     0%
<16> pcomb

Pcomb (Iter=  16) (Phase 3) (MAX_COST= -8.40986)

SPECIFICATION    PRESENT   GOOD              G          B            BAD
01   p_act_rate -9.82e+01  0.00e+00 <==       |          |            1.00e+00
02   r_act_rate -6.42e+01  0.00e+00 <==       |          |            1.00e+00
```

```
03   y_act_rate -8.41e+01  0.00e+00 <==       |          |            1.00e+01
C1   stable all -3.22e-02  0.00e+00 <--       |          |            1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--       |          |            1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C5   rol Td     -2.90e-01  0.00e+00 <--       |          |            1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |            2.00e-03
C7   yaw Td     -3.57e-01  0.00e+00 <--       |          |            1.00e-04
C8   pit damp    0.00e+00  1.00e-07 ==========*          |            1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |            1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |            1.00e-03
C11  pit quick1  3.18e-02  0.00e+00           *==================== -1.00e+05
C12  pit quick2  2.86e-01  0.00e+00           *==================== -1.00e+05
C13  pit quick3  3.57e-01  0.00e+00           *==================== -1.00e+05
C14  rol quick1  2.86e-01  0.00e+00           *==================== -1.00e+05
C15  rol quick2  6.63e-01  0.00e+00           *==================== -1.00e+05
C16  rol quick3  1.58e-01  0.00e+00           *==================== -1.00e+05
C17  yaw quick1  4.46e-03  0.00e+00           *==================== -1.00e+05
C18  yaw quick2  3.91e-01  0.00e+00           *==================== -1.00e+05
C19  yaw quick3  5.10e-01  0.00e+00           *==================== -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |            2.00e-03
FC1  pit dec up -1.40e-01  0.00e+00 <==       |          |            5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <============================== -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |            5.00e-02
FC4  rol dec lo  8.63e-02  0.00e+00 <============================== -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |            5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <============================== -5.00e-01
FC7  p gust p u -2.21e+00  0.00e+00 <==       |          |            3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <============================== -3.00e-01
FC9  p gust t u -3.92e-01  0.00e+00 <==       |          |            3.00e-02
FC10 p gust t l  2.61e-01  0.00e+00 <============================== -3.00e-02
FC11 r gust p u -1.02e+00  0.00e+00 <==       |          |            3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <============================== -3.00e-01
FC13 r gust t u -4.97e-01  0.00e+00 <==       |          |            3.00e-02
FC14 r gust t l  2.47e-01  0.00e+00 <============================== -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |            3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <============================== -3.00e-01
FC17 y gust t u -4.30e-01  0.00e+00 <==       |          |            3.00e-02
FC18 y gust t l  3.00e-01  0.00e+00 <============================== -3.00e-02
<16> sim
Enter MATLAB, type 'back' to leave
>> specs
>> prspecs
>> print
>> back
back to SOLVE

% Return to the second objective, now that the third is at the boundary

<16> setgb 0 3 =0,1e00
<16> setgb 0 2 =0,10

% Return C8 good value to 0 now that the optimization has made it <= 0

<16> setgb C 8 =0,1e-03
<16> run 0
<16> pcomb

Pcomb (Iter=  16) (Phase 3) (MAX_COST= -6.42203)

SPECIFICATION    PRESENT   GOOD              G          B            BAD
01   p_act_rate -9.82e+01  0.00e+00 <==       |          |            1.00e+00
02   r_act_rate -6.42e+01  0.00e+00 <==       |          |            1.00e+01
03   y_act_rate -8.41e+01  0.00e+00 <==       |          |            1.00e+00
```

```
C1   stable all -3.22e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.90e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.57e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  3.18e-02  0.00e+00          *========================= -1.00e+05
C12  pit quick2  2.86e-01  0.00e+00          *========================= -1.00e+05
C13  pit quick3  3.57e-01  0.00e+00          *========================= -1.00e+05
C14  rol quick1  2.86e-01  0.00e+00          *========================= -1.00e+05
C15  rol quick2  6.63e-01  0.00e+00          *========================= -1.00e+05
C16  rol quick3  1.58e-01  0.00e+00          *========================= -1.00e+05
C17  yaw quick1  4.46e-03  0.00e+00          *========================= -1.00e+05
C18  yaw quick2  3.91e-01  0.00e+00          *========================= -1.00e+05
C19  yaw quick3  5.10e-01  0.00e+00          *========================= -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.40e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  8.63e-02  0.00e+00 <================================= -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.21e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -3.92e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.61e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -1.02e+00  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.97e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  2.47e-01  0.00e+00 <================================= -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.30e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  3.00e-01  0.00e+00 <================================= -3.00e-02
<16> run 2

<18> print

Name        Value        Variation  wrt 0   Prev   Iter=18
Ktet        9.08983e+00  1.0e+00       0%     0%   frozen
Kphi        9.55674e+00  1.0e+00       0%     0%   frozen
Kpsi        7.35924e+00  1.0e+00       0%     0%   frozen
Kq          7.24229e+00  1.0e+00       0%     0%   frozen
Kp          2.12648e+00  1.0e+00       0%     0%   frozen
Kr          4.52575e+00  1.0e+00       0%     0%   frozen
Mtet        1.32703e+00  1.0e+00       0%     0%
Mphi        5.98778e+00  1.0e+00       0%     0%
Mpsi        8.13820e-01  1.0e+00       0%     0%
Kr_p        2.21514e-01  1.0e+00       0%     0%
Kp_r       -3.83921e-01  1.0e+00       0%     0%
Kc_y       -7.57832e-02  1.0e+00       0%     0%
<18> pcomb

Pcomb (Iter=  18) (Phase 3) (MAX_COST= -6.42418)

SPECIFICATION    PRESENT   GOOD              G          B             BAD
01   p_act_rate -9.82e+01  0.00e+00 <==      |          |             1.00e+00
02   r_act_rate -6.42e+01  0.00e+00 <==      |          |             1.00e+01
03   y_act_rate -8.41e+01  0.00e+00 <==      |          |             1.00e+00
```

```
C1   stable all -3.21e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.90e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.57e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  3.22e-02  0.00e+00          *========================= -1.00e+05
C12  pit quick2  2.86e-01  0.00e+00          *========================= -1.00e+05
C13  pit quick3  3.57e-01  0.00e+00          *========================= -1.00e+05
C14  rol quick1  2.86e-01  0.00e+00          *========================= -1.00e+05
C15  rol quick2  6.63e-01  0.00e+00          *========================= -1.00e+05
C16  rol quick3  1.59e-01  0.00e+00          *========================= -1.00e+05
C17  yaw quick1  4.44e-03  0.00e+00          *========================= -1.00e+05
C18  yaw quick2  3.91e-01  0.00e+00          *========================= -1.00e+05
C19  yaw quick3  5.10e-01  0.00e+00          *========================= -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.40e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  8.64e-02  0.00e+00 <================================= -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.21e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -3.92e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.61e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -1.03e+00  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.97e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  2.47e-01  0.00e+00 <================================= -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.30e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  2.99e-01  0.00e+00 <================================= -3.00e-02

% Freeze all dp's but the one you are trying to push to the boundary in
% order to help/speed up the optimization. Note slow progress above, 0%
% change wrt 0 for all parameters in previous print

<18> freeze Mtet Mpsi Kr_p Kp_r Kc_y
<18> print

Name        Value        Variation  wrt 0   Prev   Iter=18
Ktet        9.08983e+00  1.0e+00       0%     0%   frozen
Kphi        9.55674e+00  1.0e+00       0%     0%   frozen
Kpsi        7.35924e+00  1.0e+00       0%     0%   frozen
Kq          7.24229e+00  1.0e+00       0%     0%   frozen
Kp          2.12648e+00  1.0e+00       0%     0%   frozen
Kr          4.52575e+00  1.0e+00       0%     0%   frozen
Mtet        1.32703e+00  1.0e+00       0%     0%   frozen
Mphi        5.98778e+00  1.0e+00       0%     0%
Mpsi        8.13820e-01  1.0e+00       0%     0%   frozen
Kr_p        2.21514e-01  1.0e+00       0%     0%   frozen
Kp_r       -3.83921e-01  1.0e+00       0%     0%   frozen
Kc_y       -7.57832e-02  1.0e+00       0%     0%   frozen

<18> run 1
Segmentation fault (core dumped)
```

```
% Console terminated at this point a number of times.  In order to
% return to the same point, the dp's above were introduced into the
% adocs.dp.fin file ('included' in the adocs file) and then adocs was
% converted ('convert adocs') in order to create the adocs.o and adocs.d
% files necessary to run solve. Rather than trying to force the roll
% channel down to the boundary from above (thereby introducing
% segmentation faults as before), the parameter Mphi was reduced to
% bring the roll channel below the level I/level II boundary and thus
% the optimization back into phase 2

archimedes:/software/control/adocs/pjpotter/adocs/solve/Tischler: \
? /software/control/adocs/bin/solve -matlab adocs


                    Welcome to CONSOL-OPTCAD (TM)
                  SOLVE Version 1.7 (Released 8/92)

      Copyright (c) 1991, University of Maryland at College Park.
                          All Rights Reserved.
            (developed by  Michael K.H. Fan,  Andre L. Tits,
             Jian L. Zhou,  Li-Sheng Wang and  Jan Koninckx)

[loading/reading adocs.o and ...]
[reading adocs.d]
[calling simulator initialization (if any)]
[connecting to MATLAB engine ...]
[including file init.m ...]
[calling problem initialization (if any)]

type "help" for help
type "help info" for information

<0> print

Name         Value         Variation  wrt 0   Prev    Iter=0
Ktet         9.08983e+00   1.0e+00
Kphi         9.55674e+00   1.0e+00
Kpsi         7.35924e+00   1.0e+00
Kq           7.24229e+00   1.0e+00
Kp           2.12648e+00   1.0e+00
Kr           4.52575e+00   1.0e+00
Mtet         1.32703e+00   1.0e+00
Mphi         5.00000e+00   1.0e+00
Mpsi         8.13820e-01   1.0e+00
Kr_p         2.21514e-01   1.0e+00
Kp_r        -3.83921e-01   1.0e+00
Kc_y        -7.57832e-02   1.0e+00

<0> set Mphi = 4.5
<0> setgb C11 =0,-1.00e+05
<0> setgb C12 =0,-1.00e+05
<0> setgb C13 =0,-1.00e+05
<0> setgb C14 =0,-1.00e+05
<0> setgb C15 =0,-1.00e+05
<0> setgb C16 =0,-1.00e+05
<0> setgb C17 =0,-1.00e+05
<0> setgb C18 =0,-1.00e+05
<0> setgb C19 =0,-1.00e+05
<0> run 0
<0> pcomb

Pcomb (Iter=  0) (Phase 2) (MAX_COST_SOFT= 0.550734)

SPECIFICATION    PRESENT   GOOD              G          B            BAD
01   p_act_rate -9.82e+01  0.00e+00 <==      |          |            1.00e+00
```

```
O2   r_act_rate -7.49e+01  0.00e+00 <==       |          |             1.00e+00
O3   y_act_rate -8.41e+01  0.00e+00 <==       |          |             1.00e+00
C1   stable all -3.21e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.79e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.57e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  3.22e-02  0.00e+00          *========================= -1.00e+05
C12  pit quick2  2.86e-01  0.00e+00          *========================= -1.00e+05
C13  pit quick3  3.57e-01  0.00e+00          *========================= -1.00e+05
C14  rol quick1  1.07e-01  0.00e+00          *========================= -1.00e+05
C15  rol quick2  3.27e-01  0.00e+00          *========================= -1.00e+05
C16  rol quick3 -3.86e-02  0.00e+00          *========================= -1.00e+05
C17  yaw quick1  4.44e-03  0.00e+00          *========================= -1.00e+05
C18  yaw quick2  3.91e-01  0.00e+00          *========================= -1.00e+05
C19  yaw quick3  5.10e-01  0.00e+00          *========================= -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.45e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  8.64e-02  0.00e+00 <================================= -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.21e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -3.92e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.61e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -1.03e+00  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.97e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  2.47e-01  0.00e+00 <================================= -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.30e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  2.99e-01  0.00e+00 <================================= -3.00e-02

<0> freeze Ktet Kphi Kpsi Kq Kp Kr Mtet Mpsi Kr_p Kp_r Kc_y
<0> print

Name        Value        Variation  wrt 0   Prev   Iter=0
Ktet        9.08983e+00  1.0e+00                   frozen
Kphi        9.55674e+00  1.0e+00                   frozen
Kpsi        7.35924e+00  1.0e+00                   frozen
Kq          7.24229e+00  1.0e+00                   frozen
Kp          2.12648e+00  1.0e+00                   frozen
Kr          4.52575e+00  1.0e+00                   frozen
Mtet        1.32703e+00  1.0e+00                   frozen
Mphi        4.50000e+00  1.0e+00
Mpsi        8.13820e-01  1.0e+00                   frozen
Kr_p        2.21514e-01  1.0e+00                   frozen
Kp_r       -3.83921e-01  1.0e+00                   frozen
Kc_y       -7.57832e-02  1.0e+00                   frozen

<0> run 1

<1> pcomb

Pcomb (Iter=  1) (Phase 2) (MAX_COST_SOFT= 0.000109968)

SPECIFICATION    PRESENT   GOOD              G          B             BAD
```

```
O1   p_act_rate -9.82e+01  0.00e+00 <==       |          |             1.00e+00
O2   r_act_rate -7.28e+01  0.00e+00 <==       |          |             1.00e+00
O3   y_act_rate -8.41e+01  0.00e+00 <==       |          |             1.00e+00
C1   stable all -3.21e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.81e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.57e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  3.22e-02  0.00e+00          *========================= -1.00e+05
C12  pit quick2  2.86e-01  0.00e+00          *========================= -1.00e+05
C13  pit quick3  3.57e-01  0.00e+00          *========================= -1.00e+05
C14  rol quick1  1.59e-01  0.00e+00          *========================= -1.00e+05
C15  rol quick2  3.97e-01  0.00e+00          *========================= -1.00e+05
C16  rol quick3 -7.70e-06  0.00e+00          *========================= -1.00e+05
C17  yaw quick1  4.44e-03  0.00e+00          *========================= -1.00e+05
C18  yaw quick2  3.91e-01  0.00e+00          *========================= -1.00e+05
C19  yaw quick3  5.10e-01  0.00e+00          *========================= -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.43e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  8.64e-02  0.00e+00 <================================= -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.21e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -3.92e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.61e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -1.03e+00  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.97e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  2.47e-01  0.00e+00 <================================= -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.30e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  2.99e-01  0.00e+00 <================================= -3.00e-02

<1> run 1

<2> print

Name        Value        Variation  wrt 0   Prev   Iter=2
Ktet        9.08983e+00  1.0e+00       0%     0%   frozen
Kphi        9.55674e+00  1.0e+00       0%     0%   frozen
Kpsi        7.35924e+00  1.0e+00       0%     0%   frozen
Kq          7.24229e+00  1.0e+00       0%     0%   frozen
Kp          2.12648e+00  1.0e+00       0%     0%   frozen
Kr          4.52575e+00  1.0e+00       0%     0%   frozen
Mtet        1.32703e+00  1.0e+00       0%     0%   frozen
Mphi        4.77189e+00  1.0e+00       6%     0%
Mpsi        8.13820e-01  1.0e+00       0%     0%   frozen
Kr_p        2.21514e-01  1.0e+00       0%     0%   frozen
Kp_r       -3.83921e-01  1.0e+00       0%     0%   frozen
Kc_y       -7.57832e-02  1.0e+00       0%     0%   frozen

<2> pcomb

Pcomb (Iter=  2) (Phase 2) (MAX_COST_SOFT= 3.13266e-09)
```

```
SPECIFICATION    PRESENT   GOOD              G          B             BAD
O1   p_act_rate -9.82e+01  0.00e+00 <==       |          |             1.00e+00
O2   r_act_rate -7.28e+01  0.00e+00 <==       |          |             1.00e+00
O3   y_act_rate -8.41e+01  0.00e+00 <==       |          |             1.00e+00
C1   stable all -3.21e-02  0.00e+00 <--       |          |             1.00e-03
C2   pit bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C3   pit Td     -2.03e-01  0.00e+00 <--       |          |             1.00e-04
C4   rol bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C5   rol Td     -2.81e-01  0.00e+00 <--       |          |             1.00e-04
C6   yaw bw pd   0.00e+00  0.00e+00 ==========*          |             2.00e-03
C7   yaw Td     -3.57e-01  0.00e+00 <--       |          |             1.00e-04
C8   pit damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C9   rol damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C10  yaw damp    0.00e+00  0.00e+00 ==========*          |             1.00e-03
C11  pit quick1  3.22e-02  0.00e+00          *========================= -1.00e+05
C12  pit quick2  2.86e-01  0.00e+00          *========================= -1.00e+05
C13  pit quick3  3.57e-01  0.00e+00          *========================= -1.00e+05
C14  rol quick1  1.59e-01  0.00e+00          *========================= -1.00e+05
C15  rol quick2  3.97e-01  0.00e+00          *========================= -1.00e+05
C16  rol quick3 -2.19e-10  0.00e+00          *========================= -1.00e+05
C17  yaw quick1  4.44e-03  0.00e+00          *========================= -1.00e+05
C18  yaw quick2  3.91e-01  0.00e+00          *========================= -1.00e+05
C19  yaw quick3  5.10e-01  0.00e+00          *========================= -1.00e+05
C20  yaw dec2 d  0.00e+00  0.00e+00 ==========*          |             2.00e-03
FC1  pit dec up -1.43e-01  0.00e+00 <==       |          |             5.00e-02
FC2  pit dec lo  2.50e-01  0.00e+00 <================================= -5.00e-02
FC3  rol dec up -2.50e-01  0.00e+00 <==       |          |             5.00e-02
FC4  rol dec lo  8.64e-02  0.00e+00 <================================= -5.00e-02
FC5  yaw dec1 u -4.26e+00  0.00e+00 <==       |          |             5.00e-01
FC6  yaw dec1 l  4.88e+00  0.00e+00 <================================= -5.00e-01
FC7  p gust p u -2.21e+00  0.00e+00 <==       |          |             3.00e-01
FC8  p gust p l  4.00e+00  0.00e+00 <================================= -3.00e-01
FC9  p gust t u -3.92e-01  0.00e+00 <==       |          |             3.00e-02
FC10 p gust t l  2.61e-01  0.00e+00 <================================= -3.00e-02
FC11 r gust p u -1.03e+00  0.00e+00 <==       |          |             3.00e-01
FC12 r gust p l  3.90e+00  0.00e+00 <================================= -3.00e-01
FC13 r gust t u -4.97e-01  0.00e+00 <==       |          |             3.00e-02
FC14 r gust t l  2.47e-01  0.00e+00 <================================= -3.00e-02
FC15 y gust p u -3.09e+00  0.00e+00 <==       |          |             3.00e-01
FC16 y gust p l  3.97e+00  0.00e+00 <================================= -3.00e-01
FC17 y gust t u -4.30e-01  0.00e+00 <==       |          |             3.00e-02
FC18 y gust t l  2.99e-01  0.00e+00 <================================= -3.00e-02

% OPTIMAL SOLUTION (note that we did not wait for the C-O
% CONGRATULATIONS message)

<2> sim
Enter MATLAB, type 'back' to leave
>> specs
>> prspecs
>> print nomoptnew.ps
>> back
back to SOLVE
<2> store "adocs.dp.nomoptnew"
<2> quit

% The optimal dp's were stored in adocs.dp.nomoptnew before we quit
```