DESIGN OF FLIGHT CONTROL LAWS FOR AGILE AND HIGHLY SWEPT AIRCRAFT CONFIGURATIONS

Kuchar, R.; Steinhauser, R.; Looye, G.
German Aerospace Center (DLR), Institute of System Dynamics and Control (SR)
Oberpfaffenhofen, D-82234 Wessling, Germany

Abstract

In the course of the FaUSST project scope, a versatile and flexible method for the synthesis of flight control laws (FCL) for agile and highly swept aircraft configurations was required in preparation for upcoming closed-loop assessments of these configurations within the succeeding MEPHISTO project. Therefore methods and tools were required to allow for the integration within the commonly applied, distributed workflow-driven integration environment tool RCE, project specific interfaces and defined aircraft datasets (such as aircraft geometry, aerodynamics, mass and propulsion) together with control synthesis parameters (tuners) as defined within the "Common Parametric Aircraft Configuration Scheme" (CPACS). The resulting control law design philosophy is characterized by the use of model based control strategies – namely the Nonlinear Dynamic Inversion (NDI) method for the inner control loops. This allows the respective outer control loops to be set up as a linear controller structure, as the non-linear control path (the actual aircraft) can be treated as a continuously linearized system – widely independent from the actual aircraft configuration. In addition to this behavior, the described approach allows the partial use of existent methods and tools, currently in use for the generation of open-loop 3-DOF and 6-DOF flight dynamics models – based on the Modelica physical equation language for the actual implementation of the flight dynamics model equations. Thus the respective inner control loops can be automatically derived (auto-coded) by the inversion of the overall model equation system. The outer control loop can then be optimized for a multitude of target functions and tuners as requested by closed-loop flight dynamics assessment capabilities within the project scope. This is achieved by using the DLR-SR optimization tool MOPS (Multi-Objective Parameter Synthesis) being controlled by externally applied control synthesis parameters within CPACS. In the context of the FaUSST project, a "Proof-of-Concept" setup – regarding the described inner/outer control law structure and methods – has been implemented. Ongoing tasks within the MEPHISTO project scope are the further investigation, overall automation and enhancement of the used methods and derived control laws in conjunction with the respective assessment applications.

1. INTRODUCTION

Within the German Aerospace Center (DLR), a growing number of projects focus on the utilization of integrated aircraft design toolchains – as a collaborative venture of relevant disciplines and DLR institutes. A vital part within this context is the capability to perform overall assessments of flight performance, dynamic behavior and handling qualities – thus requiring the generation of detailed flight dynamics models – based on the respective configuration under design. Furthermore, the generation of respective flight control laws is of great interest in order to perform benchmark studies with a predefined ("reference") behavior.

Since these described efforts are of major relevance for the success of these addressed design and analysis projects and the overall toolchain capabilities – DLR-SR focusses on the implementation of tools for flight dynamic model and control law synthesis, utilizing the existent CPACS datasets (the "Common Parametric Aircraft Configuration Schema", an xml based format, see [7] for more information) in a collaborative tool environment (RCE, see [8]) for these purposes.

Within the context of the FaUSST project – as well as with the succeeding MEPHISTO project, the design and analysis of agile and highly swept configurations is the major focus (e.g. configurations similar to the DLR F-19 model, see FIG. 1). The described methods and processes within this paper therefore specifically focus on the model generation and flight control law synthesis for such configurations – although the underlying methods are as well capable to be utilized in any other design scenario.

Therefore the following paper contains a description of the implemented methods and algorithms for the model generation and specifically the control law synthesis, interfaces and implementation. Furthermore the application of the DLR tool MOPS ("Multi Objective Parameter Synthesis", see [4]) within the described context is presented in order to allow the optimization of the outer control loops. Finally the overall integration and fully automated execution of these tools within the DLR framework tool RCE is described.

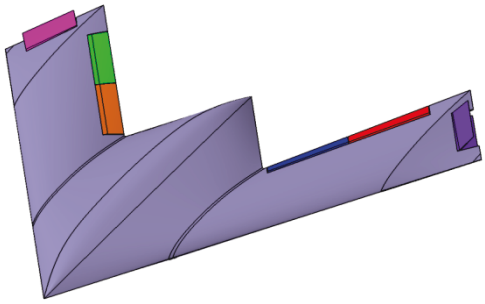

FIG. 1 The DLR F-19 configuration as used for the initial analysis

2. FLIGHT DYNAMICS MODEL GENERATION USING THE DLR MODELICA FLIGHT DYNAMICS LIBRARY

Based on a pre-calculated set of respective aircraft data within the CPACS database, a complete set of flight dynamics models (3-DOF and 6-DOF) can be obtained by utilizing the DLR Modelica Flight Dynamics Library [2] in collaboration with the Dymola modeling and simulation environment [9]. Modelica itself is an object-oriented, equation based, multi-domain physical modeling language for the component oriented modeling of complex systems.

The Flight Dynamics library framework offers a multitude of advantages regarding model specifications for various applications: 3-DOF models for mission simulation, 6-DOF models for full-featured flight simulation and finally automatic generation of inverted 6-DOF models for flight control law synthesis. Especially these inverted models (based on the Nonlinear Dynamic Inversion concept) can be used as a core element in the implementation of a Rapid Control Prototyping (RCP, see [3]) process as proposed in this paper.

Utility methods regarding the integration of external databases into the frame of the Flight Dynamics library are existent and utilized within the described project context – see FIG. 2 for an impression.

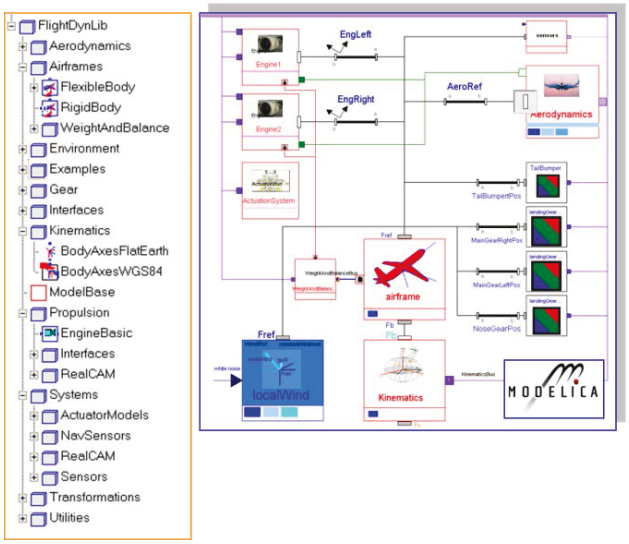

FIG. 2 Structure of the Modelica based DLR Flight Dynamics Library

Once a Modelica model has been defined within the Dassault Dymola modeling and simulation environment, the underlying mathematical software core can assemble and solve the defined model equations (ODE, PDEs) with the aid of dedicated analytical and numerical methods.

The complete flight dynamics model then resembles the solved Newton-Euler (rigid and flexible body) equations of motion based on the non-linear behavior of aerodynamics, propulsion and actuation. This process results in fairly optimized model code that is exported into ANSI ("American National Standards Institute") C-code by the respective model environment.

In order to fulfill project requirements in terms of a generic approach, the model resembles a multitude of interface possibilities – harmonized with the respective database content within CPACS – see FIG. 3 for an impression.

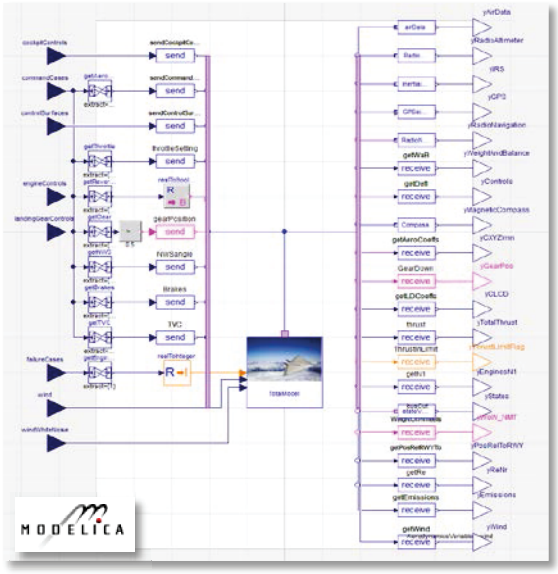

FIG. 3 Top level structure of the generic 6-DOF flight dynamics model

Subcomponents are integrated in a data driven manner: The aerodynamic database for example is directly extracted from CPACS – where it has been inserted by prior process steps – into an ANSI-C based interpolation scheme, which successively integrates natively into the Modelica code. Thus it is required to export each configuration model, as soon as all preprocessing steps have been accomplished.

The resulting ANSI C-code is exported with a standardized interface – the "Functional Mockup Interface" (FMI). This interface can be used in order to setup code for a multitude of applications – namely as MathWorks Simulink S-Function. See section 4 and [9] for more information on the setup and use of FMI units.

A complete discussion of the DLR Flight Dynamics library and its applications is available in [2] and [3].

3. FLIGHT CONTROL LAW SYNTHESIS

This section is dedicated to a description of the respective methods and steps to be taken for the synthesis of flight control laws in the described project scope – commonly referred to as "Rapid Prototyping of Flight Control Laws" – see [3].

In general the overall control architecture consists of three core elements: A reference model for prescribed and deterministic model response, inner control loops (model based NDI control structure) and outer control loops (interference suppression, PI control).

3.1. Inner Control Loops - Model based Flight Control Laws (NDI)

Based on the previously discussed 6-DOF nonlinear flight dynamics model it is possible to derive so-called “inverted models” – thus utilizing the given relationship between actuator input and system reaction in order to permute the equations of motion towards the direct relationship of a given system reaction and the related actuator input. This approach is formalized within the “Nonlinear Dynamic Inversion” approach in control law theory [1] – thus allowing the encapsulation of the nonlinear and configuration specific part of the control law within a purely model dependent assembly. Hence the use of standard control architectures (e.g. PI control) for the outer control loops can be considered throughout the entire modelled flight envelope – see section 3.2.

The following figure shows an exemplary control law structure containing the respective NDI inner loops.

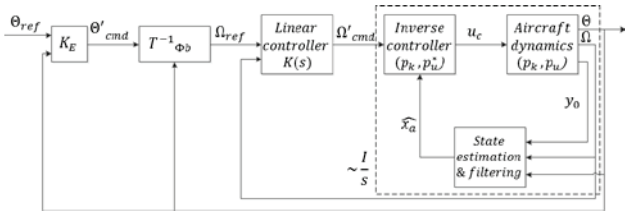

FIG. 4 Control structure, containing NDI based inner control loop – e.g. as attitude control setup

Mathematically, dynamic inversion can be derived by the known relation of earth fixed angular rates Ω_b, the kinematic transformation matrix $T_{\phi b}$ (3x3, conversion from earth fixed intoto aircraft body fixed frame) and attitude rates $\dot{\Theta}$:

$$\dot{\Theta} = T_{\phi b}(\Theta)\Omega_b \tag{1}$$

This equation can be differentiated once more:

$$\ddot{\Theta} = \dot{T}_{\phi b}(\Theta)\Omega_b + T_{\phi b}(\Theta)\dot{\Omega}_b \tag{2}$$

Given the direct dependency of $\dot{\Omega}_b$, states x and input u, dynamic inversion can be described as:

$$\dot{\boldsymbol{\Omega}}_{\boldsymbol{b}} = \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) \rightarrow \boldsymbol{u} = \boldsymbol{f}^{-1}\left(\boldsymbol{x}, \dot{\boldsymbol{\Omega}}_{\boldsymbol{bcmd}}\right) \tag{3}$$

This relationship can be used in order to adapt the Newton-Euler equation of motion in order to resemble this dependency under ideal conditions:

$$\begin{aligned} u_c = M_{A_u}(\widehat{x_a}, p_k, p_u{}^*)^{-1}\big[& I(p_k, p_u{}^*)\Omega'_{bcmd} \\ & - M_{A_x}(\widehat{x_a}, p_k, p_u{}^*) \\ & - M_T(\widehat{x_a}, u_0, p_k, p_u{}^*) - \widehat{\Omega_b} \\ & \times I(p_k, p_u{}^*)\widehat{\Omega_b}\big] \end{aligned} \tag{4}$$

Fortunately, this correlation can be implemented with the direct alteration of the previously generated 6-DOF flight dynamics model within the Modelica equation scheme. As the Dymola modeling environment can resolve this equation system – the actual inner loop controller can be derived simultaneously with the respective flight dynamics model.

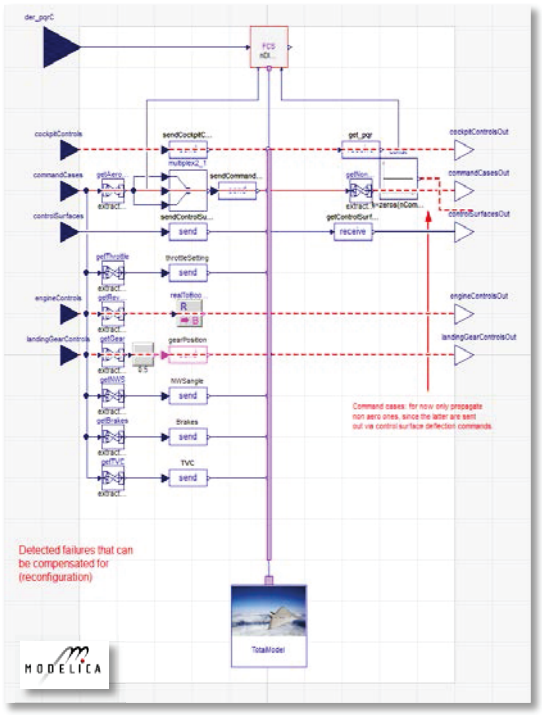

FIG. 5 Inner Control Loop (generic NDI block) – with enforced inversion structure, note the similar 6-DOF model core at the bottom

Compare FIG. 5 with FIG. 3 for actual differences on the model top-level – describing the respective adaptations for the enforced model inversion.

Likewise the previously discussed 6-DOF flight dynamics model, the NDI control block can be exported into ANSI-C code complying the “Functional Mockup Interface” (FMI) standard [9]. Therefore it is possible to use this code as a MathWorks Simulink S-Function within a respective simulation context. See section 4.1 for a description of the combined simulation model setup.

3.2. Outer Control Loops – Structure

On the contrary to the inner control loops, the outer control loops focus exclusively on the suppression of external influences onto the aircraft.

As the inner control loops (NDI based) are describing an entirely linearized system from an external perspective – it is feasible to consider the application of PI control structures for the setup of the outer control loops.

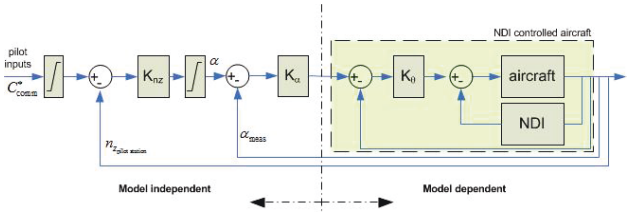

FIG. 6 Outer control loops structure – linear PI controls with envelope protections – e.g. for C* (load factor) functionality, K represents respective controller gains

The setup of FIG. 6 shows an exemplary setup – as currently under development within the FaUSST/MEPHISTO setup – characterizing a C*-based architecture, using the following relations:

$$C^*(t) = \frac{V}{g}\dot{\theta}(t) + n_{z_{pilot}}(t) \qquad (5)$$

Therefore the C* command can be considered as a speed scaled modification of a direct load factor driven command, commonly used for the control of agile flight vehicles.

All outer loop controller modules resemble PI-control structures, which can be basically characterized with the following transfer function $G(s)$:

$$G(s) = K_P \cdot \frac{T_N \cdot s + 1}{T_N \cdot s} \qquad (6)$$

Therein K_P represents the proportional gain, whereas T_N represents the respective time constant of the system. The proportional gain represents one of the tuners that can be optimized for various conditions – see section 3.4.

In the same manner as for C*, also the underlying control laws for load factor and angle-of-attack can be utilized and extracted in order to work as "stand-alone" control laws for various applications.

In order to take the restricted operational range of the aircraft configuration into consideration, "envelope protections" have been introduced within the MEPHISTO project scope. These allow the control of the aircraft to be primarily within a defined operational space – evading potentially dangerous flight conditions. This is an important factor, especially for unconventional configurations and potentially unavailable data in the early stages of design.

3.3. Integrated Reference model

Based on general control law design rules, the overall control law setup follows two basic principles – the independent treatment of a predefined behavior and tracking functionality versus the suppression of external disturbances.

The implemented approach follows therefore the known "Model Reference Control" (MRC) principles – thus providing the ability to enforce the complete setup (model and controller) to precisely follow the prescribed reference characteristics within physical limitations and with minimized tracking error. These characteristics can for example be derived by the analysis of A/C certification rules, operational requirements or respective handling quality constraints.

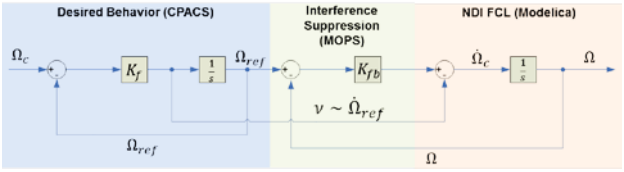

FIG. 7 Combined Setup of Reference model, Outer and Inner Control Loops – e.g. attitude control law

The availability of the reference model can be used for a detailed investigation of the overall dynamic properties of different aircraft designs, as similar ("benchmarking") system reactions are enforced. This feature is especially valuable within a design iteration context, as it allows comparable analysis results: E.g. commanded system reaction vs. actuator activity vs. energy consumption, structural loads, etc.

3.4. Optimization of the Outer Loops using MOPS

As outlined in the previous section, the design of the outer loops offers various opportunities for the adaptation of the given flight control law in order to achieve a given design characteristic. Therefore these parameters are so-called "tuners", which can be iterated and optimized within a respective optimization setup.

In order to optimize the respective dynamics of the outer control loops, the tool MOPS [5] can be used. The tunable content within the controller setup can be identified (e.g. control gains, parameters) and further criteria can be selected respectively. Finally one of the multiple available optimization strategies can be selected before the actual optimization is started – see FIG. 8 for an impression of the MOPS setup GUI which allows the manual setting of the optimization parameters.

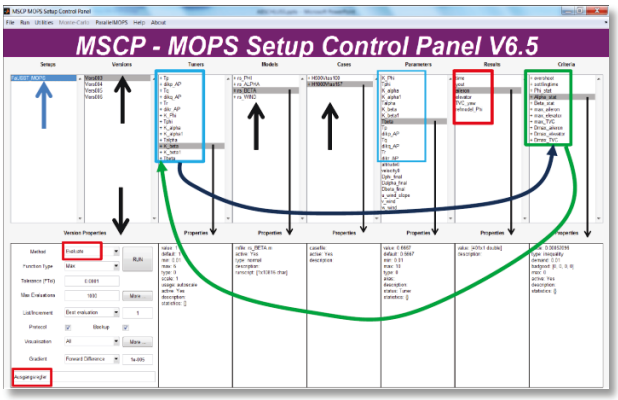

FIG. 8 MOPS Setup GUI – allows the selection of embedded tuners, conditions and parameters

In the scope of the FaUSST project a first manual MOPS optimization setup has been derived. It is planned to extend this capability within the succeeding MEPHISTO project towards a fully automated process – requiring an automated scripted infrastructure that is attached to CPACS database and the RCE runtime environment.

The actual automated setup for the overall synthesis of Flight Control Laws is controlled within a dedicated section of CPACS. This section contains entries for the principal selection of a flight control law type, tuner settings and control design criteria – e.g. the limitation of rise time,

overshoot and so forth. See FIG. 9 for an impression on possible control parameters within the CPACS context.

```
<mops>
  <optimisationSettings/>
  <rollControlSetup>
    <controlLawFunctionUID>rollControlWithNormalLaw</controlLawFunctionUID>
    <tuners/>
    <cases>
      <case>
        <name>nominal</name>
        <parameters>
          <altitude0>2000</altitude0>
          <velocity0>120</velocity0>
        </parameters>
        <criteria>
          <overshoot_Phi>
            <type>inequality</type>
            <demand>0.05</demand>
          </overshoot_Phi>
          <settlingtime_Phi/>
          <resp_error_Phi/>
          <max_aileron/>
        </criteria>
      </case>
      <case/>
  </rollControlSetup>
  <pitchControlSetup/>
  <yawControlSetup/>
</mops>
```

FIG. 9 MOPS Setup section in the CPACS database file

During the optimization runs, the actual trends of the various selected tuners can be monitored online – see FIG. 10 for an impression on the ongoing optimization progress.

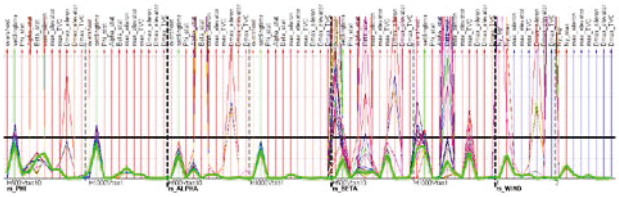

FIG. 10 MOPS Tuner representation at runtime: 71 active (6 minimal conditions, 65 constraints), 11 passive conditions

The following FIG. 11 shows a comparison of derived initial results for different optimization stages within the optimization process. Further results based on the analysis aircraft configuration DLR F-19 can be seen section 5.

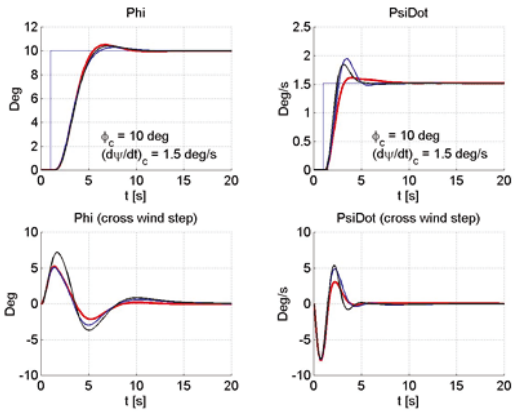

FIG. 11 Exemplary comparison of results in an optimization loop (MOPS)

4. TOOL IMPLEMENTATION AND – INTEGRATION INTO THE OVERALL RCE FRAMEWORK

This section is dedicated to the description of the practical implementation for respective model and controller setups addressing the various purposes within the FaUSST and the succeeding MEPHISTO project scopes.

4.1. Integrated model setup

As described in the previous sections, the actual integration into a common simulation environment of the 3- and 6-DOF flight dynamics models, as well as the generated flight control laws depends heavily on the flexible integration of the Modelica derived ANSI-C code within the Simulink environment. For this purpose the standardized FMI interface has been utilized, providing a flexible, yet robust interface backbone for the automated process [9]. See FIG. 12 for an impression of the implemented software wrapper structure, covering the exported Modelica models. In order to utilize the standard Simulink solvers, the Modelica setup has been exported as "FMI for Model Exchange" version – stripped from any native model solvers.

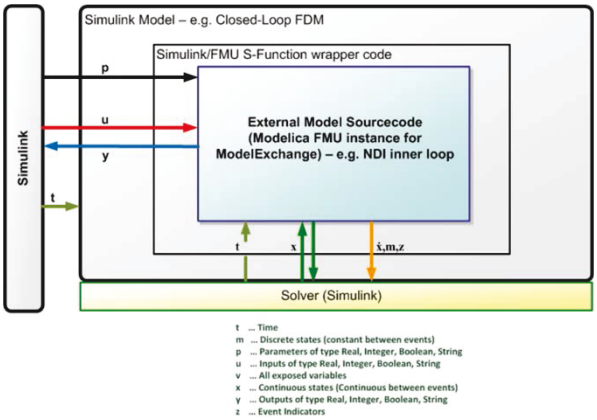

FIG. 12 Modelica FMI based Simulink S-Functions

The implemented FMI S-Function approach allows the use of the derived code in many ways:

- Plain simulation in "Normal" mode within Simulink
- Batch simulations in "Accelerator" and "Rapid Accelerator" modes within Simulink
- The export of the complete setups – e.g. with Simulink Coder or Simulink Real Time Workshop in order to use the derived code within other environments or hardware platforms – namely for the direct integration within distributed simulation environments or on experimental Flight Control computers (e.g. for UAV applications)

Based on these derived S-Functions a complete simulation setup within Simulink can be set up and is used for the respective aircraft configuration analysis. See FIG. 13 for an impression on the assembled Simulink model – as returned to all relevant users within the FaUSST/MEPHISTO project scope via the RCE "Return-ZIP" mechanism.

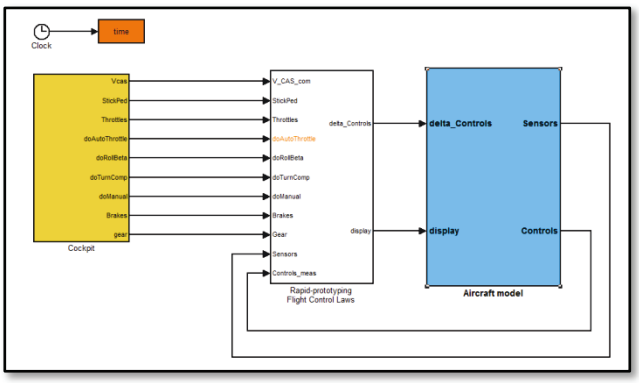

FIG. 13 Complete Simulink setup – containing 6-DOF flight dynamics model and NDI FCL

4.2. Operational steps for the generation of a valid simulation configuration for aircraft analysis

As the respective model generation, flight control law synthesis and mission simulation activities are encapsulated and packaged within the two available RCE workflow tools “flightSim” and “misSim” – the following section is dedicated to a presentation of the actual workflow order and integration within RCE.

Within the FaUSST project scope – and even more within the MEPHISTO scope – the following workflow has been implemented – based on RCE triggered actions on the local “Analysis Server”:

1. Selection of the respective FCL target type in CPACS by the prior process steps (MEPHISTO)
2. Triggering the respective “flightSim” and “misSim” blocks within the RCE environment
3. Generation of 3-DOF and 6-DOF flight dynamics models (FaUSST, MEPHISTO) – based on available CPACS datasets
4. FCL synthesis – based on 6-DOF FDM and triggered by RCE (MEPHISTO)
5. Internal generation of the respective FDM: Modelica model exported as Functional Mockup Unit (FMU, the actual FMI instance) and integrated into top-level Simulink model via FMI S-Function mechanism (FaUSST, MEPHISTO)
6. Internal FCL synthesis: Modelica controller (NDI) exported as Functional Mockup Unit (FMU) and integrated into top-level Simulink model via FMI S-Function mechanism (MEPHISTO)
7. Setup of the integrated simulation: FDM and FCL combined within Simulink environment
8. Tuner setup for MOPS optimization runs in CPACS (MEPHISTO)
9. Automated MOPS optimization runs (MEPHISTO)
10. Export of optimized controller setup within the RCE framework: Return-ZIP containing FDM and optimized FCL setups

See the following FIG. 14 for a top-level impression of the actual RCE workflow.

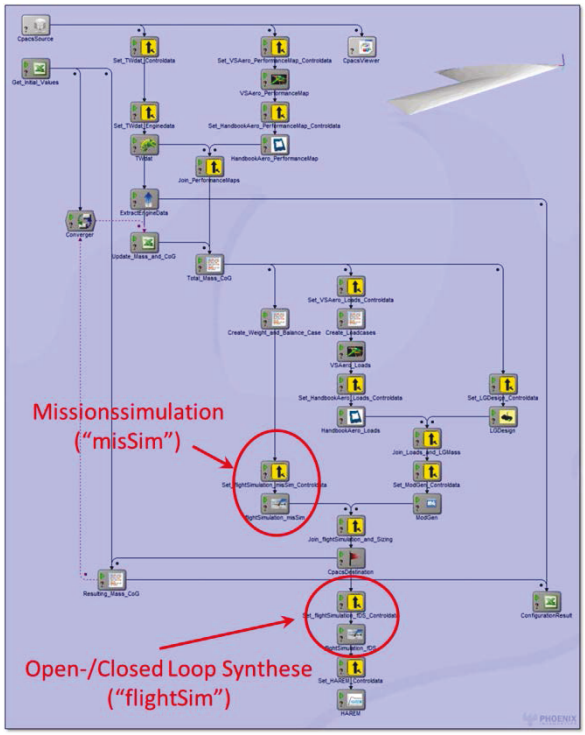

FIG. 14 “flightSim” and “misSim” Tools embedded in the FaUSST RCE/ModelCenter workflow

5. OUTER LOOP OPTIMIZATION RESULTS FOR THE DLR F-19 CONFIGURATION

This section is intended to show some optimization results – derived with an initial MOPS setup within FaUSST for the DLR F-19 configuration. The respective final optimization results are visualized by the solid black lines.

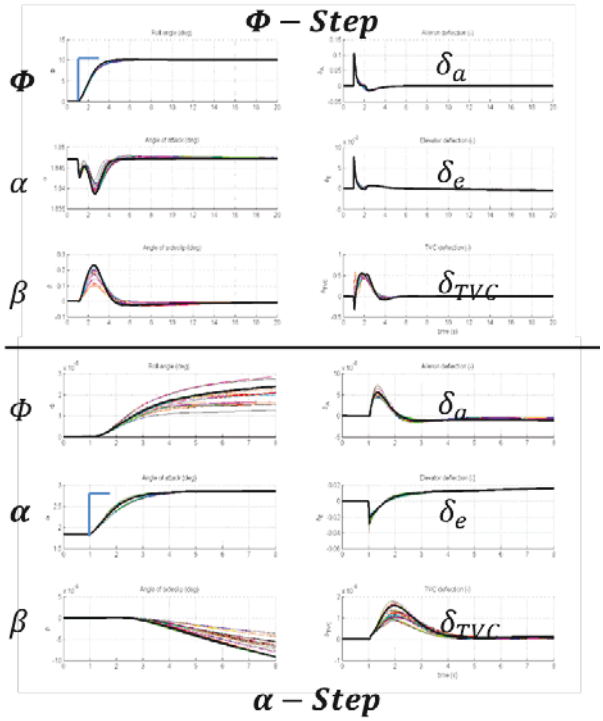

FIG. 15 Roll-Command and AoA step response @ H=1000m and V_{TAS} = 157m/s

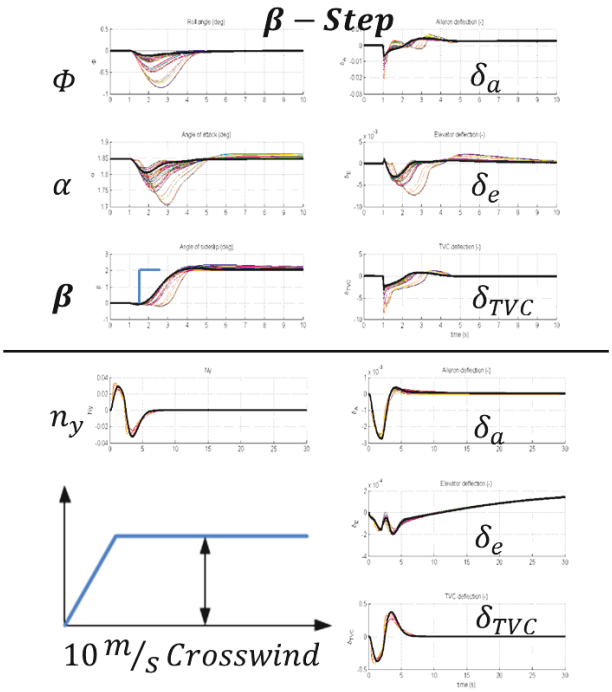

FIG. 16 AoS and Crosswind response @ H=1000m and V_{TAS} = 157m/s

The results of FIG. 15 and FIG. 16 show a conservative and stable behavior throughout a wide range of parameter variations for the given flight conditions. On the contrary, it is rather challenging to derive a robust setup for the following flight conditions with significantly lower flight speeds:

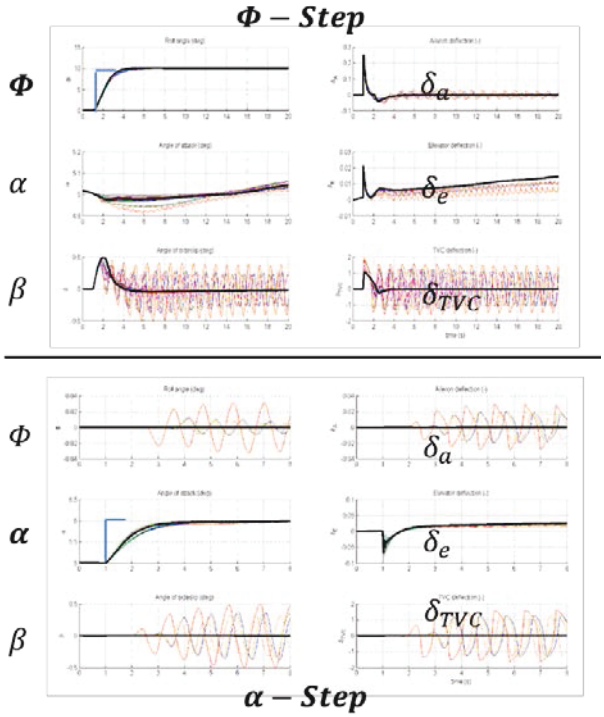

FIG. 17 Roll-Command and AoA step response @ H=500m and V_{TAS} = 100m/s

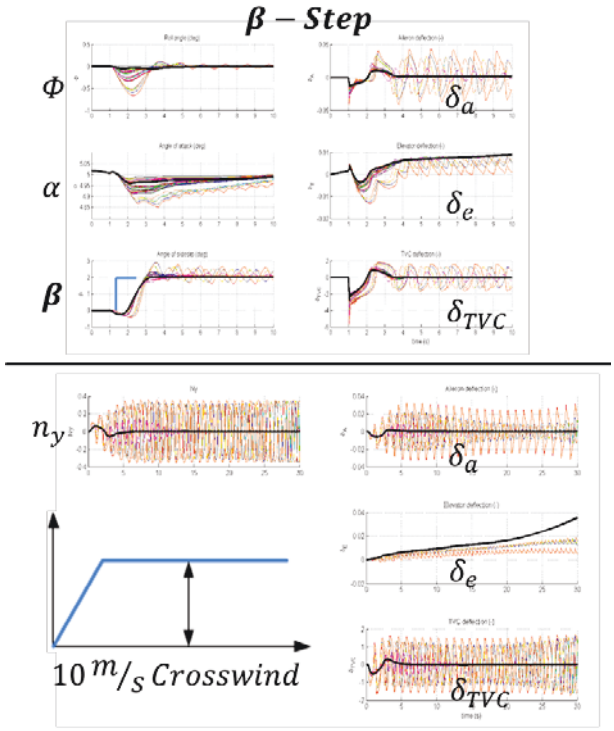

FIG. 18 AoS and Crosswind response @ H=500m and V_{TAS} = 100m/s

In FIG. 17 and FIG. 18 it can be seen, that marginal stability occurs at the given flight conditions. Nevertheless, MOPS is able to find a robust and stable solution for the complete defined flight envelope.

6. SUMMARY AND OUTLOOK

The presented methods for flight dynamics modeling and flight control law synthesis have been designed and implemented with a strong perspective on allowing for a fully automated process – based on Rapid Prototyping methods. Therefore it is possible to automatically generate Open- and Closed Loop models for analysis and simulation purposes within the aircraft pre- and conceptual design phase. Special focus has been given on the synthesis of the respective Inner- and Outer Loop, with a dedicated focus on a strict separation of the model specific (inner) loops and dynamics relevant part (outer loops). In addition to that, the utilization of a reference model has been proposed, in order to allow a specific benchmark comparison between different upcoming aircraft configurations during design iterations.

Finally the actual implementation and specific code relevant features have been discussed, which are responsible for the flexible use of the respective tools within the widely used RCE framework.

In the currently succeeding MEPHISTO project, the described setup is going to be further enhanced – focusing on improved flight control law functionalities (e.g. “dynamic protections”), as well as the implementation of an even deeper level of automation. In the same manner, improvements to the robustness of the derived solutions are planned – including uncertainty modelling within the model generation setup and extensions towards fully flexible “Weight & Balance” handling in order to allow discrete loading and unloading events.

7. REFERENCES

[1] Looye, G.: An Integrated Approach to Aircraft Modeling and Flight Control Law Design. Dissertation, Delft University of Technology, Delft, Netherlands, 2008.

[2] Looye, G.: The new DLR Flight Dynamics Library, Modelica Conference 2008, Germany.

[3] Looye, G.: Rapid prototyping using inversion based control and object-oriented modeling. Chapter 8, Lecture notes in Control and Information Sciences. Springer Verlag, Berlin, 2007.

[4] H.-D. Joos. MOPS Multi-Objective Parameter Synthesis User's Guide V 5.3. Internal report, DLR, 2008.

[5] Joos H.-D., Bals J., Looye G., Schnepper K., Varga A. A multi-objective optimization-based software environment for control systems design. IEEE International Conference on Control Applications and International Symposium on Computer Aided Control Systems Design, Glasgow, Scotland, UK,2002.

[6] A. Rizzi, M. Zhang, B. Nagel, D. Böhnke, P. Saquet: Towards a Unified Framework using CPACS for Geometry Management in Aircraft Design, AIAA Aerospace Sciences Meeting, Nashville, USA, 2012.

[7] CPACS – A Common Language for Aircraft Design and related tools (Open Source): http://code.google.com/p/cpacs/

[8] RCE/Chameleon – distributed, collaborative problem solving environment software (OpenSource): http://code.google.com/a/eclipselabs.org/p/rce/

[9] Otter, M., Blochwitz T., Elmqvist H., Junghanns A., Mauss J., Olsson H., Functional Mockup Interface – Overview, Modelisar, 2010. http://synchronics.inria.fr/lib/exe/fetch.php/modelica-fmi-elmqvist.pdf