Robustness Analysis Applied to Autopilot Design Part 3: Physical Modeling of aircraft for automated LFT generation applied to the Research Civil Aircraft Model

D. Moormann, **A. Varga**, **G. Looye**, and **G. Grübel**

German Aerospace Center, DLR Oberpfaffenhofen, Institute of Robotics and System Dynamics, D-82234 Weßling, Germany, URL: `http://www.op.dlr.de/FF-DR-ER/`, *E-mail:* `Dieter.Moormann@dlr.de`

Abstract

In this contribution the automated generation of LFT-based parametric uncertainty descriptions from a generic nonlinear aircraft-dynamics model, as used for the GARTEUR RCAM Design Challenge on Robust Flight Control, is described. For this purpose an object-oriented, equation-based modeling approach using the modeling environment Dymola was applied. Using this technique allows the modeling of physical systems as physical objects and phenomena, which are connected according to their physical interactions. This modeling in form of equations (not assignments!), as required for automated LFT generation, is different from modeling via signal flows or input-output block diagrams, as traditionally used for controller modeling. All necessary components are taken from an aircraft object library developed for this purpose. Different representations of one component may be present to allow model building of different complexity or various functionalities.
By automatic equation manipulation a symbolic model code is generated from the parameter instantiated equations of each object and from the equations derived from the interconnection structure. This code is the base for an automated generation of the LFT-based parametric uncertainty description.

1. Introduction

Within the GARTEUR[*] Design Challenge on Robust Flight Control (1995-1997) 18 teams from 7 European countries investigated the applicability of modern control concepts for developing robust flight control systems. One of the benchmarks solved there was an autopilot design for the Research Civil Aircraft Model (RCAM), whose data have been provided by Aerospatiale.

A unique simulation model for RCAM has been made available to all design teams either as Matlab/Simulink[12, 13] simulation code or as Fortran/C codes. The different codes describing the same aircraft dynamics model were built automatically from a 'generic' physical aircraft description, using the object oriented modeling and simulation code generation environment Dymola[4, 5] . This procedure guaranteed that groups working with different simulation environments still used the same aircraft model.

In the final phase of the GARTEUR project the applied design methods and their results were evaluated. One part of the evaluation was a μ-analysis to assess stability robustness of all designs. μ-Analysis requires a standard system representation of uncertainties by the so-called *Linear Fractional Transformation* (LFT). The generation of LFT models for parameterized nonlinear dynamics models is a very demanding and time consuming process for large systems like an aircraft.

In this paper we describe an highly automated procedure to generate LFT-based uncertainty descriptions, starting from the Dymola object model. The same model has been employed to build the nonlinear simulation models of RCAM used by the design teams. The LFT description is obtained in several steps. First the Dymola generated symbolic code is converted into Maple[3]-code, which in turn is used to symbolically produce a linear state space model with explicit dependencies on uncertain model parameters. In the final step the Matlab toolbox PUM[15] is used to obtain the LFT uncertainty description of the RCAM model.

[*] GARTEUR = **G**roup of **A**eronautical **R**esearch and **T**echnology in **EUR**ope

The paper is organized as follows. The flight dynamics modeling of RCAM in form of an object diagram is introduced first and two example objects are presented: a 6-degrees-of-freedom aircraft body and a tailplane actuator model. Then, from the symbolic object description of the aircraft model, a nonlinear symbolic simulation model with explicit parametric dependencies is obtained. This model serves to generate an LFT-based linear uncertainty description which is appropriate for the post-design μ-robustness assessment (see Ref[10] in this proceeding).

2. Flight dynamics as an object diagram

The most natural way of modeling physical systems is the modeling of physical *objects* and *phenomena*, which are connected according to their physical energy-flow interaction. This is different from modeling all phenomena as functions and signal flows, as it is common use in modeling for controller design.

An aircraft consists of a variety of different components and systems. These represent the interacting disciplines which are involved in aerospace engineering (e.g. flight mechanics, aerodynamics, propulsion).

One way to describe an aircraft is as follows: An aircraft consists of a **body** (fuselage and wing), which is powered by one or more **engine**s. **Gravity** is acting at an aircraft in the way as it is acting at any object with mass. The **aerodynamics** describe the effects of the air, which is influenced by the surrounding **atmosphere** and additional **wind**s.

Each of these phenomena is most conveniently described as one physical object. All objects which form the aircraft are connected according to Fig. 1.

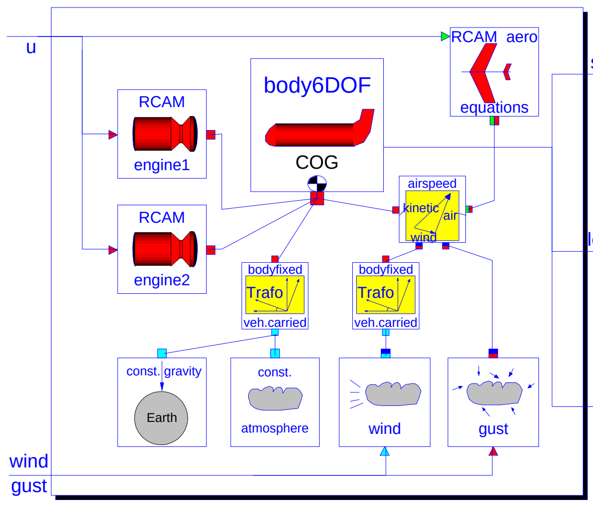


Fig. 1. Object diagram of RCAM

In order to make the understanding of each object easy, each component is described in its own coordinate system. Gravitation, wind, and atmosphere are conveniently described in an earth related coordinate system, aerodynamics in an aerodynamic coordinate system, and engines in a system which is related to the body-fixed coordinate system. Hence coordinate transformations are needed in between, when connecting all those subsystems. Therefore, in addition to basic aircraft components, coordinate transformations are also detailed and handled as objects in the aircraft library, Fig. 2.

In the physical aircraft library different representations of one component can be found. There is a class Body with six degrees of freedom (Body6DOF) and a class with three degrees of freedom (BodyLong), which can be used to generate a nonlinear simulation model for the longitudinal motion only. There are also engine, atmosphere and gravitation models of different complexity.

In a graphical view the interconnection structure of an aircraft can be most easily understood. If a more complex gravity model shall act at the aircraft, this object can simply be taken from the aircraft library and replace the simple gravitation object. In the same way one or more engines can be added or removed from the aircraft or can be modified. This is the most transparent user layer with no need of thinking about the structure of a simulation code.

The objects, which form the physical model, contain equations (and not assignments as common in programming languages or simulation languages). This makes the understanding and the reuse much easier than looking at simulation code, which is put in a form and an order that has to be understood by a computer. The object equations are sorted automatically by a symbolic equation handler rather than by a human.

Objects, formulated in that way do not contain causalities. Depending on its application one object can fulfill different tasks. For example, the object which does the transformations between the bodyfixed and the aerodynamic coordinate system, is used for the transformation of the velocities from the bodyfixed system to the aerodynamic system, as they are required within the aerodynamics. The same object is used for the transformation of the forces and moments from the aerodynamic to the bodyfixed system. When connecting components as objects, only the relation between them is defined and not the order, in which those equations are finally solved.

Components are coupled by drawing a line between the defined 'coupling' points of the objects, which are called 'cuts' in the Dymola notation. These couplings

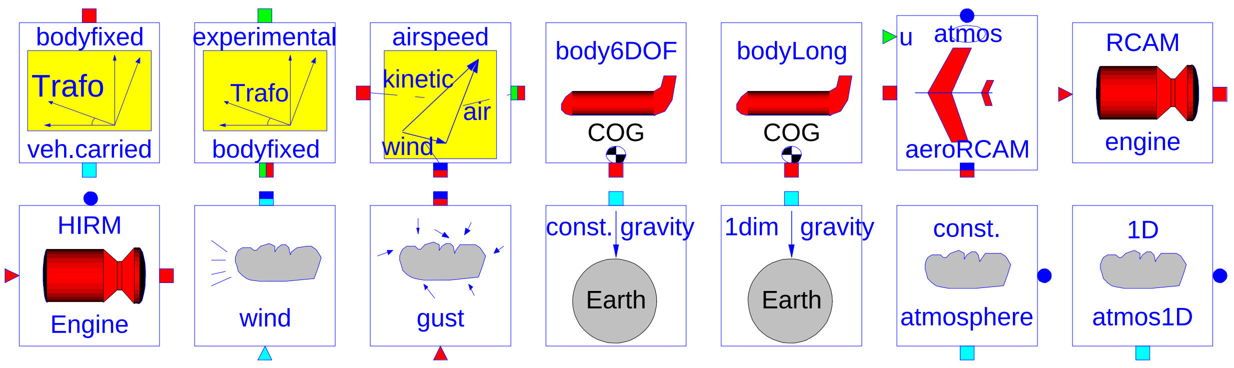


Fig. 2. Flight dynamics library as used for GARTEUR FM(AG08)

represent either energy or signal flow. The cut *bsystem*, e.g., has the following structure:

$$\text{terminal } {}^vT^b[3,3], r[3], v_b[3], w_b[3], a_b[3], z_b[3], F_b[3], M_b[3]$$

$$\text{cut bsystem } ({}^vT^b, r, v_b, w_b, a_b, z_b / F_b, M_b)$$

The first matrix ${}^vT^b$ defines the orientation of the bodyfixed system with respect to the vehicle carried (Earth) system, the vector r is the aircraft's inertial position in the vehicle carried frame. The symbols v_b and a_b are the velocity and acceleration in the bodyfixed frame and F_b and M_b are the Forces and Moments, also formulated in the bodyfixed frame. In the same way there are cuts defined for the vehicle carried system (*vcsystem*) and for the aerodynamic system (*asystem*)

This cut structure represents physical connections. When objects are connected, Dymola adds equations for the cut variables: all quantities before the slash operator ('́) are set equal when connected, as it is reasonable for positions, velocities and accelerations; quantities after the slash operator are summed up to zero, as it is reasonable for Forces and Moments.

This object oriented equation based form of describing physical systems helps to understand the physical system and enables the user to modify the model most conveniently.

An important aspect in object oriented modeling of physical systems is the encapsulation of objects. The internal implementation of details, e.g. of the aerodynamics, are not visible, when viewing the RCAM object model as depicted in Fig. 1. By encapsulation, the implementation of an object can be changed without affecting the functionality of the whole model.

Fig. 3 demonstrates, how the RCAM model is structured. Here only the aerodynamics model is extracted. In the same way details of the engine, gravity, wind, and atmospheric models can be displayed.

Using the graphical interface, 'double clicking' on **body** displays the parameter window of this object. This window allows the parameters to be modified. In the same way, all of the other objects (body, engines) can be instantiated with their parameters.

Two example objects of RCAM model will be detailed in the following sections.

3. Example object: Body6DOF

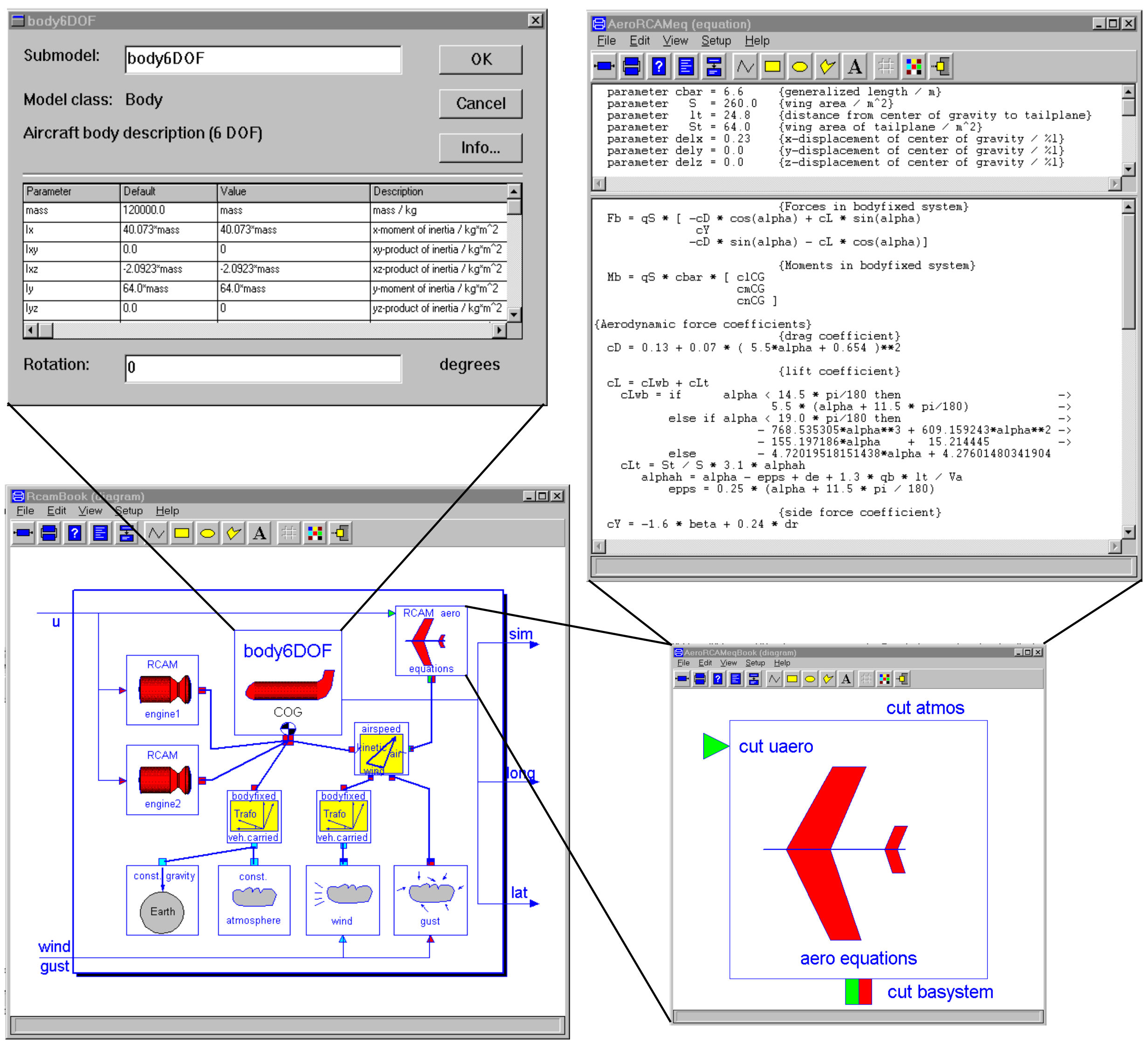
The object **Body6DOF** describes the differential equations of motion for a rigid body with six degrees of freedom and other motion relevant equations. For a more detailed derivation and explanation of these equations a reference such as Ref$^{(2)}$ or Ref$^{(6)}$ should be consulted.

Translational motion

The equations for the translational movement can be given by the force equation:

$$F = m\ (\ a_b\ +\ \omega\ \times\ V_b\ ) \qquad (1)$$

with F as the sum of forces due to the engine, the aerodynamics and gravity, m is the mass of the aircraft, V_b is the airspeed and ω is the rotation rate in bodyfixed coordinates. The acceleration a_b (in the bodyfixed system) is the time derivative of velocity V_b; the velocity $V_v$is the time derivative of the position vector X_v expressed in the vehicle carried (earth) vertical frame:



Submodel: body6DOF

Model class: Body

Aircraft body description (6 DOF)

Parameter	Default	Value	Description
mass	120000.0	mass	mass / kg
Ix	40.073*mass	40.073*mass	x-moment of inertia / kg*m^2
Ixy	0.0	0	xy-product of inertia / kg*m^2
Ixz	-2.0923*mass	-2.0923*mass	xz-product of inertia / kg*m^2
Iy	64.0*mass	64.0*mass	y-moment of inertia / kg*m^2
Iyz	0.0	0	yz-product of inertia / kg*m^2

Rotation: 0 degrees

```
parameter cbar = 6.6     {generalized length / m}
parameter   S  = 260.0   {wing area / m^2}
parameter   lt = 24.8    {distance from center of gravity to tailplane}
parameter   St = 64.0    {wing area of tailplane / m^2}
parameter delx = 0.23    {x-displacement of center of gravity / %l}
parameter dely = 0.0     {y-displacement of center of gravity / %l}
parameter delz = 0.0     {z-displacement of center of gravity / %l}
```

```
                         {Forces in bodyfixed system}
 Fb = qS * [ -cD * cos(alpha) + cL * sin(alpha)
              cY
             -cD * sin(alpha) - cL * cos(alpha)]

                         {Moments in bodyfixed system}
 Mb = qS * cbar * [ clCG
                    cmCG
                    cnCG ]

{Aerodynamic force coefficients}
                         {drag coefficient}
 cD = 0.13 + 0.07 * ( 5.5*alpha + 0.654 )**2

                         {lift coefficient}
 cL = cLwb + cLt
   cLwb = if      alpha < 14.5 * pi/180 then                 ->
                        5.5 * (alpha + 11.5 * pi/180)         ->
          else if alpha < 19.0 * pi/180 then                 ->
                      - 768.535305*alpha**3 + 609.159243*alpha**2 ->
                      - 155.197186*alpha    +  15.214445     ->
          else        - 4.72019518151438*alpha + 4.27601480341904
   cLt = St / S * 3.1 * alphah
      alphah = alpha - epps + de + 1.3 * qb * lt / Va
          epps = 0.25 * (alpha + 11.5 * pi / 180)

                         {side force coefficient}
 cY = -1.6 * beta + 0.24 * dr
```

Fig. 3. Structure of aircraft physical model

$$a_b = \frac{d\,V_b}{dt} \tag{2}$$

$$V_v = \frac{d\,X_v}{dt} \tag{3}$$

Rotational motion

The equations of motion for the rotational movement of a rigid body in the body-fixed axis system from the moment equation,

$$M = I\,\dot{\omega}\; + \;\omega\; \times\; I\,\omega \tag{4}$$

M is the sum of moments about the center of gravity due to engine and aerodynamics, ω is the inertial rotational velocity, and $\dot{\omega}$ is the inertial rotational acceleration in the body-axis system. Using the standard notation (Ref$^{(2,6)}$) we get:

$$\dot{\omega} = \frac{d\,\omega}{dt} \tag{5}$$

Again using the standard notation, the relation between the rotational velocities and the Euler angles ϕ, θ, and ψ is:

$$\frac{\dot{\mathbf{\Phi}}}{dt} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix} \cdot \omega \tag{6}$$

In Dymola-syntax the object **Body6DOF** is given as:

```
model class (TrafoBV) Body6DOF
{parameter declarations}
parameter mass =  120000.0 {mass / kg},
      Ix = 4808400.0 {x-moment of inertia / kg*m^2},
      Ixy =       0.0 {xy-product of inertia / kg*m^2},
      Ixz = -251076.0 {xy-product of inertia / kg*m^2},
       Iy = 7680000.0 {y-moment of inertia / kg*m^2},
      Iyz =       0.0 {yz-product of inertia / kg*m^2},
       Iz =11990400.0 {z-moment of inertia / kg*m^2}

{local variables}
local I[3,3] = [ Ix , -Ixy,  Ixz ;
                -Ixy,  Iy , -Iyz ;
                 Ixz, -Iyz,  Iz  ]

{ translational equations of motion }
   F = mass * ( ab + cross(wb,vb) )

   ab = der(vb)
   vv = der(r)

{ rotational equations of motion }
   M = I * zb + cross(wb,(I*wb))

   zb = der(wb)
   der (Phi) = Mfphi * wb
   Phi = [phi, theta, psi]
   Mfphi = [1,sin(phi)*tan(theta),cos(phi)*tan(theta);
            0,      cos(phi)     ,    -sin(phi)      ;
            0,sin(phi)/cos(theta),cos(phi)/cos(theta)]

{ height }
   h = -r(3)

{ flight path angle: gamma / radian }
   tan (gamma)  = -vv(3) / sqrt ( vv(1:2)' * vv(1:2) )

{ flight path heading angle: chi / radian }
   tan (xhi)   =  vv(2) / vv(1)

{ ground speed / (m/s) }
   vground = sqrt ( vv' * vv )
end
```

Typically for an aircraft body is that part of the translational differential equations are formulated in the body-frame (e.g. body-fixed speed V_b in 2) and another part in the vehicle carried (Earth)-frame (e.g. vehicle carried speed V_v in 3). In order to cope with the speed in those different coordinate systems the object TrafoBV of Fig. 2, which includes **all** transformation equations between them, is inherited to the object *body*.

4. Example object: Tailplane Actuator

Simplified models of actuators are often built from combinations of basic control block components (e.g. Transfer functions, Gains, Summations).

Fig. 4 shows a block-oriented description of the RCAM tailplane as a first order transfer function with the commanded tailplane deflection dt_c as input and the tailplane deflection dt as output. The rate of change of the actuator deflection is limited by 0.262 rad/s (=15 deg/s). A limited integrator block bounds the tailplane deflection between 0.175 rad (= 10 deg) and -0.436 rad (= -25 deg).

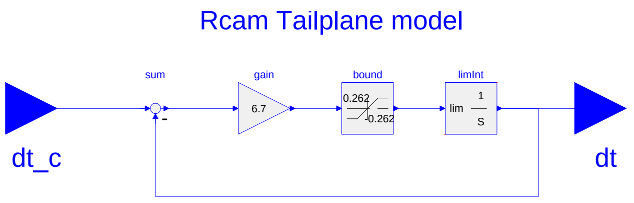


Fig. 4. Block-oriented description of RCAM tailplane

The structural difference between physical modeling and block-oriented modeling is that blockoriented models consist of functions with predefined input-output behavior. Fig. 5 gives some basic components of such a block library.

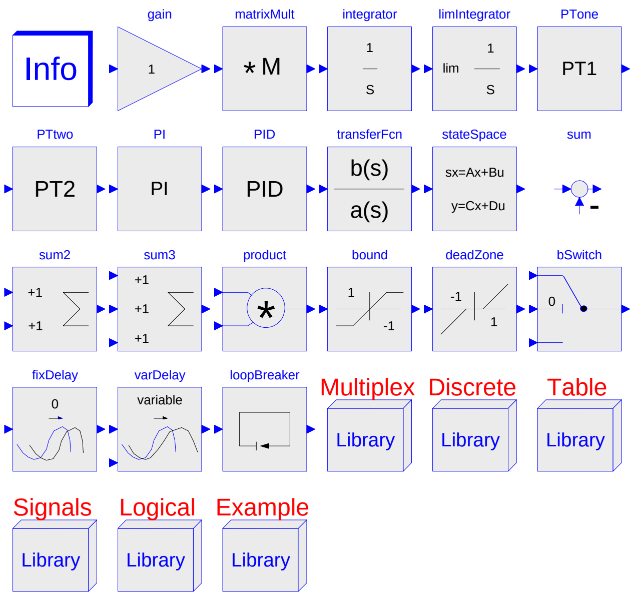


Fig. 5. Block library

When using an object oriented equation based approach both physical and block-oriented modeling methodologies can be used in the same model.

5. The RCAM benchmark

The RCAM formed the basis of the GARTEUR civil benchmark problem. The designers had to cope with a number of uncertain parameters in the model, see table 1.

In this table, m is the mass, X_{cg} and Z_{cg} are the horizontal and vertical center of gravity shifts respec-

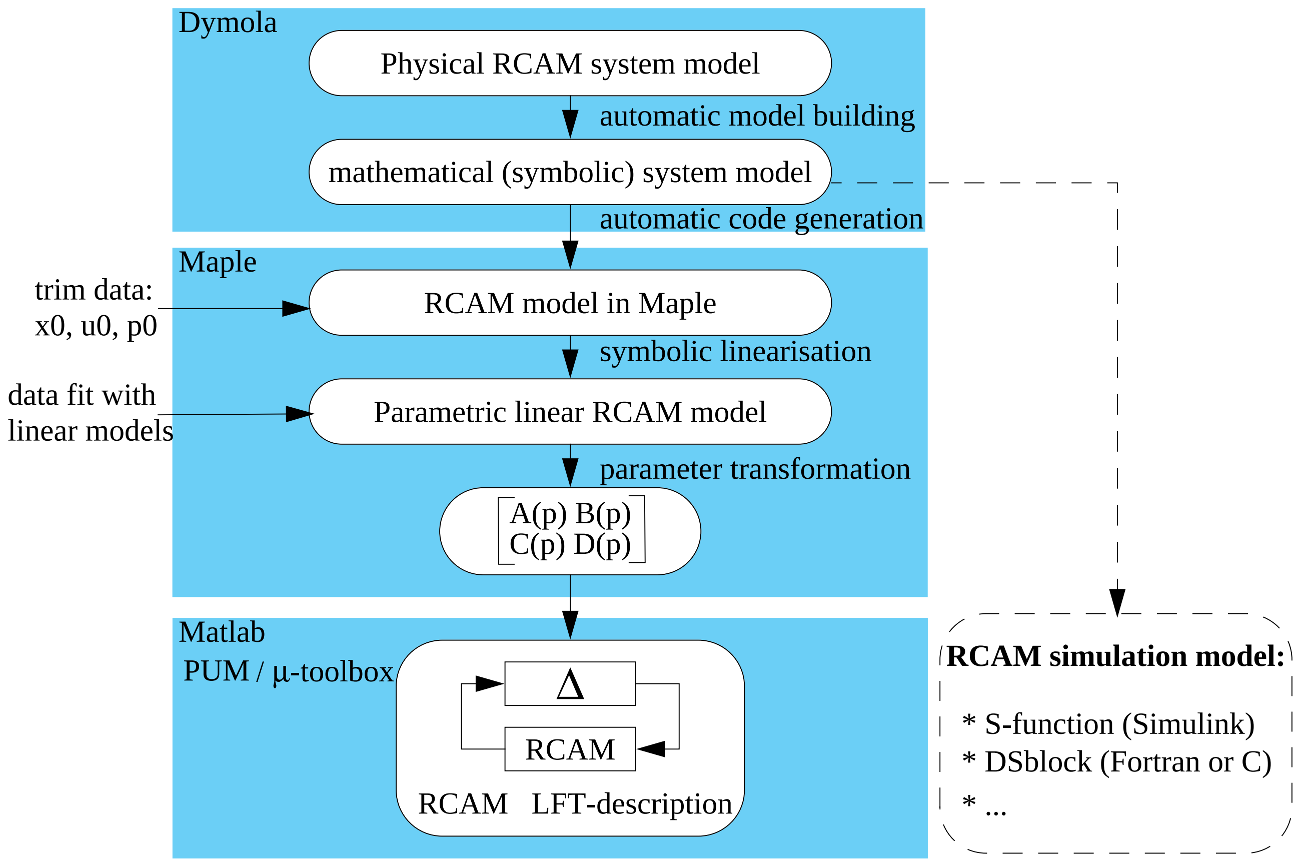


Fig. 6. Generation of LFT description

parameter	unit	min	max	nominal
m	kg	100 000	150 000	120 000
X_{cg}	m	$0.15\bar{c}$	$0.31\bar{c}$	$0.23\bar{c}$
Z_{cg}	m	$0.0\bar{c}$	$0.21\bar{c}$	$0.0\bar{c}$
τ	s	0.05	0.10	0.075
(V_A)	m/s	$1.23\ V_{stall}$	90	80

Table 1. Parameter ranges RCAM

tively, τ is the time delay of the controller, and $\bar{c}$ is the mean aerodynamic chord. Although the designers had to consider airspeed as well, they were allowed to use it as a scheduling parameter. A detailed description of the RCAM benchmark definition can be found in Ref(7).

6. Model building & Code generation

From the model, specified in form of objects as given in Fig. 1 together with the uncertain parameters of Tab. 1 a simulation model of the aircraft is generated automatically. Equations, which are formulated in the object, but not needed for the specified configuration, are automatically removed. The result is a model with a minimum number of equations for this task.

In this way it is possible to generate e.g. Matlab/Simulink Cmex-code, which was used by most design teams within the GARTEUR design challenge, or C-Code or Fortran-Code according to the DSblock standard(14) , which was used within the ANDECS design environment for control engineering(8).

6.1 Generation of LFT-description

In the final phase of the GARTEUR project the applied design methods were evaluated with μ-analysis to assess stability robustness of the designs. μ-Analysis is based on a standard system LFT representation. The process to derive such an LFT description in an automated way from the same object model, which was used to generate the nonlinear simulation models, is described in this chapter. The process is illustrated in Fig. 6.

Starting from the physical aircraft model the equation handler of Dymola is solving the equations according to inputs and outputs of the system. The result is a nonlinear symbolic state space description, with $p = [m,\ X_{cg},\ Z_{cg}]^T$:

$$\begin{aligned} \dot{x} &= f(x,u,p) \\ y &= h(x,u,p) \end{aligned} \qquad (7)$$

In a code generation process this description is transfered to a Maple[3]-syntax. For symbolic linearisation additional trim data is required, which is given by the state vector x_0, the input vector u_0 and the nominal parameter set p_0. This trimdata is obtained using the nonlinear simulation model.

The above detailed approach alone is only valid in a small neighborhood around the linearisation point and therefore not appropriate for obtaining an LFT description, which covers all flight conditions. If the trimming could be performed symbolically by solving the nonlinear equations 7 for x and u, we would obtain these vectors as explicit functions of the parameters p. Unfortunately, for complex systems like aircraft, symbolic trimming is not generally feasible.

For that reason a parameter fitting for the whole flight envelope is performed. Textbooks[2,6] can give useful hints to guess the proper form of the parametric dependencies of the effected state-space elements. A multidimensional data fitting routine from PUM[15] is used for this purpose. The result of this step, the parameter-dependent system matrices is shown in the appendix. A detailed description of the process and the required software and procedures is given in Ref[16].

7. Conclusions

It has been shown that an automated generation of LFT-based parametric uncertainty descriptions is possible for aircraft models. A requirement for this approach is that the model, from which the description is generated, is formulated by equations and not by assignments. The reason for that is, that a symbolic manipulation is later necessary to generate the parameter dependent state space description.

A modeling tool that fulfills these requirements, is the object-oriented equation-based modeling environment Dymola. The basis of Dymola is the clear separation between modeling and code generation. The modeling is done most user-friendly in form of hierarchical structured objects, which are connected according to their physical energy-flow-interaction rather than transferring all phenomena to a certain input/output behavior, as required for block-oriented tool like Simulink.

This modeling approach has the further advantage that an efficient nonlinear simulation code for different simulation and analysis environments can be automatically generated as well. Since both the LFT description and the simulation model, are generated automatically from the same source model, the consistency is guaranteed without any time-consuming verification process.

The further advantage of this modeling technique is the flexibility with respect to uncertain parameters. Within the object diagrams uncertain parameters can be easily chosen and by automatic code generation they will be incorporated automatically in the symbolic state space description. This is especially helpful for parameters, whose influence on the state space description cannot be found in textbooks.

The proposed approach is also of generic value for similar model classes encountered in practice. To problems with non-explicit parameter dependencies, as for example those with parameters defined by table look-up procedures, the proposed approach is applicable provided rational approximations can be found to replace the non-analytical functional dependencies.

8. References

[1] G. J. Balas, J. C. Doyle, K. Glover, A. Packard, and R. Smith. *μ-Analysis and Synthesis TOOLBOX, For Use with MATLAB.* The Math Works Inc., July 1993.

[2] R. Brockhaus. *Flugregelung.* Springer Verlag, Berlin Heidelberg, 1994.

[3] B.W. Char et al. *Maple V Language Reference Manual.* Springer Verlag, 1991.

[4] H. Elmqvist. *Dymola – Dynamic Modeling Language User's Manual.* Dynasim AB, S–22370 Lund, Sweden, 1993.

[5] H. Elmqvist. Object-Oriented Modeling and Automatic Formula Manipulation in Dymola. In *Scandinavian Simulation Society SIMS'93*, Kongsberg, Norway, June 1993.

[6] B. Etkin. *Dynamics of Flight - Stability and Control.* John Wiley & Sons, USA, November 1958.

[7] Flight Mechanics Action Group 08. Robust Flight Control Design Challenge Problem Formulation and Manual: the Research Civil Aircraft Model (RCAM). Technical Report TP-088-03, GARTEUR, April 1997. Version 3.

[8] G. Grübel, D. Joos H, M. Otter, and R. Finsterwalder. The ANDECS Design Environment for Control Engineering. In *12th IFAC World Congress, Sydney, Australia, July 19-23*, volume 6, pages 447–454, 1993.

[9] J. F. Magni and S. Bennani and J. Terlouw. *Robust Flight Control, A Design Challenge.* Springer Verlag, 1997.

[10] G. Looye, A. Varga, S. Bennani, D. Moormann, and G. Grübel. Robustness analysis applied to autopilot design part 1: μ-analysis of design entries to a robust flight control benchmark. In *Proceedings of the 21st ICAS Congress*, Melbourne, Australia, September 1998.

[11] G. Looye, A. Varga, D. Moormann, and S. Bennani. Post-design stability robustness assessment of the rcam controller design entries. Technical Report TP-088-35, GARTEUR, April 1997.

(12) The Math Works Inc. *Simulink User's Guide*, 1992.
(13) The Math Works Inc. *MATLAB Reference Guide*, 1993.
(14) M. Otter. DSblock: A neutral description of dynamic systems. Version 3.2. In *Technical Report TR R81-92, DLR Oberpfaffenhofen, Institute f"ur Robotik und Systemdynamik, D-82234 Wessling, Germany*, May 1992.
(15) J.C. Terlouw and P.F. Lambrechts. A MATLAB Toolbox for Parametric Uncertainty Modelling. Technical Report CR 93455 L, National Aerospace Laboratory, NLR, Amsterdam, 1993.
(16) A. Varga, G. Looye, D. Moormann, and G. Grübel. Automated Generation of LFT-Based Parametric Uncertainty Descriptions from Generic Aircraft Models. Technical Report TP-088-36, GARTEUR, April 1997.

9. Appendix

The nomenclature used for the model description and detailed information on RCAM can be found in Ref.[9] The following table gives the RCAM state-vector as used for the symbolic description of the parameter dependent A system matrix on the following page.

In this table, 'CoG' denotes 'Centre of Gravity'. F_E denotes the earth-fixed reference frame, F_B denotes the body-fixed reference frame, F_V denotes the vehicle-carried vertical frame, and F_M denotes the measurement reference frame. All state variables of the nonlinear RCAM are expressed in SI units.

Symbol			Name	Unit
p	x(1)	=	roll rate (in F_B)	*rad/s*
q	x(2)	=	pitch rate (in F_B)	*rad/s*
r	x(3)	=	yaw rate (in F_B)	*rad/s*
ϕ	x(4)	=	roll angle (Euler angle)	*rad*
θ	x(5)	=	pitch angle (Euler angle)	*rad*
ψ	x(6)	=	heading angle (Euler angle)	*rad*
u_B	x(7)	=	x component of inertial velocity in F_B	*m/s*
v_B	x(8)	=	y component of inertial velocity in F_B	*m/s*
w_B	x(9)	=	z component of inertial velocity in F_B	*m/s*
x	x(10)	=	x position of aircraft CoG in F_E	*m*
y	x(11)	=	y position of aircraft CoG in F_E	*m*
z	x(12)	=	z position of aircraft CoG in F_E	*m*

Table 2. RCAM states

Rational Parameter-dependent System Matrix A(p):

$$
A(p) = \left[\begin{array}{cccccc}
-117.05\,\frac{1}{C_w\,V_A} & 0 & 50.807\,\frac{1}{C_w\,V_A} & 0 & 0 & 0 \\
0 & \frac{0.70528\,Z_{cg} - 96.507 + 24.879\,X_{cg}}{C_w\,V_A} & 0 & 0 & 0 & 0 \\
4.8192\,\frac{1}{C_w\,V_A} & 0 & -48.116\,\frac{1}{C_w\,V_A} & 0 & 0 & 0 \\
1.0 & 0 & \alpha & 0 & 0 & 0 \\
0 & 1.0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1.0004 & 0 & 0 & 0 \\
0 & \frac{-1.9860\,\tilde{b}_{72} - 1.0\,{V_A}^2\alpha\,C_w}{C_w\,V_A} & 0 & 0 & -9.8061 & 0 \\
V_A\,\alpha & 0 & -V_A & 9.8061 & 0 & 0 \\
0 & \frac{-241.25 + 0.0040000\,C_w\,V_A + {V_A}^2 C_w}{C_w\,V_A} & 0 & 0 & -9.8100\,\alpha & 0 \\
0 & 0 & 0 & 0 & 0.000043244 & 0 \\
0 & 0 & 0 & -V_A\,\alpha & 0 & V_A \\
0 & 0 & 0 & 0 & -V_A & 0
\end{array}\right.
$$

$$
\left.\begin{array}{cccccc}
0 & \frac{-2.2278 - 0.054189\,X_{cg} + 2.5880\,Z_{cg}}{C_w\,V_A} & 0 & 0 & 0 & 0 \\
0.061601\,\frac{\tilde{a}_{27}}{C_w\,V_A} & 0 & 0.061601\,\frac{\tilde{a}_{29}}{C_w\,V_A} & 0 & 0 & 0 \\
0 & 0.061601\,\frac{\tilde{a}_{38}}{C_w\,V_A} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
-0.061601\,\frac{\tilde{a}_{77}}{C_w\,V_A} & 0 & -0.061601\,\frac{\tilde{a}_{79}}{C_w\,V_A} & 0 & 0 & 0 \\
0 & -15.697\,\frac{1}{C_w\,V_A} & 0 & 0 & 0 & 0 \\
-0.061601\,\frac{\tilde{a}_{97}}{C_w\,V_A} & 0 & -0.061601\,\frac{\tilde{a}_{99}}{C_w\,V_A} & 0 & 0 & 0 \\
1 & 0 & \alpha & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
-\alpha & 0 & 1 & 0 & 0 & 0
\end{array}\right]
$$

with $C_w = \frac{mg}{\frac{1}{2}\rho V_A^2 S}$ and

$$
\begin{aligned}
\tilde{a}_{27} &= 2.1451\,X_{cg}\,{C_w}^2 Z_{cg} + 0.058556\,X_{cg}\,C_w\,Z_{cg} - 20.291\,X_{cg}\,C_w + 1.1425\,X_{cg}\,{C_w}^2 \\
&\quad -0.90635\,{C_w}^2 - 9.5334 + 9.2389\,C_w + 18.030\,X_{cg} - 5.7399\,Z_{cg} - 5.6075\,{C_w}^2 Z_{cg} \\
&\quad -0.97164\,X_{cg}\,Z_{cg} + 5.7418\,C_w\,Z_{cg} \\
\tilde{a}_{29} &= 1.6726\,X_{cg}\,{C_w}^2 Z_{cg} - 0.17230\,{X_{cg}}^2 C_w - 3.9324\,X_{cg}\,C_w\,Z_{cg} - 0.28903\,{X_{cg}}^2 {C_w}^2 Z_{cg} - 46.850 \\
&\quad -0.070972\,{X_{cg}}^2 Z_{cg} + 0.29652\,{X_{cg}}^2 C_w\,Z_{cg} + 4.9667\,X_{cg}\,C_w - 2.7036\,X_{cg}\,{C_w}^2 + 0.58292\,{C_w}^2 \\
&\quad -0.25564\,{X_{cg}}^2 - 1.3439\,C_w + 100.13\,X_{cg} - 14.251\,Z_{cg} - 1.9116\,{C_w}^2 Z_{cg} + 1.1243\,X_{cg}\,Z_{cg} \\
&\quad +24.656\,C_w\,Z_{cg} + 0.45703\,{X_{cg}}^2 {C_w}^2 \\
\tilde{a}_{38} &= 0.096425\,{X_{cg}}^2 C_w - 0.086069\,{X_{cg}}^2 + 1.6082\,X_{cg}\,C_w - 16.591\,X_{cg} - 7.0577\,C_w + 18.418 \\
\tilde{a}_{77} &= 1.5667\,{C_w}^2 - 16.241\,C_w + 65.449 \\
\tilde{a}_{79} &= -201.39\,C_w + 121.84 \\
a_{97} &= 144.91\,C_w + 171.66 \\
\tilde{a}_{99} &= 24.355\,{C_w}^2 + 6.0937\,C_w + 962.75 \\
\alpha &= -0.041088\,X_{cg}\,C_w - 0.0053886\,X_{cg} + 0.17559\,C_w - 0.16287 \\
\tilde{b}_{72} &= 4.9092\,X_{cg}\,C_w + 0.73956\,X_{cg} - 21.270\,C_w + 19.721
\end{aligned}
$$