Guidance command generation and nonlinear dynamic inversion control for reusable launch vehicles

Paul Acquatella B., Lâle Evrim Briese, Klaus Schnepper

Future launch vehicle concepts and technologies for expendable and reusable launch vehicles are currently investigated by the DLR research projects AKIRA and X-TRAS. In particular, the winged *Liquid Fly-back Booster* concept LFBB based on an LOX/LH2 propellant combination for vertical takeoff and vertical landing (VTVL), as well as the delta-winged horizontal takeoff and horizontal landing (HTHL) concept Aurora based on an LOX/Kerosene propellant combination are considered in these projects. Because of the complexity and risks involved in on-line trajectory optimization, off-line reference trajectories are still considered important for tracking purposes. In that sense, the goal of this paper is to investigate an off-line and general-purpose guidance and control (G&C) architecture for preliminary studies of reusable launch vehicles. This is done by using trajectory optimization combined with MODELICA models for the generation of optimal guidance commands, and then trajectory tracking is performed by means of inner-loop feedback controls in terms of nonlinear dynamic inversion with prescribed desired dynamics. We showcase the advantages of this baseline G&C architecture in terms of early stability and controllability aspects during the preliminary design studies of an example configuration of a reusable launch vehicle investigated in the context of the research projects above mentioned.

Guidance Command Generation and Nonlinear Dynamic Inversion Control for Reusable Launch Vehicles

Paul Acquatella B.*, Lâle Evrim Briese, Klaus Schnepper

DLR, German Aerospace Center
Institute of System Dynamics and Control
D-82234 Oberpfaffenhofen, Germany

Abstract

Future launch vehicle concepts and technologies for expendable and reusable launch vehicles are currently investigated by the DLR research projects Akira and X-tras. In particular, the winged *Liquid Fly-back Booster* concept Lfbb based on an LOX/LH2 propellant combination for vertical takeoff and vertical landing (VTVL), as well as the delta-winged horizontal takeoff and horizontal landing (HTHL) concept Aurora based on an LOX/Kerosene propellant combination are considered in these projects. Because of the complexity and risks involved in on-line trajectory optimization, off-line reference trajectories are still considered important for tracking purposes. In that sense, the goal of this paper is to investigate an off-line and general-purpose guidance and control (G&C) architecture for preliminary studies of reusable launch vehicles. This is done by using trajectory optimization combined with Modelica models for the generation of optimal guidance commands, and then trajectory tracking is performed by means of inner-loop feedback controls in terms of nonlinear dynamic inversion with prescribed desired dynamics. We showcase the advantages of this baseline G&C architecture in terms of early stability and controllability aspects during the preliminary design studies of an example configuration of a reusable launch vehicle investigated in the context of the research projects above mentioned.

Keywords: trajectory optimization, guidance, nonlinear control, dynamic inversion, reusable space launchers

*Corresponding author.

Paper presented at the 69th International Astronautical Congress (IAC), October 1-5, 2018, Bremen, Germany. Paper-Nr: IAC–18–C1.5.12.

Email addresses: `paul.acquatella@dlr.de` (Paul Acquatella B.), `lale.briese@dlr.de` (Lâle Evrim Briese), `klaus.schnepper@dlr.de` (Klaus Schnepper)

Accepted 1 April 2020, Acta Astronautica

1. Introduction

1.1. Background

Several studies on future launch vehicle configurations and technologies for expendable and reusable launch vehicles have been extensively conducted in the past at DLR [1, 2, 3, 4, 5, 6, 7, 8]. Currently, partly or fully reusable launch vehicles using different return methods are investigated at DLR in the context of the research projects AKIRA and X-TRAS [9, 10, 11].

Reusability of launch vehicles strongly impacts the launch servicing market whenever sufficient reliability and low refurbishment costs can be achieved. Thus, keeping up with such rapidly evolving international launcher market is essential for Europe, and therefore the need for continuous investigation of different methods and technologies for reusability [5, 8, 12].

In particular, the winged *Liquid Fly-back Booster* concept LFBB, studied extensively during the early 2000's [13, 14] and more recently in [9], based on an LOX/LH2 propellant combination for vertical takeoff and vertical landing (VTVL), as well as the more recent study of the delta-winged horizontal takeoff and horizontal landing (HTHL) concept AURORA [10, 11] based on an LOX/Kerosene propellant combination have been considered.

For the launcher concepts and configurations to consider and optimize in preliminary design studies, early stability and controllability aspects are necessary. This leads to the following motivation for this paper.

1.2. Motivation

This paper focuses on early stability and controllability aspects during the preliminary design studies of launcher concepts. Identifying the impact of such aspects on performance, reaction control system (RCS) design, and actuator sizing (RCS, aerodynamic control surfaces, thrust vector control), among many others, is of great importance.

In particular, for each reusable launcher design study we ask ourselves these questions:

- *What is the optimal reference trajectory according to the mission constraints and requirements?*
- *Is this configuration controllable?*
- *What is the impact of the controllability on the design (impulse budget, reaction control system sizing, aerodynamic control surfaces, etc.)?*

Because of the complexity and risks involved in on-line trajectory optimization, off-line reference trajectories are still considered important for tracking purposes. In that sense, to answer the questions above, we focus on a guidance and control (G&C) architecture by using an optimal trajectory generator to find an off-line reference trajectory, and then trajectory tracking is performed by means of inner-loop feedback controls using Nonlinear Dynamic Inversion (NDI) and linear control (LC) methods.

1.3. Previous work

Trajectory optimization problems are often treated by means of *direct methods* [15, 16] since they can be converted from an infinite-dimensional (optimal control) problem into a finite-dimensional approximation by means of transcription [17, 18]. Solving these approximated trajectory optimization problems is widely considered in the literature. For a comprehensive survey of numerical methods for trajectory optimization the reader is referred to [17]. Transcribing the optimal control problem into a direct formulation often results in a multi-objective and multi-criteria (nonlinear) optimization problem that can be mapped into weighted *min-max* optimization problems. In particular, the optimization problem considered in this paper can be readily solved with the *Trajectory Optimization Package 'trajOpt'* [19] of DLR-SR's optimization tool Mops (*Multi-Objective Parameter Synthesis*) [20], which contains several *sequential quadratic programming* (SQP) algorithms [21, 18].

Nonlinear Dynamic Inversion, based on *feedback linearization* [22, 23, 24], is very common in the aerospace field where some applications of flight control include [25, 26, 27, 28, 29]. More advanced methods involving robustness and improvements of the method for NDI-based flight control applications are considered, among many others, in [28, 29, 30, 31, 32, 33, 34].

NDI methods also found their application for the control of spacecraft and re-entry vehicles, see for example [35, 36, 37, 38] and the references therein. Early works on NDI for space applications include [35], where a nonlinear flight control system for a winged re-entry vehicle was designed that accurately tracks attitude commands while being subject to significant aerodynamic uncertainties, and [36], where a general purpose two-loop flight control architecture for attitude control was designed based on time-scale separation for a lifting body re-entry vehicle using nonlinear dynamic inversion.

The work here presented is largely based on these last references [35, 36, 37, 38], however more oriented towards an integrated approach as in [26] combining trajectory optimization, nonlinear models implemented in the acausal modeling language Modelica, and NDI control; such synergy between these methods is not widely considered in the literature of reusable launch vehicles which leads to the following objectives.

1.4. Objectives

The goal of this paper is therefore to develop a baseline and general-purpose G&C architecture for reusable launch vehicles involving the combination of trajectory optimization and Modelica models for nonlinear control design, see Fig. 1. We do this by combining the following three separate methods:

1. **Trajectory Optimization**. An off-line reference trajectory can be generated by transcribing the trajectory optimization problem into a multi-criteria optimization problem. Solutions are found with a direct approach using the trajectory optimization package '*trajOpt*' of DLR-SR's optimization tool Mops (*Multi-Objective Parameter Synthesis*).

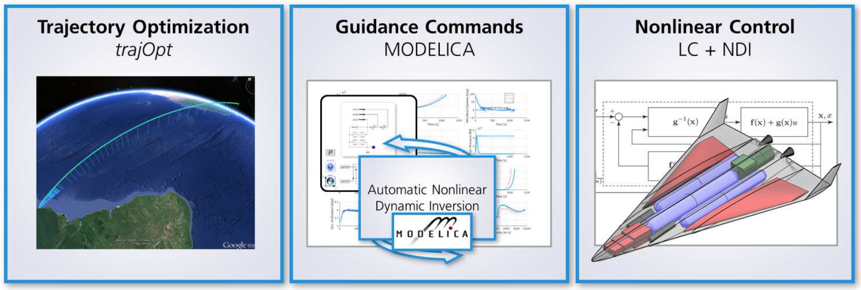


Figure 1: Workflow of the proposed G&C design architecture.

2. **Guidance Command Generation**. Guidance commands are generated via combination of *trajOpt* and nonlinear models implemented with the object-oriented, equation-based, multi-physical, and acausal modeling language Modelica. These commands consists of the optimal flight path reference and their corresponding reference values (aerodynamic angles) for the inner-loop attitude control.

3. **Nonlinear Dynamic Inversion Control**. Lastly, inner-loop attitude control is based on nonlinear dynamic inversion (NDI). NDI cancels out nonlinearities in the system via state feedback, and then desired dynamics can be prescribed to track the optimal reference trajectory. The nominal performance is therefore considered as a benchmark for the controllability analysis of the launch vehicle along the reference trajectory.

To demonstrate the feasibility of using this integrated approach, we showcase the advantages of this baseline G&C architecture in terms of early stability and controllability aspects during the preliminary design studies of an example configuration of a reusable launch vehicle.

The remainder of the paper is organized as follows. Section 2 presents trajectory optimization problem formulation and its solution. In Sections 3 the optimal guidance commands that are obtained with the trajectory optimization in combination with Modelica models are explained. Section 4 briefly explains the control design method behind the nominal trajectory tracking, and Section 5 presents the application of these methods to an exemplary reusable launcher configuration. Conclusions are discussed in Section 6.

2. Optimal Trajectory Generation

In this section, the general trajectory optimization problem that can be treated with the *Trajectory Optimization Package 'trajOpt'* [19] is specified. Following this, the transcription to a problem handled by Mops *(Multi-Objective Parameter Synthesis)*[20] is shown. Mops solves the transcribed multi-objective design problems by mapping them to weighted *min-max* optimization problems.

The *trajOpt* structure and its classes supports this transcription process by its implementation as an object-oriented MATLAB [39] package within MOPS.

The trajectory optimization problem formulation in this paper is categorized as a "multiple shooting method" formulation [18]. The main reason to use a multiple shooting method is to improve the robustness that arises with the sensitivity problems associated with direct shooting [18], which in turn may lead to numerical instability during optimization.

2.1. Optimal Trajectory Optimization Problem Formulation

The description of trajectory optimization problems follows the notation used in MOPS. In particular constraints and optimization criteria are defined by just one category of functions: MOPS criteria. Mathematically the trajectory optimization problems covered can be described as:

Given m phases with possibly optimizable phase times

$$t^j \in \{\ t^0 < t^1 < \ldots < t^m\ \} \tag{1}$$

the states $x^j(t)$ for each phase j obey initial value problems of the form:

$$\dot{x}^j = f^j(t, x^j, u^j, p^j), \quad x^j(t^{j-1}) = s^j, \quad j \in 1 \ldots m. \tag{2}$$

Here $u^j(t)$ are (optimizable) control functions in phase j and p^j are constant scalar modeling parameters (design parameters). The differential equations for each of the multiple phases can differ completely. A well known example is the ascent optimization for multistage rockets, where each stage configuration defines a phase of the problem.

For each phase there can be criteria specified at the phase's final time (right side)

$$\begin{array}{ll} \min {}_r\Psi^j_{k_j}(t^j, x^j(t^j), u^j(t^j), p^j) & j, k_j \in S_m \\ {}_r\Psi^j_{k_j}(t^j, x^j(t^j), u^j(t^j), p^j) \le 1 & j, k_j \in S_i \\ {}_r\Psi^j_{k_j}(t^j, x^j(t^j), u^j(t^j), p^j) = 1 & j, k_j \in S_e \\ \qquad \text{for all} \ \begin{cases} j \in 1 \ldots m \\ k_j \in 1 \ldots n^r_j \end{cases} & \end{array} \tag{3}$$

and the phase's initial time (left side)

$$\begin{array}{ll} \min {}_l\Psi^j_{k_j}(t^j, x^j(t^j), u^j(t^j), p^j) & j, k_j \in S_m \\ {}_l\Psi^j_{k_j}(t^j, x^j(t^j), u^j(t^j), p^j) \le 1 & j, k_j \in S_i \\ {}_l\Psi^j_{k_j}(t^j, x^j(t^j), u^j(t^j), p^j) = 1 & j, k_j \in S_e \\ \qquad \text{for all} \ \begin{cases} j \in 1 \ldots m \\ k_j \in 1 \ldots n^l_j \end{cases} & \end{array} \tag{4}$$

In this notation, for phase j, we have n^r_j final and/or n^l_j initial criteria. The initial values for state differential equations in phase j are $x^j(t^{j-1}) = s^j$.

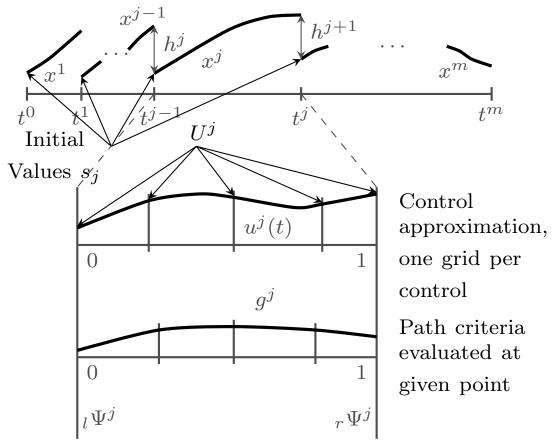


Figure 2: Multi-phase trajectory optimization problem with control discretization.

Optionally there can be additional path criteria evaluated at specified discrete times in the phase

$$\begin{aligned} \min g^j_{o_j}(t_{k_j}, x^j(t_{k_j}), u^j(t_{k_j}), p^j) \quad & j, k_j \in S_m \\ g^j_{o_j}(t_{k_j}, x^j(t_{k_j}), u^j(t_{k_j}), p^j) \leq 1 \quad & j, k_j \in S_i \\ g^j_{o_j}(t_{k_j}, x^j(t_{k_j}), u^j(t_{k_j}), p^j) = 1 \quad & j, k_j \in S_e \\ \text{for all} \begin{cases} j & \in \quad 1 \dots m \\ o_j & \in \quad 1 \dots n_j \\ t_{k_j} & \in \quad [t^{j-1}, t^j] \end{cases} & \end{aligned} \tag{5}$$

and phase connect constraints of the form

$$\begin{bmatrix} x^{j+1}(t^j) \\ u^{j+1}(t^j) \\ p^{j+1} \end{bmatrix} = h^{j+1}(t^j, x^j(t^j), u^j(t^j), p^j) \qquad \text{for all} \quad j \in 1, \dots, m-1. \tag{6}$$

Here, S_m denotes the set of criteria to be minimized, and S_e and S_i are the sets of equality and inequality criteria from Equations (3), (4), and (5), and the equality criteria defined by Equation (6).

A graphical representation of this general problem is shown in Figure 2 including the control approximation and path criteria formulation.

2.2. Transcription into a direct approach

The trajectory optimization problem as posed in the previous section is an optimal control problem in function space for the control functions u^j. In order to solve trajectory optimization problems from Equations (1) to (6) the control functions are discretized by approximation functions $u^j(t) = u^j(U^j, t)$, like piecewise polynomial functions with discretization parameters $U^j, j \in 1, \ldots, m$. These discretization parameters are added to the initial values s_j for the state equations, modeling parameters p^j, and the phase times t^j to form the optimization parameters (and tuners) of the rewritten optimization problem.

This transcription of the original trajectory optimization problem results in defining k design objectives as positive criteria c_k to be minimized against demanded values d_k by considering the following *min-max* constrained multi-criteria optimization problem (see MOPS [20])

$$\min_{\mathcal{T}} \left\{ \max_{k \in S_m} \left\{ \frac{c_k(\mathcal{T})}{d_k} \right\} \right\}, \tag{7a}$$

$$\begin{aligned} \text{subject to } c_k(\mathcal{T}) &= d_k, \quad k \in S_\mathrm{e}, \\ c_k(\mathcal{T}) &\le d_k, \quad k \in S_\mathrm{i}, \end{aligned}$$

with:

$$\mathcal{T}_\mathrm{min} \le \mathcal{T} \le \mathcal{T}_\mathrm{max}. \tag{7b}$$

Here, $\mathcal{T}$ is a vector containing the tuning parameters to be optimized, which is constrained by upper and lower bounds $\mathcal{T}_\mathrm{min}$ and $\mathcal{T}_\mathrm{max}$. $c_k \in S_m$ is the $k-$th normalized criterion and d_k its corresponding demand value which serves as a criterion weight; lastly, $c_k \in S_\mathrm{e}, S_\mathrm{i}$ are normalized criteria which are used as equality and inequality constraints. This multi-criteria optimization problem can then be solved using standard nonlinear programming (NLP) methods such as the widely used *sequential quadratic programming* (SQP) algorithms [21, 18], implemented internally in MOPS [20]. To solve multiple shooting nonlinear optimization problems, MOPS uses an efficient SQP implementation with linear least squares sub-problems [40] based on the works of Schittkowski, Lawson, and Hanson [41, 42].

As already mentioned, to support the transcription process, MOPS was augmented by the object-oriented MATLAB package *trajOpt* [19]. *trajOpt* defines base classes for specifying the ODE right-hand sides from Equation (2) and the criteria functions from Equations (3) to (6). These base classes handle much of the detail of criteria definition and evaluation within MOPS. A user needs to derive classes from these base classes for specifying only the actual criteria functions. This can be particularly easy when using Funtional Mockup Units (FMUs) as models for the ODE and criteria functions where this can reduce to a purely declarative process.

In addition *trajOpt* defines classes for handling the simulation of the actual model within the different phases and the correct evaluation of criteria functions. In particular, classes exist that hide the intricacies of using FMUs as models within the trajectory optimization framework. Additionally, using different FMU units in different phases is supported along with the ability to use MOPS and *trajOpt* in MATLAB parallel computation environments.

3. Guidance Command Generation

In this section we focus on a nominal off-line guidance method to generate an optimal reference trajectory which keeps the launch vehicle's mission and physical constraints within its optimal values. These guidance reference commands are generated via combination of the trajectory optimization package *trajOpt* with nonlinear models implemented with the object-oriented, equation-based, multi-physical, and acausal modeling language MODELICA, which is briefly introduced in the next subsection. The MODELICA models used in this study regarding trajectory optimization (3-DOF) and trajectory tracking with nonlinear control (6-DOF), together with their implementation using an advanced launch vehicle modeling framework are presented in more detail in [43, 44].

3.1. MODELICA

MODELICA [45, 46, 47, 48, 49] is a modern object-oriented, equation-based *modeling language* well suited to model complex physical systems containing, e.g., mechanical, electrical, electronic, hydraulic, thermal, control, power or process–oriented subsystems and components.

Models in MODELICA are described using differential, algebraic, and discrete equations which are then mapped into a mathematical description form called hybrid *Differential Algebraic Equations* (DAEs). A DAE system on its implicit form is generally expressed as

$$\mathbf{F}\big(\dot{\boldsymbol{x}}(t), \boldsymbol{x}(t), \mathbf{u}(t), \boldsymbol{y}(t), \boldsymbol{\rho}, t\big) = \mathbf{0}, \tag{8}$$

where $\dot{\boldsymbol{x}}$ are the state derivatives, $\boldsymbol{x}$ the state variables, $\mathbf{u}$ the inputs, $\boldsymbol{y}$ the algebraic variables, $\boldsymbol{\rho}$ the parameters and constants, t the time variable, and the dimension $dim(\mathbf{F}) = dim(\boldsymbol{x}) + dim(\boldsymbol{y})$. Systems are then solved and simulated by MODELICA simulation environments. When these systems are represented in the DAE implicit form, they can be solved directly by a DAE solver such as DASSL. Alternatively, the system can be sorted out according to specific inputs and outputs and mapped into an explicit ODE (Ordinary Differential Equation) form by solving for the derivatives and the algebraic variables, and then subsequently solved numerically by an ODE solver. The process and details of MODELICA's code compilation is out of the scope of this paper.

3.1.1. Main features [49]

In contrast to imperative languages, in which statements and algorithms are assigned in explicit steps, MODELICA is *declarative*, meaning that declarations are given through equations. These declarations most often describe model's first-principles at their lowest levels without explicit orders or *how* to compute them, hence why MODELICA is said to be *equation based*. By means of specialized algorithms, these declarative models are translated into efficient computer executable code. This allows *acausal* modeling capabilities that give better reuse of classes since equations do not specify a certain data flow direction. This is therefore one of the most important features of the language.

Modelica is *domain neutral*. In other words, it has *multi-domain* modeling capability, meaning that model components corresponding to physical objects from several different domains can be described and connected. This interaction between components is defined by means of physical ports, called connectors, and the interconnection is given accordingly to their physical meaning. This meaning is typically represented by flow variables, which describe quantities whose values add up to zero in a node connection (Kirchhoff's first rule); and by non-flow (or potential) variables, which in contrast remain equal (Kirchhoff's second rule).

Modelica is an *object-oriented* language. This helps to model systems and their physical meaning within an object-oriented structure, facilitating the reuse of component models and the evolution of the structure itself. Thus, object-orientation is primarily used as a *structuring* concept which exploits the declarative feature of the language, as well as the re-usability of models.

Modelica has a strong software component model with constructs for creating and connecting components in a *modular* fashion. Systems' individual components are defined separately as objects, and their interconnection is given accordingly to their physical meaning. Thus the language is ideally suited as an *architectural* description language for complex physical systems.

3.2. Flight path guidance

Position and flight path control loops are readily obtained from the trajectory optimization package *trajOpt* depending on problem-specific optimization goals, requirements, and constraints. These can be such as maximizing the payload to a desired orbit and maximizing downrange for the descent vehicle while minimizing accelerations and dynamic pressure, and thus mechanical and thermal loads, for instance.

In this sense, the reference trajectory in terms of position and flight path provides the guidance commands that have to be tracked by the attitude control subsystem, which in turn commands the launch vehicle in terms of moments that are actuated by the aerodynamic surface deflections, the thrust vector control (TVC), or by the reaction control system (RCS) thrusters, depending on the configuration and the phase considered.

To that end, a Modelica-based 3-DOF launch vehicle model with phase-dependent configuration parameters is exported as a Functional Mock-up Unit (FMU) from Modelica as shown graphically in Figure 3. The FMU, according to the the Functional Mock-up Interface (FMI) [50], defines a model interface which is tool–independent and can therefore be used to access and simulate the model. Subsequently, this FMU is imported separately for each phase into *trajOpt*. Depending on the chosen configuration and flight phase of the launch vehicle, multiple control input variables like the aerodynamic angle of attack α, the aerodynamic sideslip angle β, the aerodynamic bank angle μ, as well as a throttle factor c_s can be active during the trajectory optimization; these FMU inputs are also shown graphically in Figure 3, where all output values from the model are denoted by $\mathbf{R}$. As a result, the optimal reference trajectory can be obtained for several quantities such as positions, velocities, transformation matrices, forces, or even some corresponding atmospheric parameters. The launch vehicle modeling framework as shown in Figure 3 as well as the methodology to export these Modelica-based launch vehicle models as FMUs for use in trajectory optimization are explained in detail in [43, 44]. To conclude the model interfaces, the overall workflow behind the Modelica-based launch vehicle model export as Functional Mock-up Units (FMUs) into the *'trajOpt'*/Mops environment is shown graphically in Figure 4.

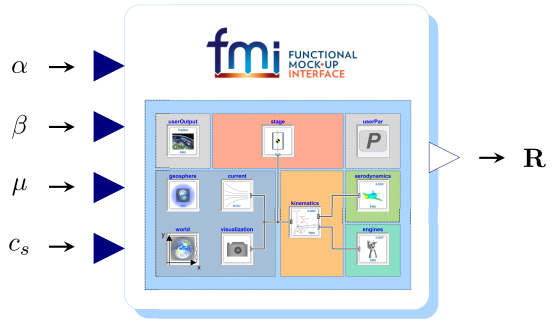


Figure 3: Input-Output Structure of an FMU containing the Launch Vehicle Modeling Framework [43, 44].

The guidance command generation for this loop consists of the resulting flight path reference commands that are given in terms of the reference flight path parameters

$$\begin{bmatrix} V \\ \gamma \\ \chi \end{bmatrix}_{\text{ref}}$$

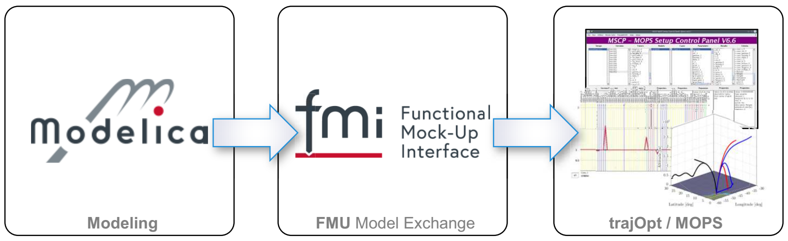

Modelica
Modeling
fmi
Functional Mock-Up Interface
FMU Model Exchange
MSCP – MOPS Setup Control Panel V6.6
trajOpt / MOPS

Figure 4: Workflow of the model exchange, from Modelica to FMU to trajOpt/MOPS.

and its corresponding control commands for the aerodynamic angles

$$\begin{bmatrix} \mu \\ \alpha \\ \beta \end{bmatrix}_{\text{cmd}}$$

respectively. The additional throttle factor c_s is taken from the reference trajectory and used as a feedforward command in the attitude control.

One major advantage of the trajectory optimization and guidance command generation approach as discussed in detail in [43] is, that by considering multiphase trajectory optimization, the computation of each trajectory phase with its respective objectives and constraints can be parallelized. This is useful when the ascent and upper stage phases have different objectives in contrast to the descent phase, although the overall trajectory must fulfill the overall mission objectives. This allows the rapid prototyping and analyses of different concepts and mission profiles.

Having found the off-line reference trajectory providing the nominal guidance commands, the final step for the baseline G&C architecture of this work is the design of the attitude control subsystem (ACS). The ACS is designed to track this reference trajectory within prescribed desired dynamics together with nonlinear dynamic inversion control, which are presented next.

4. Nonlinear Dynamic Inversion Control

Without loss of generality, consider a general multiple-input and multiple-output (MIMO) system whose number of inputs are equal to the number of outputs in order to avoid control allocation problems. Let's also assume momentarily that the nonlinear system can be described affine in the inputs as

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} \tag{9a}$$

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}) \tag{9b}$$

where $\boldsymbol{x} \in \mathcal{R}^n$ is the state vector, $\boldsymbol{u} \in \mathcal{R}^m$ is the control input vector, and $\boldsymbol{y} \in \mathcal{R}^m$ is the system output vector, the functions $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x})$ are assumed to be smooth vector fields on $\mathcal{R}^n$, and $\boldsymbol{g}(\boldsymbol{x}) \in \mathcal{R}^{n\times m}$ is a matrix whose columns are also assumed as smooth vector fields $\boldsymbol{g}_j$. Moreover, we consider $\boldsymbol{y} = \boldsymbol{x}$ so that the relative degree of each of the outputs $y_i,\ i = \{1, \ldots, m\}$ is one.

The idea of Nonlinear Dynamic Inversion (NDI) consists on canceling the nonlinearities in such nonlinear system so that the closed-loop dynamics is in a linear form. In other words, the nonlinear system is inverted by means of state feedback into a linear structure, and hence conventional linear controllers can be applied. A fundamental assumption is that the model of the system is exactly known, which gives NDI a great disadvantage from the point of view of uncertainties. Moreover, we also assume to have complete and accurate knowledge about the state of the system, which is hard to achieve in practice. NDI consists on the application of the following input transformation [51, 52, 53, 22]

$$\boldsymbol{u}_{\mathrm{cmd}} = \boldsymbol{g}^{-1}(\boldsymbol{x})\,(\boldsymbol{\nu} - \boldsymbol{f}(\boldsymbol{x})) \tag{10}$$

which cancels all nonlinearities in closed-loop, and a simple linear input-output relationship between the new virtual control input $\boldsymbol{\nu}$ and the output $\boldsymbol{y}$ is obtained

$$\dot{\boldsymbol{y}} = \boldsymbol{\nu}. \tag{11}$$

Apart from being linear, an interesting result from this relationship is that it is also decoupled since the input ν_i only affects the output y_i. From this fact, the input transformation (10) is called a *decoupling control law*, and the resulting linear system (11) is called the *single-integrator* form. This single-integrator form (11) can be rendered exponentially stable with

$$\boldsymbol{\nu} = \dot{\boldsymbol{y}}_{\mathrm{des}} = \dot{\boldsymbol{y}}_{\mathrm{cmd}} + \boldsymbol{K}_P\,\boldsymbol{e} \tag{12}$$

where $\dot{\boldsymbol{y}}_{\mathrm{des}}$ defines the desired dynamics for the output vector or control variables, $\dot{\boldsymbol{y}}_{\mathrm{cmd}}$ is the feedforward term for tracking, $\boldsymbol{e} = \boldsymbol{y}_{\mathrm{cmd}} - \boldsymbol{y}$ is the error vector, $\boldsymbol{y}_{\mathrm{cmd}}$ denotes the smooth desired output vector (at least one time differentiable), and $\boldsymbol{K}_P \in \mathcal{R}^{m\times m}$ is a diagonal matrix, whose $i-$th diagonal elements K_{P_i} are chosen so that the polynomials

$$s + K_{P_i}, \qquad i = \{1, \ldots, m\} \tag{13}$$

may become Hurwitz. This results in the exponentially stable and decoupled *desired* error dynamics

$$\dot{\boldsymbol{e}} + \boldsymbol{K}_P\,\boldsymbol{e} = \boldsymbol{0}, \tag{14}$$

which implies that $e_i(t) \to 0,\ i = \{1, \ldots, m\}$. From this typical tracking problem, and as illustrated in Figure 5, it can be seen that the entire control system will have two control loops [51, 52, 38]: the inner linearization loop based on Equation (10), and the outer control loop in Equation (12) based on linear control.

4.1. Multi-loop NDI control

For preliminary controllability studies, we will be interested in multi-loop cascaded control architectures. Regarding the attitude control concept, designed to track the reference trajectory and its guidance commands, it is composed of two control loops assuming a sufficient time-scale separation between the attitude kinematics (aerodynamic angles outer-loop) and the rotational dynamics (angular rates inner-loop). In other words, the inner-loop dynamics is assumed to be so fast, that from the outer loop perspective the angular rate commands

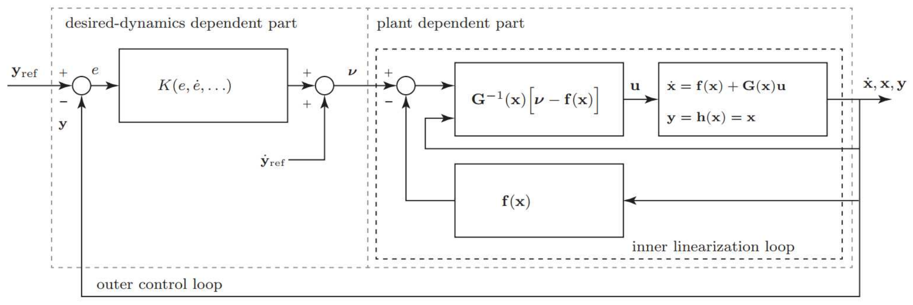

desired-dynamics dependent part
plant dependent part
$\mathbf{y}_{\mathrm{ref}}$
e
$K(e, \dot{e}, \ldots)$
$\boldsymbol{\nu}$
$\dot{\mathbf{y}}_{\mathrm{ref}}$
$\mathbf{y}$
$\mathbf{G}^{-1}(\mathbf{x})\left[\boldsymbol{\nu} - \mathbf{f}(\mathbf{x})\right]$
$\mathbf{u}$
$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}$
$\mathbf{y} = \mathbf{h}(\mathbf{x}) = \mathbf{x}$
$\dot{\mathbf{x}}, \mathbf{x}, \mathbf{y}$
$\mathbf{f}(\mathbf{x})$
inner linearization loop
outer control loop

Figure 5: Nonlinear dynamic inversion tracking control for a nonlinear MIMO system (here, ref = cmd) [38].

are achieved instantaneously. With this assumption, the attitude controller is therefore performed in terms of nonlinear dynamic inversion NDI for each loop.

The outer-loop inversion of the attitude kinematics is very commonly done in attitude control to obtain reference commands for the inner-loop dynamics. In terms of the equations of angular motion, depending on the launcher or re-entry vehicle in consideration, the rotational dynamics can take different forms, especially when considering multi-body and variable mass dynamics. In this paper, we assume that we have an accurate model to invert, and for the preliminary design studies considering stability aspects, we don't consider the effects of uncertainties and disturbances but we rather focus on the nominal behaviour and performance of the plant. In what follows, we denote the states

$$\boldsymbol{x}_1 = \begin{bmatrix} p \\ q \\ r \end{bmatrix}, \quad \boldsymbol{x}_2 = \begin{bmatrix} \mu \\ \alpha \\ \beta \end{bmatrix}, \quad \boldsymbol{x}_3 = \begin{bmatrix} V \\ \gamma \\ \chi \end{bmatrix}$$

with p, q, r being the body roll, pitch, and yaw rates, respectively; μ, α, β, the aerodynamic bank, angle of attack, and aerodynamic sideslip angles, respectively; and V, γ, χ, the relative velocity of the launch vehicle, the flight path angle, and the flight path azimuth, respectively.

The following two-loop NDI attitude control architecture is largely based on [35, 36, 37, 38].

4.2. Body angular rate control loop

Regarding the body angular rate control loop, we are interested in the variable-mass attitude equations of motion as obtained by Eke [54]

$$\boldsymbol{I}\,\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}\,\boldsymbol{\omega} + \Big(\frac{{}^{R}\mathrm{d}\boldsymbol{I}}{\mathrm{d}t}\Big)\boldsymbol{\omega} = \boldsymbol{M}_B + \boldsymbol{M}_V \tag{15}$$

where $\mathbf{M}_B \in \mathcal{R}^3$ is the external moment vector in body axes, $\mathbf{M}_V \in \mathcal{R}^3$ is the internal moment vector due to variable mass dynamics in body axes, $\boldsymbol{\omega} \in \mathcal{R}^3$ is the angular velocity vector, $\boldsymbol{I} \in \mathcal{R}^{3\times 3}$ the inertia matrix of the rigid body, and the left superscript indicates that the time derivative is taken in a frame 'R' on the solid portion of the variable mass system.

The external moments in $\boldsymbol{M}_B$ are considered as the sum of moments partially generated by the aerodynamics of the airframe $\boldsymbol{M}_a$ and moments generated by control surface deflections $\boldsymbol{M}_c$, and we describe $\boldsymbol{M}_B$ linearly in the deflection angles $\boldsymbol{\delta}$ assuming the control derivatives to be linear as in [30] with $(\boldsymbol{M}_c)_\delta = \frac{\partial}{\partial \delta}\boldsymbol{M}_c$; therefore

$$\boldsymbol{M}_B = \boldsymbol{M}_a + \boldsymbol{M}_c = \boldsymbol{M}_a + (\boldsymbol{M}_c)_\delta \boldsymbol{\delta} \tag{16}$$

where

$$\boldsymbol{M}_B = \begin{bmatrix} L \\ M \\ N \end{bmatrix}, \ \boldsymbol{M}_a = \begin{bmatrix} L_a \\ M_a \\ N_a \end{bmatrix}, \ \boldsymbol{M}_c = \begin{bmatrix} L_c \\ M_c \\ N_c \end{bmatrix}$$

with L, M, N, the roll, pitch, and yaw moments, respectively; and

$$\boldsymbol{\delta} = \begin{bmatrix} \delta_a \\ \delta_e \\ \delta_r \end{bmatrix}$$

corresponding to the control inputs: aileron, elevator, and rudder deflection angles, respectively. Furthermore, let $\boldsymbol{M}_V$ be the sum of internal moments generated by the variable mass dynamics as described in [54], where $\boldsymbol{M}_{V_1}$ is the so-called jet damping, $\boldsymbol{M}_{V_2}$ is due to the Coriolis effect (which can be neglected for axisymmetric motion as well as for negligible internal flow), $\boldsymbol{M}_H$ represents the rate of decrease of the system's angular momentum inside its boundary, and $\boldsymbol{M}_{thr}$ the moment of the thrust vector about the mass center; therefore

$$\boldsymbol{M}_V = \mathbf{M}_{V_1} + \mathbf{M}_{V_2} + \mathbf{M}_H + \mathbf{M}_{thr}. \tag{18}$$

The details of these terms are left to the reader and can be found in [54].

Since we will be interested in the body angular rate inversion, which is a state-input inversion problem [51, 52, 53], after a differentiation of the output variable

$$\boldsymbol{y}_1 = \boldsymbol{x}_1 = \boldsymbol{\omega}, \tag{19}$$

we obtain the dynamics of the rotational motion rewritten as the following set of differential equations

$$\dot{\boldsymbol{\omega}} = \boldsymbol{I}^{-1}\boldsymbol{M}_B + \boldsymbol{I}^{-1}\left[\ \boldsymbol{M}_V - \boldsymbol{\omega} \times \boldsymbol{I}\,\boldsymbol{\omega} - \Big(\frac{{}^R\mathrm{d}\boldsymbol{I}}{\mathrm{d}t}\Big)\boldsymbol{\omega}\ \right] \tag{20}$$

which inverted analytically yields

$$\bar{\boldsymbol{M}}_B = \boldsymbol{I}\,\dot{\boldsymbol{\omega}} + \left[\ \boldsymbol{\omega} \times \boldsymbol{I}\,\boldsymbol{\omega} + \Big(\frac{{}^R\mathrm{d}\boldsymbol{I}}{\mathrm{d}t}\Big)\boldsymbol{\omega} - \boldsymbol{M}_V\ \right] \tag{21}$$

where we have used the notation $\bar{\boldsymbol{M}}_B$ to denote that these moments are still commanded to the launch vehicle and that are to be produced by the aerodynamic surface deflections, the TVC, or by the RCS thrusters, depending on the configuration or the phase considered. Introducing the virtual control input

$$\boldsymbol{\nu}_\omega = \dot{\boldsymbol{\omega}}_{\text{des}} = \begin{bmatrix} \dot{p}_{\text{des}} \\ \dot{q}_{\text{des}} \\ \dot{r}_{\text{des}} \end{bmatrix} \tag{22}$$

and denoting the internal variable-mass terms as

$$\boldsymbol{M}_i = \Big(\frac{{}^R\mathrm{d}\boldsymbol{I}}{\mathrm{d}t}\Big)\boldsymbol{\omega} - \boldsymbol{M}_V$$

then the NDI control consists in the following transformation [52, 53, 22]

$$\bar{\boldsymbol{M}}_{B_{\text{cmd}}} = \boldsymbol{I}\,\boldsymbol{\nu}_\omega + \boldsymbol{\omega} \times \boldsymbol{I}\,\boldsymbol{\omega} + \boldsymbol{M}_i. \tag{23}$$

In other words

$$\begin{bmatrix} \bar{L}_{\text{cmd}} \\ \bar{M}_{\text{cmd}} \\ \bar{N}_{\text{cmd}} \end{bmatrix} = \begin{bmatrix} \bar{M}^x_{B_{\text{cmd}}} \\ \bar{M}^y_{B_{\text{cmd}}} \\ \bar{M}^z_{B_{\text{cmd}}} \end{bmatrix} = \boldsymbol{I} \begin{bmatrix} \dot{p}_{\text{des}} \\ \dot{q}_{\text{des}} \\ \dot{r}_{\text{des}} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \boldsymbol{I} \begin{bmatrix} p \\ q \\ r \end{bmatrix} + \boldsymbol{M}_i.$$

Notice that whenever the variable-mass dynamics in $\boldsymbol{M}_i$ are not considered, then the Newton-Euler equations of motion for a rigid body are recovered and the NDI control design is further simplified. In general, depending on the nature of the propulsion system and its corresponding shape or assumed burn profiles, these terms can be further simplified and implemented in closed form for simulation an control aspects, see [54, 55, 56]. In this way, these loads can be included explicitly in the formulation of the dynamic equations of motion of the corresponding element of the vehicle so that their effect can be included in attitude control system as model-based feedforward terms.

The desired dynamics in Equations (22) and (23) are specified by prescribing the exponentially stable and decoupled desired error dynamics

$$\dot{\boldsymbol{e}}_\omega + \boldsymbol{K}_\omega \boldsymbol{e}_\omega = \boldsymbol{0}, \tag{24}$$

where

$$\boldsymbol{e}_\omega = \boldsymbol{\omega}_{\text{cmd}} - \boldsymbol{\omega},$$

and

$$\boldsymbol{K}_\omega(s) = \begin{bmatrix} K_{\omega_p}(s) & 0 & 0 \\ 0 & K_{\omega_q}(s) & 0 \\ 0 & 0 & K_{\omega_r}(s) \end{bmatrix}.$$

Here, $\boldsymbol{\omega}_{\text{cmd}}$ is obtained from the aerodynamic angles outer loop, and we have introduced $\boldsymbol{K}_\omega(s)$ as a diagonal matrix while assuming that the control law in Equation (23) is fully decoupling each input-output channel, which is not generally the case. These diagonal terms can be selected, for instance as a classical proportional-integral (PI) control [36, 38] with gains

$$K_{\omega_i}(s) = K_{P_i} + \frac{1}{s} K_{I_i}, \quad i = \{p, q, r\},$$

resulting in the closed loop system

$$\dot{\boldsymbol{\omega}}_{\text{des}} = \dot{\boldsymbol{\omega}}_{\text{cmd}} + \boldsymbol{K}_{P_\omega} (\boldsymbol{\omega}_{\text{cmd}} - \boldsymbol{\omega}) + \boldsymbol{K}_{I_\omega} \int (\boldsymbol{\omega}_{\text{cmd}} - \boldsymbol{\omega}) \mathrm{d}t \tag{25}$$

with the gains

$$\boldsymbol{K}_{P_\omega} = \begin{bmatrix} K_{P_p} & 0 & 0 \\ 0 & K_{P_q} & 0 \\ 0 & 0 & K_{P_r} \end{bmatrix}, \quad \boldsymbol{K}_{I_\omega} = \begin{bmatrix} K_{I_p} & 0 & 0 \\ 0 & K_{I_q} & 0 \\ 0 & 0 & K_{I_r} \end{bmatrix}.$$

Whenever aerodynamic control surfaces are used, the aerodynamics of the airframe and the moments generated by the control surface deflections plays an important role in the dynamic inversion since these terms are hardly known exactly for model inversion. Since we assumed in (16) that the control derivatives are linear, the dynamics can be rewritten as the following

$$\dot{\boldsymbol{\omega}} = \boldsymbol{I}^{-1} (\boldsymbol{M}_c)_\delta \boldsymbol{\delta} + \boldsymbol{I}^{-1} \left[\, \boldsymbol{M}_a - \boldsymbol{M}_i - \boldsymbol{\omega} \times \boldsymbol{I} \boldsymbol{\omega} \, \right] \tag{26}$$

and then the NDI control consists in the following expression

$$\boldsymbol{\delta}_{\text{cmd}} = (\boldsymbol{M}_c)_\delta^{-1} \left[\, \boldsymbol{I} \boldsymbol{\nu}_\omega + \boldsymbol{\omega} \times \boldsymbol{I} \boldsymbol{\omega} + \boldsymbol{M}_i - \boldsymbol{M}_a \, \right]. \tag{27}$$

Here, we have also assumed that the control derivatives are invertible in the whole domain of operation, and that $dim(\boldsymbol{\delta}) = dim(\boldsymbol{y})$, meaning that the number of control variables and control effectors are equal. In the usual case where $dim(\boldsymbol{\delta}) \geq dim(\boldsymbol{y})$, meaning that there are more aerodynamic control surfaces than variables to be controlled, control allocation is required. The opposite case, meaning $dim(\boldsymbol{\delta}) \leq dim(\boldsymbol{y})$, leads to internal dynamics that must be studied in terms of stability, and the system is said to be underactuated. These aspects are however out of the scope of this paper.

Moreover, whenever the aerodynamic model in $(\boldsymbol{M}_c)_\delta$ is not directly invertible (as a multi-dimensional look-up table, for instance), a gain-scheduled PI control control law might be employed for the aerodynamic surface controls simply as $\boldsymbol{\delta}_{\text{cmd}} = \boldsymbol{I} \boldsymbol{\nu}_\omega$. In such cases, a relationship between PI control gain-tuning and (incremental) nonlinear dynamic inversion in the context of flight control systems can be established [57]. This is useful, for instance, when making a composite control low comprised of moments $\bar{\boldsymbol{M}}_{B_{\text{cmd}}}$ generated by the RCS as in (23) which is based on NDI, together with moments generated by the aerodynamic surfaces which is based on PI controls.

4.3. Aerodynamic angles outer-loop

The aerodynamic angles outer-loop inversion procedure is the same as shown before. Since we will be interested in the attitude kinematics inversion, which is also a state-input inversion considering the body angular rate as intermediate control inputs, denoting the output vector

$$\boldsymbol{y}_2 = \boldsymbol{x}_2,$$

the differentiation of this output variable yields the attitude kinematics in terms of the aerodynamic angles as expressed in [37, 58, 59] by

$$\dot{\boldsymbol{x}}_2 = \begin{bmatrix} \dot{\mu} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \boldsymbol{f}_2 + \boldsymbol{G}_2 \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{28}$$

where the angular velocity terms in $\boldsymbol{f}_2 = \boldsymbol{f}_2(\boldsymbol{x}_2, \boldsymbol{x}_3)$, omitted here, are nonlinear functions of the translational terms $\boldsymbol{x}_2,\ \boldsymbol{x}_3$ and their derivatives [37, 58, 59, 60] and

$$\boldsymbol{G}_2 = \frac{1}{\cos\beta} \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ -\cos\alpha\sin\beta & \cos\beta & -\sin\alpha\sin\beta \\ \sin\alpha & 0 & -\cos\alpha \end{bmatrix}.$$

Since this kinematic equation is nonlinear but affine in the angular rates, and in the case that the angular velocity terms contained in $\boldsymbol{f}_2$ are assumed or regarded as very small and negligible for the attitude control subsystem, as it is commonly done in the literature [37, 59], this inner-loop can be readily found by applying the following simple inversion

$$\boldsymbol{\omega}_{\text{cmd}} = \boldsymbol{G}_2^{-1} \boldsymbol{\nu}_{\text{att}} \tag{30}$$

where we have introduced the virtual control input for this loop as

$$\boldsymbol{\nu}_{\text{att}} = \dot{\boldsymbol{x}}_{2_{\text{des}}} = \begin{bmatrix} \dot{\mu}_{\text{des}} \\ \dot{\alpha}_{\text{des}} \\ \dot{\beta}_{\text{des}} \end{bmatrix}. \tag{31}$$

Otherwise, the angular velocity quantities $\boldsymbol{f}_2$ can be added as model-based feedforward terms to the guidance command generation. In other words, we have obtained the outer-loop rate commands as

$$\begin{bmatrix} p_{\text{cmd}} \\ q_{\text{cmd}} \\ r_{\text{cmd}} \end{bmatrix} = \boldsymbol{G}_2^{-1} \begin{bmatrix} \dot{\mu}_{\text{des}} \\ \dot{\alpha}_{\text{des}} \\ \dot{\beta}_{\text{des}} \end{bmatrix} = \begin{bmatrix} \cos\alpha\cos\beta & 0 & \sin\alpha \\ \sin\beta & 1 & 0 \\ \sin\alpha\cos\beta & 0 & -\cos\alpha \end{bmatrix} \begin{bmatrix} \dot{\mu}_{\text{des}} \\ \dot{\alpha}_{\text{des}} \\ \dot{\beta}_{\text{des}} \end{bmatrix}.$$

To finish the attitude control design, the desired aerodynamic angles are specified by prescribing the exponentially stable and decoupled desired error dynamics

$$\dot{\boldsymbol{e}}_{\text{att}} + \boldsymbol{K}_{\text{att}} \boldsymbol{e}_{\text{att}} = \boldsymbol{0}, \tag{32}$$

where

$$\boldsymbol{e}_{\mathrm{att}} = \boldsymbol{x}_{2\mathrm{cmd}} - \boldsymbol{x}_2,$$

and

$$\boldsymbol{K}_{\mathrm{att}}(s) = \begin{bmatrix} K_{\mathrm{att}_\mu} & 0 & 0 \\ 0 & K_{\mathrm{att}_\alpha} & 0 \\ 0 & 0 & K_{\mathrm{att}_\beta} \end{bmatrix},$$

with

$$K_{\mathrm{att}_i} = K_{P_i} + \frac{1}{s} K_{I_i}, \quad i = \{\mu, \alpha, \beta\},$$

resulting in the closed loop system

$$\dot{\boldsymbol{x}}_{2_{\mathrm{des}}} = \dot{\boldsymbol{x}}_{2_{\mathrm{cmd}}} + \boldsymbol{K}_{P_{\mathrm{att}}}(\boldsymbol{x}_{2\mathrm{cmd}} - \boldsymbol{x}_2) + \boldsymbol{K}_{I_{\mathrm{att}}} \int (\boldsymbol{x}_{2\mathrm{cmd}} - \boldsymbol{x}_2)\mathrm{d}t, \tag{33}$$

with the gains

$$\boldsymbol{K}_{P_{\mathrm{att}}} = \begin{bmatrix} K_{P_\mu} & 0 & 0 \\ 0 & K_{P_\alpha} & 0 \\ 0 & 0 & K_{P_\beta} \end{bmatrix}, \quad \boldsymbol{K}_{I_{\mathrm{att}}} = \begin{bmatrix} K_{I_\mu} & 0 & 0 \\ 0 & K_{I_\alpha} & 0 \\ 0 & 0 & K_{I_\beta} \end{bmatrix},$$

and which concludes the attitude control design.

5. Nonlinear Flight Control Simulation

A nonlinear flight control simulation for the position and attitude control of the horizontal takeoff and horizontal landing launch vehicle concept Aurora [10, 11] is here presented.

5.1. Mission profile

The Aurora two-stage-to-orbit (TSTO) concept has been studied at the *Space Launcher System Analysis* (SART) department of the DLR Institute of Space Systems (DLR-RY). This concept considered iterative studies regarding mass budget, propulsion, aerodynamics, and structural optimization amongst many others.

The concept, shown in Figure 6, aims to reduce operational costs while increasing launch frequency [11]. This is done by considering a more 'aircraft-like' operation providing a high lift-to-drag ratio and a propellant combination of LOX/Kerosene allowing placement of the kerosene tanks in the wing structure.

The trajectory optimization of this concept has been performed and shown in [43], where the following goals, requirements, and constraints were considered:

- The ascent of the launch vehicle starts at a launch site located at -52.77° latitude, 5.24° longitude, and zero altitude.

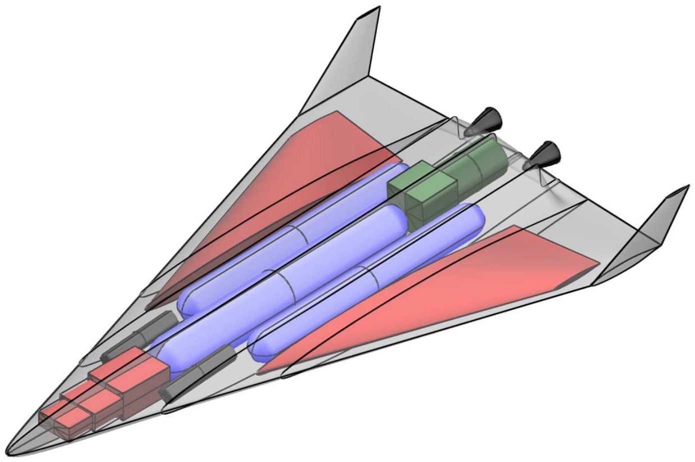


Figure 6: Aurora-RLV concept [11]

- The descent of the launch vehicle to the landing site (-64.68° latitude, 32.36° longitude, zero altitude) has to be guaranteed within a radius of approximately 25 km.
- The payload mass shall be maximized while the upper stage propellant mass is traded for the payload mass.
- The polar orbit with an apogee altitude of 1200 km has to be reached at an inclination of 90° and maximum perigee.
- The following constraints have to be considered to reduce mechanical or thermal loads on the structure:
  - Maximum acceleration n_x lower than 4.5 g.
  - Maximum acceleration n_z lower than 1.75 g (ascent).
  - Maximum acceleration n_z lower than 4.25 g (descent).
  - Maximum dynamic pressure lower than 50 kPa (ascent).
  - Maximum dynamic pressure lower than 60 kPa for the re-entry and the flight to the landing site.
  - Maximum heat flux lower than 900 kW/m^2 for a theoretic reference nose radius of 0.15 m.

The trajectory phases considered with *trajOpt* are listed in Table 1. *Phase P1* considers the horizontal liftoff powered by rocket and air-breathing engines up to $M_a \approx 1$ followed by an ascent *Phase P2* powered only by the rocket engines. *Phase P3* represents a ballistic phase up until the separation of the upper stage stored in the payload bay, initiated at a separation velocity of approximately 5 km/s. Consequently, the *Phase P4* represents the ascent of the upper stage.

Table 1: Trajectory optimization phases considered for AURORA [43].

Phase	Stages	Description
P1	US+MS	Horizontal liftoff
P2	US+MS	Ascent phase (rocket engines)
P3	US+MS	Ballistic phase & separation
P4	US	Ascent of the upper stage
P5	MS	Descent maneuver & return

Table 2: Sub-phases for AURORA descent maneuver.

Phase	Actuators	Description
P5-a	RCS (+ Fins & Flaps)	Re-entry
P5-b	Fins & Flaps	Skipping
P5-c	Fins & Flaps	Final approach

Phase P5 represents the unpowered re-entry maneuver and the return flight of the launch vehicle to the chosen landing site. The ferry flight from the landing site back to the launch site is not considered. Furthermore, *Phase P5* is divided in three sub-phases for trajectory optimization as shown in Table 2.

5.2. Nonlinear Descent Flight Control

Flight simulations on the full 6-DOF nonlinear system are performed for the *Phase P5* since it covers interesting scenarios, such as the re-entry flight and the potential to study the combination of RCS with aerodynamic surface controls during descent. Moreover, there are no variable mass dynamics since this is an unpowered descent maneuver, making the control study much simpler. The simulations are done with the double-loop NDI-based attitude control system to track the generated optimal trajectory of the launch vehicle. We do this preliminarily without being subject to any disturbances or uncertainties, and under the nominal conditions to verify if the plant is controllable during the descent, and within which range in terms of RCS budget and aerodynamic surface controls.

Figure 7 shows the descent guidance and control results for the re-entry maneuver of *Phase P5-a* using only RCS control. This is the baseline scenario considering the attitude control entirely actuated by the RCS thrusters. Since only attitude control is performed, Fig. 7-a shows the resulting ‘open loop’ kinematic position trajectory which is entirely done by means of the nominal attitude tracking control (the relative velocity is shown normalized according to the whole *Phase P5*). Fig. 7-b shows the resulting tracking performance of the attitude control system in terms of the commanded aerodynamic angles. In Fig. 7-c the resulting and commanded rates from the dynamic inversion can be seen; here, the pitch rate q and its commanded values differ because of the highly cross products involved, which are solved for automatically in the inner loop to obtain the required moments and which are not accounted for in the outer-loop commands. The re-entry maneuver demands quite high pitching moments as demonstrated by Fig. 7-d, where the required commanded moments are shown normalized with respect to the complete *Phase P5*. These results show that the system is controllable under the nominal conditions if the obtained bounds of commanded moments are achievable in practice.

As a test-case scenario, we investigate what happens whenever this *Phase P5-a* can be performed in combination with aerodynamic surface controls. This can be done only after $t = 500$ s when the launch vehicle has already entered in the atmosphere below an altitude of $h = 120$ km and therefore commandable in terms of aerodynamic forces and moments. Since the aileron and the elevator commands the flaps simultaneously, we have to restrict these commands such that the combined maximum deflection limit for each flap does not exceed ± 30 deg. The same limits apply for the fins which are actuated with the same limits of ± 30 deg. In that sense, we limit the elevator commands to ± 20 deg and the aileron commands to ± 10 deg (as an initial guideline, not optimized). To achieve this, simulation experiments were done to obtain a good compromise for allocating the moments provided by the RCS thrusters (based on NDI control) vs. the moments provided by the control surfaces (based on PI control) by allowing saturation only in the elevator commands; therefore, the NDI control in the inner loop is only done for the allocated moments to be provided by the RCS. Figure 8 shows the descent guidance and control results for the re-entry maneuver of *Phase P5-a* of the combined aerodynamic surface control and RCS. Besides the open-loop flight path and the nominal attitude tracking performance results, this figure shows the resulting impact of the demanded pitching moments of Fig. 8-d as compared to the ones with RCS thrusters only in Fig. 7-d (normalized). These results show that, while the system is still controllable under the nominal conditions considered, the impact on the RCS budget can be significant while maintaining certain bounds on the aerodynamic actuator efforts. This also showcase the potential benefit in launch vehicle's design that improvements in terms of impulse budgeting (and therefore propellant mass) can already be obtained at preliminary design levels.

In that sense, Fig. 9 shows the resulting aerodynamic control surfaces for *Phase P5-a* in combination with RCS control. The allocation of control surfaces vs. RCS thrust could be further optimized to avoid actuator saturation or to minimize fuel consumption within some actuation limits; however, this subject is not further investigated here. This scenario considering the combined RCS thrusters and aerodynamic control surfaces showcase the potential to reduce by more than half the angular impulse budget for the RCS as shown in Fig. 10. This impact on the RCS budget can lead to further improvements in terms of the launch vehicle preliminary design, since the sizing and location of the RCS thrusters can also have a considerable impact on the vehicle configuration.

To conclude the study, Fig. 11 shows the descent guidance and control results for the *Phase P5-c* which is the final approach of the descent. The attitude control of this phase is entirely performed by aerodynamic surfaces since they can produce the aerodynamic moments required. Once again, these results show that the system is controllable under the nominal conditions and within the bounds of the aerodynamic control surfaces.

6. Conclusions

This paper presented a baseline and general-purpose off-line G&C architecture for reusable launch vehicles for the early stability and controllability studies during preliminary design phases of generic launcher conceptual designs.

Optimal reference trajectories and guidance commands were obtained with a direct approach using the trajectory optimization package '*trajOpt*' in combination with MODELICA models, while inner-loop attitude control was designed in terms of nonlinear dynamic inversion together with prescribed desired error dynamics. Such optimal reference trajectory tracking helps to answer the motivating questions presented in the introduction.

To demonstrate our integrated approach, the AURORA reusable launch vehicle concept was investigated in the context of the methods presented here. The nonlinear control system, simulated for the descent phase including the re-entry flight and covering a wide flying envelope ranging from Mach 18 to Mach 5 and angles of attack between 50 and 9 deg, demonstrate the controllability of the launch vehicle as well as the potential to reduce more than half the impact on the angular impulse budget for the RCS by combining it with aerodynamic surface controls during the re-entry phase.

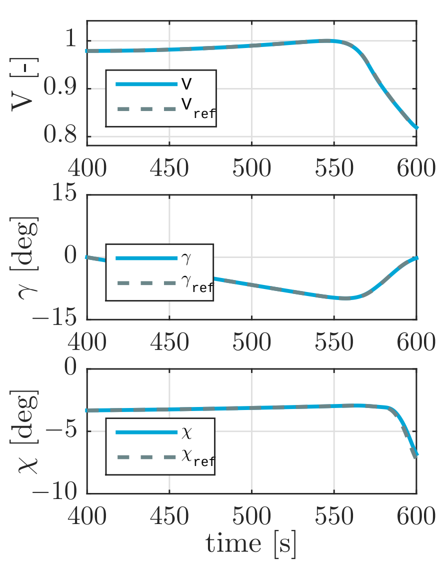
Flight simulations show that the control system accurately tracks commands in aerodynamic angles but preliminarily without being subject to significant aerodynamic uncertainties. This will be part of future work, which will consider and include more detailed analysis of the effect of parametric and aerodynamic uncertainties, as well as external perturbations such as wind and turbulence on the overall G&C and control performance.

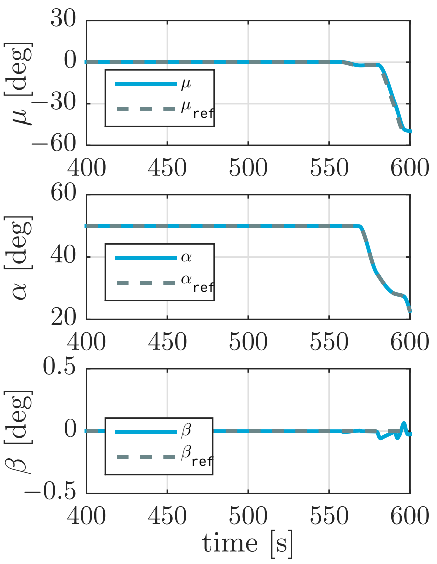
Acknowledgments

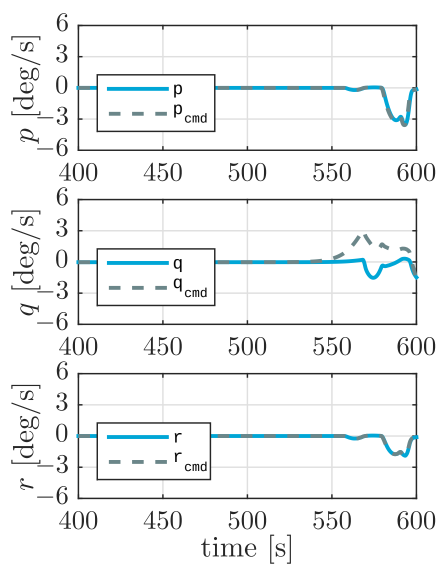
This study was performed within the DLR projects AKIRA and X-TRAS regarding preliminary system studies and evaluation of key technologies for future reusable launch vehicles. The authors would like to thank DLR colleagues from these projects, in particular the colleagues from the *Space Launcher System Analysis* (SART) department of the DLR Institute of Space Systems (DLR-RY) in Bremen for their suggestions and discussions, and for providing data of the considered launch vehicle configurations.

Declaration of Interest

The authors declare that they have no competing interests.


(a) Kinematic position trajectory results, uncontrolled version, nominal case. Reference values from trajectory optimization.


(b) Aerodynamic angles – attitude control results, nominal case. Reference values from trajectory optimization.


(c) Outer-loop angular rates guidance and control results, nominal case. Guidance obtained from model inversion, NDI control.

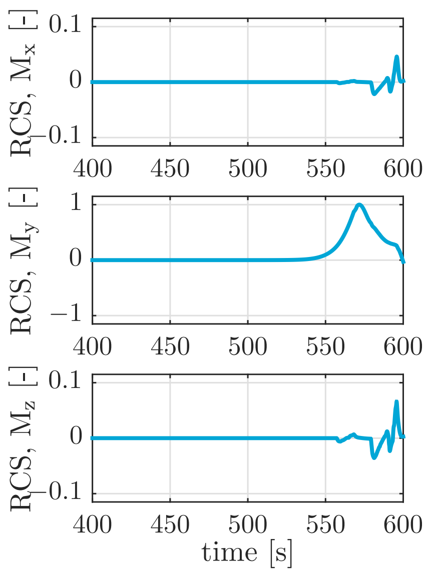
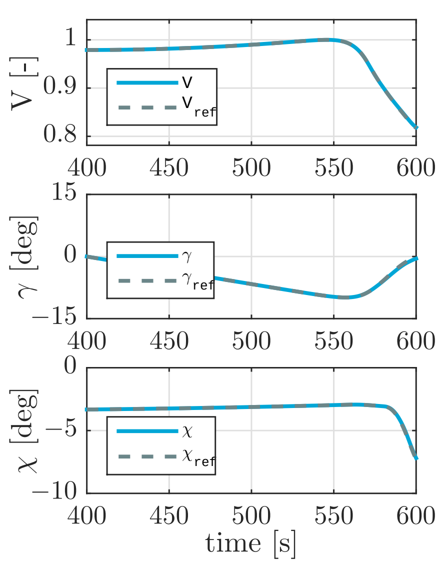
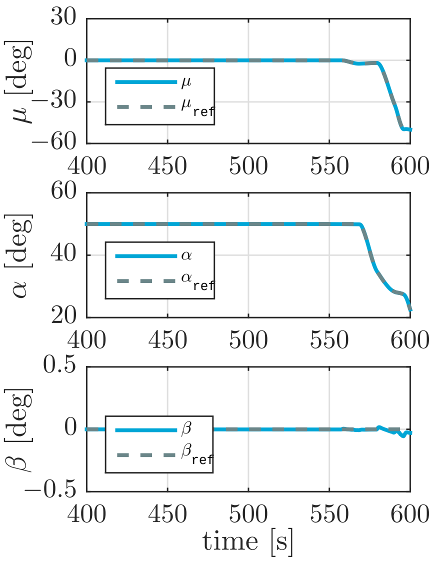

(d) Inner-loop attitude control results, nominal case. Obtained moments required for RCS control in body-frame.

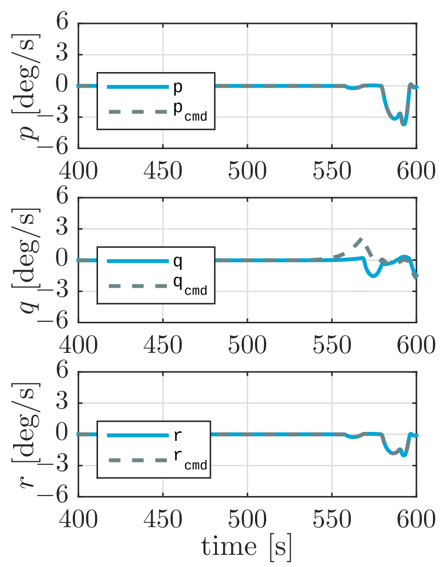
Figure 7: **Phase P5-a** – Descent guidance and control results for the re-entry maneuver using RCS control.



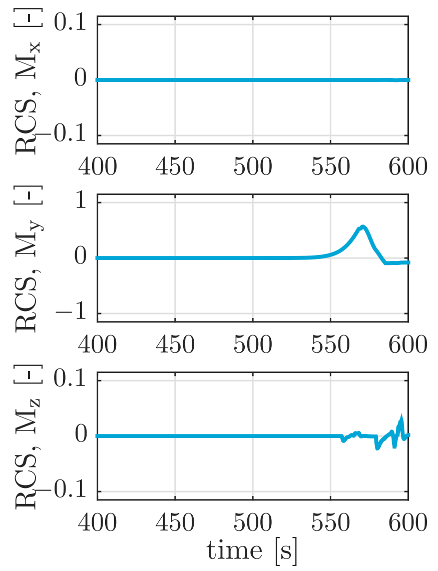
(a) Kinematic position trajectory results, uncontrolled version, nominal case. Reference values from trajectory optimization.



(b) Aerodynamic angles – attitude control results, nominal case. Reference values from trajectory optimization.



(c) Outer-loop angular rates guidance and control results, nominal case. Guidance obtained from model inversion, NDI control.



(d) Inner-loop attitude control results, nominal case. Obtained moments required for RCS control in body-frame.

Figure 8: **Phase P5-a** – Descent guidance and control results for the re-entry maneuver using RCS and aerodynamic surface control.

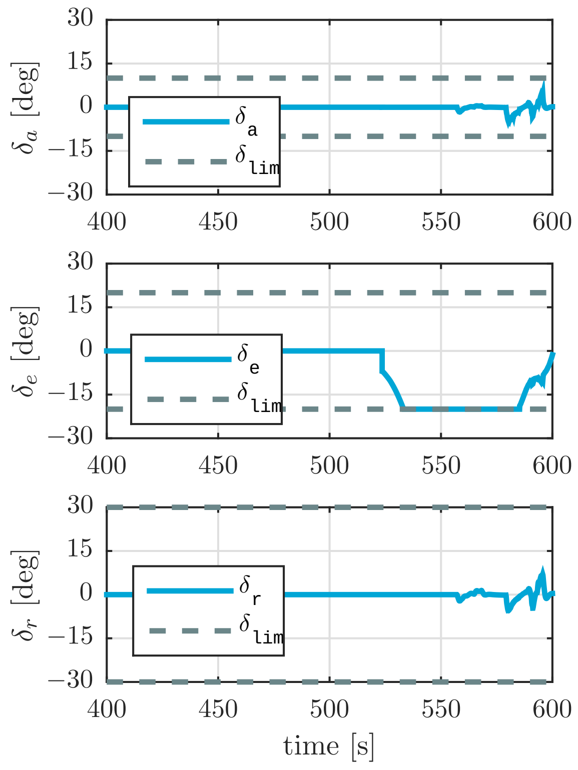


Figure 9: Aerodynamic control surface deflections used in combination with RCS control during re-entry at Phase P5-a.

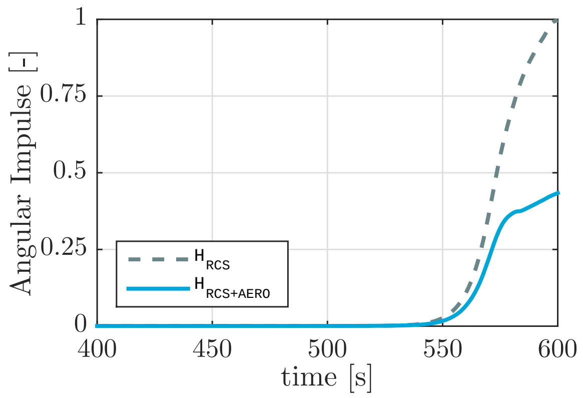
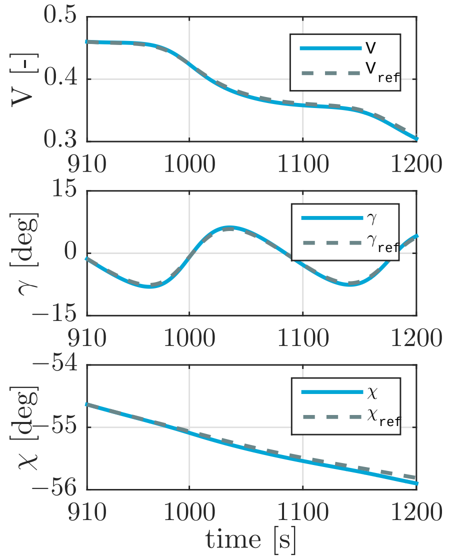
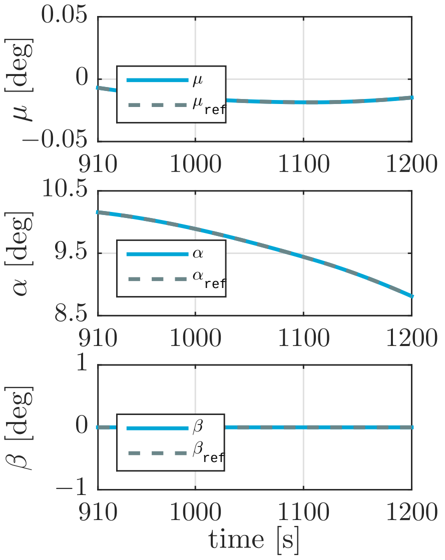


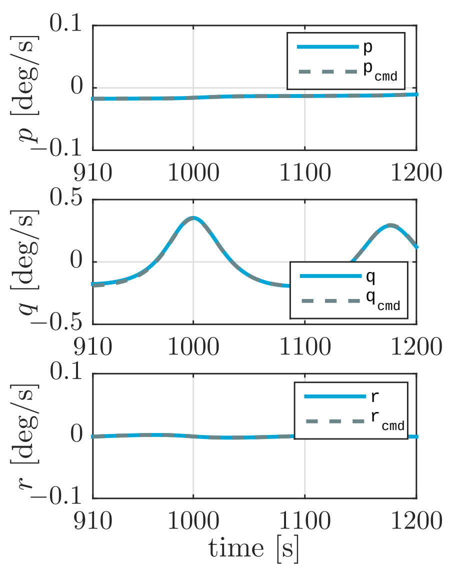
Figure 10: **Phase P5-a** – Impact on the resulting angular impulse required for RCS when using with and without aerodynamic controls.



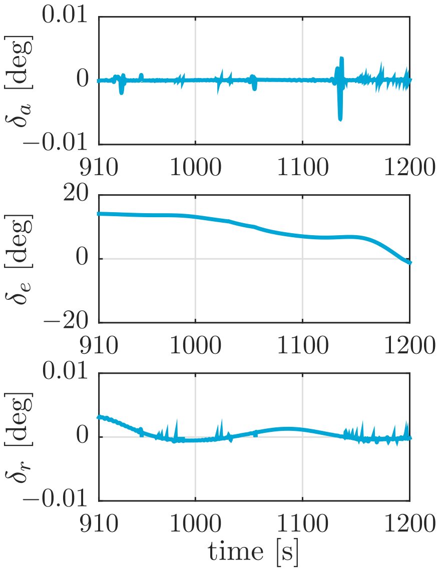
(a) Kinematic position trajectory results, uncontrolled version, nominal case. Reference values from trajectory optimization.



(b) Aerodynamic angles – attitude control results, nominal case. Reference values from trajectory optimization.



(c) Outer-loop angular rates guidance and control results, nominal case. Guidance obtained from model inversion, NDI control.



(d) Inner-loop attitude control results, nominal case. Obtained aerodynamic control surface deflections.

Figure 11: **Phase P5-c** – Descent guidance and control results for final approach of the descent.

References

[1] M. Sippel, C. Manfletti, H. Burkhardt, Longterm/strategic Scenario for Reusable Booster Stages, Acta Astronautica 58 (4) (2006) 209–221. doi:DOI:10.1016/j.actaastro.2005.09.012.

[2] M. Sippel, Promising roadmap alternatives for the Spaceliner, Acta Astronautica 66 (11-12) (2010) 1652–1658. doi:doi:10.1016/j.actaastro.2010.01.020.

[3] M. Sippel, E. Dumont, I. Dietlein, Investigations of Future Expendable Launcher Options, in: proceedings of the 63rd International Astronautical Congress IAC, 2011.

[4] E. Dumont, L. Bussler, S. Karl, D. Porrmann, C. Manfletti, D. Krause, J. Klevanski, V. Clark, O. Bozic, G. Poppe, Analysis of the Ariane 62/64 and Vega-C Launcher Family, Tech. Rep. DLR-IB-RY-HB-2016-63, DLR German Aerospace Center (2016).

[5] E. Dumont, S. Stappert, J. Wilken, Evaluation of a Future Reusable Ariane VTOL Booster, in: proceedings of the 68th International Astronautical Congress IAC, no. IAC-17-D.2.4.3, 2017.

[6] M. Sippel, S. Stappert, L. Bussler, Systematic Assessment of a Reusable First-stage Return Options, in: proceedings of the 68th International Astronautical Congress IAC, no. IAC-17-D2.4.4, 2017.

[7] M. Sippel, C. Valluchi, L. Bussler, A. Kopp, N. Garbers, S. Stappert, S. Krummen, J. Wilken, Spaceliner Concept as Catalyst for Advanced Hypersonic Vehicles Research, in: proceedings of the 7th European Conference for Aeronautics and Space Sciences (EUCASS), 2017.

[8] S. Stappert, J. Wilken, M. Sippel, E. Dumont, Assessment of a European Reusable VTVL Booster Stage, in: SART-TBC, 2018.

[9] L. Bussler, M. Sippel, Comparison of Return Options for Reusable First Stages, in: 21st AIAA International Space Planes and Hypersonics Technologies Conference, 2017.

[10] A. Kopp, M. Sippel, S. Stappert, N. Darkow, J. Gerstmann, S. Krause, D. Stefaniak, M. Beerhorst, T. Thiele, A. Gülhan, R. Kronen, K. Schnepper, L. E. Briese, J. Riccius, Forschung an Systemen und Technologien für wiederverwendbare Raumtransportsysteme im DLR-Projekt AKIRA, in: Deutscher Luft- und Raumfahrtkongress, 2017.

[11] A. Kopp, Das AURORA-R2 RLV-Konzept, in: Deutscher Luft- und Raumfahrtkongress, 2017.

[12] S. Stappert, E. Dumont, Reusability of Launcher Vehicles by the Method of SpaceX, Tech. Rep. SARTTN007/2016, DLR German Aerospace Center (2016).

[13] Patentschrift (Patent Specification), Verfahren zum bergen einer stufe eines mehrstufigen raumtransportsystems (released 2003).

[14] M. Sippel, J. Klevanski, A. van Foreest, A. Gülhan, B. Esser, M. Kuhn, The Spaceliner Concept and its Aerothermodynamic Challenges, in: proceedings of the 1st ARA-Days, 2006.

[15] O. von Stryk, R. Bulirsch, Direct and Indirect Methods for Trajectory Optimization, Annals of Operations Research 37 (2) (1992) 357 – 373. doi:doi:10.1007/BF02071065.

[16] P. Lu, Optimal Feedback Control Laws Using Nonlinear Programming, Journal of Optimization Theory and Applications 71 (3).

[17] John T. Betts, Survey of Numerical Methods for Trajectory Optimization, Journal of Guidance, Control, and Dynamics 21 (2) (1998) 193 – 207.

[18] John T. Betts, Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, SIAM, 2010.

[19] K. Schnepper, Trajektorienoptimierung in MOPS - Das Paket trajOpt Version 1.0, Tech. rep., DLR German Aerospace Center (2014).

[20] H.-D. Joos, MOPS - Multi-Objective Parameter Synthesis, Tech. Rep. DLR-IB-SR-OP-2016-128, DLR German Aerospace Center (2016).

[21] P. Boggs, J. Tolle, Sequential Quadratic Programming, Acta Numerica 4 (1995) 1–51. doi:10.1017/S0962492900002518.

[22] J. J. Slotine, W. Li, Applied Nonlinear Control, Prentice Hall Inc, 1990.

[23] A. Isidori, Nonlinear Control Systems, 3rd Edition, Springer, 1985.

[24] H. Nijmeijer, A. van der Schaft, Nonlinear Dynamical Control Systems, Springer, 1990.

[25] G. H. Looye, An Integrated Approach to Aircraft Modelling and Flight Control Law Design, PhD thesis, Delft University of Technology, Faculty of Aerospace Engineering, 2008.

[26] G. Looye, Design of Robust Autopilot Control Laws with Nonlinear Dynamic Inversion, Automatisierungstechnik 49 (12) (2001) 523–531.

[27] T. J. Lombaerts, H. O. Huisman, Q. P. Chu, J. A. Mulder, D. A. Joosten, Flight Control Reconfiguration based on Online Physical Model Identification and Nonlinear Dynamic Inversion, in: proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit, 2008.

[28] J. Reiner, G. J. Balas, W. L. Garrard, Flight Control Design Using Robust Dynamic Inversion and Time-scale Separation automatica, Automatica 32 (11) (1996) 1493–1504.

[29] P. R. Smith, A Simplified Approach to Nonlinear Dynamic Inversion Based Flight Control, in: proceedings of the AIAA Atmospheric Flight Mechanics Conference, 1998.

[30] S. Sieberling, Q. P. Chu, J. A. Mulder, Robust Flight Control Using Incremental Nonlinear Dynamic Inversion and Angular Acceleration Prediction, Journal of Guidance, Control and Dynamics 33 (6) (2010) 1732–1742.

[31] P. R. Smith, A. Berry, Flight Test Experience of a Nonlinear Dynamic Inversion Control Law on the VAAC Harrier, in: proceedings of the AIAA Atmospheric Flight Mechanics Conference, 2000.

[32] B. J. Bacon, A. J. Ostroff, Reconfigurable Flight Control using Nonlinear Dynamic Inversion with a Special Accelerometer Implementation, in: proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit, 2000.

[33] B. J. Bacon, A. J. Ostroff, S. M. Joshi, Nonlinear Dynamic Inversion Reconfigurable Controller utilizing a Fault-tolerant Accelerometer Approach, Tech. rep., NASA Langley Research Center (2000).

[34] B. J. Bacon, A. J. Ostroff, S. M. Joshi, Reconfigurable NDI Controller using Inertial Sensor Failure Detection & Isolation, IEEE Transactions on Aerospace and Electronic Systems 37 (2001) 1373–1383.

[35] A. J. Roenneke, K. H. Well, Nonlinear Flight Control for a High-lift Rentry Vehicle, in: proceedings of the AIAA Guidance, Navigation, and Control Conference, 1995.

[36] R. R. da Costa, Q. P. Chu, J. A. Mulder, Reentry Flight Controller Design Using Nonlinear Dynamic Inversion, Journal of Spacecraft and Rockets 40 (2003) 64–71.

[37] S. Juliana, Re-entry Flight Clearance, PhD thesis, Delft University of Technology, Faculty of Aerospace Engineering, 2006.

[38] P. Acquatella B., W. Falkena, E.-J. van Kampen, Q. P. Chu, Robust Nonlinear Spacecraft Attitude Control Using Incremental Nonlinear Dynamic Inversion, in: proceedings of the AIAA Guidance, Navigation, and Control Conference, 2012.

[39] MATLAB, Version R2016b, Natick, Massachusetts, United States, The Mathworks, Inc., 2016.

[40] D. Kraft, A Software Package for Sequential Quadratic Programming, Tech. Rep. DFVLR-FB-88-28, Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt (DFVLR) (1988).

[41] Lawson, C.L. and Hanson, R.J., Solving Least Squares Problems, Prentice-Hall, Englewood Cliffs, 1974.

[42] K. Schittkowski, The nonlinear programming method of Wilson, Han and Powell. Part 2: An efficient implementation with linear least squares subproblems, Numerische Mathematik 38 (1981) 115–127.

[43] L. E. Briese, K. Schnepper, P. Acquatella B., Advanced Modeling and Trajectory Optimization Framework for Reusable Launch Vehicles, in: Proceedings of the 2018 IEEE Aerospace Conference, 2018.

[44] L. E. Briese, K. Schnepper, P. Acquatella B., Multidisciplinary Modeling and Simulation Framework for Reusable Launch Vehicle System Dynamics and Control, Acta Astronautica (in press)`doi:https://doi.org/10.1016/j.actaastro.2019.08.022`.

[45] S. E. Mattson, H. Elmqvist, M. Otter, Physical System Modeling with Modelica, in: Control Engineering Practice, 1998, pp. 501–510.

[46] H. Elmqvist, S. E. Mattsson, M. Otter, Modelica - An International Effort to Design an Object-Oriented Modeling Language, in: Summer Computer Simulation Conference, 2003, pp. 333–339.

[47] M. Tiller, Introduction to Physical Modeling with Modelica, The Springer International Series in Engineering and Computer Science, 2001.

[48] M. Otter, H. Olsson, New features in Modelica 2.0., in: Proceedings of the 2rd International Modelica Conference, 2002.

[49] P. Fritzson, Principles of Object-Oriented Modeling and Simulation with Modelica 2.1., John Wiley & Sons, 2004.

[50] Functional Mock-up Interface (FMI), 2011, `https://fmi-standard.org/`, accessed: 2019-09-30.

[51] Q. P. Chu, Spacecraft Attitude Control Systems, Lecture notes, Delft University of Technology, Faculty of Aerospace Engineering, 2010.

[52] Q. P. Chu, Advanced Flight Control, Lecture notes, Delft University of Technology, Faculty of Aerospace Engineering, 2010.

[53] H. K. Khalil, Nonlinear Systems, 3rd Edition, Prentice Hall, 2002.

[54] F. O. Eke, Dynamics of Variable Mass Systems, Tech. Rep. CR-1998-208246, NASA (1999).

[55] M. J. Reiner, J. Bals, Nonlinear Inverse Models for the Control of Satellites with Flexible Structures, in: Proceedings of the 10th International Modelica Conference, 2014.

[56] P. Acquatella B., Launch Vehicle Multibody Dynamics Modeling Framework for Preliminary Design Studies, in: 6th International Conference on Astrodynamics Tools and Techniques, ICATT, 2016.

[57] P. Acquatella B., W. van Ekeren, Q.P. Chu, PI(D) tuning for Flight Control Systems via Incremental Nonlinear Dynamic Inversion, IFAC-PapersOnLine 50 (1) (2017) 8175 – 8180, 20th IFAC World Congress. `doi:https://doi.org/10.1016/j.ifacol.2017.08.1265`.

[58] S. Juliana, Q. P. Chu, J. A. Mulder, T. J. van Baten, The Analytical Derivation of Nonlinear Dynamic Inversion Control for Parametric Uncertain Systems, in: proceedings of the AIAA Guidance, Navigation, and Control Conference, 2005.

[59] S. Juliana, Q. P. Chu, J. A. Mulder, T. J. van Baten, Flight Control of Atmospheric Re-entry Vehicle with Nonlinear Dynamic Inversion, in: proceedings of the AIAA Guidance, Navigation, and Control Conference, 2006.

[60] E. Mooij, The Motion of a Vehicle in a Planetary Atmosphere, Tech. Rep. LR-768, Delft University of Technology, Faculty of Aerospace Engineering (1997).