DESIGN AND DEVELOPMENT OF *R2M2* – A MULTI-PHYSICS MODELING TOOL FOR REUSABLE LAUNCH VEHICLES

Björn Gäßler*[(1)], Lâle Evrim Briese[(1)], Paul Acquatella[(1)], Pedro Simplício[(2)], Samir Bennani[(3)], Massimo Casasco[(3)]

[(1)] *DLR, German Aerospace Center, Oberpfaffenhofen D-82234, Germany*
[(2)] *Aurora Technology for the European Space Agency, Noordwijk NL-2200, The Netherlands*
[(3)] *European Space Agency ESA/ESTEC, Noordwijk NL-2200, The Netherlands*

ABSTRACT

Designing a guidance and control (G&C) system for vertical take-off/vertical landing (VTVL) reusable launch vehicles is a complex and time-consuming process that requires comprehensive simulators considering the relevant system dynamics for various mission scenarios. The *Rapid Reusable Launcher Simulation via Multi-physics Modelling (R2M2)* tool aims to streamline this process. Implemented in Simulink/Simscape, R2M2 adopts a multi-physics/multi-domain modeling approach, making it adaptable to simulate a wide range of mission scenarios, including different vehicle configurations and environmental conditions. The simulation model is automatically assembled by a dedicated routine based on user-defined parameters. This paper describes the design and implementation of the R2M2 tool, its functionalities, advantages, and disadvantages, and showcases its capabilities for the ascent of a two-stage launch vehicle as a benchmark scenario.

1 INTRODUCTION

Guidance and control (G&C) design for reusable launch vehicles (RLV) is a complex and demanding process due to competing mission requirements and strong couplings between different disciplines. The G&C system generates and tracks a trajectory to fulfill mission objectives using various types of actuators which introduce additional dynamics and interactions that are complicated to model, for instance due to the couplings with aerodynamics, changes in center of mass (CoM), and the time-varying mechanical properties caused by propellant burning. External perturbations and other subsystem dynamics as well as potential failure cases must also be considered. The current industrial approach involves separate teams using different tools and (commonly non-interacting) models while a multi-physics/multi-domain modeling approach is believed to be more convenient for considering the relevant interactions between models in a single integrated tool. This motivates the development of a simulator for multi-actuated vertical take-off/vertical landing (VTVL) vehicles based on MathWorks Simscape and Simscape Multibody[1].

Previous efforts on launch vehicle dynamics modeling known in the literature have been carried out, among others, by NASA, the European Space Agency (ESA), and the German Aerospace Center (DLR); early efforts by NASA led to the development of the *Program to Optimize Simulated Trajectories (POST)* and its follow-up *POST2* for generalized trajectory simulation, guidance design, and optimization [1]. NASA also developed the tools *Treetops*, *CLVTOPS*, and *FRACTAL* for multiple applications, such as flexible body dynamic simulations, separation clearance, as well as autopilot

*Corresponding author, email: `bjoern.gaessler@dlr.de`

[1] In the following, symbols for registered trademarks® are omitted for all MathWorks products for readability reasons; these include MathWorks®, MATLAB® and Simulink® and all related products.

stability analysis and design [2–4]. Early efforts carried out by ESA were to develop the *Dynamic and Control Analysis Package (DCAP)* for modeling, simulating, and analyzing the dynamics and control performances of coupled rigid and flexible structural systems [5]. These efforts led to other launcher multi-body dynamics simulators based on DCAP as a backbone [6] and to a more recent collaboration with Astos Solutions including the commercial software ASTOS [7]. Moreover, ESA studied launcher dynamical simulations using SimMechanics (now Simscape) for an application considering worst-case analysis of separation dynamics [8]. DLR uses the acausal, declarative, and object-oriented modeling language MODELICA [9] for launcher dynamics modeling, simulation, and assessment. In this context, DLR developed a dedicated framework which is especially suitable for launch vehicle preliminary design studies including multi-disciplinary modeling and multi-objective optimization of preliminary guidance and control system design for future satellites and launchers [10–14]. More recently, ESA and DLR have considered applying these developments using Simscape for separation dynamics and RLV modeling [15, 16] where the latter is further expanded within this work. In particular, the accurate modeling of time-varying propellant mass dynamics is crucial for evaluating launcher performance during preliminary design studies [17]. In most studies, variable-mass dynamics are simplified, neglecting fundamental dynamical effects due to mass variability. Recent studies have addressed these effects and variable-mass formulations are derived analytically depending on the propellant/burn profile and geometry [18–20].
The objective of this paper is to describe the design and implementation of the *'Rapid Reusable Launcher Simulation via Multi-physics Modelling (R2M2)'* tool. The paper is structured as follows: First, in Section 2 the tool's basic architectural design is described, followed by a detailed description of the implementation of the main subsystems in Simulink and Simscape Multibody. Section 3 addresses challenges and drawbacks that arise when using Simscape for flight mechanics modeling, particularly with regard to variable-mass dynamics and physical effects specific to aerospace applications. Section 4 explains how to use the tool, including the basic workflow, automatic model assembly and Monte Carlo routines with uncertain parameters. Section 5 presents the setup and results obtained for a launch vehicle ascent benchmark. Finally, R2M2 is aimed to be used in future space transportation G&C activities and therefore is distributable and adaptable [2] .

2 MODELING AND IMPLEMENTATION

The implementation of the R2M2 tool is based on MATLAB's physical modeling toolboxes Simscape and Simscape Multibody. Simscape allows the modeling of multi-domain physical systems within the Simulink environment and provides fundamental building blocks. In contrast to signal-flow-based modeling in Simulink, Simscape is based on acausal signal flows and physical connections [21]. Simscape Multibody [22] extends Simscape to multi-body simulations of three-dimensional mechanical systems. The Mechanics Explorer is a visualization tool built into the Simscape Multibody toolbox that allows the user to explore the models visually.
R2M2 is implemented in MATLAB Release 2021b. Since Simscape Multibody does not offer the possibility to access its code nor the acausal ports, users are not able to implement custom components or building blocks in the physical multi-body domain. To implement custom functionalities, Simscape sensors are employed to measure the required parameters and these 'physical' quantities are transformed to Simulink signals using Simscape's *PS-Simulink Converter*. Custom components are implemented with MATLAB (embedded) functions or using standard Simulink blocks. Their outputs are transferred back to the physical domain using the *Simulink-PS Converter* and introduced to the multi-body system as external forces and torques, applying Simscape's *External Force and Torque* block. It is also possible to pass these outputs to Simscape joints as actuation motion variables.

2.1 Basic Architectural Design

Figure 1 (left) shows the high-level structure of the R2M2 simulator while the right-hand side displays the subsystem structure of the tool. Its high-level structure is divided into two main subsystems,

[2]To request access, please contact ESA technical officer Pedro Simplício (Pedro.Simplicio@ext.esa.int).

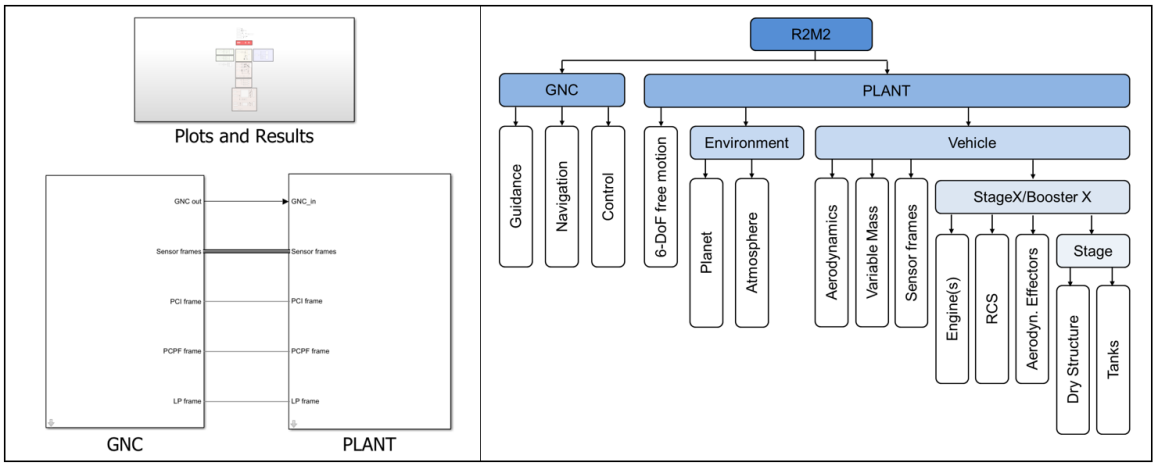

Figure 1: R2M2 simulator high level structure (left) and subsystem structure (right)

GNC and PLANT, plus a subsystem (*Plots and Results*) where the user is meant to prepare simulation data for visualization and post-processing. The PLANT subsystem contains the implementation of all physical effects. This includes the vehicle translational and rotational dynamics, launch vehicle components that are hardware in a real-world system such as vehicle structures, tanks and actuators, environmental effects such as gravitational and aerodynamic forces and moments as well as variable-mass effects and all their interactions. The subsequent sections will provide a detailed description of the PLANT subsystem implementation which is the main focus of R2M2. The GNC subsystem provides the framework for all Guidance, Navigation and Control implementations. This includes sensor models, algorithms for state estimation, trajectory planning and controller commands for trajectory tracking.

2.2 Environment Model

The environment model in R2M2 depicts three essential functionalities to simulate the motion of a launch vehicle or planetary lander in the vicinity of a central body. It provides relevant planet reference frames and contains the implementation of a gravity model and an optional atmosphere model.

The first reference frame is the Planet-Centered Inertial (PCI) frame, defined as an inertial reference frame with the planet's center as its origin. For Earth, the x–axis points towards the vernal equinox, the z–axis points North, and the y–axis completes the right hand system. The PCI frame is implemented using the Simscape *World Frame* block.

The Planet-Centered Planet-Fixed (PCPF) frame shares the same origin with the PCI frame and is fixed to the central planet, following its rotational motion. For Earth, the PCPF frame's x–axis passes through the prime meridian, and the z–axis points North. Within R2M2, the PCPF frame is defined as a pure rotation around the PCI frame's z–axis, neglecting precession and nutation motion. Simscape's *Revolute Joint* implements this z–axis rotation with the current rotation angle, angular rate, and zero angular acceleration as inputs for the joint's motion. Figure 2 shows the implementation of the PCPF frame in Simscape.

The gravity model adopted in the R2M2 simulator considers a 2nd-order spherical harmonic geopotential model including zonal harmonics [23]. Higher order gravitational perturbations were neglected due to their low impact on vehicle dynamics when considering Earth as central body. The gravitational acceleration can be calculated as $\boldsymbol{g} = \boldsymbol{g}_{\text{sph}} + \boldsymbol{g}_{\text{J2}} + \boldsymbol{g}_{\text{J3}}$ where the gravitational acceleration of the spherical reference planet $\boldsymbol{g}_{\text{sph}}$ and the perturbing $\boldsymbol{g}_{\text{J2}}$ and $\boldsymbol{g}_{\text{J3}}$ accelerations for the planet deviating from a perfectly spherical shape are implemented as in [15, 23]. These gravitational accelerations are computed using embedded MATLAB functions and resolved in PCI coordinates. They are then incorporated into the simulation model using Simscape's *Mechanism Configuration* block with time-varying gravity settings. As shown in Figure 3, this gravity implementation method restricts

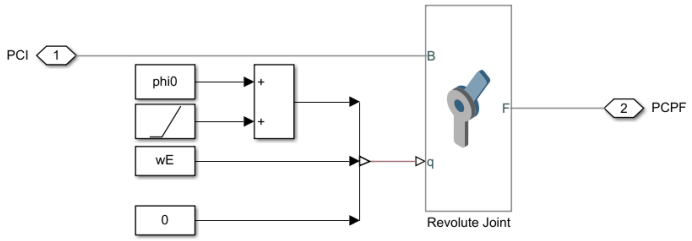

Figure 2: Implementation of the Planet-Centered Planet-Fixed (PCPF) reference frame in Simscape

the simulation model to a single-body simulation since the distance used to calculate the gravitational acceleration is measured between only one body and the PCI frame. The model can thereby be assembled of several bodies that have to be connected rigidly or by joints. This implementation offers the benefit that variable-mass effects do not need be considered explicitly, while perturbing accelerations from non-spherical planet shapes are taken into account.

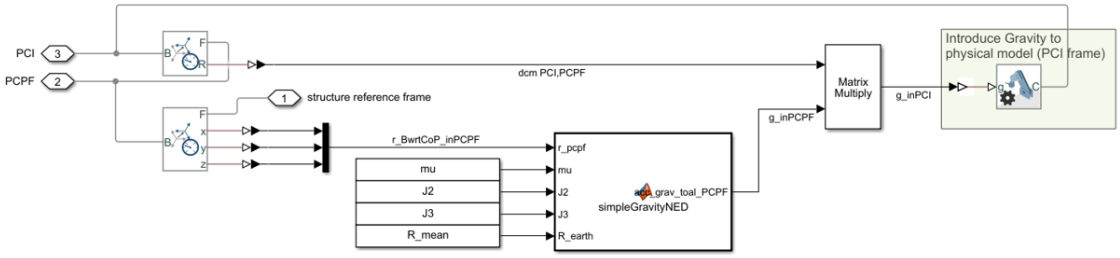

Figure 3: Implementation of the gravity model in Simscape

The atmosphere model adopted for the R2M2 tool corresponds to the 1976 Committee on Extension to the Standard Atmosphere (COESA) model [24] [25]. Atmospheric properties, in particular air density ρ, static pressure P and speed of sound a, are provided as look-up tables with respect to the vehicle's altitude h. To determine the vehicle's altitude, the body frame position with respect to PCPF frame resolved in PCPF coordinates is measured using Simscape's *Transform Sensor* and *Inertia Sensor*. This position is then used to determine the geodetic longitude, latitude and altitude of the vehicle. The conversion is not straight-forward but an analytical solution is described in [23]. Wind is included as a velocity field described by North, East and Down components with respect to the vehicle's altitude. The R2M2 tool has one representative wind profile implemented while additional samples can be added by the user.

2.3 Vehicle Dynamics

To model the launch vehicle dynamics, Simscape's *Cartesian-* and *Spherical Joint* blocks are used, as depicted in Figure 4. These joints specify the motion between the connected reference frames. The PCI frame, described in Section 2.2, was chosen as the reference frame within the R2M2 tool. The Cartesian Joint allows for three translational, and the Spherical Joint for three rotational degrees of freedom (DoF). Together, these two joints enable a 6-DoF free motion of the vehicle around the PCI frame. Additionally, these joints contain the definition of initial values as *State Targets*. The definition of the translational initial values is not as straight-forward as expected, as further explained in Section 3.1. Dynamics are governed by external accelerations acting on the vehicle, which were implemented in the environment, aerodynamics, and actuator models.

Figure 4: Implementation of launch vehicle dynamics using *Cartesian-* and *Spherical Joints*

2.4 Aerodynamics

Aerodynamic forces and moments generated by the launch vehicle's main body have a significant influence on the vehicle's dynamics. They mainly depend on the vehicle's aerodynamic states and shape. Figure 5 shows the general approach for the determination of the aerodynamic forces and moments in the R2M2 tool.

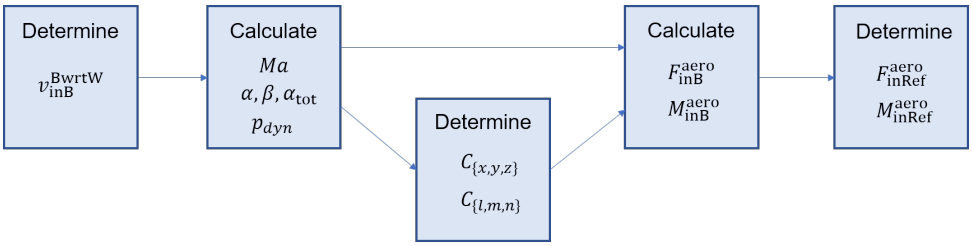

Figure 5: General approach for aerodynamics implementation

First, the aerodynamic velocity $v_{\text{B}}^{\text{B/W}}$ is determined which describes the velocity of the vehicle (body frame 'B') with respect to the atmosphere and includes the impact of wind:

$$\boldsymbol{v}_{\text{B}}^{\text{B/W}} = \boldsymbol{v}_{\text{B}}^{\text{B/PCPF}} - T_{\text{B,PCPF}} \cdot \boldsymbol{v}_{\text{PCPF}}^{\text{Wind}} \tag{1}$$

The wind velocity $\boldsymbol{v}_{\text{PCPF}}^{\text{Wind}}$ is given by the atmosphere model and Simscape's *Transform Sensor* is applied to measure the velocity of the vehicle with respect to the PCPF frame $\boldsymbol{v}_{\text{B}}^{\text{B/PCPF}}$. Measuring this quantity is not as straight-forward as expected and requires an additional coordinate transformation and velocity compensation since the *Transform Sensor* is not capable of providing the pure translational velocity of the body frame. This problem is described in detail in Section 3.2. With the velocity $v_{\text{B}}^{\text{B/W}}$ given, the vehicle's aerodynamic properties, in particular angle of attack α, angle of sideslip β, total angle of attack α_{tot}, Mach number Ma and dynamic pressure p_{dyn}, can be calculated as

$$\alpha = \text{atan2}(w, u) \tag{2a}$$

$$\beta = \text{asin}\left(v/\left|\boldsymbol{v}_{\text{B}}^{\text{B/W}}\right|\right) \tag{2b}$$

$$\alpha_{\text{tot}} = \text{acos}\left(\cos(\alpha) \cdot \cos(\beta)\right) \tag{2c}$$

$$Ma = \frac{\left|\boldsymbol{v}_{\text{B}}^{\text{B/W}}\right|}{a} \tag{3a}$$

$$p_{\text{dyn}} = \frac{1}{2}\rho\left|\boldsymbol{v}_{\text{B}}^{\text{B/W}}\right|^2 \tag{3b}$$

with the speed of sound a and air density ρ determined by the atmosphere model and u,v,w being the velocity's individual components of $\boldsymbol{v}_{\text{B}}^{\text{B/W}}$. In a next step the aerodynamic force and moment coefficients $c_{\text{x,y,z}}$ and $c_{\text{l,m,n}}$ resolved in body frame coordinates are determined from interpolation using 2-dimensional look-up tables as follows:

$$c_{\text{x}} = c_{\text{x}}(Ma, \alpha_{\text{tot}}) \tag{4a}$$

$$c_{\text{y}} = c_{\text{y}}(Ma, \beta) \tag{4b}$$

$$c_{\text{z}} = c_{\text{z}}(Ma, \alpha) \tag{4c}$$

$$c_{\text{l}} = c_{\text{l}}(Ma, \alpha_{\text{tot}}) \tag{5a}$$

$$c_{\text{m}} = c_{\text{m}}(Ma, \alpha) \tag{5b}$$

$$c_{\text{n}} = c_{\text{n}}(Ma, \beta) \tag{5c}$$

A database of aerodynamic coefficients based on [26] is included in the R2M2 tool. This database contains one set of coefficients for a full vehicle in ascent and a second set of coefficients for a generic first stage in descent. Using these coefficients, the aerodynamic forces and moments resolved in body-coordinates can be calculated as

$$\boldsymbol{F}_{\mathrm{B}}^{\mathrm{aero}} = -p_{\mathrm{dyn}} \cdot A_{\mathrm{ref}} \cdot \left[c_{\mathrm{x}}\ c_{\mathrm{y}}\ c_{\mathrm{z}}\right]^T \qquad (6a) \qquad \boldsymbol{M}_{\mathrm{B}}^{\mathrm{aero}} = p_{\mathrm{dyn}} \cdot A_{\mathrm{ref}} \cdot L_{\mathrm{ref}} \cdot \left[c_{\mathrm{l}}\ c_{\mathrm{m}}\ c_{\mathrm{n}}\right]^T \qquad (6b)$$

with the reference area A_{ref} and reference length L_{ref}. Before the aerodynamic forces and moments can be applied to the physical system using Simscape's *External Force and Torque* block, the aerodynamic moment has to be compensated for the unphysical moment that results when the aerodynamic force is applied to a reference point which is not the CoM. A detailed description of this problem can be found in Section 3.3. The dependency of the aerodynamic coefficients on the center of pressure is included implicitly in the structure of the aerodynamic database.

2.5 Structure, Tanks and MCI Properties

The implementation of the mechanical setup can be roughly divided into three main categories. The first category is the dry structure with the implementation of all mechanical components except for the actuated/movable parts and 'wet masses'. The second category considers the variable-mass elements, which mainly contain the 'wet masses' such as propellant, fuel, and oxidizer. The last category are the actuators representing the movable part of the vehicle which are mainly used for control purposes such as the aerodynamic control surfaces or engines capable of thrust vector control (TVC).
Before describing the implementation of the dry structure and the variable mass components, the concept of an assembly frame needs to be defined. Physical modeling in general, and the R2M2 tool in particular, enables the definition of the mass, center of mass, and moment of inertia of individual vehicle components, instead of calculating and defining the overall MCI properties a-priori. Therefore, individual stage and submodel MCI properties can be specified. Simscape then internally calculates the properties of the entire vehicle. This requires the definition of the submodels' positions within the vehicle with respect to an assembly frame. This assembly frame is oriented the same way as the vehicle's body frame, with its origin freely chosen by the user. It is recommended to use the first stage's lower end or the first stage's engine gimbal frame as the assembly frame. However, other prominent structure-fixed points, such as the nose, are also conceivable.

2.5.1 Dry Structure

The dry structure is composed of all non-actuated mechanical components that have a fixed mass. This includes the main structure, payloads, empty tanks, onboard instrumentation, and Reaction Control System (RCS) thrusters, among others. To implement the dry structure in Simscape, two native blocks are used: a *Cylindrical Solid* and an *Inertia* block. The former contains the geometrical parameters of the stage for visualization purposes, while the latter represents a mass element with fixed inertial properties to define the stage's mass and inertia. This subdivision is necessary to accurately define the moment of inertia with respect to the stage's center of mass.

2.5.2 Variable-Mass Elements

A detailed description of the implementation and investigation of variable-mass systems in Simscape can be found in [16]. In the context of launch vehicles, the most significant variable-mass elements are fuel, oxidizer, and propellant. These substances are burned in a combustion chamber and ejected from rocket engines through their nozzles, depleting the tanks. For this study, the focus was on cylindrical variable-mass bodies, as most rocket solid boosters and fuel/oxidizer tanks have a cylindrical shape. Additionally, five different *burn types* were considered, as shown in Figure 6. These burn types can be broadly classified by their application for solid boosters or liquid fuel/oxidizer tanks. *End Burn* and *Centrifugal Burn* are simplified models of the most common burn types in solid boosters, while *Uniform Burn* and *Inverse End Burn* types can be used to model liquid fuel/oxidizer tanks where the engines are pump- or pressure-fed.

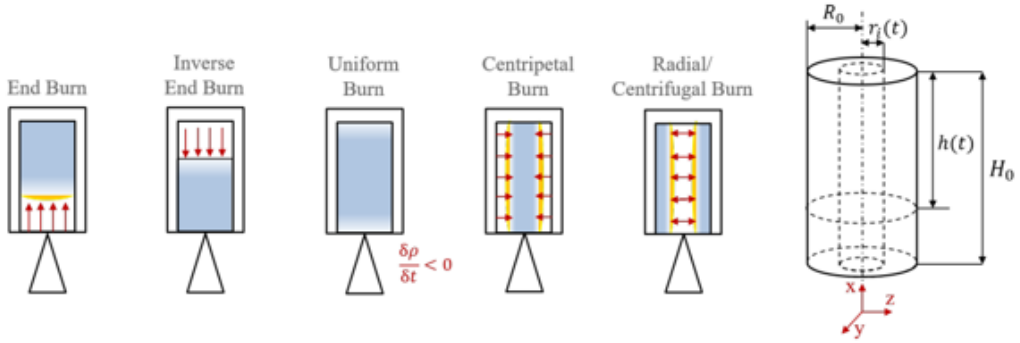

Figure 6: Different burn types and geometrical properties

To model the respective burn types, two different types of Simscape variable-mass elements were used, namely the *Variable Cylindrical Solid* and *General Variable-Mass* blocks. The *Variable Cylindrical Solid* block was used to model Uniform burn, End burn, and Centripetal burn profiles. However, for the Centrifugal burn profile, the *General Variable-Mass* block was used since the *Variable Cylindrical Solid* block did not offer enough flexibility. Geometrical inputs to these blocks were analytically calculated using Simulink blocks and MATLAB functions. Afterwards they were converted to physical signals using Simulink-PS Converters.

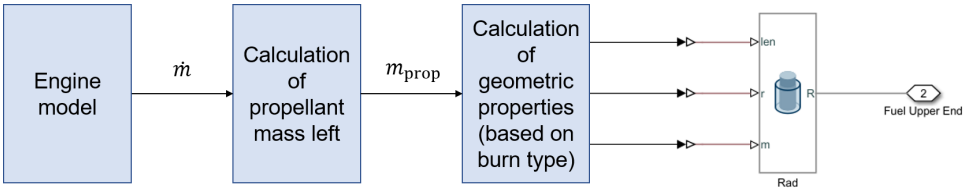

Figure 7: Procedure to calculate variable-mass geometric properties

Figure 7 depicts the implementation of variable-mass elements in the R2M2 tool and the procedure for calculating the required geometric properties. In a first step the propellant mass left within a tank is calculated based on the mass flow provided by the engine model as:

$$m\left(t\right) = m_0 + \int_{t_0}^{t} \dot{m}\ \mathrm{d}\tau \tag{7}$$

with the mass flow being defined as $\dot{m} < 0$ if $m(t) > m_{\text{residual}}$. If the tanks are empty, the mass flow is set to zero. Information about tank depletion is forwarded to the engine model which ensures that the engine's thrust is set to zero if one tank is empty. In case of a liquid engine individual mass flows for fuel and oxidizer have to be calculated from the engine's total mass flow based on the respective oxidizer to fuel mass ration $\gamma_{\text{OF}} = m_{\text{oxidizer}}/m_{\text{fuel}}$ as:

$$\dot{m}_{\text{Fuel}} = \frac{\dot{m}_{\text{total}}}{(1+\gamma)} \tag{8a}$$

$$\dot{m}_{\text{Oxidizer}} = \frac{\gamma \cdot \dot{m}_{\text{total}}}{(1+\gamma)} \tag{8b}$$

Using the left-over mass from Equation 7, the geometrical properties are calculated analytically depending on the respective burn type [16].
In a last step, geometric properties are converted to physical signals using *Simulink-PS Converters* and utilized as inputs for the native Simscape variable-mass elements. Considering the extended equations of translational and rotational motion for variable-mass systems [18]:

$$m(t)\boldsymbol{a}(t) = \boldsymbol{F}_{\text{ext}}(t) + \boldsymbol{F}_{\text{C}}(t) + \boldsymbol{F}_{\text{T}}(t) + \boldsymbol{F}_{\text{L}}(t) \tag{9a}$$

$$\boldsymbol{I}(t)\dot{\boldsymbol{\omega}}(t) + \boldsymbol{\omega}(t) \times \boldsymbol{I}(t)\boldsymbol{\omega}(t) = \boldsymbol{M}_{\text{ext}}(t) + \boldsymbol{M}_{\text{C}}(t) + \boldsymbol{M}_{\text{H}}(t) + \boldsymbol{M}_{\text{T}}(t) \tag{9b}$$

where (dropping the time dependency for readability) m and $\boldsymbol{I}$ are the vehicle's mass and moment of inertia, $\boldsymbol{a}$ is the translational acceleration, $\boldsymbol{\omega}$ and $\dot{\boldsymbol{\omega}}$ are the angular velocity and acceleration, and the rest of variable-mass dynamical effects can be observed as additional forces and moments; in particular, $\boldsymbol{F}_{\mathrm{C}}$ and $\boldsymbol{M}_{\mathrm{C}}$ describe the Coriolis force and moment, $\boldsymbol{F}_{\mathrm{T}}$ and $\boldsymbol{M}_{\mathrm{T}}$ are the thrust force and moment, whereby the thrust force is often already included in the external forces $\boldsymbol{F}_{\mathrm{ext}}$, $\boldsymbol{F}_{\mathrm{L}}$ is a force created when the system's linear momentum changes due to particle motion within the system and $\boldsymbol{M}_{\mathrm{H}}$ is a moment due to the system's decrease in angular momentum inside its boundaries (e.g. due to particle motion within the system). For a detailed description and derivation of these variable-mass dynamical effects, the reader is referred to [16, 18, 27]. Applying reasonable simplifications in the context of launch vehicles [18], the calculation of variable-mass additional forces and moments reduces to analytical equations of the Coriolis forces $\boldsymbol{F}_{\mathrm{C2}}$, a moment du to the change in inertia $\boldsymbol{M}_{\mathrm{Idot}}$ and the jet-damping moment $\boldsymbol{M}_{\mathrm{jet}}$:

$$\boldsymbol{F}_{\mathrm{C2}} = -2\,|\dot{m}|\,\boldsymbol{\omega} \times \begin{bmatrix} -l_{\mathrm{e}} \\ \Delta y \\ \Delta z \end{bmatrix} \tag{10a}$$

$$\boldsymbol{M}_{\mathrm{Idot}} = \frac{\mathrm{d}\boldsymbol{I}}{\mathrm{d}t}\boldsymbol{\omega} \tag{10b}$$

$$\boldsymbol{M}_{\mathrm{jet}} = -\dot{m} \begin{bmatrix} \frac{1}{2}R_{\mathrm{N}}^2 + \Delta y^2 + \Delta z^2 & -l_{\mathrm{e}}\Delta y & -l_{\mathrm{e}}\Delta z \\ -l_{\mathrm{e}}\Delta y & l_{\mathrm{e}}^2 + \frac{1}{4}R_{\mathrm{N}}^2 + \Delta z^2 & -\Delta y \Delta z \\ -l_{\mathrm{e}}\Delta z & -\Delta y \Delta z & l_{\mathrm{e}}^2 + \frac{1}{4}R_{\mathrm{N}}^2 + \Delta y^2 \end{bmatrix} \boldsymbol{\omega} \tag{10c}$$

with the mass flow $\dot{m}$, change in overall moment of inertia $\frac{\mathrm{d}\boldsymbol{I}}{\mathrm{d}t}$ and the vehicle's angular velocity $\boldsymbol{\omega}$. Detailed steps for the derivation of analytical equations can be found in [16, 18].

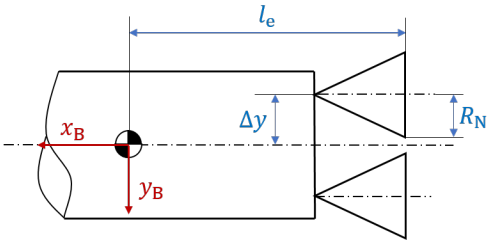

Figure 8: Geometric properties for the calculation of variable-mass dynamical effects

The geometric properties required in these equations are shown in Figure 8. In the R2M2 simulator the implementation of variable-mass dynamical effects takes place 'where they happen', meaning that $\boldsymbol{F}_{\mathrm{C2}}$ and $\boldsymbol{M}_{\mathrm{jet}}$ are implemented in each engine model since they depend on the respective mass flow over the engine's nozzle while the calculation of $\boldsymbol{M}_{\mathrm{Idot}}$ is implemented only once for the complete vehicle due to its dependence on the change in overall MoI. Figure 9 shows the implementation of $\boldsymbol{M}_{\mathrm{Idot}}$ in Simscape. A *Transform Sensor* and *Inertia Sensor* are used to determine the parameters required. $\boldsymbol{M}_{\mathrm{Idot}}$ is then calculated in a MATLAB function and introduced to the physical system as input to an *External Force and Torque* block. The option to break the algebraic loop and its implication to the simulation are discussed in Section 3.5.

2.6 Actuators

Within the R2M2 framework three different actuator types are implemented. This includes the main engine(s) with the possibility of Thrust Vector Control (TVC), aerodynamic control surfaces and a Reaction Control System (RCS).

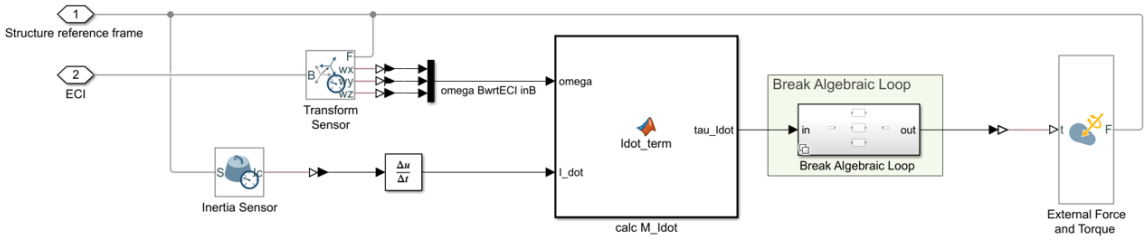

Figure 9: Implementation of M_{Idot} in Simscape

2.6.1 Engine(s)

The most prominent actuator in the context of launch vehicles are the main engine(s), which directly influence the variable-mass properties of the vehicle by consuming fuel/oxidizer or propellant. The R2M2 tool is not limited to one single engine but also allows for clustering of several engines. Each engine model includes four main functionalities:

- Definition of engine frames and MCI properties
- Definition of engine geometrical properties for visualization
- Calculation of thrust force and introduction of this force to the physical environment
- Implementation and actuation of a movable engine gimbal for thrust vector control

MCI properties of the engine are implemented using Simscape's *Inertia* block. The user has to define mass and MoI of the engine with respect to an engine frame located in the engine's CoM. For visualization Simscape's *Revolved Solid* is used visualizing the engine as cone based on its length and radius. Thrust is applied to a pivotable gimbal frame located at the engine pivot point. A simplified thrust model is implemented where the thrust of each engine is calculated as $F_{\text{thr}} = \delta \cdot F_{\text{thr,max}}$ with δ being the throttle factor of the respective engine with $\delta \in [0, 1]$ and $F_{\text{thr,max}}$ is the maximum thrust force the engine is able to produce. Changes in thrust due to variations in ambient pressure are not considered. In real engines, changes in thrust are not immediately realized as they have their own internal dynamics; to account for this, the tool includes an 'Engine Dynamics' block where users can choose between neglecting engine dynamics or modeling them using a transfer function, for instance. Additionally, the tool offers the possibility to inject failure cases. Failure cases are implemented as a derating factor that reduces the realized thrust of an engine by a fixed value as $F_{\text{thr,real}} = \xi \cdot F_{\text{thr}}$ where F_{thr} is the unperturbed thrust and $\xi \in [0, 1]$ is the derating factor with $\xi = 1$ being the unperturbed thrust and $\xi = 0$ resembling a total loss of engine power. The total mass flow of all active engines is required to calculate the geometric properties of the variable-mass elements. Therefore the mass flow of each engine is calculated as

$$\dot{m} = -\frac{F_{\text{thr,applied}}}{g_0 \cdot I_{\text{sp}}} \tag{11}$$

with the standard gravity g_0 and the specific impulse I_{sp}. In case of an engine cluster, individual engine mass flows are combined and forwarded to the respective variable-mass elements. To enable thrust vector control (TVC) Simscape's *Universal Joint* is used which allows for two rotational degrees of freedom. Engine deflections are enabled around the body frame's y- and z-axes. Simscape's *Rigid Transform* blocks are used to align the *Universal Joint's* frame with the corresponding body frame axes. Deflection dynamics are modeled as MATLAB functions whereby the user can decide between immediate deflection realization (no dynamics) or a PT-2 behavior. The latter is recommended for faster simulations since not only angle information but also rates and accelerations are provided to the joint. Figure 10 shows the gimbal and TVC implementation in Simscape.
Moving mass effects such as the 'tail wag dog' effect which occurs when deflecting the main engine do not have to be implemented separately since they are internally computed by Simscape when

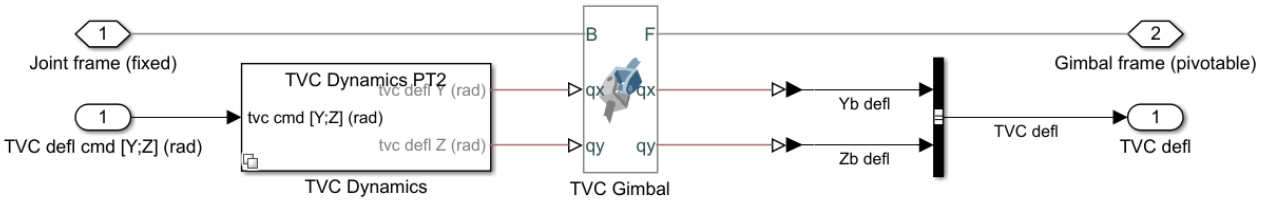

Figure 10: TVC implementation

moving the corresponding bodies with mass and inertia. The gimbal and TVC models offer the possibility to introduce uncertainty and failure cases. Rotational and translational mounting errors can be introduced using Simscape's *Rigid Transform* block to evaluate their effect on control system performance and robustness. TVC actuator failure is implemented as time-triggered jamming which results in the respective gimbal deflection freezing at its current value.
The last functionality implemented in the engine model is the variable-mass Coriolis force $\boldsymbol{F}_{C2}$ and jet-damping moment $\boldsymbol{M}_{jet}$ which are directly related to the engines as described in Section 2.5.2. Similar to the aerodynamic force in Equation 6a, the variable-mass Coriolis force $\boldsymbol{F}_{C2}$ derived in Equation 10a has to be applied to the vehicle's overall CoM. Since no CoM frame is available in Simscape, the variable-mass forces and moments are applied to a structure fixed reference frame and the moment resulting from applying the $\boldsymbol{F}_{C2}$ to a frame out of the CoM has to be counter-acted as will be described in Section 3.3.

2.6.2 Reaction Control System

To control the vehicle's attitude in low atmospheric density phases most launch vehicles are equipped with several small thrusters creating the Reaction Control System (RCS). In R2M2 the RCS implementation contains two main functionalities: (a) definition of thruster geometrical properties for visualization, and (b) calculation of RCS thrust forces and introduction of these forces to the physical environment. Simscape's *Revolved Solid* which determines the geometry based on thruster radius and length is used to visualize the thrusters as cones. Thruster positions and orientations need to be defined by the user, translations and rotations required to correctly place and orient the thrusters and their frames are then automatically computed. Thruster MCI properties are not defined individually but included in the stage dry mass and inertia (Section 2.5). Figure 11 shows an exemplary RCS assembly with 8 thrusters. The RCS frames are always rotated such that their z-axes point outwards.

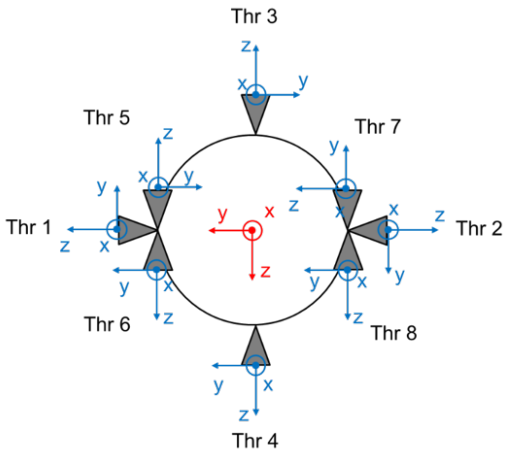

Figure 11: RCS assembly with 8 thrusters

The thrust force of each RCS thruster is calculated as:

$$F_{\text{thr,RCS}} = \begin{cases} F_{\text{thr,RCS,max}} & \text{if } \sigma_{\text{cmd}} = 1 \\ 0 & \text{else} \end{cases} \tag{12}$$

where σ_{cmd} is the controller's on/off-command for the respective RCS thruster. When opening the valves of an RCS thruster it usually takes a short transient in time until a stationary flow through the pipes and nozzle is established. The same applies when closing the valves until no gas is left in the pipes and nozzle and the flow runs dry. The user can therefore decide between an implementation without dynamics or a PT-1 behavior. The tool offers the possibility to inject an RCS failure case which causes the valve to stay closed even when commanded to open. The thrust force is introduced to the physical environment using Simscape's *External Force and Torque* block and applied in negative z-direction of the RCS thruster frame.

2.6.3 Aerodynamic Control Surfaces

Aerodynamic control surfaces, also referred to as fins, are mainly used to ensure controllability of the launch vehicle during phases within the atmosphere. This is of particular interest for reusable launch vehicles during their aerodynamic descent phase. In R2M2 the aerodynamic control surface implementation contains four main functionalities:

- Definition of control surface frames and MCI properties
- Definition of control surface geometrical properties for visualization
- Implementation of a control surface gimbal to enable deflections
- Calculation of the aerodynamic force produces by the control surface and introduction of this force to the physical system

MCI properties of the aerodynamic control surfaces as well as their visualization are implemented using one of Simscape's *Brick Solids*. The user has to specify mass, MoI as well as fin length, height and thickness. The position of each individual control surface has to be defined with respect to a reference frame chosen by the user. Figure 12 summarizes the main frames of the aerodynamic control surface implementation for an assembly of four control surfaces.

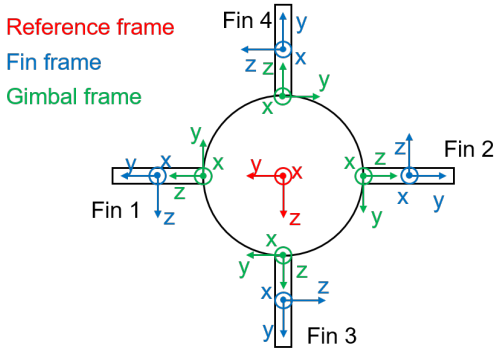

Figure 12: Aerodynamic control surface assembly main frames

The gimbal frame is defined with the z-axis always pointing outwards such that Simscape's *Revolute Joint* can be used to enable one-dimensional rotational motion of each aerodynamic effector. A dynamics model provides the rotation angle of this joint. Additionally, a potential actuator failure is implemented as time-triggered jamming of the aerodynamic control surface.
For all aerodynamic control surfaces the fin frame is aligned such that the y-axis always points outwards with respect to the main body and the x-axis always points upwards if the effector is undeflected.

Within the aerodynamic force implementation only normal forces produced by the aerodynamic control surfaces are considered while axial force contribution are assumed to be negligible due to the small reference area of the control surfaces compared to the main body [26]. Due to the orientation of the fin frame, the normal component of the aerodynamic force acts in negative z-direction.

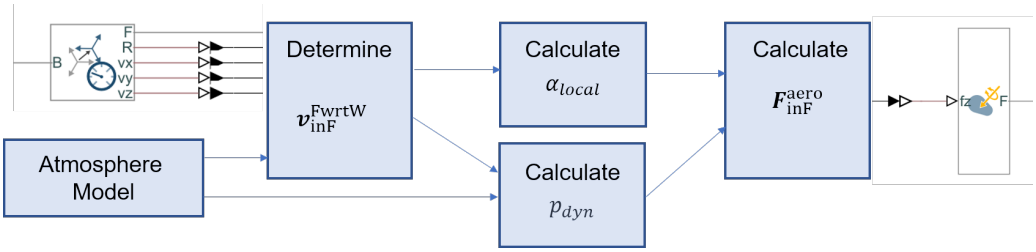

Figure 13: General approach for aerodynamic effector force calculation

The approach for the calculation of the aerodynamic force of an aerodynamic control surface is shown in Figure 13. Based on wind information from the atmosphere model and velocity measurements using Simscape's *Transform Sensor* the relative velocity of the fin with respect to the surrounding air resolved in fin frame coordinates $\boldsymbol{v}_{\mathrm{F}}^{\mathrm{F/W}}$ is calculated as:

$$\boldsymbol{v}_{\mathrm{F}}^{\mathrm{F/W}} = \boldsymbol{v}_{\mathrm{F}}^{\mathrm{F/PCPF}} - \boldsymbol{v}_{\mathrm{F}}^{\mathrm{wind}} = \boldsymbol{T}_{\mathrm{F,PCPF}} \cdot \boldsymbol{v}_{\mathrm{PCPF}}^{\mathrm{F/PCPF}} - \boldsymbol{T}_{\mathrm{F,PCPF}} \cdot \boldsymbol{v}_{\mathrm{PCPF}}^{\mathrm{wind}}. \tag{13}$$

The transformation matrix $\boldsymbol{T}_{\mathrm{F,PCPF}}$ as well as the velocity of the aerodynamic effector with respect to PCPF frame resolved in PCPF coordinates $\boldsymbol{v}_{\mathrm{PCPF}}^{\mathrm{F/PCPF}}$ are determined using Simscape's *Transform Sensor*. In a second step the local dynamic pressure and local angle of attack of the aerodynamic control surface are calculated as:

$$p_{\mathrm{dyn,F}} = \frac{1}{2} \cdot \rho \left| \boldsymbol{v}_{\mathrm{F}}^{\mathrm{F/W}} \right|^2 \tag{14a}$$

$$\alpha_{\mathrm{F}} = \mathrm{atan2}(w_{\mathrm{F}}, u_{\mathrm{F}}) \tag{14b}$$

with u_{F} and w_{F} being the first and third component of the velocity vector $\boldsymbol{v}_{\mathrm{F}}^{\mathrm{F/W}}$ and ρ is the air density. Finally, the aerodynamic force is calculated according to [26, 28] as:

$$F_{\mathrm{aero,F}} = -p_{\mathrm{dyn,F}} \cdot c_{\mathrm{N,max}} \cdot A_{\mathrm{ref,F}} \cdot \sin(\alpha_{\mathrm{F}}) \tag{15}$$

with the aerodynamic effector reference area $A_{\mathrm{ref,F}}$ and the maximum normal force coefficient $c_{\mathrm{N,max}}$. The aerodynamic force is applied in the fin frame's z-direction. Torques resulting from these forces are internally calculated and applied by Simscape.

3 CHALLENGES OF FLIGHT DYNAMICS MODELING IN SIMSCAPE MULTIBODY

During the flight dynamics implementation for the R2M2 simulator several challenges have occurred, whereof the main challenges will be described in the following section including possible workarounds. All these challenges add up to the overall problem of flight dynamics implementation in Simscape, which is described in the end of this section.

3.1 Applying Initial Values

Initial values are defined as *State Targets* in either the *Cartesian Joint* for translation or the *Spherical Joint* for rotation (see Section 2.3). The definition of translational initial values, especially for velocity requires special attention. In contrast to conventional simulators, initial values in Simscape can not be defined with respect to the body's center of mass (CoM) but are defined with respect to the frame connected to the joint. Since Simscape does not provide a generic CoM frame this is usually a structure-fixed reference frame (referred to as assembly frame in R2M2 - see Section 2.5). For the

initial position this means that the offset between assembly frame and CoM has to be explicitly considered. For the same reason, the relative velocity resulting from an initial angular rate and the lever arm between assembly frame and CoM has to be considered when defining the initial translational velocity as follows:

$$\boldsymbol{v}_{\text{init}} = \boldsymbol{v}_{\text{init,CoM}} + \boldsymbol{\omega}_{\text{init}} \times \boldsymbol{r}_{\text{CoM}} \tag{16}$$

where $\boldsymbol{v}_{\text{init}}$ is the initial velocity as applied to the *Cartesian Joint*, $\boldsymbol{v}_{\text{init,CoM}}$ is the initial velocity of the body frame (origin in CoM), $\boldsymbol{\omega}_{\text{init}}$ is the initial angular rate and $\boldsymbol{r}_{\text{CoM}}$ is the initial vector from the assembly frame's origin to the CoM. An initial angular rate $\boldsymbol{\omega}_{\text{init}}$ is almost always present as soon as a rotation of the PCPF frame with respect to the PCI frame is considered. The compensation requires initial knowledge of the overall CoM position which either has to be analytically computed a-priori or measured using Simscape's *Inertia Sensor* in a pre-simulation.

3.2 Measurements of Translational Parameters with respect to the Center of Mass

Simscape does not offer the possibility to directly measure properties with respect to the overall CoM for multi-body models. This is because Simscape's *Transform Sensor* only allows to measure quantities relative to the frames connected to it. Therefore the translational properties have to be compensated for relative velocities and accelerations resulting from the lever arm between CoM and measurement frame in the presence of rotations.

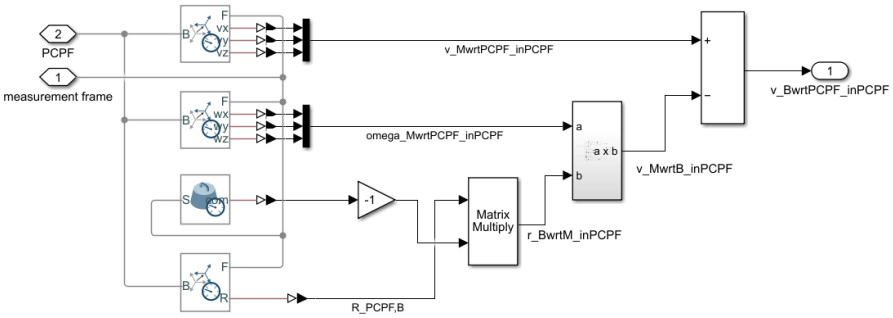

Figure 14: Approach to determine translational velocity resolved in body frame coordinates

Figure 14 shows the approach to determine the translational velocity with respect to the body frame. *Transform Sensors* are used to measure the velocity of the measurement frame 'M' (which is connected to Simscape's *Transform Sensor*) with respect to the PCPF frame $\boldsymbol{v}_{\text{PCPF}}^{\text{M/PCPF}}$ as well as the angular rate $\boldsymbol{\omega}_{\text{PCPF}}^{\text{M/PCPF}}$ and the transformation matrix $\boldsymbol{R}_{\text{PCPF,B}}$ to rotate from body to PCPF frame. In addition, an *Inertia Sensor* measures the CoM offset $\boldsymbol{r}_{\text{B}}^{\text{CoM}}$. The translational velocity with respect to the body frame is then calculated as:

$$\boldsymbol{v}_{\text{PCPF}}^{\text{B/PCPF}} = \boldsymbol{v}_{\text{PCPF}}^{\text{M/PCPF}} - \boldsymbol{\omega}_{\text{PCPF}}^{\text{M/PCPF}} \times \left(\boldsymbol{R}_{\text{PCPF,B}} \cdot \boldsymbol{r}_{\text{B}}^{\text{CoM}}\right) \tag{17}$$

Especially for the aerodynamics model (see Section 2.4) but also for control purposes correct knowledge of translational velocity is essential. The determination of translational accelerations with respect to CoM becomes even more complicated since not only relative accelerations, but also Coriolis- and centripetal accelerations have to be taken into account [29].

3.3 Applying Forces in the Center of Mass

The structure of the aerodynamic model and database requires the aerodynamic force and moment to be applied in the CoM of the vehicle. This is a common approach in flight dynamics modeling. Additionally the equations for variable-mass dynamical effects are derived such that these forces and moments have to be applied to the vehicle's overall CoM (see Section 2.5.2). Simscape does not offer the possibility to apply forces and moments to an overall CoM but only to a frame attached to

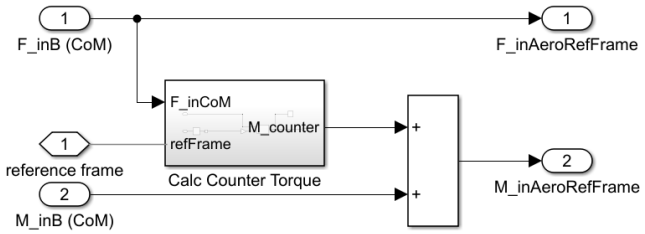

Figure 15: Approach to apply forces to a frame with an origin other than the CoM and to compensate the resulting nonphysical moment

its *External Force and Torque* block. This is fine for moments as long as the attached frame has the same orientation as the body frame. Applying forces to a frame with an origin other than the CoM on the other hand results in the generation of moments due to the lever arm. Since these moments are unintended and nonphysical they need to be compensated for. Figure 15 depicts the implemented approach to apply forces to a frame with an origin other than the CoM while compensating for the resulting non-physical moment. The counter-moment is calculated as:

$$\boldsymbol{M}_{\text{Ref}}^{\text{counter}} = \boldsymbol{F}_{\text{B}} \times \boldsymbol{r}_{\text{Ref}}^{\text{CoM}} \tag{18}$$

where the CoM position $\boldsymbol{r}_{\text{Ref}}^{\text{CoM}}$ is measured using Simscape's *Inertia Sensor*.

3.4 Measurements of Translational Parameters resolved in Body Frame Coordinates

Using Simscape's *Transform Sensor* for relative translational velocity and acceleration measurements is not as straight-forward as anticipated and might result in unexpected measurement. In a first implementation of the R2M2 tool a *Transform Sensor* was used to directly measure the velocity of the body frame with respect to the PCPF frame resolved in body coordinates. Therefore, the sensor was connected as shown in Figure 16 with the sensor's base frame connected to the model's PCPF frame and the sensor's follower frame connected to the model's body frame. The follower frame was chosen as 'Measurement Frame' option within the block.

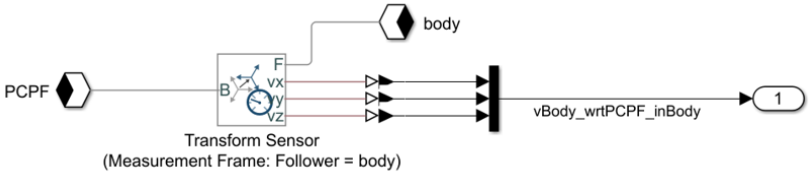

Figure 16: Faulty approach to measure $\boldsymbol{v}_{\text{PCPF}}^{\text{B/PCPF}}$

Using a *Transform Sensor* with these settings resulted in faulty/unexpected velocity measurements and therefore in wrong aerodynamic states that were derived based on this velocity measurement. For investigating the cause of this issue, the velocity of the body frame B with respect to PCI, resolved in PCPF coordinates was decomposed as follows:

$$\boldsymbol{v}_{\text{PCPF}}^{\text{B/PCI}} = \boldsymbol{v}_{\text{PCPF}}^{\text{B/PCPF}} + \boldsymbol{\omega}_{\text{PCPF}}^{\text{PCPF/PCI}} \times \boldsymbol{r}_{\text{PCPF}}^{\text{B/PCPF}} \tag{19}$$

which is a typical approach in flight dynamics modeling. Resolving quantities in body frame, splitting up the angular rate $\boldsymbol{\omega}_{\text{PCPF}}^{\text{PCPF/PCI}}$ in its components with respect to the body frame and exploiting the fact that PCI and PCPF share the same origin (therefore it applies that $\boldsymbol{r}_{\text{PCPF}}^{\text{B/PCPF}} = \boldsymbol{r}_{\text{PCPF}}^{\text{B/PCI}}$) Equation 19 becomes:

$$\boldsymbol{v}_{\text{B}}^{\text{B/PCPF}} = \underbrace{\boldsymbol{v}_{\text{B}}^{\text{B/PCI}} - \boldsymbol{\omega}_{\text{B}}^{\text{B/PCI}} \times \boldsymbol{r}_{\text{B}}^{\text{B/PCI}}}_{\boldsymbol{v}_{\text{out,Sensor (Figure 16)}}} + \underbrace{\boldsymbol{\omega}_{\text{B}}^{\text{B/PCPF}} \times \boldsymbol{r}_{\text{B}}^{\text{B/PCPF}}}_{\boldsymbol{v}_{\text{missing}}}. \tag{20}$$

Comparing the results, it became apparent that the *Transform Sensor* with the settings shown in Figure 16 neglects the second cross term in Equation 20 which caused the unexpected measurement results. This problem was circumvented by determining the velocity $\boldsymbol{v}_{\mathrm{PCPF}}^{\mathrm{B/PCPF}}$ as follows:

$$\boldsymbol{v}_{\mathrm{B}}^{\mathrm{B/ECEF}} = \boldsymbol{R}_{\mathrm{B/ECEF}} \cdot \boldsymbol{v}_{\mathrm{ECEF}}^{\mathrm{B/ECEF}}. \tag{21}$$

Figure 17 shows the respective implementation in Simulink/Simscape resulting in correct measurement results.

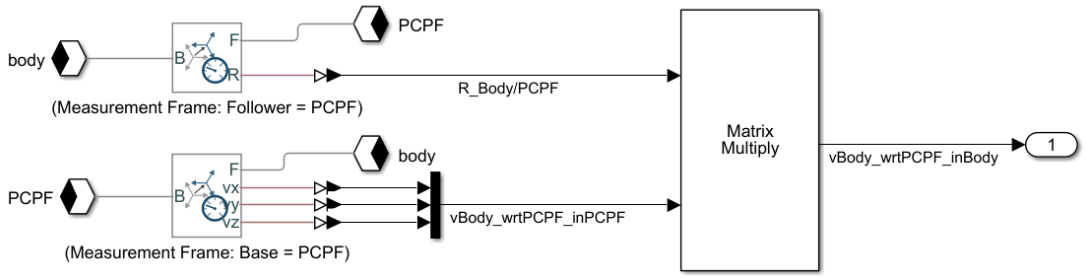

Figure 17: Approach to determine $\boldsymbol{v}_{\mathrm{PCPF}}^{\mathrm{B/PCPF}}$ correctly in Simulink/Simscape

3.5 Algebraic Loops

Simscape encounters algebraic loops, especially when using its *Inertia Sensor*. Algebraic loops either cause the simulation to break or result in significantly increased simulation times, which is particularly unfavorable for large Monte Carlo campaigns. In the R2M2 tool, the user has the option to break algebraic loops, either by applying a *Memory* block or by using a transfer function with a small time constant.

3.6 Summary of Challenges

The implementation of flight dynamics, in particular in the presence of aerodynamic forces and moments, is very challenging and not straight-forward in Simscape. The reason for this lies in the variable-mass nature of the system in combination with aerodynamics and other forces and moments (such as variable-mass dynamical effects). Since a large part of a launch vehicle's mass is fuel and oxidizer the vehicle's MCI properties change significantly during flight. Thereby the change in CoM is of particular importance. During atmospheric flight the launch-vehicle's body creates aerodynamic forces and moments. Calculating these aerodynamic forces and moments requires knowledge about the body frame's translational velocity which cannot be measured directly (see Sections 3.2 and 3.4). Depending on the structure of the aerodynamic database these forces (and moments) are either applied to the CoM of the system or to its center of pressure (CoP), both of which are not fixed and vary over time. Since there is neither the possibility to apply forces directly to a multi-body's CoM frame nor does a 'movable' reference frame exist in Simscape, the aerodynamic forces and moments are applied to a structure-fixed reference frame causing the problems described in Sections 3.3 and 3.5. Since, in contrast to the basic Simscape toolboxes or other available multi-body environments, the code of the Simscape Multibody toolbox is not accessible to the user, investigations as in Section 3.4 and the implementation of individual multi-body blocks are particularly difficult.

4 TOOL WORKFLOW

4.1 General Workflow

The R2M2 tool combines the following main functionalities:

- 6-degree of freedom simulations of VTVL launch vehicles
- Generation of the parameter structure required for these simulations, enabling different vehicle and actuator configurations
- Automatic assembly of a simulation model
- Monte Carlo simulations with perturbed parameters
- Consideration / insertion of actuator failure cases

In the following section the general workflow of the tool is described. The first step is to define the parameters of the simulation model.

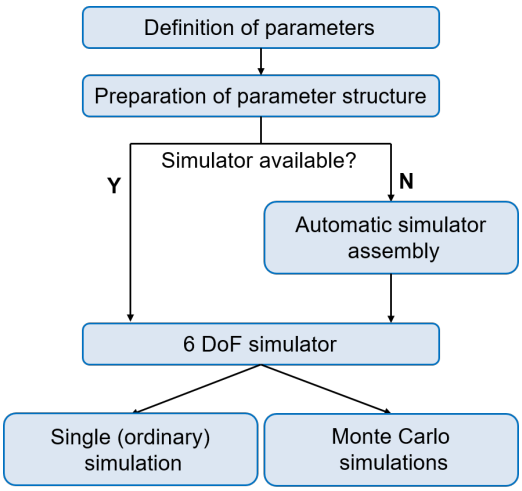

Figure 18: General workflow

Therefore, the user creates new MATLAB files in the existing configurations folders and specifies the parameters according to a predefined structure [30]. When the parameters of all subsystems are defined, the user is supposed to generate a main file in the applications section of the tool to determine the combination of configuration-files to be used. In a next step the basic parameter structure is generated by applying a dedicated script. The basic parameter structure loaded to the MATLAB workspace is essential for using the R2M2 simulator and all its applications. This basic parameter structure consists of four individual structures (config, GNC, PLANT and simulation) which are fundamental for all other applications. After the basic parameter structure is prepared the user can either perform simulations with a pre-existing simulator, let the automatic assembly routine create a new simulator or perform Monte Carlo simulations with perturbed parameters.

4.2 Automatic Model Assembly

The automatic assembly routine aims at simplifying model generation and to reduce the possibility for errors by manual assembly or modification of models. It is based on the basic parameter structure in which the user defines all mechanical properties and parameters needed for the respective simulation. The routine begins by creating a new folder for the simulator and copying a basic simulator model into it. This simulator model already contains the main structure with all basic subsystems but needs to be extended to include the user defined structure, actuators, atmosphere and aerodynamics as well as the GNC system. The automatic assembly routine extends the basic simulator by means of the information saved in the basic parameter structure loaded to the workspace. For every step of the assembly routine the corresponding Simulink/Simscape blocks are added to the basic simulator and the respective parameters are assigned to the subsystem. The automatic assembly script is mainly using the following four MATLAB commands which can be used to manipulate models in Simulink: *add_block()*; *add_line()*; *set_param()*; and *get_param()*. All subsystems that are added to the simulator by the assembly routine are pre-defined in the R2M2 library. It contains subsystems for the

environment, free motion, actuators, aerodynamics, different types of stages, variable-mass effects and simulation-related subsystems. After the simulator is successfully assembled it is automatically saved in the 'simulators'-folder of the tool under a user defined name.

4.3 Monte Carlo Simulation Routine

The automatic assembly, as described in the previous section, results in a ready-to-run Simscape model for VTVL launch vehicles (within Simulink). This model can either be used stand-alone or it can be embedded in a Monte Carlo (MC) routine. For the latter a script is available that covers all steps concerning the definition and execution of MC simulations. The number of MC simulations and the name of the model that should be used for these simulations have to be defined by the user as well as the perturbed parameter names and the corresponding distribution type. Normal distribution and uniform distribution are implemented and the user has to define the respective distribution parameters. The Monte Carlo script then automatically creates perturbed parameter sets and loads the respective parameter set before each Monte Carlo run. It further executes all simulations and saves the results after each simulation for further data processing.

5 LAUNCH VEHICLE ASCENT BENCHMARK

5.1 Benchmark Description

A benchmark scenario based on [26] of a multi-actuated launch vehicle operating on Earth was implemented to verify the simulator and to demonstrate the tool's capabilities.

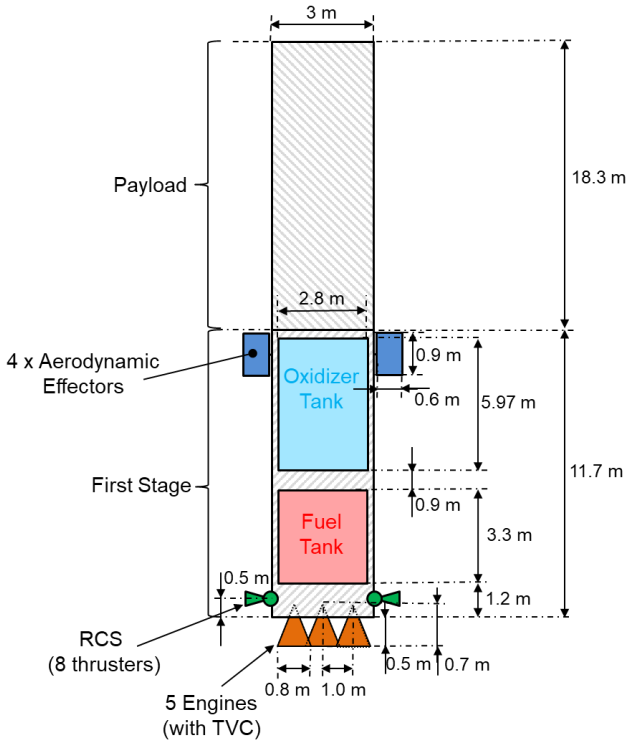

Figure 19: Geometrical properties of the benchmark launch vehicle adapted from [26]

The benchmark is a 2-staged rocket launching from Guiana Space Center in Kourou with a latitude of approx. 5.24 deg, a longitude of approx. -52.78 deg and zero altitude. The benchmark sequence covers the flight phase from launch until separation of the upper stage. As environment model, the implementation described in Section 2.2 was applied with a non-spherical Earth gravitational model and the COESA atmosphere. For the results shown in the next section no wind has been applied,

though the user has the option to use a representative wind profile. Geometrical properties of the vehicle are shown in Figure 19. The second stage - termed 'Payload' - includes every element of the launch vehicle above the first stage. This comprises the actual payload, the second stage's structure, its engine(s), fairing etc. MCI properties of the payload and the dry first stage without variable and actuated/movable masses are summarized in Table 1.

Table 1: Benchmark stage main MCI properties

Stage	**Parameter**	**Value**
First Stage (dry)	Mass	2750 kg
	Longitudinal MoI	≈ 3980 kgm^2
	Lateral MoI	≈ 40270 kgm^2
	CoM (w.r.t. lower end)	$(4.6\ 0\ 0)^T$ m
Payload	Mass	43000 kg
	Longitudinal MoI	$\approx 4.4 \cdot 10^4$ kgm^2
	Lateral MoI	$\approx 3 \cdot 10^6$ kgm^2
	CoM (w.r.t. lower end)	$(12.91\ 0\ 0)^T$ m

Table 2: Propulsion system characteristics

Parameter	**Value**
Engine max. thrust	561.15 kN
Specific Impulse	282 s
Max. TVC deflection	5 deg
Ox./Fuel mass ratio	2.56
Initial fuel mass	25913 kg
Initial oxidizer mass	66337 kg

The vehicle is assumed to be a liquid-propellant rocket with both tanks being modeled as inverse end-burns. The two tanks feed a cluster of five main engines oriented in a cross-shaped configuration. Each engine produces a throttleable thrust with a maximum of 561.15 kN and is capable of thrust vector control (TVC). Table 2 summarizes the main propulsion system characteristics.
In addition to TVC, the launch vehicle is equipped with a reaction control system (RCS) and aerodynamic control surfaces to manipulate its attitude. The RCS consists of 8 thrusters, oriented as shown in Figure 11 with a thrust of 400 N each. Unfolded aerodynamic control surfaces are very uncommon for launch vehicle in ascent. They are usually folded during the ascent phase to reduce aerodynamic drag and unfold only for descent to control the vehicle during unpowered aerodynamic flight phases. To demonstrate the implementation of aerodynamic control surfaces, four fins, mounted as depicted in Figure 12, were added to the launch vehicle. They have a surface area of 0.54 m^2, a maximum deflection of 35 deg and a maximum aerodynamic force coefficient in normal direction of $c_{\text{N,max}} = 6$. To minimize their impact during ascent, their aerodynamic drag is minimized by controlling their local angles of attack to be zero. The reference profiles for main engine throttle factor, pitch, roll and yaw angle adopted in the benchmark are presented in Figure 20.
The benchmark reference profiles consist of a throttle-factor command that is the same for all five engines as well as attitude roll, pitch and yaw angle commands [26]. After $\approx$ 90 s the engines are continuously throttled down until engine cut-off at 110 s. The simulation ends at $t = 112$ s when the upper stage is separated. The launch vehicle's attitude is commanded as reference pitch, yaw and roll angles in open-loop guidance. Additional translational guidance parameters are not considered. This approach is often applied during the atmospheric ascent of a launch vehicle in order

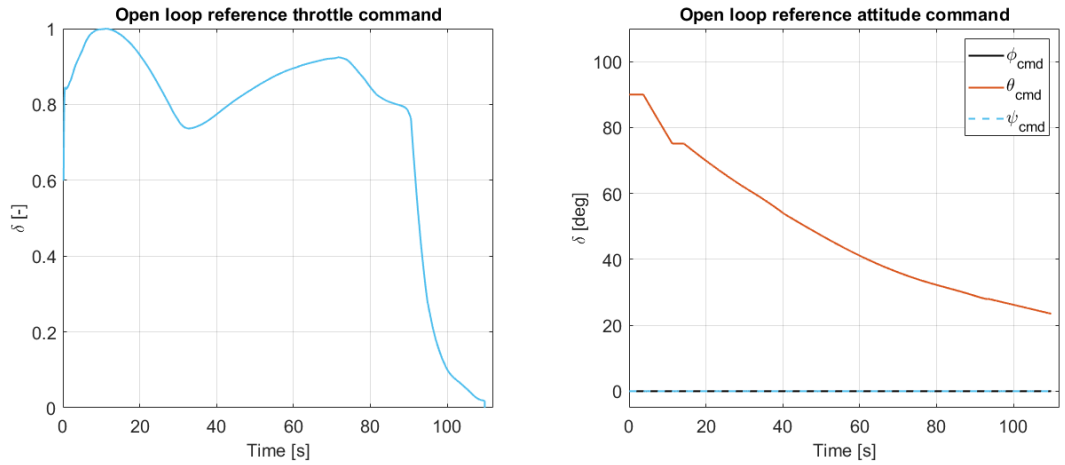

Figure 20: Benchmark reference trajectories

to reduce thermal and mechanical loads. Dispersion resulting from environmental disturbances or system parameter uncertainties are compensated later during exoatmospheric flight. The goal of the implemented control system is therefore to track the pitch angle commands while keeping roll and yaw angle at zero.

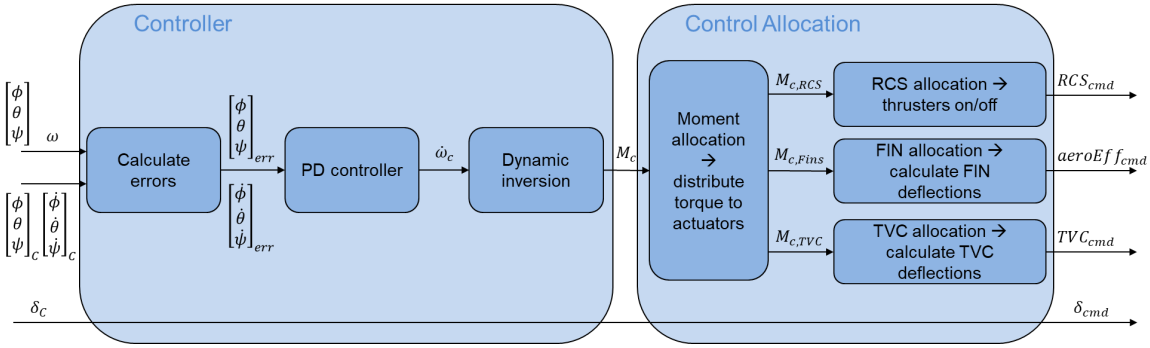

Figure 21: Benchmark control system architecture

Figure 21 shows the benchmark's control system architecture. A PD-controller in combination with a simple dynamic inversion calculates a control moment. The individual controller gains were tuned manually for the nominal trajectory without disturbances. Moment of inertia estimations for the dynamic inversion are scaled linearly with the total vehicle mass. Applying a more sophisticated controller design method would likely result in better tracking performance and higher control system robustness. A control allocation distributes the commanded moments between the three types of actuators and then calculates the required TVC and aerodynamic control surface deflections as well as RCS thruster on/off sequences to generate these moments. Since the controller design was not the main task of the R2M2 project, the control system will not be further discussed here. The current implementation still offers huge potential for improvement and more realistic implementations. However, it is sufficient to stabilize the vehicle on its nominal trajectory and offers a starting point for future implementations and investigations.

5.2 Benchmark Results

Figure 22 shows the visualization of the benchmark launch vehicle in both plots on the left resulting from the automatic assembly routine including all actuators, structure and tanks in Simulink's *Mechanics Explorer*. Based on the user input configurations, the model was correctly assembled by the

automatic assembly routine. On the right different engine, structure and booster configurations are shown, demonstrating the modularity of the R2M2 tool.

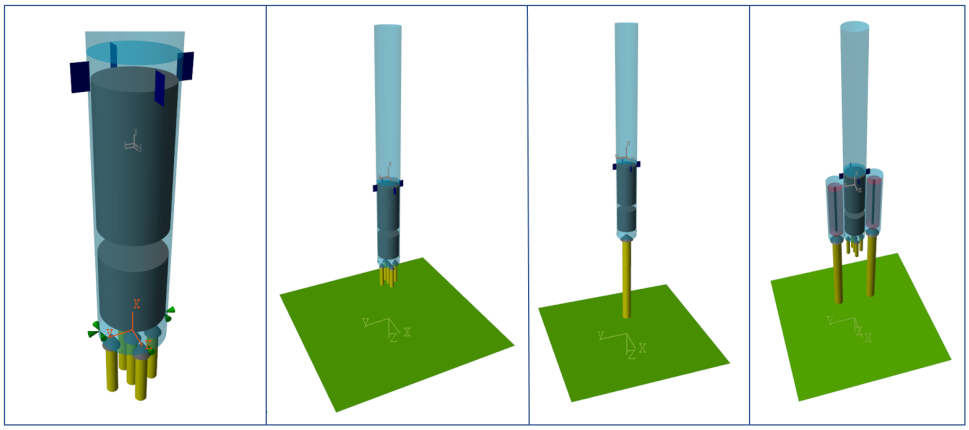

Figure 22: Visualization of the benchmark launch vehicle (left) and different engine, structure and booster configurations (right)

As the launch vehicle burns fuel and oxidizer to produce thrust for its ascent, the overall mass steadily decreases until end of burn at approximately 110 s. Since the thrust is produced by the engines of the first stage, fed by its fuel and oxidizer tanks, the CoM moves upwards in positive body axis x-direction with time as shown in the second plot of Figure 23. At the same time the change in mass and its distribution results in a change in moment of inertia (MoI). For the longitudinal MoI component (around the body's x-axis) this change is almost linear while the lateral MoI components change non-linearly as discussed in more detail in [16].

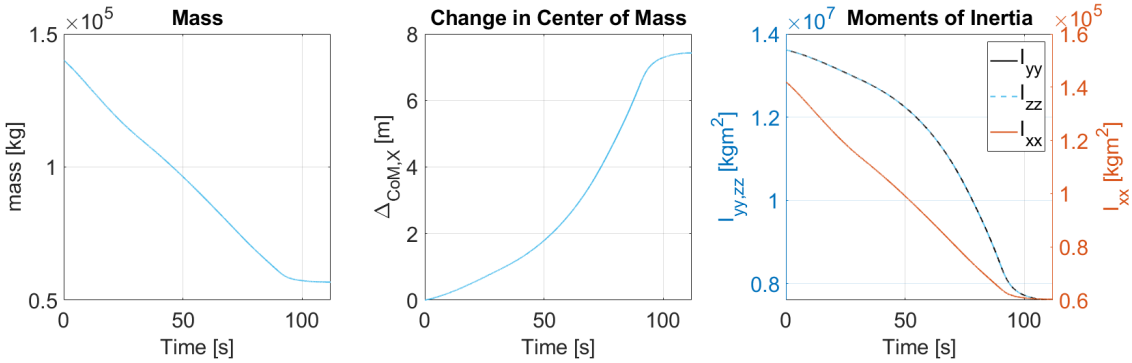

Figure 23: Benchmark MCI properties

The GNC system tracking performance is shown in Figure 24 for the nominal trajectory. Roll and yaw angle errors remain small while the pitch profile is tracked well with minor errors after the pitch-over maneuver showing the controller's capability to stabilize the vehicle and track the commanded trajectory.
The actuator commands to follow the reference trajectory are depicted in Figure 25. TVC deflection commands are exemplary shown for the middle engine. Since the allocation distributes the commanded moments evenly to all engines, TVC deflection commands are the same for all five engines. Especially during the pitch-over maneuver, shortly after lift-off high TVC action δ_Y is required to follow the pitch profile. TVC is the main actuator until the engines are severely throttled down and finally cut off, as TVC control effectiveness then becomes zero and the RCS is applied for vehicle attitude control. This is shown in the third plot of Figure 25 for one of the RCS thrusters which is operated only at the end of the simulation. The aerodynamic control surfaces are commanded such that

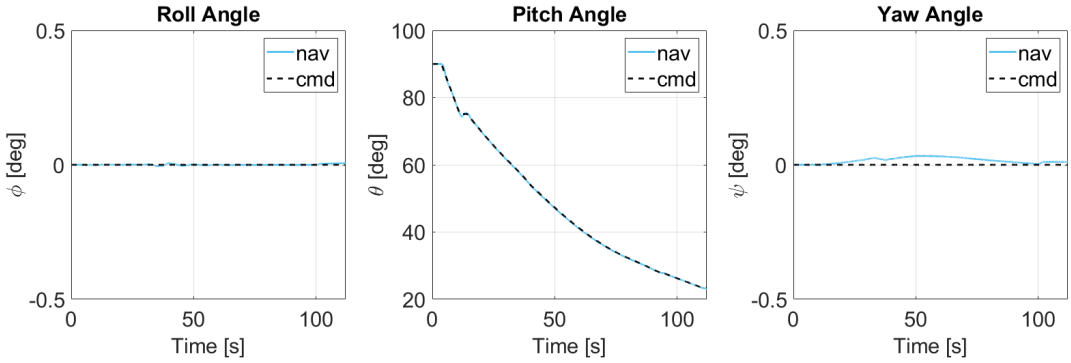

Figure 24: Benchmark control system tracking performance

they produce as little aerodynamic moment as possible. Their deflections are depicted in the middle plot of Figure 25.

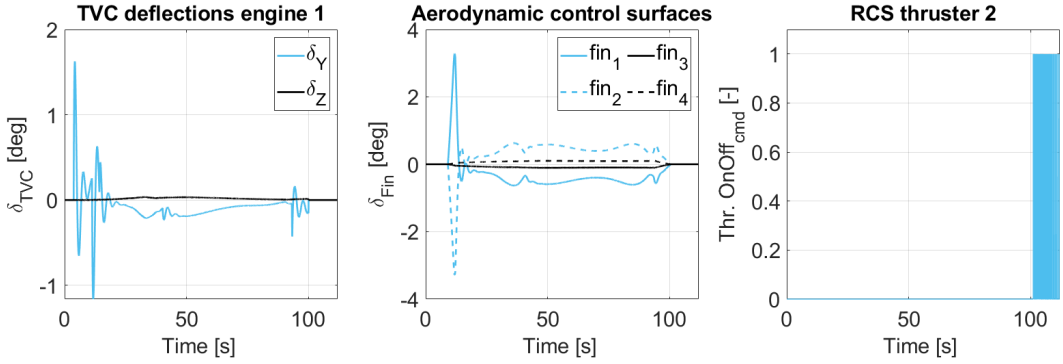

Figure 25: Benchmark actuator actions

The CPU time for a launch vehicle ascent flight of 112 s was approximately 60 s on a regular personal computer which is comparably long and particularly unfavorable for large Monte Carlo campaigns. A comparison with other simulation tools (Modelica, Simulink,etc.) could provide an assessment of Simscape's performance in flight dynamics modeling.

6 CONCLUSIONS AND OUTLOOK

Within the *Rapid Reusable Launcher simulation via Multiphysics Modelling (R2M2)* project, a simulation tool has been developed to streamline the process of setting up simulations of vertical takeoff and landing (VTVL) vehicles using MathWorks physical simulation environment Simscape and Simscape Multibody. The R2M2 tool is highly adaptable and can simulate a wide range of VTVL mission scenarios, including different vehicle and actuator configurations, as well as various environmental conditions. It is capable of considering external influences such as gravitational and aerodynamic effects, as well as internal dynamics from time-varying and moving masses. The tool can also simulate common actuator failure cases and investigate the effects of uncertain parameters.
Simscape's flight dynamics implementation offers great modularity, allowing for automatic model assembly routines based on user input configurations, which can significantly reduce modeling efforts and expedite guidance, navigation, and control developments. Different actuator configurations can be quickly implemented and tested, which is especially beneficial for multi-actuated systems. At the same time, Simscape is compatible and interoperable with all MATLAB & Simulink products, facilitating GNC design and analysis.
However, Simscape's flight dynamics simulations including variable-mass dynamics and all relevant dynamical effects (aerodynamics, environment, etc.) turned out to be relatively slow, which can be

particularly unfavorable for large Monte Carlo campaigns. Furthermore, flight dynamics implementation in Simscape is not as straightforward as expected, requiring several workarounds that overall reduce confidence in the implementation and results. In future work, the performance in terms of CPU time and ease of implementation should be compared to other modeling and simulation environments such as Modelica, Simulink, or Modia. Additionally, physical modeling of flexibilities, slosh, and external perturbations (winds, gusts) shall be addressed to reveal interactions and propagations of these loads in the system. Finally, the development of uncertainty models for robustness analyses and robust synthesis problems shall also be addressed.

ACKNOWLEDGMENTS

This work was financed by the ESA study *R2M2 - Rapid Reusable Launcher Simulation via Multi-Physics Modelling*, contract No. 4000136360/21/NL/CRS. The authors would like to thank Steve Miller from MathWorks® for his technical support. The views expressed in this paper can in no way be taken to reflect the official opinion of the European Space Agency.

REFERENCES

[1] G. L. Bauer, D. E. Cornick, and R. Stevenson, "Capabilities and Applications of the Program to Optimize Simulated Trajectories (POST)," NASA, Tech. Rep. CR-2770, 1977.

[2] R. M. Yates, "TREETOPS Structural Dynamics Controls Simulation System Upgrade," NASA, Tech. Rep. Contract NAS8-40194, Marshall Space Flight Center, August 1996.

[3] J. Orr, M. Johnson, J. Wetherbee, and J. McDuffie, "State Space Implementation of Linear Perturbation Dynamics Equations for Flexible Launch Vehicles," in *AIAA 2009-5962*, August 2009.

[4] J. S. Orr, "Space Launch System Flight Control," in *Aerospace Control and Guidance Systems Committee (ACGSC) Meeting 110*, October 2012.

[5] S. Portigliotti, M. Dumontela, G. Baldesi, and D. Sciacovelli, *DCAP (Dynamics and Control Analysis Package) - an Effective Tool for Modelling and Simulating of Coupled Controlled Rigid Flexible Structure in Space Environment*, 2004.

[6] G. Baldesi and M. Toso, "European Space Agency's Launcher Multibody Dynamics Simulator used for System and Subsystem Level Analyses," *CEAS Space Journal*, vol. 3, pp. 27–48, 2012.

[7] Astos Solutions. Astos User Manual. [Online]. Available: https://www.astos.de/products/astos

[8] P. Simplicio and S. Bennani, "Worst-case Launch Vehicle Stage Separation Analysis," 04 2015, 3rd CEAS Specialist Conference on Guidance, Navigation and Control, CEAS, 2015.

[9] S. E. Mattson, H. Elmqvist, and M. Otter, "Physical System Modeling with Modelica," *Control Engineering Practice*, vol. 6, pp. 501–510, 1998.

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

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

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

[13] L. E. Briese, P. Acquatella B., and K. Schnepper, "Multidisciplinary modeling and simulation framework for launch vehicle system dynamics and control," *Acta Astronautica*, vol. 170, pp. 652–664, 2020.

[14] P. Acquatella B., L. E. Briese, and K. Schnepper, "Guidance command generation and nonlinear dynamic inversion control for reusable launch vehicles," *Acta Astronautica*, vol. 174, pp. 334–346, 2020.

[15] B. Gäßler, P. Acquatella B., C. Yábar Vallès, A. Rinalducci, and S. Bennani, "Preliminary Design and Development of MAST, a Multivehicle Analysis and Separation Tool," 2021, 8th International Conference on Astrodynamics Tools and Techniques (ICATT).

[16] B. Gäßler, L. E. Briese, P. Acquatella B., P. Simplício, S. Bennani, and M. Casasco, "Variable-Mass Dynamics Implementation in Multi-Physics Environment for Reusable Launcher Simulations," 2022, 9th European Conference for Aeronautics and Space Sciences (EUCASS).

[17] F. O. Eke, T. Tran, and J. Sookgaew, "Dynamics of a Spinning Rocket with Internal Mass Flow," *Nonlinear Dynamics and Systems Theory*, vol. 6, no. 2, 2006.

[18] F. O. Eke, "Dynamics of Variable Mass Systems," NASA, Tech. Rep. CR-1998-208246, NASA Ames Research Center, January 1999.

[19] F. O. Eke and T. C. Mao, "On the dynamics of variable mass systems," *International Journal of Mechanical Engineering Education*, 2000.

[20] E. Mooij, "The Motion of a Vehicle in a Planetary Atmosphere," Delft University of Technology, Tech. Rep. LR-768, 1994.

[21] MathWorks. (2023) Simscape Documentation. [Online]. Available: https://mathworks.com/help/physmod/simscape/

[22] MathWorks. (2023) Simscape Multibody Documentation. [Online]. Available: https://mathworks.com/help/physmod/sm/

[23] F. L. Markley and J. L. Crassidis, *Fundamentals of Spacecraft Attitude Determination and Control*. Springer, Space Technology Library, 2014.

[24] Committee on Extension to the Standard Atmosphere, "U.S. Standard Atmosphere," 1976, NASA TM-X-74335.

[25] MathWorks. (2017) MATLAB Aerospace Toolbox Documentation. [Online]. Available: https://www.mathworks.com/help/aerotbx/

[26] P. Simplício, A. Marcos, and S. Bennani, "Reusable Launchers: Development of a Coupled Flight Mechanics, Guidance, and Control Benchmark," *In: Journal of Spacecraft and Rockets*, vol. 57, 2020.

[27] E. Mooij, "The Motion of a Vehicle in a Planetary Atmosphere," Tech. Rep. LR-768, 1997. [Online]. Available: http://resolver.tudelft.nl/uuid:e5fce5a0-7bce-4d8e-8249-e23293edbb555

[28] A. De Oliveira and M. Lavagna, "Reusable Launchers Re-entry Controlled Dynamics Simulator," 2022, 9th European Conference for Aeronautics and Space Sciences (EUCASS).

[29] W. Fichter and W. Grimm, "Flugmechanik," *Institute of Flight Mechanics and Control, University of Stuttgart*, 2009.

[30] B. Gäßler, L. E. Briese, and P. Acquatella B., "TN6 - R2M2 User Manual," German Aerospace Center, DLR-SR, Tech. Rep., January 2023.