GNC 2017 : 10th International ESA Conference on Guidance, Navigation & Control Systems
29 May - 2 June 2017, Salzburg, Austria
European Space Agency

FORCES-RTTO: A TOOL FOR ON-BOARD REAL-TIME AUTONOMOUS TRAJECTORY PLANNING

Juan Jerez[1], Sandro Merkli[1], Samir Bennani[2], Hans Strauch[3]

[1] *Embotech GmbH, Physikstrasse 3, 8092 Zürich, Switzerland, +41 44 632 79 10, jerez@embotech.com, merkli@embotech.com*
[2] *ESA ESTEC, Keplerlaan 1, Noordwijk, The Netherlands, samir.bennani@esa.int*
[3] *Airbus DS GmbH, Airbus Allee, 28361 Bremen, Germany, hans.strauch@airbus.com*

ABSTRACT

The capability for generating re-optimized guidance trajectories on-board in real-time based on current flight conditions promises to improve the system performance, add fault tolerance capabilities, and reduce the mission preparation costs. This project is building a tool to analyze the impact of recent advances in optimization algorithms, embedded software packages, and novel mathematical formulations on the viability of an optimal real-time guidance concept. The developed software tool provides state-of-the-art implementations of custom solvers for two space GNC applications: large angle reorientation maneuvers for an orbiting spacecraft under multiple attitude-constrained zones; and a powered descent and pinpoint landing scenario. Various guidance methods, ranging from a nonlinear programming formulation to a convexified problem description, can be evaluated. A closed-loop validation engine with Airbus simulators will allow to explore the reliability-speed-performance trade-off for the different guidance methods. Initial tests on flight-representative hardware show that the optimal trajectory generation code is fast enough for real-time implementation.

1 INTRODUCTION

The goal of the guidance function is to generate a reference trajectory that can be followed by a low-level controller to meet the mission goals. A generic optimal trajectory generation problem for space flight can be written as:

$$\begin{aligned}
\text{minimize}_{t_f,x,u} \quad & J(x(t),u(t),t) := \int_{t_0}^{t_f} L(x(t),u(t),t)\,dt && && \text{(1a)}\\
\text{subject to} \quad & x(t_0) = x_{\text{current}}, && && \text{(1b)}\\
& x(t_f) = x_{\text{target}}, && && \text{(1c)}\\
& \dot{x}(t) = f(x(t),u(t),t) && \forall t \in [t_0,t_f] && \text{(1d)}\\
& x(t) \in \mathcal{X}(t) && \forall t \in [t_0,t_f] && \text{(1e)}\\
& u(t) \in \mathcal{U}(t) && \forall t \in [t_0,t_f] && \text{(1f)}
\end{aligned}$$

where $x(t)$ and $u(t)$ represent the state and actuator trajectories, t_0 is the current time, and t_f is the maneuver completion time or the flight time. For the guidance function, the computed actuator

trajectory is often discarded, while the state trajectory is given as a reference to a feedback low-level controller that generates the real-time actuation commands. Equations (1b) and (1c) ensure that the mission goals are reached from the current state of the vehicle, while equations (1d), (1e) and (1f) encode the vehicle's dynamical model and limitations and should ensure that the state trajectory can actually be followed by the low-level controller. Finally, (1a) encodes the performance metric that one wishes to improve, e.g. the fuel used, the completion time, the final error, the safety margin, or a trade-off between various objectives.

In the classical *offline* design setting nominal reference trajectories are computed on the ground by solving problem (1) departing from initial conditions ($t_0 = 0$) and assuming no disturbance or uncertainty affecting the vehicle's model, no failure in the vehicle, and no changes in the original mission. These *open-loop* trajectories are optimized once using powerful computers with minor compute time constraints. If the process fails it can be started again with different parameters until a satisfactory trajectory can be found. The results are stored in a table that is read in real-time to provide the reference for the low-level controller [1].

There are several shortcomings with this approach:

- Due to the limited on-board correction capabilities, current offline mission preparations for GNC software, for ascent, endo-atmospheric and exo-atmospheric flight, has to take many possible eventualities into account and, as a result, takes many months and resources [1].
- Despite the time consuming design process, the computed open-loop trajectories assume ideal conditions, hence they are unavoidably suboptimal in real flight conditions. The on-board real-time control system is designed to follow this suboptimal trajectory regardless.
- The current framework does not allow for fast trajectory re-planning in the case of failures, abort scenarios or mission changes. These capabilities are needed for future autonomous vehicles.

In order to cope with operational uncertainty, like atmospheric wind, vehicle component failures, mission changes, or modeling errors, it is necessary to bring optimal guidance trajectory generation from being an off-board design tool to becoming a real-time autonomous *on-board* operational tool. Such capabilities would afford unprecedented benefits:

Operational flexibility: Real-time automated guidance and control would require a drastically reduced amount of labor intensive pre-mission analysis and re-planning whenever the mission profile changes. Last minute changes of the target orbit in time critical missions could be easily handled by the system.

In-flight change-adoption of target orbit: The same features of the closed-loop ascent guidance that provide operational flexibility also result in greatly reduced need for human intervention. As a result, the reoccurring operational cost related to ascent guidance could be reduced to a minimum.

Fault tolerance: An online guidance and control system is by design adaptive in managing unforeseen off-nominal conditions. Future vehicle mission and health management systems for autonomous flight will have to accommodate for system failures within the physical recoverability limits towards a successful mission completion.

Performance enhancement: The guidance and control commands shall be the result of an online re-optimized trajectory at every sampling instant, re-targeting depending on current conditions. In this way optimal flight will be reached in the face of the physical flight conditions instead of an artificial pre-programmed reference trajectory.

Flight envelope extension: Conventional rocket flights use disposable technology. With the upcoming availability of computational power, autonomous boost-back flight schemes, already stemming from the Kistler concept in the 60s, are now considered as a viable usability concept. The concept is being matured by US companies, such as SpaceX, Blue-origin, Masten, Firefly, RocketLab, or United Launch Alliance (Lockheed-Boeing). There are currently no large-scale ESA programs in this direction, with the exception of small initiatives such as this one.

Despite the many potential advantages, on-board trajectory optimization is not yet an available technology due to the difficulty in solving trajectory planning problems (1) on a flight computer in real-time without any human intervention. Finding these optimal trajectories relies on iterative numerical optimization techniques that require substantial computing resources and are not guaranteed to terminate in a fixed time. Hence, for applications that require real-time *closed-loop* guidance capabilities it is common practice to derive explicit guidance rules, which are simple to implement, assuming simplified dynamics and constraints [2, 3, 4, 5].

So far it has not been possible to execute complex iterative optimization algorithms reliably on resource-constrained embedded platforms such as the ones available in a typical flight computer. However, in recent years we have seen incredible progress in novel mathematical formulations and more reliable and efficient optimization algorithms designed for autonomous operations. In addition, many powerful software tools have been developed to significantly ease the process of deploying these algorithms on embedded computers. The tool that we are developing, the FORCES Real-Time Trajectory Optimizer (RTTO), leverages on these recent technological advances to explore the capabilities of the current state-of-the-art.

Our approach towards the goal of creating a universal real-time optimal trajectory generation software has been to start with a benchmark-based study that includes powered descent scenarios and a satellite re-orientation scenario under multiple attitude constraints. In powered descent applications the uncertainty during the non-powered descent phase is often so large that it is impossible to compute a useful pre-programmed trajectory offline. Hence, we will compare against a heuristic online navigation solution. In contrast, the attitude guidance scenarios that we are considering are currently solved using the classical offline trajectory generation framework, hence, we will show how these trajectories can also be computed on-board in real-time.

There are several optimization-based approaches to solving problem (1), which is often *non-convex* and hard to solve. There have been studies proposing to solve the non-convex problem online using a calculus of variations approach [6] and using nonlinear programming. Alternatively, one can apply approximation techniques to instead solve *convex* problems, for which robust and reliable algorithms exist. In recent years, practical convex reformulations of the constrained soft-landing problem [7, 8, 9] and a constrained rendezvous problem [10] have been efficiently solved using second-order cone programming (SOCP) solvers. The choice of solution approach has an impact on the reliability, the computational speed and the overall GNC performance, which will be discussed in the next sections. The goal of FORCES RTTO is to provide various state-of-the-art implementations to enable the explo-

ration of the reliability-speed-performance design space for various space flight scenarios. FORCES RTTO provides a unified framework for validating the various approaches in software-in-the-loop with high fidelity simulators provided by Airbus. There is also the option for hardware-in-the-loop validation on flight representative hardware.
The work has been performed under ESA's Future Launcher Preparatory Program (FLPP3) within the study "On-board Real-time Trajectory Generation". It has close links with two other FLPP3 studies, namely "Upper Stage Attiude Control Development Framework" (USACDF I, II) and "Demonstrator for Technologies Validation" (DTV). The simulator for the constrained attitude guidance task has been taken from USACDF I and the simulator for the constrained powered descend and hopping task used the simulator developed in USACDF II and DTV. The Romanian National Institute for Aerospace Research (INCAS) develops within the FLPP activity "Demonstrator for Technologies Validation" (DTV) a vertical take-off and landing demonstrator (VTVL). It is planned that this VTVL vehicle will be used to test the algorithms described in this paper.

2 GUIDANCE PROBLEMS: MODELS AND OBJECTIVES

This section describes the two GNC problems that are being studied in this activity.

2.1 Constrained powered descent

The situation studied in this benchmark is the following: a space vehicle has entered the atmosphere and, after an initial passive slowdown phase with a parachute, the vehicle needs to be steered towards the landing site and slowed down using thrusters so that it can land gracefully. The goal is to compute guidance trajectories that minimize the fuel needed to complete the maneuver. In cases where it is not possible to reach the target given the available fuel it becomes necessary to compute guidance trajectories that instead minimize the landing error [9]. In addition, it is required that these trajectories satisfy the dynamics of the space vehicle, do not exceed velocity limitations, and are achievable given the available fuel and the characteristics of the thrusters. Furthermore, the computed trajectory must ensure that the planned flight path stays within a safe distance from the ground. In some cases, the attitude of the vehicle must also remain within a mandatory region, for example, when relying on an on-board camera to scan the landing surface.
During the passive slowdown phase errors accumulate with respect to the pre-programmed trajectory due to wind and other atmospheric uncertainty, hence it is necessary to recompute the landing trajectory on-board given the current position and velocity once the parachute gets cut out. It is also necessary to compute this trajectory within one or two seconds to prevent the vehicle from accelerating uncontrollably towards the ground. As a result, current practice is to use simple trajectory generation methods and check the constraints a posteriori. However, this approach has been shown to reduce the re-targeting range of the spacecraft up to one order of magnitude compared to the optimal flight trajectory [11]. Future space applications require pin-point landing accuracy, which cannot be provided with simple methods. For Mars exploration, scientifically valuable sites are often in hazardous terrain, hence only reachable with high landing accuracy capabilities. For reusable launcher technology, it is necessary to be able to land the launch vehicle within a few meters of the target for security and recoverability reasons.

For this initial study we use the simulator for the vertical take-off and landing (VTVL) vehicle developed in the USACDF and DTV projects. The simulator model described in [12] includes attitude dynamics coupled with the translational dynamics affected by aerodynamic drag. Dual quaternions can be used to represent both attitude and translational motion using eight parameters. In [13] the use of dual quaternions is justified for the powered descent and pinpoint landing problem because the problem involves both attitude and position constraints simultaneously. However, for real-time computability reasons it is necessary to use the simplest model that captures the main dynamics of the vehicle. Hence, it is common practice to decouple the translation and rotational dynamics of the landing vehicle, which is often a good approximation in practice because the attitude can be changed relatively quickly compared to translational motion [14]. For the translational motion we use the following point-mass model without aerodynamic effects:

$$\ddot{r}(t) = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{u(t)}{m(t)}\,, \tag{2}$$

$$\dot{m}(t) = -\,\alpha \|u(t)\|_2\,, \tag{3}$$

where g is the gravitational constant of the planet in ms^{-2}, r is the position of the center of mass of the vehicle (with the z-direction representing the vertical axis), u is the applied thrust forces in the reference inertial coordinates, and m is the mass of the vehicle. For our approximation of the mass depletion dynamics we set the constant α to be the fuel flow per thrust of the main engine (in liters per Newton second) times the kerosene density (in kilograms per liter).
The translational motion of the vehicle can be subject to various constraints. Assuming that the target is set to the origin, the following constraint ensures that the planned trajectory stays within an angle θ from the vertical axis. This prevents the vehicle from getting too close to the ground, which would be equivalent to $\theta = 90°$ for flat terrain surfaces.

$$r_z(t)\tan\theta \geq \sqrt{r_x(t)^2 + r_y(t)^2}\,. \tag{4}$$

The mass should always be larger than the dry mass (mass of the vehicle without fuel) plus an additional safety fuel margin left for the final landing mechanism.

$$m(t) \geq m_{\text{dry}} + \epsilon\,. \tag{5}$$

In addition, there are limitations on the lateral and vertical forces exerted by the main engine. These limitations apply in the body frame, hence one needs to consider the current attitude of the vehicle through a direction cosine matrix (DCM) $M(\theta(t))$:

$$-F_{\text{lat,max}} \leq M(\theta(t))u_{x,y}(t) \leq F_{\text{lat,max}}\,, \tag{6}$$

$$F_{\text{vert,min}} \leq M(\theta(t))u_z(t) \leq F_{\text{vert,max}}\,. \tag{7}$$

Notice that $F_{\text{vert,min}}$ is a positive quantity, i.e. the vertical thrust can only be applied upwards and, once the engine is ignited, can only be switched off at the end of maneuver.

Attitude constraints can be indirectly considered by adding constraints on the thrust direction:

$$\mathbf{d}^T \frac{u(t)}{\|u(t)\|_2} \geq \cos\gamma\,, \tag{8}$$

where $\mathbf{d}$ is a direction in the inertial frame and γ is the maximum angle that the thrust is allowed to deviate from that direction. These constraints can also be added to make sure that the vehicle points mostly upwards such that our modeling approximations are more accurate.
For the VTVL Earth simulator, the optimal trajectory generation problem for position guidance can be formally described as:

$$\text{maximize } m(t_f) = \text{minimize} \int_{t_0}^{t_f} \|u(t)\|_2\, dt \tag{9a}$$

$$\text{subject to } r(t_0) = r_{\text{current}},\ \dot{r}(t_0) = \dot{r}_{\text{current}}, m(t_0) = m_{\text{current}}, \tag{9b}$$

$$r_{x,y}(t_f) = 0,\ \dot{r}_{x,y}(t_f) = 0, \tag{9c}$$

$$r_z(t_f) = 3.925,\ \dot{r}_z(t_f) \geq -0.5, \tag{9d}$$

$$\ddot{r}(t) = \begin{bmatrix} 0 \\ 0 \\ -9.81 \end{bmatrix} + \frac{u(t)}{m(t)}, \qquad \forall t \in [t_0, t_f] \tag{9e}$$

$$\dot{m}(t) = -\alpha\|u(t)\|_2, \qquad \forall t \in [t_0, t_f] \tag{9f}$$

$$m(t_f) \geq 23.3, \tag{9g}$$

$$(r_z(t) - 3.925)\tan 85^\circ \geq \sqrt{r_x(t)^2 + r_y(t)^2}, \qquad \forall t \in [t_0, t_f] \tag{9h}$$

$$-65.1257 \leq u_{x,y}(t) \leq 65.1257, \qquad \forall t \in [t_0, t_f] \tag{9i}$$

$$10 \leq u_z(t) \leq 350, \qquad \forall t \in [t_0, t_f] \tag{9j}$$

$$\frac{u_z(t)}{\|u(t)\|_2} \geq \cos 5^\circ, \qquad \forall t \in [t_0, t_f] \tag{9k}$$

Notice that the target altitude is set to be three meters above the ground (plus some additional distance to account for the position of the center of mass). A dedicated landing mechanism performs the landing maneuver for the final meters. Also notice that the thrust limits have been simplified. Since the model does not include attitude information, it is not possible to predict the DCM into the future. Instead we use the thrust angle constraint (9k) to tighten the lateral and vertical thrust limits (9i)–(9j) by considering the expected worst-case attitude.
Figure 1 describes the original guidance and control architecture with decoupled functions for translational motion and attitude. The new position guidance functionality is meant to replace the forward flight and descent guidance modes. The current forward flight mode makes the vehicle fly along the x-direction at a fixed horizontal speed until it reaches a flight envelope from where it attempts to descent towards the target on a 15° angle with a pre-defined velocity profile based on the altitude. It is expected that the new guidance function, based on solving (9) on-board in real-time, will lead to a trajectory that uses less fuel, which directly translates into an enlarged flight envelope in limited fuel situations.

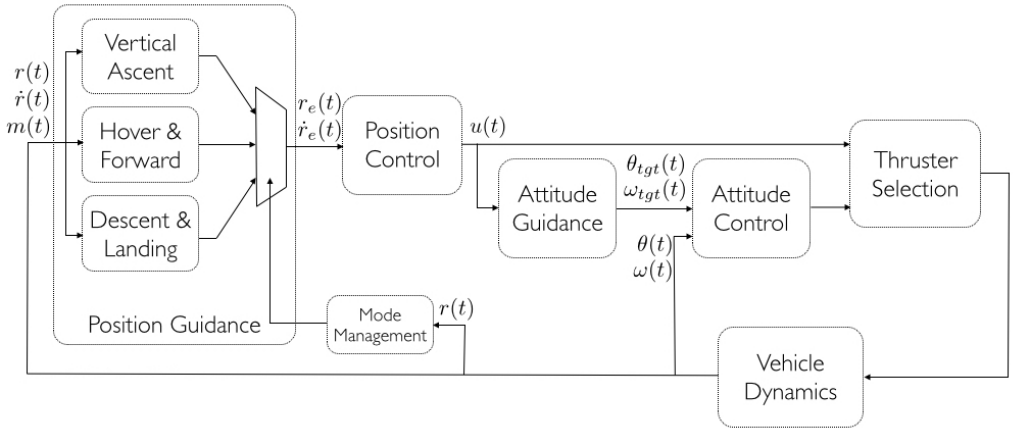

Figure 1: VTVL ascent and powered descent GNC architecture. r_e and $\dot{r}_e$ are the errors with respect to the position and velocity commands given by the guidance function. The Euler angles θ and angular rates ω are controlled by other systems. The position controller, which implements a clipped LQR controller, and the remaining blocks in the GNC architecture have not been modified significantly. Full details on the original control architecture are given in [12].

2.2 Constrained attitude guidance

This benchmark studies the situation of an orbiting spacecraft that needs to change its orientation. The goal is to provide functionality to compute guidance trajectories for this maneuver that either minimize the completion time or the fuel needed, or a weighted combination of both objectives. In addition, it is required that these trajectories satisfy the rotational dynamics of the spacecraft, do not exceed limitations of the torque actuators, and respect the limits on the angular rate bandwidth such that the low-level attitude controller is able to follow the computed guidance trajectory. Furthermore, the computed trajectory must make sure that the spacecraft avoids certain forbidden attitude zones, usually representing the need to avoid that certain sensitive instruments are damaged by the sun's radiation. In some cases, the computed trajectory must also ensure that the spacecraft always keeps its attitude within a mandatory region, representing the need for a communication antenna to maintain communication with other spacecraft or with the Earth at all times.
Currently these guidance trajectories are computed offline in a lengthy, experience-based process. The need to enforce constraints leads to multiple design iterations and exhaustive simulations. The goal is to dramatically simplify this design process and enable just-in-time trajectory re-planning for time-critical missions. In addition, real-time on-board trajectory re-generation capabilities will increase the autonomy of the spacecraft and would increase the chances of automatic recovery in case of noncritical component failures when coupled with a health management system.
The rotational rigid body dynamics of the spacecraft are described by Eulers equations:

$$\begin{aligned} J_x\dot{\omega}_x(t) - (J_y - J_z)\omega_y(t)\omega_z(t) &= u_x(t)\,, \\ J_y\dot{\omega}_y(t) - (J_z - J_x)\omega_z(t)\omega_x(t) &= u_y(t)\,, \\ J_z\dot{\omega}_z(t) - (J_x - J_y)\omega_x(t)\omega_y(t) &= u_z(t)\,, \end{aligned} \tag{10}$$

where J_x, J_y and J_z are the moments of inertia of the vehicle for the principal axis, expressed in kg m^2. ω_x, ω_y and ω_z are the roll, pitch and yaw angular rates, respectively, expressed in rad s^{-1}, and u_x, u_y, and u_z are the control torque inputs in Nm. It is common that the moment of inertia is the same along the pitch and yaw axis, but different along the roll axis, hence the equations remain bilinear with a linear subsystem.
The rigid body rotational dynamics do not include information about the direction the spacecraft is pointing at. We use unit quaternions to represent attitude, which are defined as

$$q(t) := \begin{bmatrix} \mu(t)^T & \epsilon(t) \end{bmatrix}^T , \tag{11}$$

with $\mu(t) \in \mathbf{R}^3$, $\epsilon(t) \in \mathbf{R}$, and satisfy $\|q(t)\|_2 = 1$. Using unit quaternions we can describe the spacecraft's attitude kinematics as:

$$\begin{aligned} \dot{q}(t) &= \frac{1}{2} q(t) \otimes \begin{bmatrix} \omega(t) & 0 \end{bmatrix} \\ &= \frac{1}{2} \begin{bmatrix} 0 & \omega_z(t) & -\omega_y(t) & \omega_x(t) \\ -\omega_z(t) & 0 & \omega_x(t) & \omega_y(t) \\ \omega_y(t) & -\omega_x(t) & 0 & \omega_z(t) \\ -\omega_x(t) & -\omega_y(t) & -\omega_z(t) & 0 \end{bmatrix} q(t) . \end{aligned} \tag{12}$$

Notice that the attitude kinematics are always bilinear. A very important property of the attitude kinematic equations is that they preserve the unit norm property of the quaternion.
An attitude forbidden zone is an orientation that the spacecraft should avoid in order to protect sensitive instruments. For instance, consider that there is a bright object in direction x in the reference inertial frame and we have a light-sensitive instrument on direction y in the body frame of the spacecraft. In [15] the authors show how the requirement of maintaining an angle of at least θ with respect to the bright object can be written as an inequality constraint of the form:

$$q(t)^T P_f(x, y, \theta) q(t) \leq 0 , \tag{13}$$

whereas the need of keeping the spacecraft's orientation within θ degrees of a given direction x can be written as:

$$q(t) P_m(x, y, \theta) q(t) \geq 0 , \tag{14}$$

with the matrix having the following form in both cases:

$$P_{f,m}(x, y, \theta) := \begin{bmatrix} A_i & b_i \\ b_i^T & d_i \end{bmatrix} , \tag{15a}$$

$$A := xy^T + yx^T - (x^T y + \cos\theta) I_3 , \tag{15b}$$

$$b := -(x \times y) , \tag{15c}$$

$$d := x^T y - \cos\theta . \tag{15d}$$

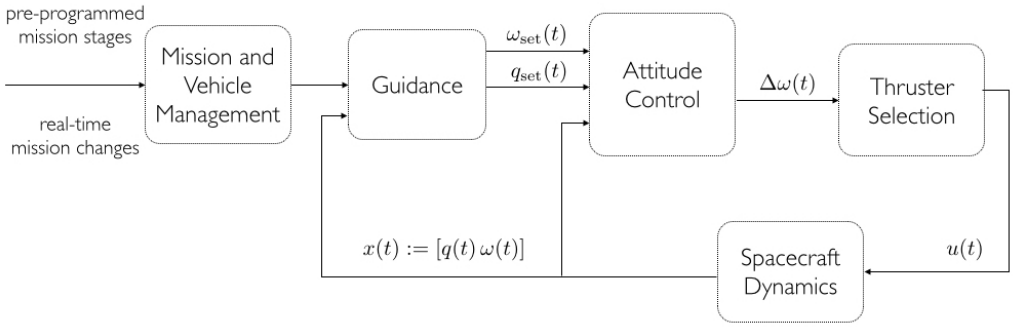

Figure 2: Spacecraft reorientation GNC architecture. See [16] for more details.

The full constrained attitude guidance trajectory generation problem with torque and angular rate limits can be formally described by:

$$\text{minimize} \int_{t_0}^{t_f} \lambda + (1-\lambda)|u(t)|\,dt \tag{16a}$$

$$\text{subject to } q(t_0) = q_{\text{current}},\ \omega(t_0) = \omega_{\text{current}}, \tag{16b}$$

$$q(t_f) = q_{\text{target}},\ \omega(t_f) = 0, \tag{16c}$$

$$\text{rigid body dynamics } (10), \qquad \forall t \in [t_0, t_f] \tag{16d}$$

$$\text{quaternion kinematics } (12), \qquad \forall t \in [t_0, t_f] \tag{16e}$$

$$-T_{\max} \leq u(t) \leq T_{\max}\,, \qquad \forall t \in [t_0, t_f] \tag{16f}$$

$$-\omega_{\max} \leq \omega(t) \leq \omega_{\max}\,, \qquad \forall t \in [t_0, t_f] \tag{16g}$$

$$q(t)P_{m,i}(x_i, y_i, \theta_i)q(t) \geq 0\,, \qquad \forall t \in [t_0, t_f], \forall i = \{1, \ldots, m\} \tag{16h}$$

$$q(t)P_{f,i}(x_i, y_i, \theta_i)q(t) \leq 0\,, \qquad \forall t \in [t_0, t_f], \forall i = \{1, \ldots, f\} \tag{16i}$$

where the fixed parameter $\lambda \in [0, 1]$ trades off between a minimum time objective ($\lambda = 1$) and a minimum fuel objective ($\lambda = 0$).
Figure 2 shows the GNC architecture in the simulator provided by Airbus. The original guidance consisted of an open-loop pre-programmed trajectory with various mission stages giving static quaternion setpoints for the low-level proportional controller. The structure was modified to accommodate for the new functionality by introducing feedback in the attitude guidance block, which now computes a new quaternion and angular rate setpoint trajectory based on the current state and the final target orientation, which can change in real-time.

3 CONVEXIFICATION

The continuous optimal control problems (9)–(16) do not have analytic solutions, hence one has to use a numerical method to solve them. Many early guidance trajectory generation problems for space applications where solved using the so-called indirect method, where the first-order optimality conditions are derived by minimizing the Hamiltonian and the resulting boundary value problem is

solved using a numerical method [17]. However, the difficulty in solving the boundary value problem efficiently and reliably motivated the development of modern direct methods, which we use in this work. We use a piece-wise constant parametrization of the control trajectory $u(t)$ and discretize the problem using direct multiple shooting [18] to approximate the state trajectories using an integration scheme and arrive at a finite-dimensional optimization problem.
The resulting nonlinear program (NLP) is non-convex and has multiple local solutions. The most popular methods for solving such problems are nonlinear interior-point methods and sequential quadratic programming (SQP) methods [19]. Both of these methods are highly sensitive to the supplied initial guess and cannot provide polynomial convergence guarantees to a local optimum, which raises reliability questions for their use in autonomous systems.
The main goal in an online real-time GNC concept is to be able to re-compute a safe flight trajectory for the vehicle whenever it is needed, with the computing resources that are available, and in the time that is available. The process must be autonomous and must never fail. For this reason, in a real-time context, it is desirable to approximate the problem with a convex relaxation to formulate a quadratic program (QP) or second-order cone program (SOCP) and be able to rely on robust convex optimization technology [20]. A wide range of convex optimization methods exist. Active-set methods and first-order methods can be an effective solution in certain situations. However, interior-point methods [21] are guaranteed to find the global optimum in polynomial time and exhibit robust and predictable behavior in practice with no dependence on the initial guess, which makes them very attractive for real-time implementation.
The goal of the *convexification* procedure is to modify the non-convex objective or constraint with a convex form that has a minimal impact on the solution of the problem. In some cases it is possible to derive reformulations that have the same solution as the original problem. This is referred to in the literature as *lossless convexification* [8]. Unfortunately, in most cases the approximation procedure leads to a change in the solution and the hope is that the loss in optimality is outweighed by the benefits of increased reliability. In the following we outline convexification procedures for both space flight benchmark problems.

3.1 Constrained powered descent

The dynamics of the point mass model with variable mass (2)-(3) are nonlinear. Since they enter the optimization problem as equality constraints they make the optimization problem non-convex. It is possible to introduce the following change of variables, described in [7],

$$w(t) := \frac{u(t)}{m(t)}\,, \quad \sigma(t) := \frac{\|u(t)\|_2}{m(t)}\,, \quad z(t) := \ln m(t)\,, \tag{17}$$

to rewrite the dynamics in the following linear form:

$$\ddot{r}(t) = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + w(t)\,, \tag{18}$$

$$\dot{z}(t) = \frac{\dot{m}(t)}{m(t)} = -\alpha\sigma(t)\,, \tag{19}$$

$$\|w(t)\|_2 \leq \sigma(t)\,, \tag{20}$$

where $\|w(t)\|_2 = \sigma(t)$ always holds when $\sigma(t)$ is minimized in the cost function.
As a result, the actuator constraints now become

$$-F_{\text{lat,max}} \leq m(t)M(\theta(t))w_{x,y}(t) \leq F_{\text{lat,max}}\,, \tag{21}$$

$$F_{\text{vert,min}} \leq m(t)M(\theta(t))w_z(t) \leq F_{\text{vert,max}}\,, \tag{22}$$

$$\mathbf{d}^T \frac{w(t)}{\sigma(t)} \geq \cos\gamma\,. \tag{23}$$

Constraints (21)-(22) have now become non-convex, but they can be relaxed by assuming that the mass will evolve as $m_{\text{current}} - \alpha F_{\text{min,norm}} t$ (or $m_{\text{current}} - \alpha F_{\text{max,norm}} t$ for the left hand side of (22)). This is an inner approximation that ensures feasibility with respect to the original problem. The approximation is very accurate for the situations that we are considering.
In addition, determining the optimum flight time is not known to be convex. If the time of flight is chosen too short, the problem may be infeasible because it is not physically possible to arrive at the target in the specified time. If it is chosen too long, the problem may also become infeasible because the thrusters can only be switched off at the end of the maneuver and there might not be enough fuel to keep the thrusters on for the entire maneuver. Hence, the time of flight that leads to feasible maneuvers is constrained as

$$t_{\min} \leq t_f \leq t_{\max}\,, \tag{24}$$

and one can employ a variety of search procedures where each iteration involves solving a convex optimization problem.

3.2 Constrained attitude guidance

The forbidden and mandatory pointing constraints (13)–(14) are non-convex because the matrices P_f and P_m are indefinite. However, in [22] it is shown that they can be rewritten as the following convex constraints:

$$q(t)^T \big(P_f(x,y,\theta) + 2\big) q(t) \leq 2\,, \tag{25}$$

$$q(t)\big(-P_m(x,y,\theta) + 2\big) q(t) \leq 2\,. \tag{26}$$

This is a surprising lossless convexification procedure, which only holds due the unit norm property of the quaternion.
The rigid body rotational dynamics (10) and the attitude kinematics (12) are bilinear. As before, since they enter the optimization problem as equality constraints, these constraints are non-convex. However, unlike in the powered descent case, it is not possible to derive a change of variables that linearizes the dynamics. The approach described in [23] eliminates the dynamics from the problem and keeps only the quaternions as decision variables. The problem with this approach is that once the dynamics are eliminated it is necessary to impose the unit norm quaternion property as a constraint, which is also non-convex and harder to handle than the bilinear dynamics. Instead we propose to use a standard procedure in the nonlinear model predictive control literature, which consists in linearizing the dynamics around a nominal trajectory $(\bar{q}(t), \bar{\omega}(t))$ and optimizing the perturbations. The success of this procedure relies on the availability of a good nominal trajectory. Assuming that we use a

forward Euler discretization of the rotational rigid body dynamics and quaternion kinematics with a guidance discretization interval T_s we can write the linear discrete perturbation dynamics as

$$\begin{aligned}\delta\omega(k+1) = &\begin{bmatrix} 1 & T_s J_x^{-1}(J_y - J_z)\bar{\omega}_z(k) & T_s J_x^{-1}(J_y - J_z)\bar{\omega}_y(k) \\ T_s J_y^{-1}(J_z - J_x)\bar{\omega}_z(k) & 1 & T_s J_y^{-1}(J_z - J_x)\bar{\omega}_x(k) \\ T_s J_z^{-1}(J_x - J_y)\bar{\omega}_y(k) & T_s J_z^{-1}(J_x - J_y)\bar{\omega}_x(k) & 1 \end{bmatrix} \delta\omega(k) \\ &+ T_s \begin{bmatrix} J_x^{-1} & 0 & 0 \\ 0 & J_y^{-1} & 0 \\ 0 & 0 & J_z^{-1} \end{bmatrix} \delta u(k) \\ &+ \left(\bar{q}(k) + \frac{T_s}{2} \begin{bmatrix} 0 & \bar{\omega}_z(k) & -\bar{\omega}_y(k) & \bar{\omega}_x(k) \\ -\bar{\omega}_z(k) & 0 & \bar{\omega}_x(k) & \bar{\omega}_y(k) \\ \bar{\omega}_y(k) & -\bar{\omega}_x(k) & 0 & \bar{\omega}_z(k) \\ -\bar{\omega}_x(k) & -\bar{\omega}_y(k) & -\bar{\omega}_z(k) & 0 \end{bmatrix} \bar{q}(k) \right) - \bar{q}(k+1)\,, \end{aligned} \tag{27}$$

$$\begin{aligned}\delta q(k+1) = &\frac{T_s}{2} \begin{bmatrix} \bar{\epsilon}(k) & -\bar{\mu}_3(k) & \bar{\mu}_2(k) \\ \bar{\mu}_3(k) & \bar{\epsilon}(k) & -\bar{\mu}_1(k) \\ -\bar{\mu}_2(k) & \bar{\mu}_1(k) & \bar{\epsilon}(k) \\ -\bar{\mu}_1(k) & -\bar{\mu}_2(k) & -\bar{\mu}_3(k) \end{bmatrix} \delta\omega(k) \\ &+ \begin{bmatrix} 1 & \frac{T_s}{2}\bar{\omega}_z(k) & -\frac{T_s}{2}\bar{\omega}_y(k) & \frac{T_s}{2}\bar{\omega}_x(k) \\ -\frac{T_s}{2}\bar{\omega}_z(k) & 1 & \frac{T_s}{2}\bar{\omega}_x(k) & \frac{T_s}{2}\bar{\omega}_y(k) \\ \frac{T_s}{2}\bar{\omega}_y(k) & -\frac{T_s}{2}\bar{\omega}_x(k) & 1 & \frac{T_s}{2}\bar{\omega}_z(k) \\ -\frac{T_s}{2}\bar{\omega}_x(k) & -\frac{T_s}{2}\bar{\omega}_y(k) & -\frac{T_s}{2}\bar{\omega}_z(k) & 1 \end{bmatrix} \delta q(k) \\ &+ \left(\bar{\omega}(k) + T_s \begin{bmatrix} \frac{1}{J_x}\left((J_y - J_z)\bar{\omega}_y(k)\bar{\omega}_z(k) + \bar{u}_x(k)\right) \\ \frac{1}{J_y}\left((J_z - J_x)\bar{\omega}_z(k)\bar{\omega}_x(k) + \bar{u}_y(k)\right) \\ \frac{1}{J_y}\left((J_x - J_y)\bar{\omega}_x(k)\bar{\omega}_y(k) + \bar{u}_z(k)\right) \end{bmatrix} \right) - \bar{\omega}(k+1)\,. \end{aligned} \tag{28}$$

In order to increase the accuracy of this approximation it may be necessary to add second-order cone constraints to limit the size of the perturbation and perform several successive linearizations as described in [24].

As in the powered descent case, determining the flight time can be obtained via a search procedure. However, in this case the problem can only become infeasible if the flight time is chosen too short, i.e. $t_{\min} \leq t_f$, hence one can use a simple backtracking procedure.

4 SOFTWARE VALIDATION FRAMEWORK

We have created a software framework to be able to compare the characteristics of different guidance modes in closed-loop with the high fidelity simulators provided by ESA and Airbus. The goal is to allow for an automatic in-depth evaluation of the reliability-speed-performance trade-off for different optimal guidance strategies (convex and non-convex) and compare them with the baseline guidance functionality. Figure 3 illustrates the architecture of the software tool:

User Interface A light-weight MATLAB-based text interface allows the user to create a (powered descent or attitude guidance) scenario object and modify several parameters in the guidance

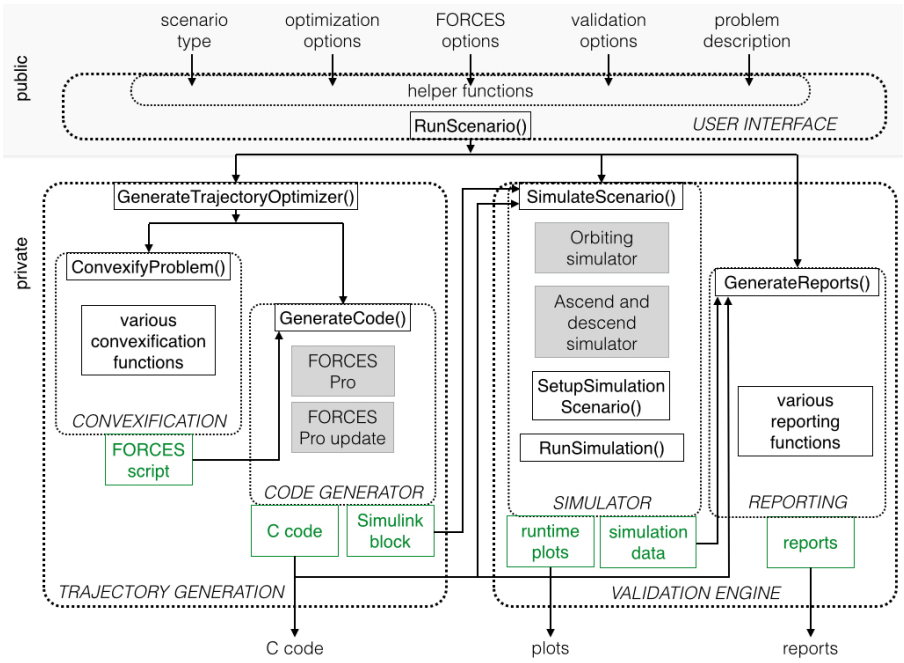

Figure 3: Software architecture of the validation tool for guidance strategies.

problem formulations (9) or (16), respectively. The interface also allows to define the guidance mode and various validation options. Once the problem description and all the options have been set one can invoke the run scenario method, which calls all the remaining functions.

Trajectory Generation This block creates code that computes guidance trajectories by solving an optimization problem. Depending on the selected guidance mode, the code has to solve the nonlinear program directly using nonlinear optimization software, or apply the convexification techniques described in Section 3 to solve a modified problem using a convex solver.

Many powerful commercial and open-source optimization packages are available. However, the code must be embeddable to run on-board in real-time, which severely restricts the available choices. FORCES Pro [25] and CVXGEN [26] are two software packages that generate customized optimization solvers that are tailored for deployment on resource-constrained embedded systems. In both cases the generated code uses only static memory allocation and does not make use of any external linear algebra libraries. CVXGEN only includes a convex quadratic programming (QP) solver and the size of the problems it can handle is very limited. FORCES Pro, on the other hand, offers a wide range of algorithmic options that cover the entire range of problem formulations that will be potentially solved in this activity. The package produces efficient code for problems with up to several thousand variables and constraints and is actively developed. Figure 4 illustrates its design concept.

FORCES Pro has various MATLAB- and Python-based interfaces to define convex and non-convex optimization problems. It can also be called using the YALMIP modeling language [27].

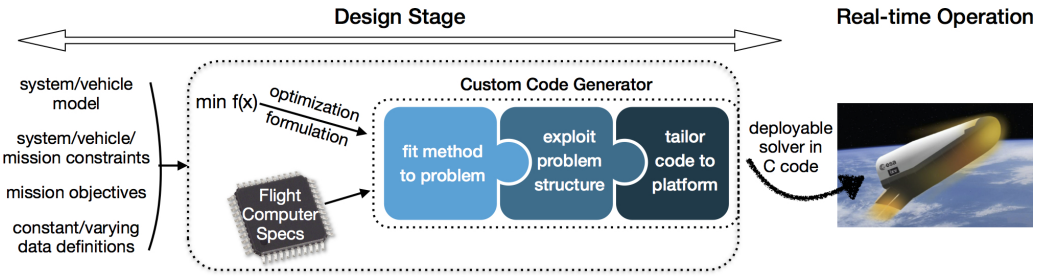

Figure 4: FORCES Pro design flow. A customized solver implemented in library-free C code is generated for every different problem description.

Depending on the selected guidance mode an appropriate FORCES script is invoked, which generates customized C code and the necessary Simulink wrapper for easy integration in the ESA-Airbus simulator.

Validation Engine The validation block provides both software- and hardware-in-the-loop simulation functionality. First, the simulator parameters are set, the generated FORCES Pro Simulink block is automatically integrated and the external hardware is initialized (if required by the validation options). Second, the scenario is simulated with the trajectory generation code running on the PC (for software-in-the-loop mode) or on a Raspberry Pi 2 board (for hardware-in-the-loop mode) with UDP communication via Ethernet. The Raspberry Pi 2 board features an ARM Cortex A7 processor, which has very similar characteristics to the vertical take-off vertical landing demonstrator vehicle being developed by FLPP3. Finally, the results of the simulation are presented in a computational performance report, a numerical reliability report and a GNC closed-loop performance report.

5 TEST RESULTS

Extensive computational and numerical tests are still to be carried out since the design of the guidance modes has not yet been finalized. In this section we include some preliminary results showing one test scenario for each of the control problem types.

5.1 Constrained powered descent

The considered scenario is as described by Figure 5. The ducted fan takes off from a position 120m away from the target in the horizontal direction and 10m below it in altitude. The ascent phase is the same for all guidance modes and lasts until the vehicle is 30m above the target. At that point the baseline guidance mode decides to fly in the horizontal direction at a constant altitude until it reaches a point where it can descend towards the target on a 15° trajectory and land gracefully. Following this path uses 0.87 kilograms of kerosene, as shown in Table 1.
Instead, the optimization-based guidance modes decide that the most fuel-efficient trajectory is to first fly higher at an angle, then descend at high speed towards the target and make a final braking burn

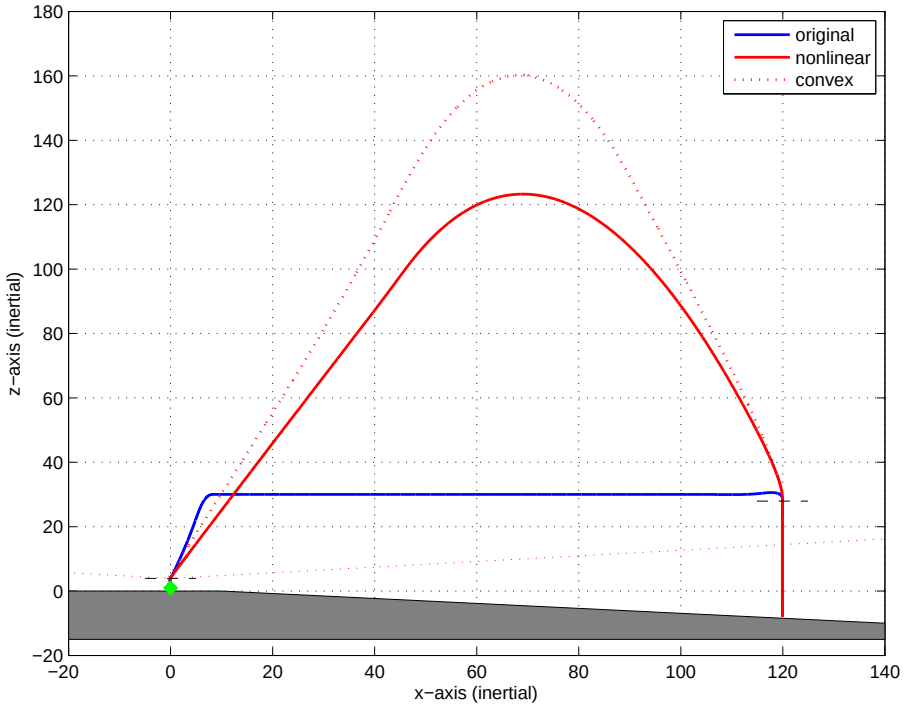

Figure 5: Comparison of flight trajectories for the different guidance modes.

Table 1: GNC closed-loop performance metrics for various guidance modes for powered descent. The fuel consumption and flight time exclude the ascent phase, which is the same for all guidance modes.

	original guidance	nonlinear optimization	convex optimization
Landing error (m)	0.0017	0.0099	0.2221
Vertical speed at landing (ms^{-1})	-0.4637	-0.5440	-0.5279
Horizontal speed at landing (ms^{-1})	0.0019	0.0006	0.0401
Angles at landing (deg)	(-0.00, -0.01, -0.00)	(-0.00, 0.02, 0.00)	(-0.00, 0.00, -0.00)
Angular rates at landing (deg s^{-1})	(0.00, 0.01, 0.00)	(-0.00, -0.11, -0.00)	(-0.00, -0.10, 0.00)
Fuel consumption (kg)	0.87	0.34	0.33
Flight time (s)	73.2	28.0	27.3

before reaching the target at the target speed. The detailed position, velocity, attitude, thrust and mass trajectories for the convex mode are shown in Figure 6. The descent thrust profile is in agreement with the theoretically optimal solution, which is known to be of bang-bang form [28]. Table 1 shows that the new guidance modes achieve a substantial reduction in fuel consumption (more than 60%), which could be used to increase the retargeting range of the vehicle.
In this scenario the guidance trajectory is only computed once after the ascent phase. The computed position and velocity trajectories are stored once and read out at each control interval. Figure 6 also shows the difference between the commanded trajectory and the actual realized trajectories. The low-level controller is able to follow the commanded trajectory very accurately, emphasizing the validity for the approximations we used for the dynamical model and constraints.
Furthermore, the optimal trajectories are being computed fast enough for real-time implementation. As a point of reference for the reader, the nonlinear optimization solution is computed on the Raspberry Pi 2 in under 0.2 seconds, leaving room for solving more complex problems in real-time in the future.

5.2 Constrained attitude guidance

The test scenario is as described by Figure 7. The body frame of the spacecraft is initially slightly displaced from the inertial reference frame (corresponding to $q_{\text{current}} = \begin{bmatrix}0.1 & 0.1 & 0.1 & 0.985\end{bmatrix}$). There is an antenna pointing towards the positive z-axis of the space vehicle that must remain pointing to within 20° of the direction $\begin{bmatrix}0 & 0 & 1\end{bmatrix}$ in the inertial frame. In addition, there is a light-sensitive instrument mounted on the positive y-axis of the space vehicle that must avoid pointing towards 20° of the direction $\begin{bmatrix}-1 & 1 & 0.2\end{bmatrix}$ in the inertial frame. The target orientation is defined by $q_{\text{target}} = \begin{bmatrix}0 & 0 & 0.643 & 0.766\end{bmatrix}$. The initial and target angular rates are zero.
The original GNC structure just passes the target quaternion to the attitude controller, which computes an angular acceleration command proportional to the difference between the current quaternion and the target. The control system is not aware of the attitude constraints, hence the resulting trajectory violates the attitude forbidden zone, as shown in Table 2. In the classical trajectory generation framework one would compute way-points for various reorientation stages such that the constraints are not violated and the trajectory improves certain objectives, such as the reorientation time. This process would involve multiple design iterations and extensive simulations.
In contrast, the optimization-based guidance modes are aware of the attitude constrained zones and are able to plan a trajectory in real-time that satisfies the constraints. In addition, one can automatically sweep between a minimum fuel to a minimum time trajectory by just tuning one parameter (see Table 2), potentially saving on mission preparation time or allowing the spaceraft to reorient itself autonomously. Figure 8 shows the detailed trajectory profile for the nonlinear optimization guidance mode when the objective is set to minimum time ($\lambda = 1$). The aggressive guidance trajectory can still be followed by the low-level controller since the dynamical constraints of the spacecraft have been considered in the formulation of the optimization problem.

6 FUTURE DEVELOPMENTS

The next stage of development in the software validation tool will involve functionality to be able to evaluate the reliability-speed-performance via extensive simulations. We have defined algorithm

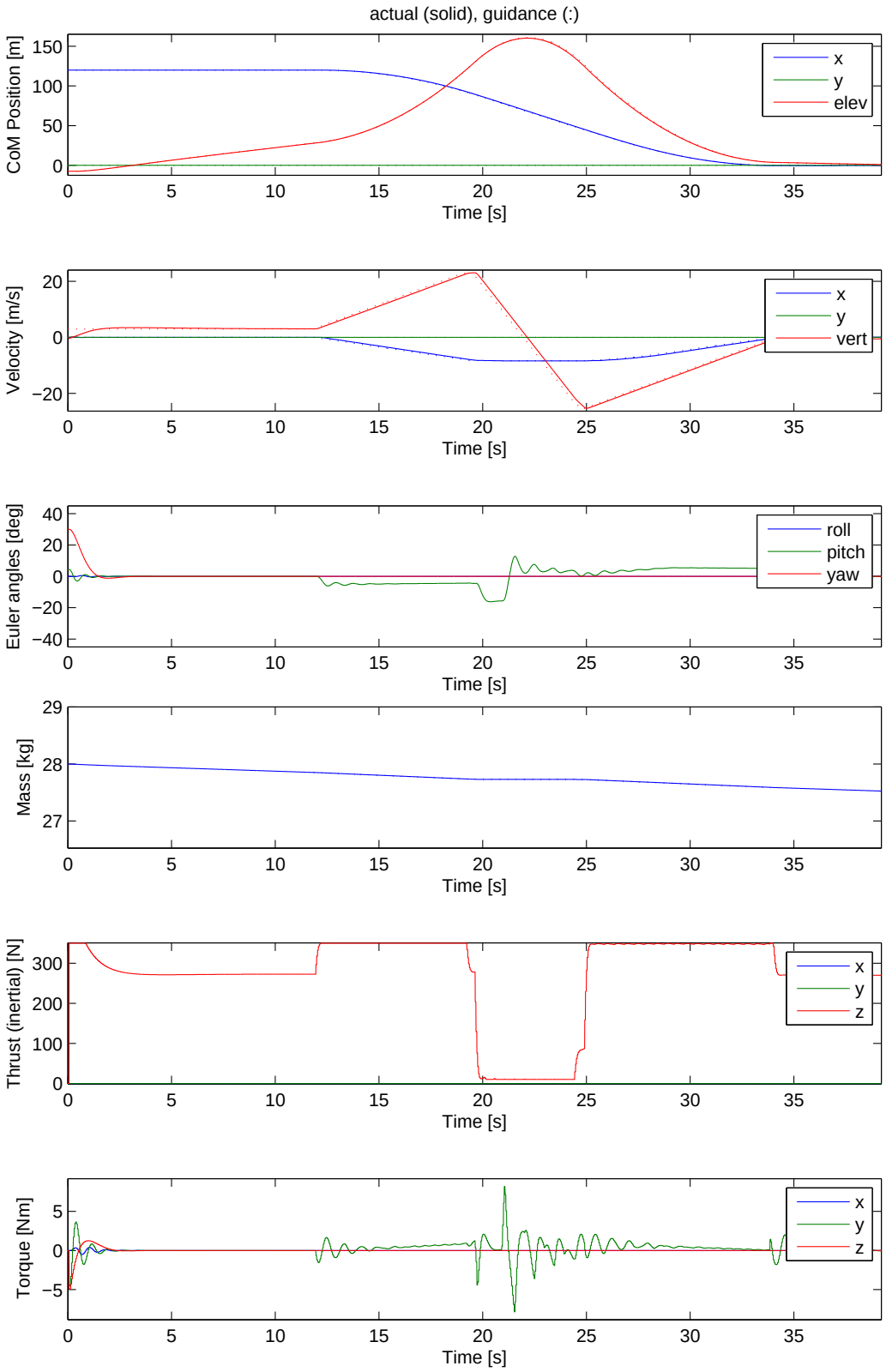

Figure 6: Realized trajectory (solid) and reference guidance trajectory (dotted) provided by the convex mode for powered descent.

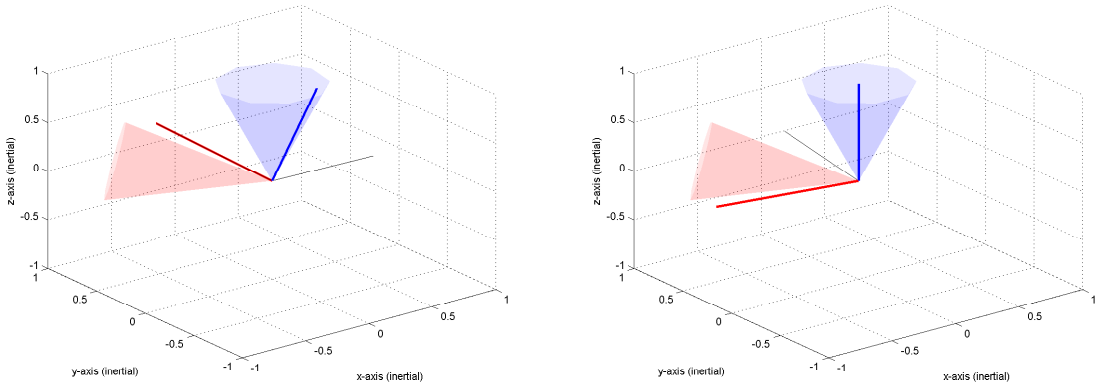

Figure 7: Initial (left) and target (right) orientations for the constrained attitude guidance scenario. Attitude mandatory (inclusion) zones are depicted in blue whereas attitude forbidden (exclusion) zones are depicted in red.

complexity, consistency, and performance metrics for each benchmark that will be evaluated using a Monte Carlo approach as suggested in [29].
We will also investigate the need for adding further extensions to the problem formulations (9) and (16). For instance, the glideslope constraint (4) can be easily extended for landing inside pits or craters [30]. It may also become necessary to consider aerodynamic drag in the point-mass dynamics (2), which adds an additional nonlinearity that requires a different convexification approach [31].
In parallel, ESA has started the developement of a vertical take-off and landing demonstrator powered by three jet engines with approximately 2500 N thrust (DTV). The DTV shall serve as a testbed for various technologies needed for future launchers. The first flights are envisaged in summer 2018. The described algorithm and software will be tested for scenarios similar to those shown in Figure 5.

7 CONCLUSIONS

This activity studies a new approach to the generation of flight guidance trajectories by solving optimization problems on-board and in real-time. The capability to generate optimal trajectories autonomously would bring unprecedented benefits in terms of cost and flight performance over the classical offline design approach.
The proposed online trajectory generation framework is thought to be applicable to any GNC problem setup. In this paper we have shown how to apply this methodology in two very different space flight scenarios. For constrained powered descent situations, where an offline trajectory generation process is not applicable due to atmospheric uncertainty, we have shown how one can compute optimization-based trajectories in real-time and significantly enlarge the flight envelope compared to online heuristic guidance approaches. For spacecraft constrained reorientation procedures we have shown how one can compute optimal trajectories, either on-board or on Earth, in a matter of milliseconds, which could significantly decrease mission preparation costs or allow for the autonomous recovery of the space vehicle.

Table 2: GNC closed-loop performance metrics for various guidance modes for constrained attitude guidance with different objectives: minimum fuel ($\lambda = 0$); balanced ($\lambda = 0.5$); minimum time ($\lambda = 1$).

	original guidance	nonlinear optimization
Attitude zone violations - time (s)	16.1	0
Attitude zone violations - integral (-)	6.15	0
Cost function integral (-) ($\lambda = 0$)	372.04	406.11
Flight time (s) ($\lambda = 0$)	82.4	164.8
Cost function integral (-) ($\lambda = 0.5$)	227.22	316.39
Flight time (s) ($\lambda = 0.5$)	82.4	132.1
Cost function integral (-) ($\lambda = 1$)	82.40	27.80
Flight time (s) ($\lambda = 1$)	82.4	27.8

To support the future development of flight-ready navigation code we have described the architecture of a software validation tool with hardware-in-the-loop capabilities that allows one to explore the design space for optimal trajectory generation algorithms and implementations. The tool uses state-of-the-art software that incorporates many recent technological advances in optimization methods for real-time systems.

REFERENCES

[1] A. J. Bordano, G. G. Mcswain, and S. T. Fernandes, "Autonomous guidance, navigation and control," in *Annual Rocky Mountain Guidance and Control Conference*, Keystone, Coloroado, USA, Jan 1991, pp. 15–37.

[2] F. Teren, "Explicit guidance equations for multistage boost trajectories," NASA Lewis Research Center, Tech. Rep. NASA-TN-D-3189, Jan 1966.

[3] E. M. Queen, "Guidance concept for a Mars ascent vehicle first stage," NASA Langley Research Center, Tech. Rep. NASA/TM-2000-210618, Nov 2000.

[4] H. N. Scofield, "A simplified explicit guidance scheme for orbital injection," *Journal of Spacecraft and Rockets*, vol. 5, no. 11, pp. 1309–1311, Nov 1968.

[5] R. Jaggers, "An explicit solution to the exoatmospheric powered flight guidance and trajectory optimization problem for rocket propelled vehicles," in *Guidance and Control Conference*, Hollywood, Florida, USA, Aug 1977, pp. 566–578.

[6] G. A. Dukeman, "Atmospheric ascent guidance for rocket-powered launch vehicles," in *AIAA Guidance, Navifation and Control Conference*, Monterey, California, USA, Aug 2002.

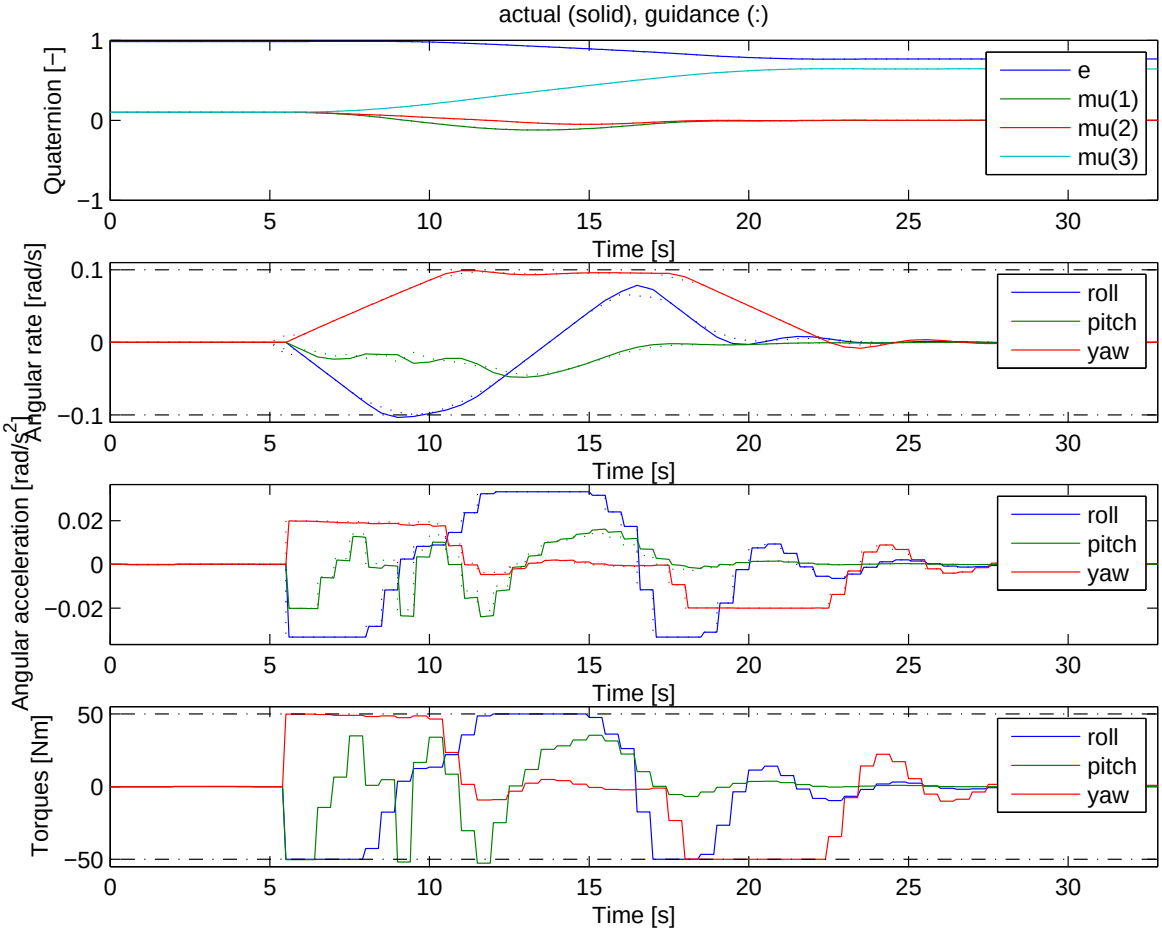

Figure 8: Realized trajectory (solid) and reference guidance trajectory (dotted) provided by the non-linear optimization mode for constrained attitude guidance when the objective is set to minimum time ($\lambda = 1$).

[7] B. Açikmeşe and S. R. Ploen, "Convex programming approach to powered descent guidance for Mars landing," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 5, pp. 1353–1366, Sep 2007.

[8] B. Açikmeşe and L. Blackmore, "Lossless convexification of a class of optimal control problems with non-convex control constraints," *Automatica*, vol. 47, no. 2, pp. 341–347, Feb 2011.

[9] L. Blackmore, B. Açikmeşe, and D. P. Scharf, "Minimum-landing-error powered-descent guidance for Mars landing using convex optimization," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 4, pp. 1161–1171, Jul 2010.

[10] P. Lu and X. Liu, "Autonomous trajectory planning for rendezvous and proximity operations by conic optimization," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 2, pp. 375–389, Mar 2013.

[11] R. R. Sostaric and J. R. Rea, "Powered descent guidance methods for the Moon and Mars," in *AIAA Guidance, Navigation, and Control Conference*, San Francisco, California, Aug 2005.

[12] H. Strauch, C. Jetzschmann, and U. Soppa, "Flight control algorithm description," Airbus DS, Tech. Rep. USACDF-TN0033, July 2016.

[13] U. Lee and M. Mesbahi, "Optimal powered descent guidance with 6-DoF line of sight constraints via unit dual quaternions," in *AIAA Guidance, Navigation, and Control Conference*, Kissimmee, Florida, USA, Jan 2015.

[14] B. Açikmeşe, M. Aung, J. Casoliva, S. Mohan, A. Johnson, D. Scharf, D. Masten, J. Scotkin, A. Wolf, , and M. W. Regehr, "Flight testing of trajectories computed by G-FOLD: Fuel optimal large divert guidance algorithm for planetary landing," *Advances in the Astronautical Sciences*, vol. 148, Jan 2013.

[15] U. Lee and M. Mesbah, "Quaternion based optimal spacecraft reorientation under complex attitude constrained zones," *Advances in the Astronautical Sciences*, vol. 150, Jan 2014.

[16] H. Strauch, K. Luig, and S. Bennani, "Model based design environment for launcher upper stage gnc development," *Workshop on Simulation for European Space Programmes (SESP), Noordwijk, The Netherlands*, 2014.

[17] R. Sargent, "Optimal control," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1, pp. 361–371, Dec 2000.

[18] H. Bock and K. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," in *9th IFAC World Congress*, Budapest, Hungary, Jul 1984.

[19] J. Nocedal and S. Wright, *Numerical Optimization*. Springer Verlag, 2000.

[20] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[21] S. J. Wright, *Primal-Dual Interior-Point Methods*. SIAM, 1997.

[22] U. Lee and M. Mesbahi, "Feedback control for spacecraft reorientation under attitude constraints via convex potentials," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 4, pp. 2578–2592, Oct 2014.

[23] U. Eren, B. Açikmeşe, and D. P. Scharf, "A mixed integer convex programming approach to constrained attitude guidance," in *European Control Conference*, Linz, Austria, July 2015, pp. 1120–1126.

[24] X. Liu and P. Lu, "Solving nonconvex optimal control problems by convex optimization," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 3, pp. 750–765, May 2014.

[25] A. Domahidi and J. Jerez. (2014-2017) Forces Professional. embotech GmbH. [Online]. Available: http://embotech.com/FORCES-Pro

[26] J. Mattingley and S. Boyd, "CVXGEN: a code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, no. 1, pp. 1–27, Mar 2012.

[27] J. Löfberg, "YALMIP: A toolbox for modmodel and optimization in MATLAB," in *IEEE Internation Symposium on Computer Aided Control Systems Design*, Taipei, Taiwan, Sep 2004, pp. 284–289.

[28] U. Topcu, J. Casoliva, and K. Mease, "Fuel efficient powered descent guidance for Mars landing," in *AIAA Guidance, Navigation, and Control Conference*, San Francisco, California, Aug 2005.

[29] B. A. Steinfeldt, R. D. Brauny, and S. C. Paschall, "Guidance and control algorithm robustness baseline indexing," in *AIAA Guidance, Navigation, and Control Conference*, Toronto, Canada, Aug 2010.

[30] N. S. Bhasin, "Fuel-optimal spacecraft guidance for landing in planetary pits," Master's thesis, Carnegie Mellon University, Apr 2016.

[31] A. W. Berning, "Verification of successive convexification algorithm," Master's thesis, The University of Texas at Austin, May 2016.