Post-Print

This is the accepted version of:

J. Guadagnini, M. Lavagna, P. Rosa
Model Predictive Control for Reusable Space Launcher Guidance Improvement
Acta Astronautica, In press - Published online 15/10/2021
doi:10.1016/j.actaastro.2021.10.014

The final publication is available at https://doi.org/10.1016/j.actaastro.2021.10.014

Model Predictive Control for Reusable Space Launcher Guidance Improvement

Jacopo Guadagnini[a], Michèle Lavagna[b] and Paulo Rosa[c]

[a] *Politecnico di Milano, Milano, Italy,* jacopo.guadagnini@mail.polimi.it
[b] *Dipartimento di Scienze e Tecnologie Aerospaziali, Politecnico di Milano, Milano, Italy,* michelle.lavagna@polimi.it
[c] *Deimos Engenharia, Lisbon, Portugal,* paulo.rosa@deimos.com.pt

Abstract

This work aims to demonstrate the benefits and limitations of an on-board Guidance for reusable launch vehicles, as well as to tradeoff different Model Predictive Control (MPC)-based Guidance and Control (G&C) architectures, exploiting, in particular, recent advances on successive convexification algorithms for optimization problems. Leading space agencies and private companies are investing on the development of reusable space launchers to reduce the cost to access the space. Indeed, that cost is one of the major deterrents in space exploration and space utilization. Reusability is, therefore, the unanimous solution to lower costs, and get a reliable and fast space access. Among many technological enhancements, the guidance, navigation, and control plays a crucial role: precise pinpoint landing capabilities or mid-air recovery, in fact, are mandatory. Indeed, the capability for generating re-optimized guidance trajectories on-board in real-time based on current flight conditions promises to improve the system performance, allows for fault tolerance capabilities, and reduces mission preparation costs. The work focuses especially on the implementation of a successive convexification Model Predictive Control guidance algorithm which solves the 6 Degree-of-Freedom (DoF) Powered Descent Guidance problem (PDG). The novelty of that work is applying a model predictive-based technique to a complex dynamic environment, trading off different solutions to the problem and relying on results obtained by using an industrial simulation framework. The robustness of the proposed approach is tested in several operative scenarios and the feasibility of real-time implementation is studied. For what concerns the trajectory optimization routine, the formulation of the problem, while initially being non-convex, is convexified. This is performed by implementing a successive convexification algorithm, which obtains a sub-optimal solution of the original problem in a fraction of the time required by a global optimizer, by solving a Second Order Cone Programming (SOCP) problem. This method allows coping with different kinds of dynamics nonlinearities, as well as cost functions and constraints. By presenting the approach and critically discussing the obtained results, the work provides an overview of the different methodologies available in the literature and assesses the limits of those approaches when applied to highly nonlinear scenarios, with large dispersions of uncertain parameters, as it is the case of reusable launch vehicles.

Keywords: Reusability, Successive Convexification, Reentry, Trajectory Optimization, MPC.

Nomenclature

i	Iteration
k	Discrete time instant
K	Number of discrete points
t_f	Time of flight (or final time)
I_s	Specific Impulse
g_0	Gravity at sea level
P_{amb}	Ambient Pressure
A_{nozzle}	Area of the nozzle
$\boldsymbol{T}_B$	Thrust vector in Body Frame
$\boldsymbol{A}_B$	Aerodynamics force vector in Body Frame
$\mathbf{R}_{i\to j}$	Rotation matrix from i to j
$\boldsymbol{e}_i$	Versor relative to i axis
T_{min}	Minimum thrust
T_{max}	Maximum thrust
γ_{gs}	Glide-slope angle
θ	Tilt angle
$\boldsymbol{\omega}$	Angular velocity
α	Angle of attack
δ	Gimbal angle
σ	Time dilatation coefficient

Acronyms/Abbreviations

VTVL	Vertical Takeoff Vertical Landing
PDG	Powered Descent Guidance
MPC	Model Predictive Control
DoF	Degree of Freedom
SOCP	Second Order Cone Programming
SCvx	Successive Convexification
FOH	First Order Hold
ZOH	Zero Order Hold
LTV	Linear Time Varying
TVC	Thrust Vector Control
STC	State-Triggered Constraints

1. Introduction

The capabilities of a precise soft-landing have multiple applications in several space missions, such as mining asteroids exploration, planetary explorations and reusable space launcher. In the last few years, reusable launch vehicles became one of the principal interests for leading space agencies and private companies, especially in what concerns the vertical takeoff vertical landing (VTVL). Reusability of launch vehicles is, indeed, the solution to lower the costs for space access. A VTVL mission involves the recovery of the launch vehicle or a part of it (usually, but not necessarily, the first stage). To satisfy this demand, precise pin-point landing or mid-air recovery is mandatory. To answer this goal, the guidance, navigation and control plays a crucial role. One of the most challenging aspects of this problem is the development of an autonomous optimal guidance technology that allows the computation of reoptimized trajectory online in real-time, based on current flight conditions. Often, the trajectory of a launcher is performed by adopting an offline open-loop guidance: the optimal trajectory problem is solved offline and during the flight a low-level control system follows a reference trajectory. Until recently, only this approach has been adopted due to the difficulty in solving trajectory planning problems in real-time without any human intervention and to execute complex iterative optimization algorithms reliably on resource-constrained embedded platforms, such as the ones available in a typical flight computer. Thanks to the continuous advancement in more efficient and higher performing computational platforms, combined with the availability of fast and reliable optimization codes and in novel mathematical formulations of the problem, the closed-loop online guidance starts to be appealing. Indeed, online closed-loop guidance can improve the performance of the vehicle by recomputing the optimal trajectory from the knowledge of the current external conditions [1],[2],[3],[4]. This approach allows to cope with the uncertainties encountered during the entry, descent and landing phases. This increases the likelihood that the launch will land safely and at the desired site and enhances the probability of the success of the mission [5]. Recently developed real-time embedded MPC guidance and control strategies have a great potential for the next generation of high-performance reusable space launchers.

This work focuses especially on the implementation of a successive convexification MPC-based guidance algorithm which solves the 6 DoF PDG problem. This approach is applied to a complex dynamic environment, by testing its robustness and underlying the limitations, comparing two G&C strategies, based on simulations obtained within an industrial simulation framework.

1.1 Organization

The organization of the paper is the following: Sec. 2 gives an overview of the successive convexification algorithm; Sec. 3 presents the 6DoF Powered Descent Guidance Problem; Sec. 4 illustrates the G&C architectures proposed in this paper; Sec. 5 reports the evaluation of the performance; Sec. 6 underlines the conclusion of the work.

2. Successive Convexification

In this section, a general overview of the successive convexification algorithm is presented; in this paper, indeed, this tool has been used to convexify and to solve the 6-DoF PDG guidance problem.

The successive convexification algorithm is a novel method to solve nonconvex optimal control problem with nonlinear dynamics and nonconvex state and control constraints. This method has been introduced firstly in [6], where the authors, after presenting the algorithm, perform the convergence analysis and demonstrate that the obtained solution of the convex subproblem will recover the optimality of the original nonconvex problem**.** One fundamental feature is that the algorithm can be initialized with a simple dynamically inconsistent guess solution. For the reasons stated previously and due to the fact that this algorithm has minimal requirements on the considered dynamical system, it can be applied to several real-world optimal control problems, such as autonomous landing of reusable launchers (as in this work and in [4] and [7]).

The successive convexification algorithm computes the solution of the original problem by iteratively solving convex optimization subproblems, preferably formulated as SOCP: these are obtained through a linearization of the nonconvex dynamics and constraints on the previous iteration solution. The method involves adding virtual controls, dynamics relaxation and trust regions to avoid artificial infeasibility and artificial unboundedness, which drive the convergence. The user can specify the tolerances within which the original nonconvex problem is solved with local optimality [6],[8].

2.1 Linearization

The first step in successive convexification is the linearization of the nonlinear dynamics, state and inputs constraints about the $(i\text{-}1)^{th}$ solution in order to eliminate the nonconvexities. The obtained convex subproblem is solved resulting in a new solution for the i^{th} iteration. The process is reiterated in succession until convergence is accomplished and the optimality is restored.

A general optimization problem with dynamics constraints is described by:

$$\begin{aligned} &\text{minimize} && J(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \\ &\text{subject to} \\ &&\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \\ &&s\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) \leq 0 \\ &&l\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big) = 0 \\ &&\boldsymbol{x}(t) \in \mathcal{X} \subseteq \mathbb{R}^{n_x} \\ &&\boldsymbol{u}(t) \in \mathcal{U} \subseteq \mathbb{R}^{n_u} \end{aligned} \tag{1}$$

The problem is linearized at each time instant about the previous iterative solution, though a first-order Taylor approximation. The linearized subproblem is:

$$\begin{aligned} &\text{minimize} && J(\boldsymbol{x}(t), \boldsymbol{u}(t), t) \\ &\text{subject to} \\ &&\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{B}(t)\boldsymbol{u}(t) + \boldsymbol{D}(t) + \boldsymbol{z}(t) \\ &&h\big(\boldsymbol{x}^{i-1}(t), \boldsymbol{u}^{i-1}(t)\big) + \boldsymbol{H}_{\boldsymbol{x}}\boldsymbol{x}(t) + \boldsymbol{H}_{\boldsymbol{u}}\boldsymbol{u}(t) \leq 0 \\ &&g\big(\boldsymbol{x}^{i-1}(t), \boldsymbol{u}^{i-1}(t)\big) + \boldsymbol{G}_{\boldsymbol{x}}\boldsymbol{x}(t) + \boldsymbol{G}_{\boldsymbol{u}}\boldsymbol{u}(t) = 0 \\ &&\boldsymbol{x}(t) \in \mathcal{X} \subseteq \mathbb{R}^{n_x} \\ &&\boldsymbol{u}(t) \in \mathcal{U} \subseteq \mathbb{R}^{n_u} \end{aligned} \tag{2}$$

X and *U* are convex set**.** The analytical expressions of the jacobian is reported in [9].

2.2 Discretization

The linear-time variant (LTV) problem obtained in the previous section must be convert in a finite dimensional parameter optimization problem. The dynamics and the time are, therefore, discretized to *K* points. The time vector is divided in this way:

$$t_k = \frac{k-1}{K-1} t_f \; k \in [1, \; K] \tag{3}$$

The discretization of the linear problem can be performed by using several discretization schemes, such as ZOH, FOH, Runge-Kutta methods or Global Pesudospectral methods. Ref. [10] provides a detailed overview of each of these methods. In this paper, the ZOH has been adopted for the discretization. In [9] the analytical expressions of the discretized state space solution of the time-varying system and the discretized input are reported.

2.3 Trust Regions and Virtual Controls

The linearization of the nonlinearities can create two issues called artificial unboundedness and artificial infeasibility. The former is mitigated with the introduction of the trust regions constraints. In this paper, they are defined as quadratic inequality constraints and the penalization in the cost function of the trust region radius. The aim of these constraints is to define a region close to the previous iteration, in order to limit the deviation between two adjacent iterations. The inequality constraints are defined in this way:

$$\delta \boldsymbol{x}_k^i \cdot \delta \boldsymbol{x}_k^i + \delta \boldsymbol{u}_k^i \cdot \delta \boldsymbol{u}_k^i \leq \Delta_k^i$$

$$\text{with} \qquad \begin{aligned} \delta \boldsymbol{x}_k^i &= \boldsymbol{x}_k^i - \boldsymbol{x}_k^i \\ \delta \boldsymbol{u}_k^i &= \boldsymbol{u}_k^i - \boldsymbol{u}_k^i \end{aligned} \tag{4}$$

The artificial infeasibility is avoided by introducing the so-called virtual controls, which are additional control inputs and allow reaching each point of the solution domain, through dynamics relaxation:

$$\boldsymbol{x}_{k+1}^i = \bar{\boldsymbol{A}}_k \boldsymbol{x}_k^i + \bar{\boldsymbol{B}}_k \boldsymbol{u}_k^i + \bar{\boldsymbol{z}}_k^i + \boldsymbol{\nu}_k^i \tag{5}$$

with $\boldsymbol{\nu} \in \mathbb{R}^{nx}$. Both trust regions and virtual controls are penalized in the cost function.

$$\text{minimize } \mathrm{P} = J + w_{\boldsymbol{\nu}} S_{\boldsymbol{\nu}}^i + w_{\boldsymbol{\Delta}} S_{\boldsymbol{\Delta}}^i \tag{6}$$

Where w_j are the weights and S_j^i are the norm of the trust region radius and the norm of $\boldsymbol{\nu}$.

2.3 Initial Guess

One of the important features of the successive convexification algorithm is that the process can be initialized with a simple dynamically inconsistent guess solution.

The simplest initialization approach is to create a linear interpolation of the discrete state variables at each time instant among the initial and final conditions. The discrete input variables can be initialized by assuming a constant value for the whole simulation. It is worth to underline that whenever there are unknown variables a priori, they can be initialized with user-defined values with the foresight to satisfy the imposed constraints.

The initial guess state and input variables solution at each time instant, are defined as:

$$\begin{aligned} \boldsymbol{x}_k^0 &= \frac{K-k}{K-1}\boldsymbol{x}_{in} + \frac{k-1}{K-1}\boldsymbol{x}_f \quad k \in [1, \; K] \\ \boldsymbol{u}_k^0 &= \boldsymbol{u}_{guess} \qquad\qquad\qquad\quad k \in [1, \; K-1] \end{aligned} \tag{7}$$

The algorithm is not particularly sensitive to the initial guess, but, poor guesses may lead to increased convergence time [7].

2.4 Algorithm

Successive Convexification
Initialization: • Compute the initial guess (i=0) $\boldsymbol{x}_k^0$ and $\boldsymbol{u}_k^0$; • for $k \in$ [1, K-1] linearize the problem around $\boldsymbol{x}^0$ and $\boldsymbol{u}^0$: $\bar{\boldsymbol{A}}_{\boldsymbol{k}}$, $\bar{\boldsymbol{B}}_{\boldsymbol{k}}$ and $\bar{\boldsymbol{z}}_{\boldsymbol{k}}$ and the terms related to the constraints • Set the maximum number of iterations and the tolerance on the trust regions radius.
Successive Convexification Loop: • **While** ($i < i_{max}$ & $\|\boldsymbol{\Delta}^i\|_2 \geq \Delta_{tol}$) 1. Solve the problem in Eq. 2 with $\boldsymbol{x}_k^{i-1}$, $\boldsymbol{u}_k^{i-1}$ and the matrices; 2. Store the new solution $\boldsymbol{x}_k^i$, $\boldsymbol{u}_k^i$, $\boldsymbol{\Delta}^i$, $\boldsymbol{v}_k^i$; 3. **If** $i \geq i_{max}$ or $\|\boldsymbol{\Delta}^i\|_2 < \Delta_{tol}$ **Exit Loop** **else** (a) Linearize the problem around the new solution; (b) Set $i = i+1$; (c) Return to step 1; **end** **end**

The algorithm has been validated by applying the method to a nonlinear optimal control problem of which it is known the analytical solution [9],[11],[12].

3. 6 DoF Landing Problem

In this section, the 6-DoF formulation for a generalized powered descent guidance problem is presented. The powered descent guidance problem is a trajectory optimization problem, in which the optimal control problem is solved by maximising desired performance, while a set of constraints is satisfied. More specifically, in this case the objective is to minimize the amount of fuel needed to safely land [13].

For the feasibility of implementing the guidance and control algorithms, certain assumptions need to be made:

- The planetary rotation effects are neglected due to the relatively short duration of the studied problem.
- A simplified formulation of the aerodynamics forces has been considered.
- The centre of pressure is considered constant in the body-fixed frame.
- The effects of the wind are not embedded into the guidance problem.
- The vehicle is equipped with a single rocket engine that can be gimballed symmetrically about two axes up to a maximum gimbal angle.
- The engine can be throttled between minimum and maximum thrust values and once the engine is ignited, it cannot be shut off until the terminal condition is reached.
- The launcher is modelled as a rigid-body.
- The center of mass is considered constant during flight.
- The moment of the inertia of the rocket is considered constant during the flight.
- Higher order terms like flexible modes and sloshing are neglected by Guidance.
- The time of flight is not fixed a priori, but it is an optimization variable.

It is worth to be noticed that these assumptions refer only to the guidance problem. Most of them are dropped for the development of the nonlinear simulator. However, any nonlinear dynamics and control configurations can be incorporated in the guidance optimization problem, making the problem more complete on one hand, but on the other more complicated [5],[16]. Indeed, by introducing more realistic models, the linearization process, introduced in Section 2, will produce denser Jacobians and the optimization problem will be more difficult to be solved by the optimizer.

It is worth to underline that the strongest assumptions that may impact on the results are the evaluation of the centre of gravity, the centre of pressure and the inertia during the flight. These terms, indeed, play a crucial role for the definition of the moments on the vehicles, and so on the control. However, for the powered descent problem the definition of both the centre of gravity and centre of pressure is not straightforward and dedicated analysis must be performed, which depends on the considered vehicle features, such as shape of the vehicle, tanks configuration, fuel and oxidizer used. The integration in the optimization is not a trivial issue. An example of formulations can be seen in [22].

By considering the trade-off between complexity of the problem and computational effort, the author decided to considered the assumptions stated above, starting from the work of [5],[16].

3.1 Reference Frames

For the development of the equations of motion, two reference frames have been considered [4]:

- **Inertial reference frame $\mathcal{F}_I$:** The inertial reference frame is centered in the landing site, and is Up-East-North reference frame, such that the x_I-axis points Up, y_I-axis East, while z_I-axis North.
- **Body-frame $\mathcal{F}_B$:** The body frame attached to the rocket's center of mass and has the x_B-

axis aligned with the thrust vector when it is not gimbaled, y_B-axis perpendicular with respect to x_B-axis and the z_B-axis completes the right-handed system, where the subscript B means "Body", such that when the rocket is landed $\mathcal{F}_B$ coincides with $\mathcal{F}_I$.

Fig. 1 illustrates these two reference frames.

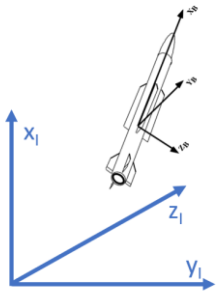

Fig. 1: Inertial (Blue) and Body-Frame (Black).

The rotation matrix between the reference frames is expressed by the quaternion formulation, with the leading scalar element convention. In addition, it is remarked that the controller actuates the TVC by rotating the nozzle of the engine by a certain angle. Ref. [9] gives the expressions of the rotation matrices and the definition of the gimbal angles.

3.2 Nonconvex Problem

Here the equations of motion, the state and inputs constraints are reported. Moreover, the environment and the aerodynamic model exploited in this work are outlined. One of the novelties in this paper is the formulation of the landing problem by considering a variable atmospheric model and a variable gravitational field.

3.2.1 Equations of motion

The mass-depletion dynamic is assumed to be an affine function of the thrust magnitude and it is defined by:

$$\dot{m}(t) = -\frac{\left\|\boldsymbol{T}_B\right\|_2}{I_s g_0} - \beta_{\dot{m}} \tag{8}$$

where $\beta_{\dot{m}}$ is is the term related to the reduction of the specific impulse due to the atmospheric backpressure and is expressed by [14]:

$$\beta_{\dot{m}} = \frac{A_{nozzle} P_{amb}}{I_s g_0} \tag{9}$$

The translational motion is ruled by:

$$\dot{\boldsymbol{r}}_I(t) = \boldsymbol{v}(t)$$
$$\dot{\boldsymbol{v}}_I(t) = \frac{1}{m}\mathbf{R}_{B\to I}(t)\left(\boldsymbol{T}_B(t) + \boldsymbol{A}_B(t)\right) + \boldsymbol{g}_I \tag{10}$$

The gravitational field $\boldsymbol{g}_I = [g\ 0\ 0]'$ depends on the altitude.

The rotational motion is governed by Euler rigid-body attitude dynamics:

$$\dot{\boldsymbol{q}}(t) = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_B(t))\boldsymbol{q} \tag{11}$$
$$\dot{\boldsymbol{\omega}}_B(t) = \mathbf{J}^{-1}(\boldsymbol{r}_{T,B} \times \boldsymbol{T}_B(t) + \boldsymbol{r}_{cp,B} \times \boldsymbol{A}_B(t) - \boldsymbol{\omega}_B(t) \times \mathbf{J}\boldsymbol{\omega}_B(t))$$

$\boldsymbol{\Omega}(.)$ is the skew-symmetric matrix of the angular velocities, while $\mathbf{J}$ is the inertia matrix. $\boldsymbol{r}_{i,j}$ are the arms with respect to the centre of mass of the launcher.

The nonlinear dynamics are summarized by:

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)) \tag{12}$$

3.2.2 Atmospheric Model

The equations of motion are largely affected by the aerodynamics forces and torques generated by the atmosphere on the vehicles. For this reason, an accurate model of the atmosphere must be selected. The choice of such a model has to be a trade-off between the accuracy of the model itself and the complexity introduced in the optimization model. By taking into account both characteristics, it has been decided to adopt the exponential atmospheric model.

The density and the pressure variations are defined as follows:

$$\rho = \rho_0 e^{-\frac{x}{H}}$$
$$P = P_0 e^{-\frac{x}{H}} \tag{13}$$

where H is the scale height factor, and x is the altitude. A deeper insight can be found in [9].

3.2.3 Aerodynamic Model

A tractable aerodynamic model is presented which approximates the relationship between the aerodynamic forces and the velocity [7], [15]. The model allows to express the aerodynamics forces in $\mathcal{F}_B$ coordinates as follows:

$$\boldsymbol{A}_B(t) = \frac{1}{2}\rho\left\|\boldsymbol{v}_{I(t)}\right\|_2 S_A C_A \mathbf{R}_{I\to B}(t)\boldsymbol{v}_I(t) \tag{14}$$

C_A is the matrix of constant aerodynamic coefficient, which is a symmetric-positive-definite matrix. Due to the axis-symmetry of most of the rocket-powered vehicles, C_A takes this form:

$$C_A = \begin{bmatrix} c_{a,x} & 0 & 0 \\ 0 & c_{a,yz} & 0 \\ 0 & 0 & c_{a,yz} \end{bmatrix}$$

(15)

If $c_{a,x}$ is equal to $c_{a,yz}$ the model is called *spherical aerodynamic model*, because $\boldsymbol{A}_B(t)$ is always anti-parallel with respect to the velocity in body frame. If $c_{a,x}$ is different to $c_{a,yz}$ the model is called *ellipsoidal aerodynamic model* and $\boldsymbol{A}_B(t)$ has also orthogonal component. A more detailed explanation is given by [7]. In Fig. 2 a representation of the model is illustrated.

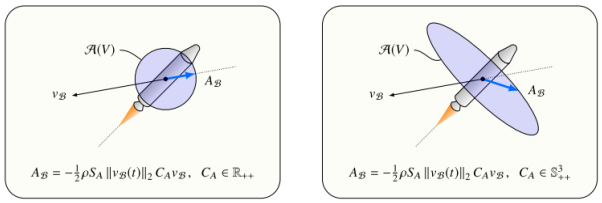

Fig. 2: Spherical aerodynamic model (left) and Ellipsoidal aerodynamic model (right) [7].

3.2.4 State Constraints

The problem embeds several state constraints. The first one is a lower bound of the mass. For each time instant, the mass cannot be smaller than the dry mass. This constraint is enforced by:

$$m(t) \geq m_{dry} \tag{16}$$

The second constraint refers to the glide scope angle, which allows to restrict the path of the vehicle to lie within an upward-facing cone:

$$\boldsymbol{e}_1 \cdot \boldsymbol{r}_I(t) \geq \tan(\gamma_{gs}) \left|\left|[\boldsymbol{e}_2\ \boldsymbol{e}_3]^T \boldsymbol{r}_I(t)\right|\right|_2 \tag{17}$$

where $\mathbf{e}_i$ are the versors. The angle $\gamma_{gs} \in [0,90°)$. The third constraint bounds the tilt angle which is the angle between the *x*-axis of $\mathcal{F}_I$ and the *x*-axis of $\mathcal{F}_B$. It takes this form:

$$\cos(\theta_{max}) \leq \boldsymbol{e}_{I1}^T \mathbf{R}_{B\to I}(t) \boldsymbol{e}_{B1} \tag{18}$$

The fourth constraint limits the maximum angular speed of the launcher, and it is given by:

$$\boldsymbol{\omega}(t) \leq \boldsymbol{\omega}_{max} \tag{19}$$

Then, an additional constraint must be considered in the formulation of the problem in order to preserve the unit norm of the quaternions:

$$\left|\left|\boldsymbol{q}(t)\right|\right| = 1 \tag{20}$$

Lastly, the boundary conditions must be set. In this paper the initial attitude has not been constrained, while it is constrained when the problem is solved during the flight in order to have continuity in the solution.

3.2.5 State-triggered Constraint

The state-triggered constraints (STC) are particular kinds of constraints enforced only when a state-dependant condition is satisfied. These constraints work like an *if*-statement on the solution variables. In this work, a continuous formulation of the STC has been used; however, in this subsection it is reported the application only. A deeper explanation of the theoretical background can be found in [6], [7], [16], [17], [18] and [19]. In this work, the STC has been used to formulate the constrain of the angle of attack of the vehicle. This is managed through a simplified *q*-*α* bound which imposes an angle of attack (α) constraint when the dynamic pressure (q) is larger than a prescribed value ($\hat{q}_{dyn}$), similar to what has been done in [6], [7], [16] and [19]. The STC has the following form:

$$\begin{aligned} &\text{if } \frac{1}{2}\rho\left|\left|\boldsymbol{v}_B(t)\right|\right|_2^2 \geq\ \hat{q}_{dyn} \\ &\rightarrow\ -\boldsymbol{e}_1 \cdot \boldsymbol{v}_B(t) \geq \cos(\alpha_{max}) \left|\left|\boldsymbol{v}_B(t)\right|\right|_2 \end{aligned} \tag{21}$$

Then, Eq. 21 is rewritten in order to have a continuous formulation for the optimization problem: it is constituted by a *trigger*-function (g_α) and a *constraint*-function (c_α):

$$\begin{aligned} h_\alpha\left(\boldsymbol{v}_I(t), \boldsymbol{q}(t)\right) &= -min(g_\alpha, 0) \cdot c_\alpha \leq 0 \\ c_\alpha\left(\boldsymbol{v}_I(t), \boldsymbol{q}(t)\right) &= \boldsymbol{e}_1 \cdot \boldsymbol{R}_{I\to B}\boldsymbol{v}_I(t) + cos(\alpha_{max}) \left|\left|\boldsymbol{v}_B(t)\right|\right|_2 \\ g_\alpha\left(\boldsymbol{v}_I(t), \boldsymbol{r}_I(t)\right) &= \hat{q}_{dyn} - \frac{1}{2}\rho\left|\left|\boldsymbol{v}_B(t)\right|\right|_2^2 \end{aligned} \tag{22}$$

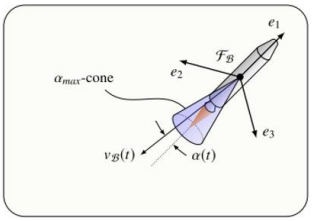

Fig. 3: Angle of attack constraint

Fig. 3 illustrates the geometrical representation of this constraint.

3.2.6 Input Constraints

The powered descent guidance problem has been formulated with two input (or control) constraints. The first one is related to the maximum and the

minimum thrust magnitude that the engine can grant. The second constraint limits the thrust vector control up to a maximum gimbal angle.
The former is defined as:

$$0 < T_{min} \leq \left|\left|\boldsymbol{T}_B(t)\right|\right|_2 \leq T_{max} \tag{23}$$

where T_{min} and T_{max} are the lower and upper bounds. The latter is given by:

$$\cos(\delta_{max}) \left|\left|\boldsymbol{T}_{B(t)}\right|\right|_2 \leq \boldsymbol{e}_1 \cdot \boldsymbol{T}_B(t) \tag{24}$$

An additional input constraint should be considered in order to have a more general and complete guidance problem. This constrain limits the throttle rate, avoiding that the thrust will change instantaneously. However, in this work it has not been embedded to maintain the problem simpler. The smoothness of the thrust profile has been checked as posteriori, and for the development of the simulator, the dynamics of the engine has been modelled by means of low-pass filter, as explained in Section 4.

3.2.7 Cost Function

The last step that must be done to properly design the 6-DoF landing problem is the definition of the cost function. the objective of the powered descent guidance is to find a trajectory to land safely in a prescribed location, by using the least amount of fuel on-board. This goal can be achieved by the selection of several cost functions, such as the maximization of the final mass or the minimization of the time-of-flight, indeed the fuel consumption is strictly increasing monotonic function with respect to the time. In this paper, the time-of-flight has been selected as cost function, since better outcomes are experienced.

3.2.8 Summary Nonconvex Problem

Nonconvex Problem
Cost Function: $\text{minimize}_{t,\boldsymbol{u}(t)}\ J = t_f$
Dynamics: $\dot{\boldsymbol{m}}(\boldsymbol{t}) = -\frac{\|\boldsymbol{T}_B\|_2}{\boldsymbol{I_s g_0}} - \boldsymbol{\beta}_{\dot{m}}$ $\dot{\boldsymbol{r}}_I(t) = \boldsymbol{v}(t)$ $\dot{\boldsymbol{v}}_I(t) = \frac{1}{m}\mathbf{R}_{B\to I}(t)\left(\boldsymbol{T}_B(t) + \boldsymbol{A}_B(t)\right) + \boldsymbol{g}_I$ $\dot{\boldsymbol{q}}(t) = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}_B(t))\boldsymbol{q}$ $\dot{\boldsymbol{\omega}}_B(t) = \mathbf{J}^{-1}(\boldsymbol{r}_{T,B} \times \boldsymbol{T}_B(t) + \boldsymbol{r}_{cp,B} \times \boldsymbol{A}_B(t) - \boldsymbol{\omega}_B(t) \times \mathbf{J}\boldsymbol{\omega}_B(t))$
State Constraints: $m(t) \geq m_{dry}$ $\boldsymbol{e}_1 \cdot \boldsymbol{r}_I(t) \geq \tan(\gamma_{gs}) \left|\left|[\boldsymbol{e}_2\ \boldsymbol{e}_3]^T \boldsymbol{r}_I(t)\right|\right|_2$ $\cos(\theta_{max}) \leq \boldsymbol{e}_{I1}^T \mathbf{R}_{B\to I}(t)\boldsymbol{e}_{B1}$ $\boldsymbol{\omega}(t) \leq \boldsymbol{\omega}_{max}$ $\left|\left|\boldsymbol{q}(t)\right|\right| = 1$
State-triggered Constraints: $h_\alpha(\boldsymbol{v}_I(t), \boldsymbol{q}(t)) = -min(g_\alpha, 0) \cdot c_\alpha \leq 0$
Input Constraints: $0 < T_{min} \leq \left|\left|\boldsymbol{T}_B(t)\right|\right|_2 \leq T_{max}$ $\cos(\delta_{max}) \left|\left|\boldsymbol{T}_{B(t)}\right|\right|_2 \leq \boldsymbol{e}_1 \cdot \boldsymbol{T}_B(t)$

3.3 Convex Problem

The original problem is convexified by mean of the Successive Convexification algorithm. The nonconvex continuous-time free-final-time optimal control problem is converted into a convex linear-time-varying discrete-time fixed-final-time subproblem, specifically an SOCP problem.

3.3.1 Free-final Time to Fixed-final Time

In order to make the original problem explicitly time dependent and to convert it from a nonconvex continuous-time free-final-time optimal control problem to an equivalent convex linear-time-varying discrete-time fixed-final-time subproblem, the dynamics outlined in Eq. 12 is expressed in terms of normalized trajectory time, $\tau \in [0, 1]$. This step is performed as follows:

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = \frac{\mathrm{d}\tau}{\mathrm{d}t}\frac{\mathrm{d}}{\mathrm{d}\tau}\boldsymbol{x}(t) \tag{25}$$

The inverse of the derivative of τ with respect to t defines a dilatation coefficient σ:

$$\sigma = \left(\frac{\mathrm{d}\tau}{\mathrm{d}t}\right)^{-1} \tag{26}$$

In this way, the dynamics is recasted such that:

$$\frac{d}{d\tau}\boldsymbol{x}(\tau) = \sigma f(\boldsymbol{x}(\tau), \boldsymbol{u}(\tau)) \tag{27}$$

3.3.2 Convexification of the Equations of Motion

The convexification of the equations of motion is performed by linearizing the nonlinear dynamics about the solution of the previous iteration. This operation guarantees the convexity of the subproblem. The original continuous time problem is transformed into an LTV subproblem:

$$\boldsymbol{x}_{k+1}^{i} = \sigma^{i}\,\bar{\boldsymbol{\Sigma}}_{k}^{i} + \bar{\boldsymbol{A}}^{i}{}_{k}\boldsymbol{x}_{k}^{i} + \bar{\boldsymbol{B}}^{i}{}_{k}\boldsymbol{u}_{k}^{i} + \bar{\boldsymbol{z}}_{k}^{i} + \boldsymbol{\nu}_{k}^{i} \quad (28)$$

where the matrices are:

$$\begin{aligned}\boldsymbol{A}(\tau) &= \sigma^{i-1}\frac{\partial f}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}^{i-1},\boldsymbol{u}^{i-1}}\\ \boldsymbol{B}(\tau) &= \sigma^{i-1}\frac{\partial f}{\partial \boldsymbol{u}}\Big|_{\boldsymbol{x}^{i-1},\boldsymbol{u}^{i-1}}\\ \boldsymbol{z}(\tau) &= -\boldsymbol{A}(\tau)\boldsymbol{x}^{i-1} - \boldsymbol{B}(\tau)\boldsymbol{u}^{i-1}\\ \boldsymbol{\Sigma}(\tau) &= f(x^{i-1}(\tau), u^{i-1}(\tau))\end{aligned} \quad (29)$$

The dynamics is relaxed with the addition of the virtual controls $\boldsymbol{\nu}$. This set of equality constraints must be satisfied in each time instant $k \in [1, K\text{-}1]$.

3.3.3 Convexification of the Constraints

The formulation of the convex problem requires the convexification of three constraints: two state constraints, more precisely the norm of the quaternions and the angle of attack constraint, and one input constraint, which is the lower bound of the thrust magnitude.

The convexification of the Eq. 20 is obtained by performing a first-order Taylor expansion approximation, evaluated around the *(i-1)*th iteration:

$$\left|\left|\boldsymbol{q}^{i-1}\right|\right|_2 + \frac{(\boldsymbol{q}^{i-1})^T}{\left|\left|\boldsymbol{q}^{i-1}\right|\right|_2}(\boldsymbol{q}^{i} - \boldsymbol{q}^{i-1}) = 1 \quad (30)$$

This expression is enforced in each $k \in [1, K]$. Even the convexification of the angle of attack constraints (Eq. 22) is obtained thanks to the first-order Taylor expansion. Due to the min function, the approximation is given as follow:

$$\begin{cases} h_{\alpha,k}(\boldsymbol{\zeta}_k^{i-1}) + \boldsymbol{H}_{\alpha,k}\delta\boldsymbol{\zeta}_k^{i} \le 0 & \text{if } g_\alpha(\boldsymbol{\zeta}_k^{i-1}) < 0 \\ 0 & \text{otherwise} \end{cases} \quad (31)$$

where $\boldsymbol{\zeta}$ is the combined vector of the variables and $\boldsymbol{H}_{\alpha}$ is the jacobian.

Lastly, the lower bound of the thrust magnitude is convexified by linearizing the equation around the previous iteration, obtaining the following expression:

$$\begin{aligned} h_{T,k}(\boldsymbol{u}_k) &= T_{min} - \left|\left|\boldsymbol{T}_{B,k}\right|\right|_2 \\ h_{T,k}(\boldsymbol{u}_k^{i-1}) &+ \boldsymbol{H}_{T,k}\delta\boldsymbol{u}_k \le 0 \end{aligned} \quad (32)$$

This constraint must be valid for $k \in [1, K\text{-}1]$.

3.3.4 Trust Regions Constraints

As anticipated in Sec. 2.3, trust regions constraints are added to problem in order to limit the deviation between two adjacent iterations. Beyond to the ones relative to the state and inputs, a constraint related to the time of flight is introduced:

$$\left|\left|\sigma^{i} - \sigma^{i-1}\right|\right|_2 \le \Delta_\sigma \quad (33)$$

Then, Δ_σ is penalized in the cost function.

3.3.5 Cost Function of the Subproblem

The original cost function is augmented with penalization terms relative to the trust region radius and the virtual controls norm. Moreover, the time of flight is replaced by σ, due to the time-normalization:

$$\text{minimize } J = \sigma^{i} + w_\nu S_\nu^{i} + w_\Delta S_\Delta^{i} + w_\sigma \Delta_\sigma^{i} \quad (34)$$

3.3.6 Summary

Convex Subproblem

Cost Function:

$$\text{minimize } J = \sigma^{i} + w_\nu S_\nu^{i} + w_\Delta S_\Delta^{i} + w_\sigma \Delta_\sigma^{i}$$

Dynamics:

$$\boldsymbol{x}_{k+1}^{i} = \sigma^{i}\,\bar{\boldsymbol{\Sigma}}_{k}^{i} + \bar{\boldsymbol{A}}^{i}{}_{k}\boldsymbol{x}_{k}^{i} + \bar{\boldsymbol{B}}^{i}{}_{k}\boldsymbol{u}_{k}^{i} + \bar{\boldsymbol{z}}_{k}^{i} + \boldsymbol{\nu}_{k}^{i} \quad k \in [1, K\text{-}1].$$

State Constraints:

$$\begin{aligned} m_k &\ge m_{dry} \\ \boldsymbol{e}_1 \cdot \boldsymbol{r}_{I,k} &\ge \tan(\gamma_{gs})\left|\left|[\boldsymbol{e}_2\ \boldsymbol{e}_3]^T\boldsymbol{r}_{I,k}\right|\right|_2 \\ \cos(\theta_{max}) &\le \boldsymbol{e}_{I1}^T\mathbf{R}_{B\to I,k}\boldsymbol{e}_{B1} \\ \boldsymbol{\omega}_k &\le \boldsymbol{\omega}_{max} \quad k \in [1, K]. \\ \left|\left|\boldsymbol{q}^{i-1}\right|\right|_2 &+ \frac{(\boldsymbol{q}^{i-1})^T}{\left|\left|\boldsymbol{q}^{i-1}\right|\right|_2}(\boldsymbol{q}^{i} - \boldsymbol{q}^{i-1}) = 1 \end{aligned}$$

State-triggered Constraints:

$$h_{\alpha,k}(\boldsymbol{\zeta}_k^{i-1}) + \boldsymbol{H}_{\alpha,k}\delta\boldsymbol{\zeta}_k^{i} \le 0 \quad \text{if } g_\alpha(\boldsymbol{\zeta}_k^{i-1}) < 0 \quad k \in [1, K\text{-}1].$$

Input Constraints:

$$\begin{aligned} \left|\left|\boldsymbol{T}_{B,k}\right|\right|_2 &\le T_{max} \\ h_{T,k}(\boldsymbol{u}_k^{i-1}) + \boldsymbol{H}_{T,k}\delta\boldsymbol{u}_k &\le 0 \\ \cos(\delta_{max})\left|\left|\boldsymbol{T}_{B,k}\right|\right|_2 &\le \boldsymbol{e}_1 \cdot \boldsymbol{T}_{B,k} \end{aligned} \quad k \in [1, K\text{-}1].$$

Trust Regions:

$$\begin{aligned} \delta\boldsymbol{x}_k^{i} \cdot \delta\boldsymbol{x}_k^{i} + \delta\boldsymbol{u}_k^{i} \cdot \delta\boldsymbol{u}_k^{i} &\le \Delta_{\mathrm{k}}^{i} \\ \left|\left|\boldsymbol{\Delta}^{i}\right|\right|_2 &\le S_\Delta^{i} \\ \left|\left|\sigma^{i} - \sigma^{i-1}\right|\right|_2 &\le \Delta_\sigma \end{aligned}$$

$k \in [1, K-1]$.
Virtual Controls: $\left\|\left\|\boldsymbol{v}_k^i\right\|\right\|_2 \leq k_{v,k}^i$ $\left\|\left\|\boldsymbol{k}_{v,k}^i\right\|\right\|_2 \leq S_v^i$ $k \in [1, K-1]$.

It is worth to underline that the optimization problem is scaled, for more detailed explanation see Ref [9].

3.4 Example of Trajectory

In this subsection, an example of a trajectory generated solving the 6-DoF guidance problem is shown, and the results in terms of method behavior are provided.

For this example, the parameters adopted are given in Tab. 1 and Tab. 2 in Sec. 5.

It is worth to note that, due to numerical stability, both the boundary conditions and the parameters of the problem are scaled with appropriately chosen scaling-factors in order to have values ∈ [-10, 10] range. The outcomes are presented in Fig. 4 – Fig. 6.

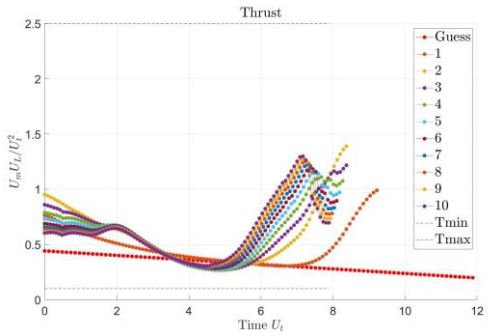

Fig. 4: Thrust Profile convergence.

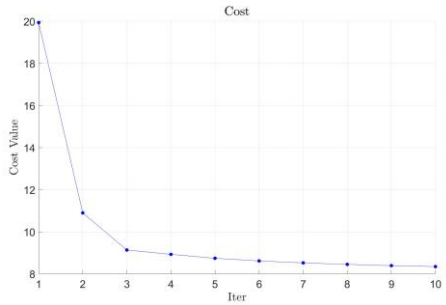

Fig. 5: Cost function convergence.

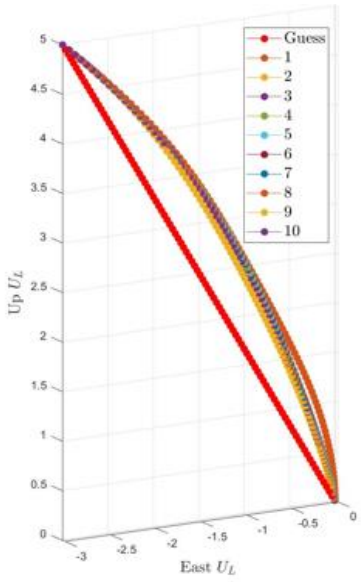

Fig. 6: Trajectory convergence

From the figures, it is clear the iterative nature of the successive convexification method: both the thrust profile (Fig. 4) and the trajectory (Fig. 6) converge to a final solution starting from a dynamically inconsistent guess solution, while satisfying the constraints (Fig. 4). This behaviour is confirmed also in Fig. 5 where the value of the cost function decreases until it flattens, meaning that the convergence is met.

Fig. 7 – 9 show the final trajectory and the final profiles.

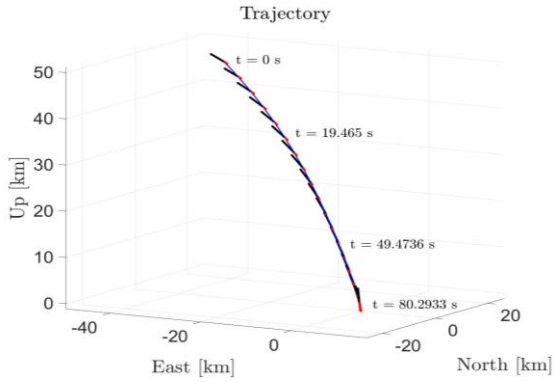

Fig. 7: Final trajectory (blue line: trajectory, red line: thrust direction, black line: attitude of the vehicle).

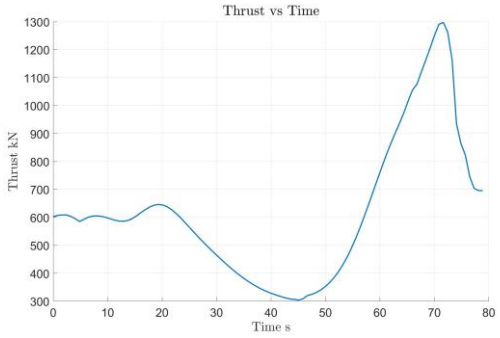

Fig. 8: Thrust profile.

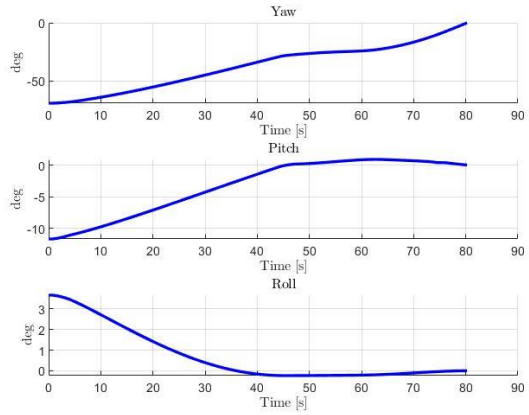

Fig. 9: Euler angles profile.

The outcomes show that the obtained solution are smooth both for the thrust profile and the Euler angles. Further details are presented in [9].

4. G&C Architectures

Once the guidance problem is set, a low-level controller is required to generate the commands so that the reference trajectory is followed by the system, in order to meet the mission goals. Indeed, one of the aims of this work is to test the guidance algorithm in highly nonlinear scenarios, with large dispersions of uncertain parameters by using a nonlinear simulator, validated against an industrial simulation framework. For this reason, two control architectures have been developed, to link the guidance with the simulator. In the closed-loop guidance frame, both cases imply a replanning online of the trajectory by solving the 6DoF guidance problem depending on the flight conditions. It is noted that, in this work the navigation problem is not addressed, so it is assumed that the full state of the system is known at each time instant. However, a simplified performance model of the navigation is considering by introducing white noise to the measurements.

The first architecture proposed in this work is based on an MPC-Guidance and a PID-Controller. The term MPC-Guidance has been adopted since the trajectory is re-optimized during the flight, taking into account the current conditions and looking ahead at a given horizon, as in a typical MPC problem. The 6DoF guidance optimal control problem is solved with a prescribed frequency in order to generate a new path, based on the past solution and the new conditions. However, in this architecture, the MPC methodology is used to implement the function of guidance only, (i.e., the frequency of the MPC optimization is considered to be much smaller than that of the fastest dynamics of the system, thus seen as a function generating reference trajectories), while the control (high-frequency) action is performed by the PID. The choice of a PID controller has been made since there is a well-established knowledge of this kind of controller in aerospace applications, and because of the robustness and the relatively simplicity of the design. Different works use other kind of controllers such as H_∞, LPV or Adaptive controller.

The second architecture proposed in this work is closer to a classical Model Predictive Control approach: the optimal control problem is solved with higher frequency with respect to the previous architecture and the solution is used to feed directly the simulator. The 6DoF landing problem, indeed, is solved repeatedly with the new flight conditions [20]. In theory, this strategy is closer to an optimal control implementation, but, as it is possible to see in the next subsections, it has some limitations.

4.1 MPC-G & PID-C

Here the design of the PID control is described. The first step requires the linearization of the model used to describe the system around a reference trajectory. Then, the 6DoF dynamics is decomposed into two 3DoF dynamics, in order to obtain two single-input single-output (SISO) systems and to design the controller based on Linear Control Theory. Indeed, the symmetry of the vehicle and the nature of the TVC control allow the decoupling of the original system. The control inputs are the deflection angles of the nozzle (δ_y and δ_z). This assumption is verified also by the eigenvalue analysis, as it is possible to see in Fig. 10.

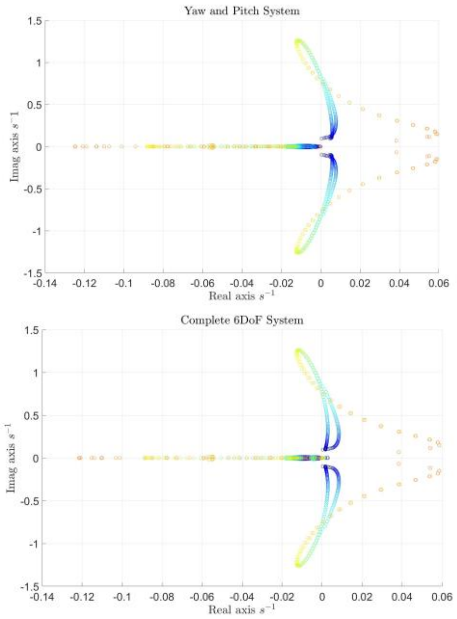

Fig. 10: Poles of the decoupled system (above) and of the original system (below).

Moreover, due to the time-varying nature of the system, a gain-scheduled PID has been adopted, in order to get stability and similar performance in each phase of the flight. The gains have been tuned by using the MATLAB app PID tuner, in order to meet the following performance: overshoot less than 10%, settling time less than 1 second and at least 60° of phase margin. For the detailed analysis see [9]. The pitch and yaw controllers have been designed by following this strategy.

For what concerns the control of the thrust magnitude, a different approach has been considered, indeed the design of a PID control able to modulate the magnitude of the thrust is not a trivial work. In fact, if the thrust is considered as an input variable, the model cannot be decoupled as before. For this reason, the thrust profile obtained by solving the SCvx problem is provided to the nonlinear simulator. This approach, however, does not compensate for the tracking errors as in a classical control implementation. This action is made by the recomputation online of the guidance problem. As stated in the previous chapter, indeed, the 6DoF powered descent guidance problem is solved online depending on the current conditions of the flight. This generates new solutions with new thrust profiles, which consider the error accumulated during the dynamics of the simulator.

The thrust profile is not provided directly to the simulator, but a simple dynamics model of the engine (low-pass filter) is added to the model in order to simulate the intrinsic physical of the device. To compensate the delay introduced by the dynamic model, a PI-controller has been designed, where the integral action needs to eliminate the offset.

Fig. 11 illustrates the schematic representation of this architecture.

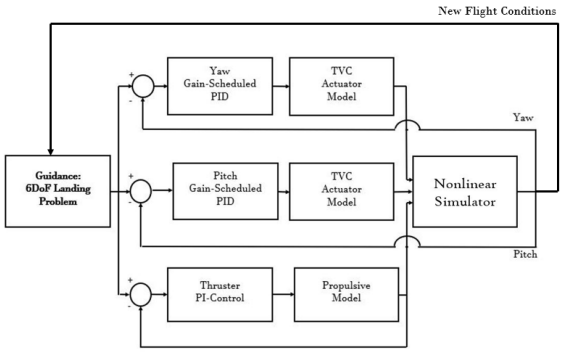

Fig. 11: Schematic representation of the first architecture.

In this paper, the recomputation frequency of the guidance is 0.05 Hz, so every 20 s a new reoptimized solution is computed.

4.2 MPC-G&C

The second architecture is presented, with a methodology similar to the classical Model Predictive Control approach. Indeed, in this case, the MPC has the role of both generating new trajectories and controlling the system. In order to cope with this demand, the 6DoF powered descent guidance is solved at a high frequency. However, it has been noticed that by solving the problem stated in Sec. 3 with a frequency larger than 0.1 Hz, the optimization problem becomes excessively sensitive to the model of the system, ultimately leading to convergence problems. To cope this issue three different approaches have been assessed. In the first one, the frequency is reduced, and the terminal state is given as hard constraint; in the second one, the frequency remains relatively high (1 Hz) and the terminal state is given with a penalization term in the cost function. The third one is a combination of the two. The choice has been driven by three parameters: computational effort, convergence issues and final state error. The third strategy has been selected, because allows to have good final state errors, with less computational effort.

Fig. 12 illustrates a schematic representation of this architecture:

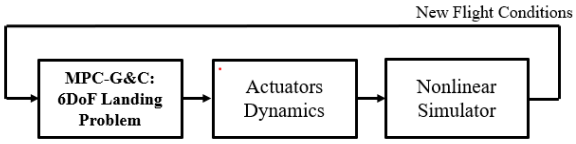

Fig. 12: Schematic representation of the second architecture.

In this paper, it has not been used a receding horizon technique, because the successive convexification tool is not suitable for this approach, so, the whole trajectory has been considered as horizon.

5. Performance Evaluation

In this section the performances of the proposed architectures are assessed with the nonlinear simulator by mean of a sensitivity analysis with respect to several parameters of the problem.

5.1 Nonlinear Simulator

The nonlinear simulator has been implemented with a 6DoF dynamics, by considering the same reference frames reported in Sec. 3.1. The simulator takes into account both constant winds and winds gusts computed by U.S. Naval Research Laboratory [21], the flexibility of the launcher with the introduction of additional forces and torques, model uncertainties and variable inertia during the flight. All the assumptions are reported in [9]. The nonlinear simulator has been validated against "DEIMOS-FES Atmospheric Flight Simulation Core": the discrepancies between the models are reported in [9].

5.2 Sensitivity Analysis

The sensitivity analyses are performed by varying the initial conditions, the features of the launch vehicles, such as initial mass and inertia, and the model uncertainties (thrust misalignment and centre of mass shift), and the atmospheric conditions, such as wind gust and uncertainties on the values of atmospheric density and pressure, with respect to the nominal conditions. Firstly, the sensitivity is performed by considering one set of dispersion at a time, and then the worst-case scenario is assessed with the dispersion of the all the sets. This final condition is the worst-case scenario, since all the data vary from the nominal ones. It is noted that the constant winds are always active, while the wind gusts are considered as a perturbation with respect to the nominal conditions. The data of the problem used to perform the sensitivity analyses are uniformly distributed within prescribed range. The bounds of the range have been set by carrying out a preliminary campaign of tests. The optimization problem parameters are kept constant. The emphasis is given to the evaluation of the final position and velocity.

5.2.1 Nominal Trajectory

Here the parameters used for the computation of the nominal trajectory have been reported (Tab. 1 and Tab. 2). Moreover, the errors on the final conditions are outlined in the tables (Tab. 3 and Tab. 4).

$\boldsymbol{r}_{I,0} = [50,\ -30,\ 0]\ km$	$\boldsymbol{r}_{I,f} = [0,0,0]\ km$
$\boldsymbol{v}_{I,0} = [-700,\ 800,\ -50]\ m/s$	$\boldsymbol{v}_{I,f} = [-5,0,0]\ m/s$
$\boldsymbol{\omega}_{I,0} = [0,0,0]\ rad/s$	$\boldsymbol{\omega}_{I,f} = [0,0,0]\ rad/s$
$m_0 = 45000\ kg$	$\boldsymbol{q}_f = [1,0,0,0]$

Tab. 1: Initial conditions and desired final conditions.

w_Δ	1	C_A	diag([0.82, 0.82, 0.82])
w_ν	1000	ρ_0	1.225 kg/m^3
w_σ	0.75	$t_{f,guess}$	120 s
Iter$_{max}$	10	L	40 m
tol_Δ	1e^{-3}	T_{max}	2.5 e^3 kN
K	100	T_{min}	1.2 e^2 kN
I_s	300 s	ω_{max}	[5 20 20] °/s
J	diag([0.08,6.04,6.04])e^6 kg/m^3	θ_{max}	90°
S_{cross}	10.52 m^2	γ_{gs}	10°
$\boldsymbol{r}_{cp}$	5 e$_1$ m	δ_{max}	10°
$\boldsymbol{r}_T$	-15 e$_1$ m	α	5°
A_{nozzle}	3.15 m^2	$\hat{q}_{dyn}$	4 e^4 Pa
g_0	9.81 m/s^2	m_{dry}	20000 kg
P_{amb}	101325 Pa		

Tab. 2: Problem data and optimization problem parameters.

	Results	Error
$\boldsymbol{r}_{I,f}$	[5.36 -0.56 0.13] m	5.39 m
$\boldsymbol{v}_{I,f}$	[4.69 -0.16 -0.01] m/s	0.35 m/s
$[\psi,\theta,\phi]_f$	[0.04 -0.21e^{-2} -0.03e^{-2}]°	[0.04 0.21e^{-2} 0.03e^{-2}]°
$\boldsymbol{\omega}_{B,f}$	[0 -0.16 0.77]°/s	0.79°/s
m_f	26332 kg	-

Tab. 3: Final conditions and errors for architecture MPC-G & PID-C.

	Results	Error
$\boldsymbol{r}_{I,f}$	[24.33 1.73 -8.07] m	25.70 m
$\boldsymbol{v}_{I,f}$	[-6.03 -0.04 -0.88] m/s	1.36 m/s
$[\psi,\theta,\phi]_f$	[-1.96 0.74 -0.19]°	[1.96 0.74 0.19]°
$\boldsymbol{\omega}_{B,f}$	[0 0.32 -1.30]°/s	1.34°/s
m_f	26567 kg	-

Tab. 4: Final conditions and errors for architecture MPC-G&C

5.2.2 Dispersions

The sensitivity analyses entail several dispersions of the data with respect to the nominal conditions. Four cases have been identified and organized as follows:

- Dispersion of the initial position and velocity.
- Dispersion of the launch vehicle features: initial mass, inertia, misalignment of the thrust and off set of the center of mass.
- Dispersion on the atmospheric conditions (wind gusts) and uncertainties of the atmospheric density and pressure.
- Worst-case scenario, where all the dispersion stated above are active.

For each case, 100 samples have been considered for the first architecture, and 50 samples for the second one, due to the higher computational effort. In Tab. 5 and 6 are reported the range of the set.

Dispersion on the initial conditions
$\boldsymbol{r}_{I,0} = \boldsymbol{r}_{I,0_{Nominal}} \pm 5\ km$
$\boldsymbol{v}_{I,0} = \boldsymbol{v}_{I,0_{Nominal}} \pm 100\ m/s$
Dispersion on the launch vehicle parameters
$m_0 = m_{0_{Nominal}} \pm 5000\ kg$
$x_{cg}\text{Shift} = x_{cg_{Nominal}} \pm 0.75\ m$
Thrust misalignement angle = $\pm 0.1°$
Dispersion on the atmospheric conditions
$\rho = \rho_{Nominal} \pm 10\%$
$P = P_{Nominal} \pm 10\%$
$\boldsymbol{V}_{Gust} = \pm 50 \frac{m}{s}$ at $h \in [2000\ 11000]\ m$
$\boldsymbol{r}_{I,0} = \boldsymbol{r}_{I,0_{Nominal}} \pm 5\ km$

Tab. 5: Set first architecture

Dispersion on the initial conditions
$\boldsymbol{r}_{I,0} = \boldsymbol{r}_{I,0_{Nominal}} \pm 2\ km$
$\boldsymbol{v}_{I,0} = \boldsymbol{v}_{I,0_{Nominal}} \pm 50\ m/s$
Dispersion on the launch vehicle parameters
$m_0 = m_{0_{Nominal}} \pm 2000\ kg$
$x_{cg}\text{Shift} = x_{cg_{Nominal}} \pm 0.3\ m$
-
Dispersion on the atmospheric conditions
$\rho = \rho_{Nominal} \pm 10\%$
$P = P_{Nominal} \pm 10\%$
$\boldsymbol{V}_{Gust} = \pm 50 \frac{m}{s}$ at $h \in [2000\ 7000]\ m$

Tab. 6: Set second architecture.

In the next subsection, the results of the analyses are reported, showing in particular the dispersions of the final position and the velocity, and the trajectories for the worst-case scenario.

5.2.3 Results

In this subsection, the outcomes of the sensitivity analyses are shown. In particular, the results regarding the worst-case scenario.

The dispersions of the final position and the final velocity for the first G&C architecture are illustrated in Fig. 13.

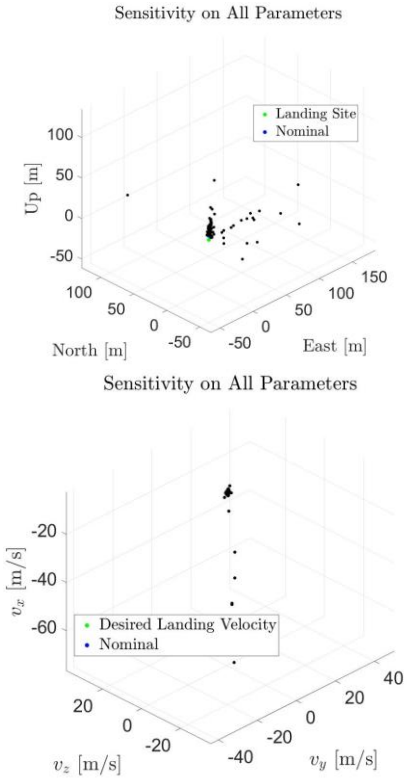

Fig. 13: Final position dispersion (above), final velocity dispersion (below), MPC-G & PID-C.

The dispersions relative to the second architecture are depicted in Fig. 14:

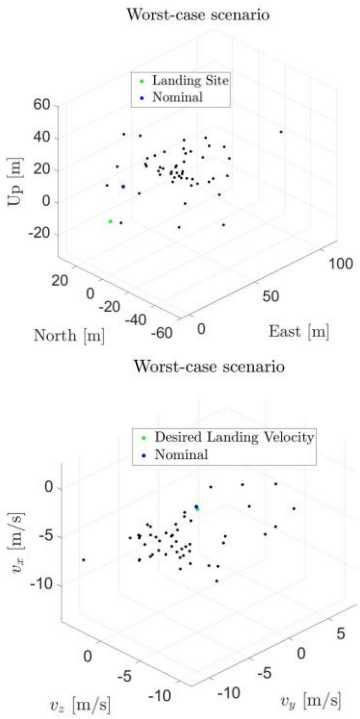

Fig. 14: Final position dispersion (above), final velocity dispersion (below), MPC-G&C.

As it is possible to see from these figures, the first architecture allows guarantees a superior performance in terms of final state errors minimization. Indeed, the mean errors on the final position are 23.50 m for the first architecture and 51.74 m for the second one; while, the mean errors on the final position are 3.05 m/s for the first strategy and 6.26 m/s for the second one. Below (Fig. 15) are reported also the trajectories generated by different initial conditions. Even by looking these plots, the first architecture is less sensitivity to the problem parameters than the second one, even if the ranges of the sets are wider. The points marked in green, indeed, means that the final error is less than 50 m for the position and 3 m/s for the velocity (6 m/s for the second architecture), otherwise the points are marked in red.

These results are confirmed also by looking the mean errors for all cases studied in Fig. 16. The labels on vertices stand for:

- IC: Initial Conditions;
- ATMO: Atmospheric conditions;
- RP: Rocket Parameters;
- COM: Worst-case scenario;

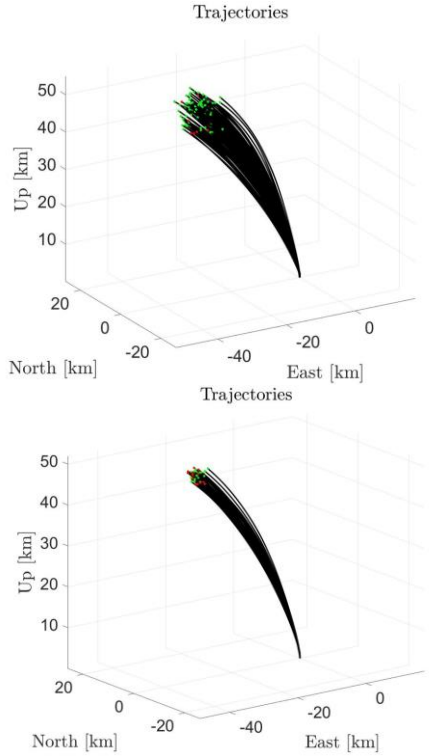

Fig. 15: Trajectory of the sensitivity analysis: above MPC-G&PID-C, below MPC-G&C.

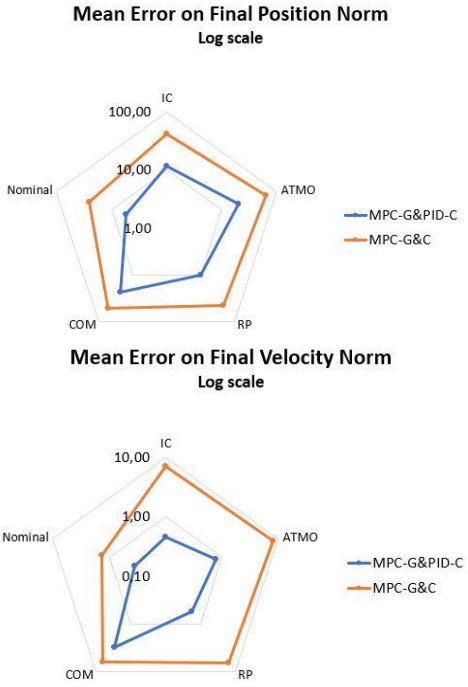

Fig. 16: Sensitivity analyses outcomes.

In all cases, the first architecture achieves better outcomes. It is interesting to note also that, without considering the worst-case scenario, the atmospheric

conditions generated the worst errors, meaning that an accurate atmospheric model is fundamental to have reliable outcomes and that the atmospheric disturbances have an important role for this kind of problems. The larger errors of the second architecture are due also for the soft constraints in the terminal state, but the main limitation is the converge issues. Indeed, for the second architecture, it has been experienced a high percentage of failure due to convergence issue: 18% on 50 samples. For the first architecture the percentage is 4%. A low-level control, indeed, allows to have better performance and reduces the likelihood of encountering convergence errors, reducing the gap between the guidance and the actual trajectory. The performance of the second architecture may be improved by considering an MPC formulation with online prediction of the disturbance and increasing the computation frequency of the guidance, requiring a larger computational effort. Ref. [9] reports also the assessment of the final attitude and the propellant mass consumption.

6. Conclusions

The obtained results from the sensitivity analyses have shown that the MPC-G&PID-C is robust to the dispersed parameters and the errors with respect to the desired final state conditions are deemed acceptable; moreover, the presence of a low-level controller allows to reduce the gap between the reference trajectory computed by the guidance and the actual trajectory followed by the vehicle, and decreases the likelihood of encountering convergence issues. The MPC-G&C, on the other hand, is much more sensitive to the dispersed parameters, while the use of a soft constraints for the terminal states leads to larger final errors. However, the main limitation of this second architecture is the relatively high probability of facing convergence issues, due to the discrepancy between the optimal trajectory and the real one, which leads to the violation of the imposed constraints. This behaviour can be mitigated by increasing the frequency of the computation online of the optimization problem, but in this case the computation time becomes the main bottleneck. For this reason, the implementation of a strategy that takes advantage of an inner-loop controller, as proposed in the first architecture, is more reliable. The drawback of the first architecture is that the PID-controller is designed around a reference trajectory, so an online tuning of the PID-gains is needed, or a reference trajectory has to be known a priori. However, different control strategies are available in the literature and can be considered in future works.

In conclusion, in the light of the obtained outcomes, the successive convexification Model Predictive Control guidance is a very promising solution for an online autonomous guidance implementation for precise pin-point landing.

6.1 Proposed future work

In this subsection, a set of topics for future investigation is summarized:

- Different versions of the successive convexification algorithms may be assessed, in particular with a deeper focus on the adopted discretization scheme, the definition of dynamical trust region radius constraints, and more accurate implementation of path constraints by exploiting additional buffer zone constraints.
- Some improvements may be considered such as a more accurate atmospheric model, more accurate computation of the inertia and the geometric features of the launch vehicle (centre of mass and centre of pressure), and more accurate model of the aerodynamic database by considering the dependence on the angle of attack and the Mach number.
- Additional path constrains may be added in the optimization problem, such as a throttle limit to ensure unfeasible thrust variation.
- The real-time implementation of the developed approach must be assessed, by coding the algorithm in C++ or converted in .mex files.

References

[1] E. M. Rocco F. Antônio da Silva Mota J. N. Hinckel and H. Schlingloff. *Trajectory Optimization of Launch Vehicles Using Object-oriented Programming.* J. Aerosp. Technol. Manag., São José dos Campos, 2018. doi: 0.5028/jatm. v10.948.

[2] S. Bennani J. Jerez S. Merkli and H. Strauch. *FORCES-RTTO: a tool for on-board realt-time autonomous trajectory planning*. Embotech GmbH, ESA ESTEC, Airbus DS GmbH, 2017.

[3] A. A. Martin. *Model predictive control for Ascent Load Management of a Reausable Launch Vehicle.* A thesis. Massachusetts Institute of Technology, 2002.

[4] B. Pan P. Lu. *Highly Constrained Optimal Launch Ascent Guidance*. In: Journal of Guidance, Control, and Dynamics 33.2 (2010), pp. 404–414. doi:10.2514/1.45632.

[5] M. Szmuk and B. Açıkmese. *Successive convexification for 6-DoF mars rocket powered landing with free-final-time.* In: AIAA Guidance,

Navigation, and Control Conference, 2018 210039 (2018), pp. 1–15. doi: 10.2514/6.2018-0617. eprint: 1802.03827.

[6] Y. Mao, M. Szmuk, and B. Acikmese. *Successive convexification of nonconvex optimal control problems and its convergence properties.* In: 2016IEEE 55th Conference on Decision and Control, CDC 2016 (2016). doi:10.1109/CDC.2016.7798816. eprint: 1608.05133.

[7] T.P.Taylor M. Szmuk and B. Acikmese. *Successive Convexification for Real-Time 6-DoF Powered Descent Guidance with State-Triggered Constraints.* University of Washington, Seattle.

[8] P. S. Lysandrou. *A Successive Convexification Optimal Guidance Implementation for the Pinpoint Landing of Space Vehicles.* MA thesis. University of Colorado, 2019.

[9] J. Guadagnini. *Model Predictive Control for Reusable Launcher Guidance Improvement.* MSc Thesis, Politecnico di Milano, Milano, 2020.

[10] D. Malyuta et al. *Discretization performance and accuracy analysis for the powered descent guidance problem.* In: AIAA Scitech 2019 Forum January (2019). doi: 10.2514/6.2019-0925.

[11] G. Fasano and J. D. Pintér. *Modeling and Optimization in Space Engineering.* Vol. 73. Springer, 2013, pp. 47–51. isbn: 9781461444688. doi: DOI10.1007/ 978-1-4614-4469-5.

[12] F. Topputo and C. Zhang. *Survey of direct transcription for low-thrust space trajectory optimization with applications.* In: Abstract and Applied Analysis 2014.June (2014). issn: 16870409. doi: 10.1155/2014/851720.

[13] H. B. Tsai. *Optimal trajectory designs and systems engineering analyses of reusable launch vehicles.* In: PhD Dissertation (2003).

[14] G. P. Sutton and O. Biblarz. *Rocket Propulsion Elements. Wiley*, 2017. isbn: 9781118753651.

[15] M. Szmuk et al. *Successive convexification for fuel-optimal powered landing with aerodynamic drag and non-convex constraints.* In: 2016 AIAA Guidance, Navigation, and Control Conference January (2016).

[16] Y. Mao et al. *Successive Convexification of Non-Convex Optimal Control Problems with State Constraints.* In: IFAC-PapersOnLine 50.1 (2017), pp. 4063–4069. issn: 24058963. doi: 10.1016/j.ifacol.2017.08.789.
eprint: 1701.00558.

[17] R. E. Stone R. W. Cottle J. S. Pang. *Linear Complementarity Problem.* In: (1992).

[18] J. M. Schumacher W. Heemels and S. Weiland. *Linear Complementarity Systems.* In: SIAM Journal of Applied Math 60.4 (2000). doi: 10.1097/01.

[19] M. Szmuk et al. *Successive convexification for 6-dof powered descent guidance with compound state-triggered constraints.* In: AIAA Scitech 2019 Forum (2019). doi: 10.2514/6.2019-0926. eprint: 1901.02181.

[20] W. S. Levine S. V. Rakovic. *Handbook of Model Predictive Control.* 2019. isbn: 978-3-319-77488-6. doi: 10.1007/978-3-030-11869-3_4.

[21] J. T. Emmert D. P. Drob and M. S. Dhadly. *Horizontal Wind Model (HWM).* 2016.

[22] P. V. Simplicio. *Guidance and Control Elements for Improved Access to Space, from Planettary Landers to Reusable Launchers*. (2019) University of Bristol.