

# Stability Guarantees for Model Predictive Controllers

Rita Nogueira Palma

Instituto Superior Técnico, Lisboa, Portugal

November 2023

## Abstract

The technological advances and their application across various sectors together with the growing interest in autonomous vehicles mark the beginning of a new era for the latter. Each of these sectors demands more robust control strategies to ensure that the vehicles can interact efficiently with external environments and can execute complex manoeuvres. Motivated by these challenges, the proposed approach implements a Model Predictive Control with specific constraints to force the system to enter into an invariant set represented as a Constrained Convex Generator, reducing the related conservatism. In doing so, the controller ensures its feasibility even with the presence of disturbances. Moreover, it is also possible to obtain a description of the nominal region of attraction for the system to reach the terminal set. By following this approach, it becomes possible to ensure the system's stability before undertaking the mission, given that the initial condition falls within the region of attraction. Simulation results validate the performance of Model Predictive Control trajectories whilst confirming that the computational times are suitable for a real-time application.

**Keywords:** Model Predictive Control, Constrained Convex Generator, Region of Attraction, Robust Positively Invariant, Stability, Recursive Feasibility

## 1. Introduction

Quadrotors, highly versatile UAV, excel in navigating tight spaces, maintaining stable hover, and offering cost-effective solutions [1]. Their use has driven the development of control methods for attitude stabilisation and trajectory tracking, emphasising precision in real-world environments. Moreover, in the quest for efficient parcel delivery, the CAPTURE project [2] employs shuttle drones for launch and capture manoeuvres with vehicles and objects. It involves two scenarios: cooperation between shuttle drones and the vehicles they launch or capture and dealing with the launch and capture of non-cooperative objects or drones.

Advanced UAVs must excel in mobility, motion accuracy, and energy efficiency [3]. Islam et al. studied various control methods for UAVs, underscoring the garnered attention in MPC for flight control. MPC impose constraints on the controlled invariant sets and optimises performance [5]. Although traditional methods often exhibit conservatism in defining these sets, which can lead to infeasibility. As MPC operates as a feedback loop, maintaining stability is crucial for robustness.

The envisaged solution ensures feasibility *a priori* by adjusting the optimisation problem and representing the robust controller invariant sets with minimal conservatism using CCGs [6]. The main contributions are: i) Development of a method to backpropagate the mRPI to determine the region of attraction. ii) Designing a robust predictive controller that adapts constraints for missions exceeding the prediction horizon. iii) Evalu-

ation of set descriptions using Constrained Zonotopes (CZ) and CCG formulations.

## 2. Literature Review

### 2.1. Model Predictive Control in Aerospace Systems

MPC potentially influences aerospace systems, promoting accurate and effective control of the vehicles [5]. Different architectures may be created by modifying constraint and cost functions. This controller can account for complex nonlinearities and couplings (e.g., aircraft dynamics and mission modes) that exhibit powerful dynamics in aerospace systems.

In [7], MPC schemes are considered to control the vehicle during the final phase of the *rendezvous* manoeuvre to meet the mission constraints. Focusing on robustness and computational efficiency, it introduces Tube-based Robust MPC (TRMPC), which tightens the constraints to minimise the cost function. It allows the trajectories of the uncertain system to lie in a tube centred on the nominal one, where each path is associated with a specific realisation of the uncertainty at each discrete-time step.

In collaboration with the CAPTURE project [2], da Silva Pinto explores optimising relay manoeuvres using multirotors for efficient parcel exchanges during flight. The strategy generates MPC optimal polynomial trajectories with acceleration constraints to ensure attitude coordination during package exchange.

Therefore, the quadrotor translation dynamics are modelled as a triple integrator, allowing the control input sequences to adhere to thrust and angular ve-

locity limits. Results show commendable tracking performance with successful mid-air parcel exchange, although there are relatively high roll and pitch rates. To further enhance trajectory generation, introducing bound constraints on position derivatives is desirable to manage saturation and rate limits, even though these additional inequality constraints may lead to numerical challenges.

## 2.2. Trajectory Generation

Robust trajectory generation ensures public trust and safety in autonomous systems. Trajectory generation algorithms compute multidimensional state and control signals that meet specifications and optimise mission objectives. A convex optimisation-based trajectory generation provides a method to deal with nonconvexities and generate a trajectory based on a convex solver.

In [9], the challenge of dynamic manoeuvring with stochastic target tracking and obstacle avoidance introduces probabilistic constraints, making path planning nonconvex and computationally demanding. A novel approach involves approximating complex shapes with circles. Thus, the search space is linearised at discrete-time intervals, replacing nonlinear probabilistic constraints with linear ones. It transforms optimisation into a convex problem and yields more efficient trajectories and robust navigation.

## 2.3. Stability and Robustness

Ensuring feasibility in real-time control applications is vital for safe and reliable performance. Often, it is preferable to anticipate system states using a nominal, linearised model for computational simplicity.

Recursive feasibility can be ensured by constraining states within a pre-computed controlled invariant set [10], emphasising the significance of sets that guarantee controller robustness. The intricate geometries of these sets can make the optimisation problem computationally expensive when state restrictions are applied.

In [11], it replaces the terminal condition with a contractive constraint derived from a sequence of reachable sets. These sets represent a progression, not necessarily invariant, wherein the system can be guided from one set to the next, eventually reaching the target set. Its computation is online, initiating from a positively invariant set. However, a drawback is the lack of assured invariance during the approximations, which may not include the previous set from the sequence. Nevertheless, the controller achieves asymptotic stability and local optimality, resulting in less computationally demanding algorithms for determining the sequence.

## 3. Problem Formulation

### 3.1. Model Predictive Control

Predictive control relies on a discrete-time model to predict future behaviour over a prediction horizon  $N$ . It applies a set of input sequences to a system, with the measured state/output as the initial condition while accounting for constraints. The objective is to optimise a

cost function by selecting the optimal sequence of control inputs. The feedback control law applies only the first element from the computed sequence at each step and repeats the process for the updated state in the next instant. It repeatedly runs the closed loop to correct discrepancies between actual and previously assumed optimal states. The prediction model in discrete-time is given by

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k)), \forall k \geq 0, \quad (1)$$

where  $\mathbf{x}(k) \in \mathbb{R}^{n_x}$  and  $\mathbf{u}(k) \in \mathbb{R}^{n_u}$  are respectively the state and control input vector at instant  $k$ ; and  $f: \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}^{n_x}$  calculates the next state  $\mathbf{x}(k+1) \in \mathbb{R}^{n_x}$ . If  $f$  is a linear function, the system is characterised by the linear state model

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k), \forall k \geq 0, \quad (2)$$

where  $A \in \mathbb{R}^{n_x \times n_x}$  represents the state-transition matrix and  $B \in \mathbb{R}^{n_x \times n_u}$  the control input matrix.

Quality of performance associated with the pair of state and control variables is measured via the stage cost  $\ell(\mathbf{x}(k), \mathbf{u}(k)): \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \rightarrow \mathbb{R}$ . It penalises the state and input predicted trajectories up to the terminal state. To drive the state towards the reference, it penalises the state at the end of the prediction horizon with the terminal cost  $V_f(\mathbf{x}(N)): \mathbb{R}^{n_x} \rightarrow \mathbb{R}^+$ . The MPC is defined as

$$\begin{aligned} \min_{\{\mathbf{u}(k)\}_{k=0}^{N-1}} J(\mathbf{x}(k), \mathbf{u}(k)) &= \sum_{k=0}^{N-1} \ell(\mathbf{x}(k), \mathbf{u}(k)) + V_f(\mathbf{x}(N)) \\ \text{s.t. } \mathbf{x}(k+1) &= f(\mathbf{x}(k), \mathbf{u}(k)), \forall k = 0, \dots, N-1, \\ \mathbf{x}(0) &= \mathbf{x}_0 \in \mathbb{X}, \\ (\mathbf{x}(k), \mathbf{u}(k)) &\in \mathbb{G}, \forall k = 0, \dots, N-1, \\ \mathbf{x}(N) &\in \mathbb{X}_f \subseteq \mathbb{X}, \end{aligned} \quad (3)$$

where the first constraint translates the system model; the second sets the first element of the trajectory as the initial state; the third guarantees that the pair is contained in the constraint set  $\mathbb{G} \subseteq \mathbb{X} \times \mathbb{U}$  (i.e.,  $\times$  represents the Cartesian Product); lastly, the terminal state belongs to the terminal set  $\mathbb{X}_f$ , which is the target region the system reaches at the end of the horizon.

The value of  $J(\mathbf{x}(k), \mathbf{u}(k))$  represents the cost-to-go from instant 0 to  $N$  under the control inputs. A common choice is a quadratic function because of its numerous advantages, like being convex, smooth and easy to tune [5]. It is given by

$$\mathbf{x}(N)^T P \mathbf{x}(N) + \sum_{i=0}^{N-1} (\mathbf{x}(i)^T Q \mathbf{x}(i) + \mathbf{u}(i)^T R \mathbf{u}(i)). \quad (4)$$

The tuning parameters are  $Q \in \mathbb{R}^{n_x \times n_x}$  and  $R \in \mathbb{R}^{n_u \times n_u}$ , chosen based on performance specifications. Large values of  $Q$  compared to  $R$  indicate a higher priority to quickly drive the state towards the origin,

which requires a more expensive control action. Conversely, a higher weight to  $R$  relative to  $Q$  reduces the control action and slows the rate at which the state approaches the origin. The final state penalty  $P \in \mathbb{R}^{n_x \times n_x}$  is calculated for exact  $Q$  and  $R$  to guarantee recursive feasibility and stability.

Dynamic Programming (DP) [12] is an efficient means for solving multistage optimisation problems. Instead of computing the solution for the optimal control problem (3), DP addresses similar sub-optimisation problems. It seeks to minimise the trajectory cost between sets  $\mathcal{X}_j$  representing all the possible states in  $\mathbb{X}$  that can be driven to  $\mathbb{X}_f$  by an admissible control sequence in  $j$  steps. In the following, some required definitions related to this technique are provided.

**Definition 3.1 (Control Invariant Set, [13]).** A set  $\mathcal{A}$  is control invariant for  $\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k))$ , if,  $\forall \mathbf{x}(k) \in \mathcal{A}, \exists \mathbf{u}(k) \in \mathbb{U} : f(\mathbf{x}(k), \mathbf{u}(k)) \in \mathcal{A} \forall k \in \mathbb{N}_0$ .

If a state belongs to a set, all subsequent states belong to the same. Thus, the system is invariant.

**Definition 3.2 (Recursive Feasibility, [13]).** If the problem is feasible at  $k = 0$ , it implies the problem's feasibility  $\forall k \geq 0$ .

This property ensures that the optimal control problem is feasible for all successor states if it is for the initial condition.

### 3.2. Constrained Convex Generators

In [6], a direct approach can over-approximate sets by a polytope, whether defined in hyper-plane or CZ.

**Definition 3.3 (Constrained Zonotope, [14]).** A set  $Z$  is a CZ defined by the tuple  $(G, c, A, b) \in \mathbb{R}^{n \times n_g} \times \mathbb{R}^n \times \mathbb{R}^{n_c \times n_g} \times \mathbb{R}^{n_c}$  such that:

$$Z = \{G\xi + c : \|\xi\|_\infty \leq 1, A\xi = b\},$$

where  $n$  is the number of states,  $n_g$  of generators and  $n_c$  of constraints.

Computations for control problems typically involve set operations such as Minkowski sums, linear maps, and intersections. Consider three CZs:  $Z = (G_z, c_z, A_z, b_z) \subset \mathbb{R}^n$ ;  $W = (G_w, c_w, A_w, b_w) \subset \mathbb{R}^n$ ;  $Y = (G_y, c_y, A_y, b_y) \subset \mathbb{R}^m$ ; matrix  $R \in \mathbb{R}^{m \times n}$  and vector  $t \in \mathbb{R}^m$ . The set operations are:

$$\begin{aligned} RZ + t &= (RG_z, Rc_z + t, A_z, b_z), \\ Z \oplus W &= \left( \begin{bmatrix} G_z & G_w \end{bmatrix}, c_z + c_w, \begin{bmatrix} A_z & 0 \\ 0 & A_w \end{bmatrix}, \begin{bmatrix} b_z \\ b_w \end{bmatrix} \right), \\ Z \cap_R Y &= \left( \begin{bmatrix} G_z & 0 \end{bmatrix}, c_z, \begin{bmatrix} A_z & 0 \\ 0 & A_y \\ RG_z & -G_y \end{bmatrix}, \begin{bmatrix} b_z \\ b_y \\ c_y - Rc_z \end{bmatrix} \right). \end{aligned} \quad (5)$$

$\mathcal{C}_{z,w,y}$  is the unit  $\ell_\infty$ -norm ball. Nothing is pushing its use as the generator, and one could resort to any unit ball following a  $p$ -norm but also extend the definition to convex cones (and other convex sets).

Therefore, Silvestre develop the CCG formulation. This concept uses  $\ell_2$  unit balls for smooth surfaces,  $\ell_\infty$  unit balls for facets, and cones for unbounded sets.

**Definition 3.4 (Constrained Convex Generators, [6]).** A CCG  $Z \in \mathbb{R}^n$  is defined by the tuple  $(G, c, A, b, \mathfrak{C})$  with  $G \in \mathbb{R}^{n \times n_g}, c \in \mathbb{R}^n, A \in \mathbb{R}^{n_c \times n_g}, b \in \mathbb{R}^{n_c}$ , and  $\mathfrak{C} := \{\mathcal{C}_1, \mathcal{C}_2, \dots, \mathcal{C}_{n_p}\}$  such that:

$$Z = \{G\xi + c : A\xi = b, \xi \in \mathcal{C}_1 \times \dots \times \mathcal{C}_{n_p}\}.$$

The sets  $Z, W$ , and  $Y$  are specified as CCGs:  $Z = (G_z, c_z, A_z, b_z, \mathfrak{C}_z) \subset \mathbb{R}^n$ ;  $W = (G_w, c_w, A_w, b_w, \mathfrak{C}_w) \subset \mathbb{R}^n$ ;  $Y = (G_y, c_y, A_y, b_y, \mathfrak{C}_y) \subset \mathbb{R}^m$ ; matrix  $R \in \mathbb{R}^{m \times n}$ , vector  $t \in \mathbb{R}^m$ , and the set operations as:

$$\begin{aligned} RZ + t &= (RG_z, Rc_z + t, A_z, b_z, \mathfrak{C}_z), \\ Z \oplus W &= \left( \begin{bmatrix} G_z & G_w \end{bmatrix}, c_z + c_w, \begin{bmatrix} A_z & 0 \\ 0 & A_w \end{bmatrix}, \begin{bmatrix} b_z \\ b_w \end{bmatrix}, \{\mathfrak{C}_z, \mathfrak{C}_w\} \right), \\ Z \cap_R Y &= \left( \begin{bmatrix} G_z & 0 \end{bmatrix}, c_z, \begin{bmatrix} A_z & 0 \\ 0 & A_y \\ RG_z & -G_y \end{bmatrix}, \begin{bmatrix} b_z \\ b_y \\ c_y - Rc_z \end{bmatrix}, \{\mathfrak{C}_z, \mathfrak{C}_y\} \right). \end{aligned} \quad (6)$$

These set operations offer efficient real-time closed-form solutions, presenting a novel concept for polytopes by enabling an arbitrary set of convex sets to cover the generator variables.

### 3.3. Vehicle Model

This research resorted to the quadrotor model as the application. The details for this formulation are presented in [15]. The drone has four identical rotors and propellers arranged in a configuration resembling a square, generating thrust and torque perpendicular to this plane [8]. The model is

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v}, \\ m\dot{\mathbf{v}} &= mg\mathbf{e}_3 - T\mathbf{R}\mathbf{e}_3, \\ \dot{\mathbf{R}} &= \mathbf{R} \times \boldsymbol{\omega}, \\ \mathbf{J}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} &= \mathbf{M}, \end{aligned} \quad (7)$$

where:  $\mathbf{p} \in \mathbb{R}^{3 \times 3}, \mathbf{v} \in \mathbb{R}^{3 \times 3}$  are the position and velocity;  $m \in \mathbb{R}^+$  is the total mass;  $g = 9.8 \text{ m s}^{-2}$  is the Earth's gravity;  $\mathbf{e}_3 = [0; 0; 1]$  is one of the basis vectors;  $T \in \mathbb{R}$  is the total thrust;  $\mathbf{R} \in \mathbb{R}^{3 \times 3}$  is the rotation transformation from the body-fixed to the inertial frame; for the body-fixed frame:  $\boldsymbol{\omega} \in \mathbb{R}^3$  is the angular velocity;  $\mathbf{J} \in \mathbb{R}^{3 \times 3}$  is the inertia tensor; and  $\mathbf{M} \in \mathbb{R}^3$  is the total moment.

### 3.4. Problem Statement

The quadrotor is modelled as a triple integrator, which fits the formulation for a linear dynamical system. The vehicle dynamics are given by

$$\mathbf{x}(k+1) = \mathbf{A}\mathbf{x}(k) + \mathbf{B}\mathbf{u}(k) + \mathbf{d}(k), \forall k \geq 0, \quad (8)$$

where  $\mathbf{x}(k) \in \mathbb{R}^{n_x}$ ,  $\mathbf{u}(k) \in \mathbb{R}^{n_u}$ , and  $\mathbf{d}(k) \in \mathbb{R}^{n_d}$  are respectively the state, input and disturbance signals at instant  $k$ . Additionally, the MPC is defined by the constrained optimal problem in (3). The system has

state and control input constraints and is affected by bounded disturbances.

The research directs attention to two separate yet interconnected problems:

i) The objective is to precisely define the region of attraction, which includes all the initial states, ensuring feasibility and stability during its mission. Thus, this set must be invariant, adhere to constraints, and ensure recursive feasibility for all states within its boundaries.

ii) Design the MPC with the following requirements:

a) Adhere to the physical and safety constraints of the quadrotor model. b) Focusing on acceleration constraints to ensure effective attitude coordination while managing saturation and rate limits in trajectory generation. c) Once the system reaches the terminal set, it remains there indefinitely. d) Ensure feasibility for a duration beyond the prediction horizon.

## 4. Proposed Solution

### 4.1. Admissible Sets

Admissible sets represent the feasible system states that satisfy specified constraints and performance criteria. As mentioned by Yang et al., these sets often exhibit intricate geometries. The inclusion of CCGs facilitates the definition of a robust geometry.

The translation dynamics are modelled as a triple integrator to access the vehicle's orientation and angular velocity. Controlling a vehicle's motion requires addressing translation aspects (e.g., position and velocity) and rotational elements (e.g., orientation and angular velocity). The orientation in vehicle dynamics refers to the vehicle's spatial positioning, including roll, pitch, and yaw angles, playing a crucial role in tasks like attitude stabilisation, autonomous flight, or precise manoeuvring. Angular velocity characterises the rate at which a vehicle rotates around its axes and is vital for sustaining stable flight, managing a vehicle's heading, and preventing undesirable spinning or tumbling. The triple integrator is implemented as

$$\dot{\mathbf{p}} = \mathbf{v}, \quad \dot{\mathbf{v}} = \mathbf{a}, \quad \dot{\mathbf{a}} = \mathbf{u}, \quad (9)$$

where  $\mathbf{p} \in \mathbb{R}^{3 \times 3}$  is the position,  $\mathbf{v} \in \mathbb{R}^{3 \times 3}$  the velocity, and  $\mathbf{a} \in \mathbb{R}^{3 \times 3}$  the linear acceleration driven by the control input  $\mathbf{u} \in \mathbb{R}^3$ .

The triple integrator is discretised for the sampling time  $T_s$  and the assumption that  $\mathbf{u}$  remain constant between sampling instances

$$\begin{aligned} \mathbf{p}(k+1) &= \mathbf{p}(k) + T_s \mathbf{v}(k) + \frac{T_s^2}{2} \mathbf{a}(k) + \frac{T_s^3}{6} \mathbf{u}(k), \\ \mathbf{v}(k+1) &= \mathbf{v}(k) + T_s \mathbf{a}(k) + \frac{T_s^2}{2} \mathbf{u}(k), \\ \mathbf{a}(k+1) &= \mathbf{a}(k) + T_s \mathbf{u}(k). \end{aligned} \quad (10)$$

The following sets are represented as CCGs.  $\mathbb{P}$  is related to the position and  $\mathbb{V}$  to the velocity. These constraints are defined as a hypercube (i.e.,  $\ell_\infty$ -norm) centred at the origin in which the diagonal of the generator matrix has the maximum of each parameter. The

set  $\mathbb{A}$  consider the maximum thrust and angular rates the vehicle achieves and the gravity force. According to (7), the acceleration is described as

$$a_x^2 + a_y^2 + (a_z - g)^2 \leq \left( \frac{T_{max}}{m} \right)^2, \quad (11)$$

where  $T_{max} \in \mathbb{R}$  is related with the rotating propellers. The acceleration is a sphere (i.e.,  $\ell_2$ -norm) centred at the origin but offset along the  $z$ -axis by  $g$ , and the diagonal of the generator matrix is its radius. In [8], the sphere's boundary is approximated by a polyhedron, underlining the benefits of using CCGs to represent this set. Lastly, the admissible states is  $\mathbb{X} := \mathbb{B} \times \mathbb{A}$ , where  $\mathbb{B} := \mathbb{P} \times \mathbb{V}$ .

$\|\mathbf{u}\|$  is bounded to control the roll  $p$  and pitch rates  $q$ , imposed by  $\boldsymbol{\omega} = [p \ q \ r]^\top$ . Considering the assumptions presented in [8] and assuming  $q_{max} = p_{max} = \Omega \in \mathbb{R}_0^+$  it comes

$$\|\mathbf{u}\| \leq \frac{\tilde{T}}{m} \Omega, \quad (12)$$

where  $\tilde{T} \leq T_{max}$ . The selection of  $\tilde{T}$  signifies the degree to which the design allows angular rate inputs to exceed the maximum achievable value. The set  $\mathbb{U}$  is a sphere (i.e.,  $\ell_2$ -norm) centred at the origin and the radius on the diagonal of the generator matrix. As stated, the approach in [8] does not consider this geometry. It samples points along the sphere's edge and computes their convex hull, resulting in a polyhedron.

The disturbance signals are bounded to ensure  $\mathbf{d}(k) \subset \mathbb{D}, \forall k \geq 0$  to calculate the terminal set considering the disturbances. It affects the measured control input, influencing the vehicle's acceleration. The set  $\mathbb{D}$  is described as an  $\ell_\infty$ -norm centred at the origin, with maximum disturbance values concerning the acceleration within the generator matrix.

### 4.2. Robust Positively Invariant Set

The concept of a RPI set is crucial for designing stable controllers despite disturbances. Once entered, the system remains within it indefinitely. In a disturbance-free stable system, the RPI is trivially a set with zero because all trajectories eventually converge to zero.

To guarantee feasibility, the approach in [10] formulates a terminal set by imposing constraints on the states, rendering it a control-invariant and recursively feasible set. This results in a positively invariant set for a stable system with unknown disturbances.

The quadrotor is closed-loop stable (i.e.,  $\rho(A+BK) < 1$ ) and contains unknown disturbances, meeting all the conditions to compute a RPI. The closed-loop dynamics are

$$\mathbf{x}(k+1) = (A+BK)\mathbf{x}(k) + \mathbf{d}(k). \quad (13)$$

Thus, the dynamics in order of  $\mathbf{x}(k)$  are

$$\mathbf{x}(k) = (A+BK)^k \mathbf{x}(0) + \sum_{i=0}^{k=N} (A+BK)^{k-i} \mathbf{d}(i). \quad (14)$$

The terminal set depicts the domain to where all states converge in infinity, computed as

$$\mathbb{X}_f := \lim_{k \rightarrow \infty} \mathbf{x}(k) = \underbrace{\mathcal{A}^k \mathbf{x}(0)}_{\rightarrow 0} + \sum_{i=0}^{k=N} \mathcal{A}^{k-i} \mathbf{d}(i), \quad (15)$$

where  $\mathcal{A} := A + BK$  represents the closed loop matrix of the system. If  $\mathbf{d}(k) \subset \mathbb{D}$  and  $\mathbb{D}$  does not change over time, the terminal set is described as

$$\mathbb{X}_f := \sum_{i=0}^{k=N} \mathcal{A}^{k-i} \mathbb{D}. \quad (16)$$

There are techniques to simplify the numerous computations of the RPI. However, computing such a set raises a trade-off between accuracy, complexity, and computation time. Therefore, inner approximations are commonly used in the set domain, which improves the practicality of these sets' representation and ensures accuracy. It restricts the set's complexity to satisfy a pre-determined upper bound on the number of generators and constraints. The approximation of RPI generates a minimal Robust Positively Invariant (mRPI).

**Definition 4.1 (The minimal Robust Positively Invariant, [16]).** The mRPI set  $F_\infty$  is the RPI set that is contained in every closed RPI set of (13) and is given by

$$F_\infty = \bigoplus_{i=0}^{\infty} \mathcal{A}^i \mathbb{D}. \quad (17)$$

Obtaining an explicit characterisation of  $F_\infty$  is challenging due to the infinite sequence of Minkowski sums. CCGs portray a new concept of CZ that enables the application of the methods outlined in [17]. In this research, the proposed  $F_s$  is calculated iteratively as

$$F_s = \left( \bigoplus_{i=0}^N \mathcal{A}^i \mathbb{D} \right) \oplus \left( \left( (I_n - \mathcal{A})^{-1} - \sum_{i=0}^N \mathcal{A}^i \right) \mathbb{D} \right). \quad (18)$$

The left side of (18) calculates the inner approximation of the RPI by only summing until  $N$ . Additionally, the right side performs a numerical evaluation of the upper limit (i.e., overbound) for the residual portion of the expression. Consequently, the impact of this upper limit on the disturbances set is considered by employing the linear map. It allows the robust estimation of the upper bound by effectively accounting for all dimensions influenced by disturbances. Finally, the Minkowski sum guarantees an overbound of the inner approximation of the RPI.

It improves the method's time by greatly reducing the iterations required to calculate the mRPI. Ultimately, it achieves a good balance between computational time and accuracy.

An accurate approximation of the RPI reduces the computational cost of offline computation for backwards reachable sets from mRPI. These successive computations result in an accurate and reliable region of

attraction, ensuring that the system eventually reaches the mRPI. As a result, enlarging the control horizon expands the zone but at a higher computational cost, whereas increasing the terminal set produces an enlargement at no extra cost [11].

### 4.3. Nominal Region of Attraction

The region of attraction contains all the initial states that guarantee the feasibility and stability of the system. It is computed through backward propagation to force the system to enter into the terminal set, defined as a mRPI. Thus, the controller can switch to the LQR and remain in the set regardless of disturbances.

The method relies on the principles outlined in Section 3.1 regarding DP to minimise the trajectory cost across sets. A key aspect of computing backwards reachable sets is ensuring invariance. Once each set contains the previous one, the region of attraction is invariant, guaranteeing the recursive feasibility of the controller. It provides a significant advantage compared to [11], which does not account for invariance sets due to the approximations made during the backward computation.

To obtain the formulation of this region, it is necessary to incorporate the computation of the backward reachable sets from the mRPI as

$$\mathbb{X}_f = A\mathcal{X}_j + BU \Rightarrow \mathcal{X}_j = A^{-1}(\mathbb{X}_f - BU), \quad (19)$$

where  $\mathbb{X}_f$  represents the mRPI and  $\mathcal{X}_j := \mathcal{X}_1$  is the set that in a single discrete-time step reaches the mRPI considering the worst-case for the disturbances that can impact the system. This process is iterative, wherein each subsequent step,  $\mathbb{X}_f := \mathcal{X}_1$  and  $\mathcal{X}_j := \mathcal{X}_2$ , and so on. It continues until  $\mathcal{X}_j$  becomes the region of attraction.

Moreover, it is noteworthy that (19) does not account for disturbances as the methodology calculates a nominal region of attraction.

The subtraction operation on the right side of (19) is a Pontryagin Difference of two sets [18].

Algorithm 1 is deployed for CCGs, based on the technique in [17]. Ultimately, the final set appended to the sequence includes all the states after a specified number of steps  $K$  (i.e.,  $K > N$ ) converge to the mRPI (i.e., the region of attraction).

---

**Algorithm 1** Sequence of backward reachable sets.

---

**Input:** mRPI

**Output:** Sequence of reachable sets

- 1:  $\mathcal{X}_j \leftarrow$  mRPI
  - 2: **for**  $i = 1 : K$  **do**
  - 3:     Set estimation:  $\hat{\mathcal{X}}_j \leftarrow A^{-1}\mathcal{X}_j \oplus (-A^{-1}B)\mathbb{U}$
  - 4:      $\mathcal{X}_j \leftarrow \hat{\mathcal{X}}_j \cap \mathbb{X}$
  - 5:     Add  $\mathcal{X}_j$  to the sequence
  - 6: **end for**
-

#### 4.4. Sequence of Reachable Sets in Model Predictive Control

This section outlines the methodology used to control a drone's translation to force it to enter into a specific set. The cost function is quadratic, chosen for its favourable characteristics. The problem resembles the one discussed in [11] due to the terminal constraint within the sequence of  $N_c \cdot N$  reachable sets, where  $N$  denotes the prediction horizon and  $N_c$  the convergence steps. The optimisation problem is

$$\begin{aligned} \min_{\{\mathbf{u}(k)\}_{k=0}^{N-1}} \quad & \mathbf{x}(N)^\top P \mathbf{x}(N) + \sum_{i=0}^{N-1} (\mathbf{x}(i)^\top Q \mathbf{x}(i) + \mathbf{u}(i)^\top R \mathbf{u}(i)) \\ \text{s.t.} \quad & \mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k), \forall k = 0, \dots, N-1, \\ & \mathbf{x}(0) = \mathbf{x}_0, \\ & \mathbf{x}(k) \in \mathbb{X}, \forall k = 0, \dots, N-1, \\ & \mathbf{u}(k) \in \mathbb{U}, \forall k = 0, \dots, N-1, \\ & \mathbf{x}(N) \in \mathcal{X}_j \subseteq \mathbb{X}, j = \max((N_c - 1) \cdot N - k, 0). \end{aligned} \quad (20)$$

In each step, the terminal desired region is updated to tend the nominal system to the mRPI. Including this terminal constraint guarantees the placement of the terminal state within the subsequent computed reachable set [11]. Figure 1 depicts the methodology implemented.

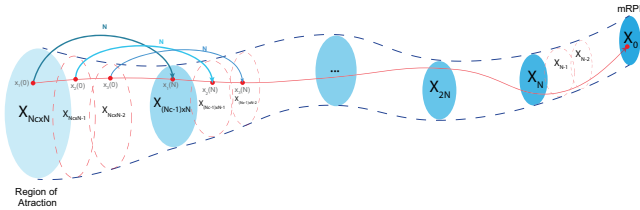


Figure 1: Illustration of the adapted MPC.

#### 4.5. Obstacle Avoidance

Obstacle avoidance is a critical challenge for autonomous vehicles, ensuring safety and adaptability in dynamic environments. These systems must perceive and react to obstacles, preventing collisions and enabling intelligent decision-making to adapt to changing surroundings. Evaluating the effectiveness of MPC-based trajectories in obstacle avoidance provides real-world validation of this approach.

Incorporating obstacles in MPC trajectories requires maintaining the convexity of each set throughout the CCG formulation. Highlighting that creating sets with voids is infeasible implies adjusting the set where the obstacle is introduced to preserve its convexity. Introducing an obstacle restricts the range of possible states, forming a subset of the previous set, essentially narrowing down the allowable space. Therefore, the subsequent sets of the obstacle introduction are adjusted considering the backward sequence, encompassing all potential states at each instant. This method is pro-

posed to generate the sequence of forward reachable sets presented in Algorithm 2.

---

#### Algorithm 2 Sequence of forward reachable sets.

---

**Input:** Obstacle set  $\mathbb{O}$  and the sequence of backward reachable sets  $\mathcal{X}_j$ ,  $j = 0, \dots, K$

**Output:** Sequence of forward reachable sets considering obstacles  $\mathcal{O}_j$ ,  $j = 0, \dots, K$

- 1: Define index  $k$  where the obstacle is introduced
  - 2: **for** the set  $i$  being the RA to the mRPI **do**
  - 3:     **if** the set is previous to the set where the obstacle is introduced **then**
  - 4:          $\mathcal{O}_i \leftarrow \mathcal{X}_i$
  - 5:     **else if** the set is where the obstacle is introduced **then**
  - 6:          $\mathcal{O}_i \leftarrow \mathcal{X}_i \cap \mathbb{O}$
  - 7:     **else if** the set is subsequent to the set where the obstacle is introduced **then**
  - 8:          $\hat{\mathcal{O}}_i \leftarrow A\mathcal{O}_i \oplus BU$
  - 9:          $\mathcal{O}_i \leftarrow \hat{\mathcal{O}}_i \cap \mathcal{X}_i$
  - 10:     **end if**
  - 11:     Add  $\mathcal{O}_i$  to the forward sequence
  - 12: **end for**
- 

Offline computation of the sequence reduces computational complexity and is essential for scenarios where MPC might encounter unexpected obstacles before predicting their presence. Real-time computations during MPC execution can be challenging for navigating unanticipated obstacles. Consequently, the MPC remains analogous to the one described in Section 4.4 with a different sequence provided to the controller.

### 5. Simulation Results and Discussion

The codebase and resources related to the research can be found in the GitHub Repository. The specifications for the quadrotor model are in agreement with the DJI Phantom 4 Pro [19].

#### 5.1. Admissible Sets Representation

Accurate definitions of the vehicle's position, velocity, and acceleration are essential for the controller's operation. Define the sets as CCG enables precise representation of the robust controlled invariant sets, enhancing performance.

Figure 2 depicts the state set  $\mathbb{X}$ , which exhibits a distinct geometric structure due to the employment of CCGs. The position and velocity resemble hypercubes, but the acceleration takes the form of a sphere (11) corresponding to the theoretical ideal.

As proven in (12), the configuration of  $\mathbb{U}$  takes the form of a sphere illustrated in Figure 3a. Additionally, as mentioned, the disturbance set  $\mathbb{D}$  displayed in Figure 3b spans all the dimensions but primarily affects the system's control input, influencing the vehicle's acceleration.

In da Silva Pinto, the acceleration and control input set are approximated as a polyhedron, resulting

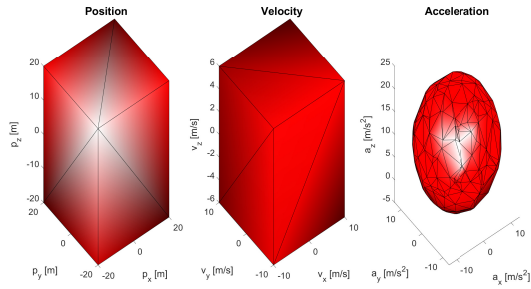


Figure 2: Projection of the admissible states set  $\mathbb{X}$ .

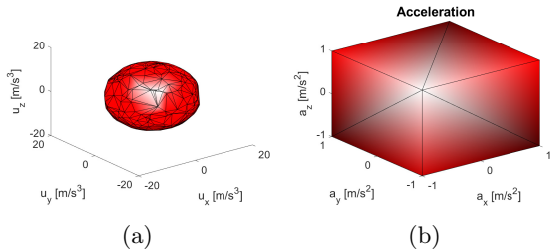


Figure 3: (a) Control input set  $\mathbb{U}$ . (b) Projection of the disturbances set  $\mathbb{D}$ .

in admissible sets that lack precision. Nonetheless, it does not accurately depict these sets, potentially compromising the vehicle's performance. This observation supports the reduction of conservatism when describing these sets using CCGs.

### 5.2. Computation of the minimal Robust Positively Invariant Set

The mRPI defines the feasible region where system states converge and remain invariant despite disturbances. An analysis of the methods described in Section 4.2 to calculate the mRPI is shown in Table 1 and depicted in Figure 4.

The computation of the  $F_\infty$  is based on (17) and given by

$$\bigoplus_{i=0}^K \mathcal{A}^i \mathbb{D}, \quad (21)$$

with  $K \gg N$ .  $K$  is chosen considering various values until there are no visible differences in the resulting set. The inner RPI is computed through (21) with  $K = N$ , corresponding to the left side of (18).  $F_s$  is calculated via (18), which is the mRPI of the proposed solution.

Table 1: Generators and computational time for different RPI computation methods.

	$F_s$	Inner RPI	$F_\infty$
Generators	39	36	30 003
Time [s]	$3.04 \times 10^{-3}$	$2.26 \times 10^{-3}$	3.98

It is evident that the inner RPI effectively serves as an inner approximation of the mRPI. While this set

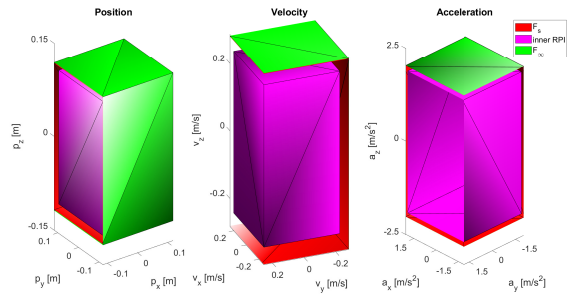


Figure 4: Projection of RPI computation methods.

offers optimal computational efficiency, it lacks the precision to represent the mRPI. On the other hand, the proposed  $F_s$  offers an outstanding approximation of  $F_\infty$ . This set has fewer generators (39 instead of 30 003) and maintains excellent computational performance.

Landing parameters depend on requirements, drone capabilities, and the environment. Typically, the drone lands near the ground, considering its height, landing pad's altitude, and potential obstacles. It ensures a smooth descent by gradually reducing vertical velocity and mainly uses velocity control for collision avoidance. The mRPI for this system is portrayed in Figure 5 (corresponds to  $F_s$  in Figure 4), which aligns with the specific specifications and constraints of the drone. This technique provides stability and respects the operational limits.

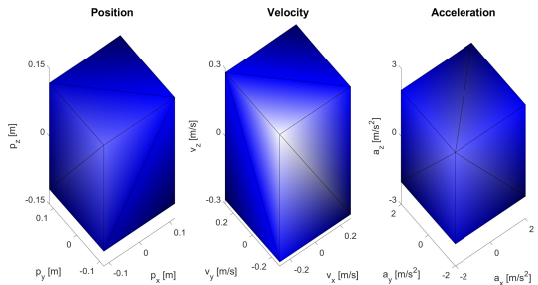


Figure 5: Projection of the mRPI.

### 5.3. Conservatism of Region of Attraction Definition

Precisely computing the region of attraction is critical for validating control system stability and feasibility. This section underscores the impact of the mRPI computation on the region of attraction and compares two distinct approaches to its computation.

The number of generators and computational time needed for the region of attraction definition is examined through backward propagation from the sets  $F_s$  and  $F_\infty$  (i.e., defined in Section 5.2). The results are shown in Table 2, highlighting the computational efficiency of the impact of the proposed  $F_s$  to describe the region of attraction with fewer generators.

Figure 6 depicts the computation of the region of attraction using CCGs and CZs to define the control input

Table 2: Generators and computational time for the region of attraction with different RPI sets.

	$F_s$	$F_\infty$
Generators	183	30 147
Time [s]	$1.12 \times 10^{-2}$	4.27

set  $\mathcal{U}$ . The strategy delivered by da Silva Pinto to determine the region of attraction tends to over-approximate the  $\ell_2$ -norm balls by a polytope, increasing the possibility of selecting an initial condition that does not converge to the terminal set, shown in Figure 6a. However, the computation with CCGs or CZ coincides when the control input is  $\ell_\infty$ -norm ball, displayed in Figure 6b.

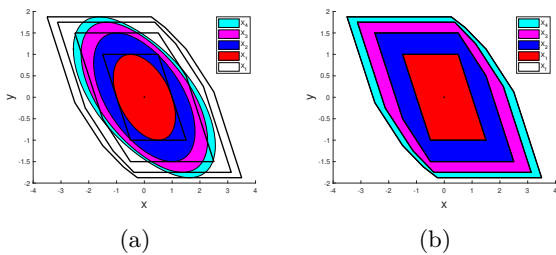


Figure 6: Region of attraction computed with CCGs (coloured filled sets) and CZs (black outline sets) with the control input set  $\mathcal{U}$  described as (a)  $\ell_2$ -norm. (b)  $\ell_\infty$ -norm.

In Figure 6a, the  $\mathcal{X}_1$  is over-approximated when computed with CZs, and then the error is propagated for the following sets. When  $\mathcal{U}$  is a  $\ell_2$ -norm, the strategy of da Silva Pinto, in a Monte Carlo simulation, has approximately 32% of states as infeasible initial points. Confirming the capacity of CCGs to depict sets with reduced conservatism.

Furthermore, the focus shifts to the practical application of this comparative analysis within the drone scenario depicted in Figure 7.

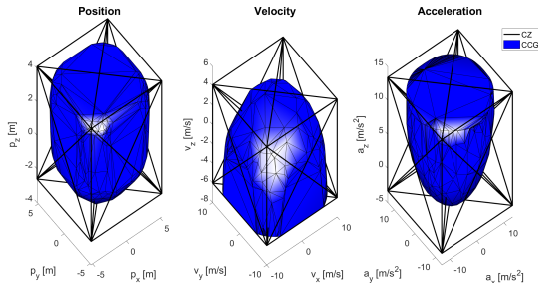


Figure 7: Comparison of the projected region of attraction when computed with CCGs and CZs.

The reduction of conservatism provided by the developed method confers a notable advantage as it en-

ables the precise description of  $\ell_2$ -norm sets. Specifically, the linear velocity dynamics (11) yields a set in the shape of a sphere for acceleration portrayed in Figure 2. Also, the control input (12) illustrated in Figure 3a is a sphere. Furthermore, regarding initial states within the CZ formulation, approximately 23% are infeasible. This percentage may affect performance, underscoring the need for less conservative strategies, as exemplified by CCG representations.

#### 5.4. Diverse Initial States in Model Predictive Control Trajectory

To explain the controller's behaviour, the simulation in Figure 8 focuses on the  $x$  and  $y$  directions but is depicted in 3D (with discrete-time instants on the  $x$ -axis). The disturbances only manifest in acceleration to illustrate the context of the attitude stabilisation.

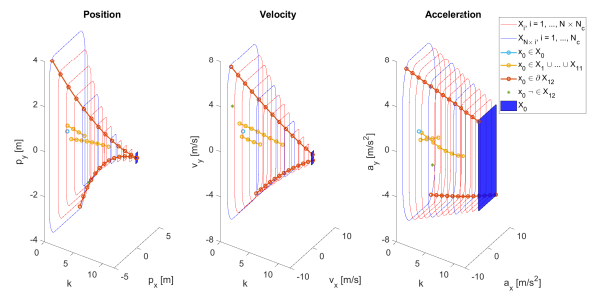


Figure 8: Projection of sequence sets in  $x$  and  $y$  directions over discrete-time instants and the MPC trajectories from different initial conditions.

Several insights can be derived:

i) The blue trajectories represent cases where  $x_0$  belongs to the mRPI, denoted as  $\mathcal{X}_0$ . Consequently, these trajectories are only one state.

ii) The yellow trajectories illustrate scenarios where  $x_0$  falls between the penultimate set  $\mathcal{X}_1$  and the second set  $\mathcal{X}_{11}$ . In scenarios where  $x_0$  resides within the mRPI in some parameters (e.g., position, velocity and acceleration), the trajectory tends to move towards the mRPI in the dimensions that have yet to be reached. Thus, it demonstrates that when  $x_0$  is further from the mRPI, it requires more discrete-time instants to reach the mRPI.

iii) To test the MPC under maximum circumstances,  $x_0$  is placed on the boundary of the region of attraction. The orange trajectories indicate that the MPC operates at its limits and demands maximum actuation in particular directions. Notably, when  $x_0$  is positioned at the edge of the region of attraction, the final condition also lies along the boundary of the mRPI, underscoring the calculation's accuracy.

iv) When  $x_0$  lies outside the region of attraction, the MPC is infeasible, represented by an asterisk.

Figure 9 depicts the MPC trajectories within the quadrotor scenario. It showcases a selection of trajectories converging towards the mRPI (smaller blue set). Only the  $\mathcal{X}_{i,N}, i = 0, \dots, N_c$  are displayed to simplify

trajectory visualisation. The controller’s performance reflects the previous observations for the  $x$  and  $y$  directions. It reaffirms the accuracy and robustness of the proposed MPC.

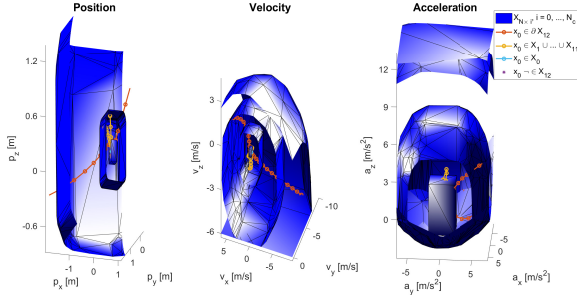


Figure 9: Zoom-in on the projection of sequence sets and the MPC trajectories from different initial conditions.

Furthermore, in a simulation employing the region of attraction with 30 147 generators (i.e., the case presented in Table 2), pre-compiling the optimisation problem takes over an hour before further integration in the MPC. In contrast, when performing with the proposed mRPI, it takes approximately 1.39 s to pre-compile the problem and 1.41 s to generate the MPC trajectory. It highlights the computational load when working with an mRPI with few generators in the MPC process, confirming that the computational times are suitable for a real-time application.

### 5.5. Obstacles on the Trajectory

A series of simulations are conducted to assess the impact of different factors on the controller’s performance to avoid obstacles. The obstacle is stationary, primarily affecting the position. Figure 10 illustrates the influence on the sets and MPC throughout the sequence, both before and after the obstacle’s introduction in set  $\mathcal{X}_8$ . There are  $\mathcal{X}_i$ ,  $i = 0, \dots, 12$  sets, with  $N = 4$  and  $N_c = 3$ . The various trajectories along this sequence are initiated from different initial conditions.

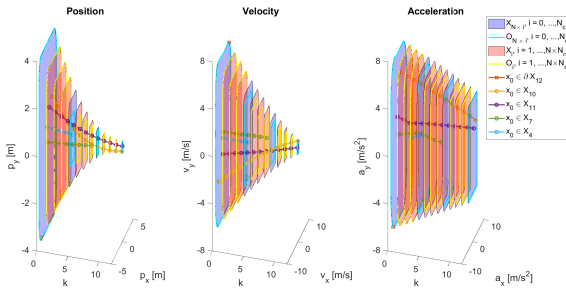


Figure 10: Projection of sequence sets in  $x$  and  $y$  directions over discrete-time instants and the MPC trajectories from different initial conditions. The obstacle is situated in  $\mathcal{X}_8$ .

The simulation portrayed in Figure 10 and various

simulations conducted for this analysis have offered insights into the varied responses when avoiding obstacles. These findings are summarised as follows:

**i) Obstacle Location:** Placing an obstacle at the beginning of the trajectory (after a minimum of  $N$  steps for MPC could detect) might allow more instants to adjust the course, yielding better performance.

**ii) Obstacle set size:** When the obstacle set occupies a larger area, it expands the available space for manoeuvring around it. Thus, the likelihood of infeasibility with a less extensive set of obstacles is greater.

**iii) Initial condition:** a) When lies on the boundary of the region of attraction demands maximum actuation to reach the edge of each terminal set in the sequence. Consequently, when considering the introduction of obstacles, the MPC immediately determines infeasibility, as the last state of the prediction  $\mathbf{x}(N)$  cannot reside within  $\mathcal{O}_s$ . This example is illustrated with the orange square. b) If it is close to the mRPI, the state might reach it without encountering the obstacle. These scenarios are depicted in blue and green. c) For the ones that are neither close to the mRPI nor on the boundary of the region of attraction, the problem’s feasibility depends on specific circumstances. The yellow and purple trajectories represent the cases where the initial condition allows reaching the new mRPI.

**iv) Prediction Horizon:** A shorter horizon restricts the controller’s capacity to navigate obstacles, reflecting poorer performance. However, significantly increasing the horizon increases computational complexity.

In the drone scenario, behaviour resembles the 2D case, but 3D visualisation is challenging.

## 6. Conclusion

MPC is often computationally demanded, and the increased number of generators in sets impairs its performance (e.g., pre-compiling the optimisation problem with 30 147 generators took one hour), which may lead to numerical issues or infeasibility. The proposed method introduces an optimal representation of the mRPI with fewer generators (e.g., 39 instead of 30 003) and an excellent computational time (e.g.,  $3.04 \times 10^{-3}$  s instead of 3.98 s).

Additionally, the MPC with a variable terminal set regarding the backward sequence of CCGs forces the system to enter into an invariant set defined as the mRPI, guaranteeing the system remains there despite disturbances. Consequently, the controller can smoothly transition to the LQR, which reduces the computational resources and maintains adherence to the constraints, conferring a distinct advantage to autonomous vehicles.

UAVs demand more robust control strategies to guarantee that the vehicles can effectively interact with external environments and execute complex manoeuvres. When evaluating the performance with CCGs and CZ to estimate the region of attraction, the results prove

that CZ yields 23% of infeasible states, which poses a risk to the aerospace system. The CCGs revealed the ability to reduce the conservatism linked with set representations whilst improving computational efficiency. Thus, incorporating CCGs into MPC to represent the control invariant sets addresses the rising demand for robustness, adaptability, and improved performance.

In summary, this research precisely defines the region of attraction subject to constraints and disturbances, ensuring mission feasibility from the outset. It ensures that if any initial condition is located within the region of attraction, the MPC successfully reaches the terminal set 100% of the time with a computational time suitable to a real-time application. The outcome of this study stands as a validation of a concept, underscoring the potential of advanced control methods in shaping the trajectory of autonomous vehicles in the future.

Nonetheless, there is ample room for future exploration and improvement. MPC is computationally demanded, mainly when dealing with more extensive trajectories and complex systems. A promising approach is the paving technique, which involves over-approximating the sets within each backward sequence. It facilitates the computation of the sets whilst guaranteeing an accurate definition. Furthermore, incorporating practical tests in a real-world context holds notable value. However, before undertaking these tests, it is crucial to conduct simulations that integrate elements, such as thrusts, sensors, onboard computers, and disturbance models (e.g., gravitational perturbations and atmospheric drag).

## References

- [1] R. Pérez-Alcocer, J. Moreno-Valenzuela, and R. Miranda Colorado. A robust approach for trajectory tracking control of a quadrotor with experimental validation. *ISA Transactions*, 2016.
- [2] B. Guerreiro, R. Cunha, A. Pascoal, and C. Silvestre. Capture project, 2023. URL <https://capture.isr.tecnico.ulisboa.pt/>. Accessed: October 12, 2023.
- [3] C. Ding and L. Lu. A tilting-rotor unmanned aerial vehicle for enhanced aerial locomotion and manipulation capabilities: Design, control, and applications. *IEEE/ASME Transactions on Mechatronics*, 2021.
- [4] M. Islam, M. Okasha, and E. Sulaeman. A model predictive control approach on unit quaternion orientation based quadrotor for trajectory tracking. *International Journal of Control, Automation and Systems*, 2019.
- [5] U. Eren, A. Prach, B. B. Koçer, S. V. Raković, E. Kayacan, and B. Açıkmeşe. Model predictive control in aerospace systems: Current state and opportunities. *Journal of Guidance, Control, and Dynamics*, 2017.
- [6] D. Silvestre. Constrained convex generators: A tool suitable for set-based estimation with range and bearing measurements. *IEEE Control Systems Letters*, 2022.
- [7] M. Mammarella, E. Capello, H. Park, G. Guglieri, and M. Romano. Tube-based robust model predictive control for spacecraft proximity operations in the presence of persistent disturbance. *Aerospace Science and Technology*, 2018.
- [8] J. D. B. da Silva Pinto. Model predictive control strategies for aggressive parcel relay maneuvers using drones. Master’s thesis, Universidade de Lisboa - Instituto Superior Técnico, 2021.
- [9] M. A. Mousavi, Z. Heshmati, and B. Moshiri. Ltvmpc based path planning of an autonomous vehicle via convex optimization. In *2013 21st Iranian Conference on Electrical Engineering (ICEE)*, 2013.
- [10] L. Yang, A. Katnik, and N. Ozay. Quickly finding recursively feasible solutions for mpc with discrete variables. In *2019 IEEE Conference on Control Technology and Applications (CCTA)*, 2019.
- [11] D. Limon, T. Alamo, and E. Camacho. Enlarging the domain of attraction of mpc controllers. *Automatica*, 2005.
- [12] R. Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences*, 1952.
- [13] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert. Model predictive control: Theory, computation, and design. *Automatica*, 2000.
- [14] J. K. Scott, D. M. Raimondo, G. R. Marseglia, and R. D. Braatz. Constrained zonotopes: A new tool for set-based estimation and fault detection. *Automatica*, 2016.
- [15] T. Lee, M. Leok, and N. H. McClamroch. Geometric tracking control of a quadrotor uav on  $se(3)$ . In *49th IEEE Conference on Decision and Control (CDC)*, 2010.
- [16] S. V. Raković, E. Kerrigan, K. Kouramas, and D. Mayne. Invariant approximations of the minimal robust positively invariant set. *Automatic Control, IEEE Transactions on*, 2005.
- [17] V. Raghuraman and J. P. Koeln. Set operations and order reductions for constrained zonotopes. *Automatica*, 2022.
- [18] M. Althoff. On computing the minkowski difference of zonotopes, 2022.
- [19] DJI. Dji phantom 4 pro v2, 2023. URL <https://www.dji.com/pt/phantom-4-pro-v2>.