Minimum-time B-spline trajectories with corridor constraints. Application to cinematographic quadrotor flight plans

Gauthier Rousseau, Cristina Stoica Maniu, Sihem Tebbani, Mathieu Babel, Nicolas Martin

Minimum-time B-spline trajectories with corridor constraints. Application to cinematographic quadrotor flight plans

Gauthier Rousseau[a,b], Cristina Stoica Maniu[a,*], Sihem Tebbani[a], Mathieu Babel[b], Nicolas Martin[b]

[a] *Laboratoire des Signaux et Systèmes, CentraleSupélec-CNRS-Univ. Paris-Sud, Université Paris Saclay, 3 rue Joliot-Curie, 91190 Gif-sur-Yvette, France*
[b] *Flight Control Department, Parrot Drones, 174 quai de Jemmapes, 75010 Paris, France*

Abstract

This paper proposes a novel strategy for completing a flight plan with a quadrotor UAV, in the context of aerial video making. The flight plan includes different types of waypoints to join, while respecting flight corridors and bounds on the derivatives of the position of the quadrotor. To this aim, non-uniform clamped B-splines are used to parameterize the trajectory. The latter is computed in order to minimize its overall duration, while ensuring the validation of the waypoints, satisfying the flight corridors and respecting the maximum magnitude on its derivatives. A receding waypoint horizon is used in order to split the optimization problem into smaller ones, which reduces the computation load when generating pieces of trajectories. The effectiveness of the proposed trajectory generation technique is demonstrated by simulation and through an outdoor flight experiment on a quadrotor.

Keywords: UAV, quadrotor, guidance, trajectory generation, B-spline

1. Introduction

Multirotors have become a new standard for video making, as it simplifies and significantly reduces the costs of aerial footage. This has raised the interest in achieving autonomous performance of cinematographic aerial shots. It usually requires the generation of smooth trajectories, satisfying high level specifications, based on the vocabulary and the specifications of a cinematographer rather than of a drone pilot. Efforts in this regard have been provided through different works Galvane et al. (2013), Joubert et al. (2016), Roberts and Hanrahan (2016), Fleureau et al. (2016), Engelhardt et al. (2016), Gebhardt et al. (2016), Joubert (2017), Nägeli et al. (2017a), Nägeli et al. (2017b), Galvane et al. (2017) and Galvane et al. (2018), and have yield impressive results. Such autonomous sequences are often specified in the form of waypoints to join successively, sometime along with speed references, flight corridors or camera behaviors. The performance of these flight plans then requires the generation of feasible trajectories, meeting the requirements of cinematography in terms of shape, speed profile and smoothness.

The use of smooth piecewise polynomials to this aim is quite popular. In Mellinger and Kumar (2011) trajectories are generated through the minimization of the root mean square of one or several derivatives of the position, for a given duration of the overall trajectory. In this method, flight corridors were satisfied on a finite number of points (gridding). It was successfully developped and applied on real systems Richter et al. (2013), Richter et al. (2016), where a bi-level optimization is used to optimize the trajectory duration and to ensure its feasibility. In Joubert et al. (2015), a similar method is used for designing a smooth path which evaluation is given by a discretized timing law, in order to deal with the dynamics of the quadrotor and its payload, and applied to cinematography. A continuous alternative to this timing law is proposed in Roberts and Hanrahan (2016), in which it was used to make a cinematographic trajectory feasible without altering its shape. It was also used in Joubert et al. (2016), as part of a method proposed to perform scripted aerial shots with quadrotors. In Rousseau et al. (2018), instead of a timing law, a bi-level optimization is used to optimize both the shape of the curve and its speed profile simultaneously.

Motion primitives and their feasibility for quadrotors were also studied in Schöllig et al. (2011), Mueller et al. (2015). In Gebhardt et al. (2016), such primitives are obtained through discretization of the trajectory and by solving an optimization problem on the control inputs and the state of the quadrotor at each step, and was especially suited for joining waypoints at specified times. In Van Loock et al. (2013), minimum-time trajectories for quadrotors are generated through convex optimization, but their shape is not suitable for cinematography (e.g. the quickest trajectory joining two points at the same altitude is not a straight line, as it would be expected for cinematography).

Model Predictive Control (MPC) can also be considered to tackle both the trajectory generation and tracking at the same time. It has been successively applied to quadrotors in Bouffard (2012), Abdolhosseini et al. (2013), Bangura and Mahony (2014) and Kamel et al. (2015) and could be a candidate to

*Corresponding author
Email addresses: gauthier.rousseau@centralesupelec.fr, gauthier.rousseau@parrot.com (Gauthier Rousseau), cristina.stoica@centralesupelec.fr (Cristina Stoica Maniu), sihem.tebbani@centralesupelec.fr (Sihem Tebbani), mathieu.babel@parrot.com (Mathieu Babel), nicolas.martin@parrot.com (Nicolas Martin)

perform both the feedback control and the smooth trajectory generation simultaneously. Its use was suggested in Engelhardt et al. (2016) for quadrotor cameras. In Nägeli et al. (2017a), Nägeli et al. (2017b), an MPC strategy is used to generate trajectories directly through the optimization of cinematographic criteria, such as framing or occlusion, with obstacle avoidance solutions. This requires the online resolution of a constrained optimization problem.

An alternative approach consists into generating first a cinematographic path by interpolation of two camera poses, see for instance Fleureau et al. (2016). Then, a steering method (such as Galvane et al. (2013)) is used to complete the path. A similar strategy is applied in Galvane et al. (2017) with a smooth path generated using a method similar to Mellinger and Kumar (2011).

Due to their useful properties to constrain the trajectory and its derivatives inside convex regions, the use of B-spline curves to parameterize the trajectory is also quite spread now Van Loock et al. (2015), Stoican et al. (2017), Van Parys and Pipeleers (2017), Mercy et al. (2017), Nguyen et al. (2018). These curves are piecewise polynomials parameterized by a set of control points and a vector of knots, defining when the curve switches between two polynomial representations. Most of the time though, a particular kind of B-spline is used, i.e. uniform B-spline (also called cardinal B-spline), for which the knots are equally spaced. Their advantage is that they can simplify some of the calculation and allow some of them to be preformed offline. This can significantly speed up the B-spline generation process which is useful for embedded systems. However, one drawback is that it is difficult to generate time-optimal trajectories using uniform B-splines, and other metrics are usually optimized rather than the duration for their generation (e.g. the length or the root mean square of a derivative). This can make the generation of trajectories with both the shape and speed profile suited for cinematography complex. In Stoican et al. (2017) it is suggested to optimize the number of control points, which can help improving the results for cinematography, but requires the resolution of a Mixed Integer Problem (MIP).

This paper proposes a new method to generate minimum-time B-spline trajectories for aerial cinematography. This strategy has the advantage to directly generate a smooth and feasible trajectory with adequate speed profiles, without computing an additional timing law. To achieve this, the work Rousseau et al. (2018) is improved by two means. Firstly, the present work avoids the use of a bi-level optimization and only minimizes the duration of the trajectory. This allows generating trajectories with better speed profiles and improves the numerical stability of the algorithm. Secondly, the framework of B-splines is used in order to guarantee the validation of the constraints over the entire trajectory, rather than on a finite set of points (gridding). The results obtained with this trajectory generation strategy, in the context of video making with quadrotors, are then confronted to a simulation. Then, the use of non-uniform clamped B-splines rather than uniform clamped B-splines is discussed. Finally, in order to assess the quality of the obtained trajectories in terms of feasibility and cinematographic quality, a minimum-time trajectory (computed off-line) is performed by a quadrotor during an outdoor flight experiment. As main contributions, this paper proposes a new strategy for generating minimum-time B-spline trajectories and then its validation both in simulation and outdoor flight.

The paper is organized as follows. First, a specification of the problem tackled by this work is given in Section 2. Then, in order to facilitate the comprehension of this paper and to introduce useful notations and preliminary results, a section about B-splines and clamped B-splines is presented in Section 3. A compact way to represent a piecewise clamped B-spline trajectory is introduced in Section 4 and further used in Section 5 to formulate the trajectory generation as an optimization problem. The viability of this strategy is illustrated by an application to quadrotor cinematography, with a simulation and an outdoor flight in Section 6.

Notations. In the sequel, a, $\mathbf{a}$, $\mathbf{A}$ respectively denote a scalar, a vector and a matrix. The transpose of a matrix $\mathbf{A}$ is denoted by $\mathbf{A}^\top$. The set of strictly positive natural numbers is represented by $\mathbb{N}^*$. The symbol $\mathbf{1}_{n\times m} \in \mathbb{R}^{n\times m}$ defines a n-by-m matrix whose elements are 1, while $\mathbf{0}_{n\times m} \in \mathbb{R}^{n\times m}$ defines a n-by-m matrix filled with zeros. $\mathbf{I}_n \in \mathbb{R}^{n\times n}$ designates the n-by-n identity matrix. For $i \in \mathbb{N}$, $j \in \mathbb{N}$, with $i < j$, it is denoted by $[\![i, j]\!]$ the set of consecutive positive integers $\{i, i+1, \dots, j-1, j\}$. Here, $\mathrm{Conv}(\mathcal{S})$ designates the convex hull of a set $\mathcal{S}$. For a vector $\mathbf{x} \in \mathbb{R}^{n\times 1}$, the following notation is used $||\mathbf{x}||_2 = \sqrt{\mathbf{x}^\top\mathbf{x}}$. For two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{3\times 1}$, the notation $\mathbf{u}^\times \in \mathbb{R}^{3\times 3}$ is the skew symmetric matrix such that $\mathbf{u}^\times\ \mathbf{v}$ is equal to the vector product $\mathbf{u} \times \mathbf{v}$.

2. Problem statement

The high level reference considered in this paper is further detailed. It consists in a 3D flight plan, typically used to specify a path for a quadrotor. Though this work was developed in the context of aerial video making, this high level reference remains generic enough to be adapted to other applications.

2.1. Flight plan description

The considered flight plan consists in a serie of $N + 1$ consecutive waypoints $\{\mathbf{w}_0, \mathbf{w}_1, \dots, \mathbf{w}_N\}$, i.e. a starting position $\mathbf{w}_0$ followed by N waypoints to join. Different types of waypoints are further considered

- *Stop waypoint.* The drone stops on the waypoint. The first and last waypoints of the flight plan are usually *stop* waypoints, but they can also be used during the mission for taking pictures, for standing at a given point in order to record a panorama etc.

- *Lock waypoint.* The drone passes on the waypoint. This kind of waypoint can be used to impose precisely the position of the drone when passing through a window or a door for instance, or for specific camera shots.

- *Sphere waypoint.* The drone passes in a neighborhood of a specified radius $r_{\mathbf{w}_i}$ around the waypoint. This allows the drone to perform wider turns around the waypoint while remaining in the flight corridors, resulting in more natural trajectories.

- *Constrained waypoint.* This is a special type of waypoint that can not be chosen by the user. It imposes the derivatives of the trajectory on the waypoint. It is used when the receding waypoint horizon strategy described in Rousseau et al. (2018) is adopted.

For each pair of waypoints, a reference speed v_i, with $i \in [\![1, N]\!]$, and a flight corridor radius $r_{\mathrm{corr}\,i}$ are also specified.

For such a flight plan, this work seeks to generate a feasible trajectory, satisfying the validation criteria of each waypoint and respecting the speed references between each pair of waypoints. In order to be suitable for video making, the trajectory should also be smooth and natural. This criterion will be achieved by limiting the norm of the acceleration and the jerk of the quadrotor (see Section 6).

2.2. *Flight plan pre-processing*

A pre-processing of the flight plan is performed in order to check that the velocity references are pertinent and to split the problem into simpler ones.

The reference speed on each piece of trajectory is clamped so that the vertical component of the velocity lies within admissible bounds and so that the lateral speed does not exceed a given limitation.

Furthermore, the flight plan is split at each *stop* waypoint other than the first (i.e. $\mathbf{w}_0$) and the last (i.e. $\mathbf{w}_N$) waypoints. Whenever two waypoints are superimposed, they are both replaced by stop waypoints and the flight plan is split. This results in several flight plans, containing only *lock* and *sphere* waypoints, except for the first and the last ones. Trajectories are separately generated for each flight plan and are then laid end to end.

In this work, an algorithm for generating trajectories statisfying the specifications given above is proposed, based on the use of B-spline curves. These latter are detailed in the next section.

3. Overview on B-splines

Basis-spline curves (commonly called B-spline curves) are often considered as an extension of the Bézier curves. The latter can be seen as one way to parameterize polynomials as convex combinations of control points, weighted by Bernstein polynomials. In a similar fashion, B-spline curves can be seen as one way to parameterize piecewise polynomials as convex combinations of control points, weighted by B-spline functions. This is useful since piecewise polynomials allow to parameterize rich trajectories while keeping the polynomial degree low.

B-spline curves are piecewise polynomial curves described by

- A set of $(n + 1)$ control points, which have a similar role as the polynomial coefficients;
- A polynomial degree k;
- A set of $(m + 1)$ knots defining where the curve switches between two polynomial representations.

An example of such a B-spline curve of degree 2 is presented on Figure 1, with the control points $\{\mathbf{p}_i\}_{i\in[\![0,4]\!]}$ in black and each polynomial piece of the curve in a different color.

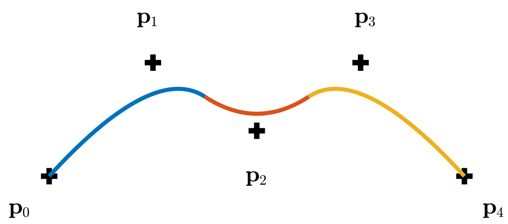

Figure 1: Example of B-spline curve

In order to facilitate the comprehension of this paper, this section gives a quick overview about B-spline curves and introduces notations and preliminary results that will be used in the other sections. It starts with notions on B-spline functions, used as weights on the control points. For more details, De Boor (1978) and Piegl and Tiller (2012) provide in-depth presentations of these objects.

3.1. *B-spline functions*

Given a polynomial degree $k \geqslant 0$ and a vector of $(m + 1) > k + 1$ increasing knots, represented by the matrix

$$\boldsymbol{\tau} = (\tau_0 \; \tau_1 \; \ldots \; \tau_m) \in \mathbb{R}^{1\times(m+1)} \tag{1}$$

a set of $(n + 1) = m - k$ B-spline functions $\{B_{i,k}\}_{i\in[\![0,n]\!]}$ can be defined, using the following recurrence (De Boor (1978), Piegl and Tiller (2012))

$\forall i \in [\![0, n]\!] \quad \forall t \in \mathbb{R}$

If $k = 0$

$$B_{i,k}(t) = \begin{cases} 1 & \text{if } \tau_i \leqslant t < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2a}$$

Else

$$B_{i,k}(t) = \omega_{i,k}(t)\; B_{i,k-1}(t) \;+\; (1 - \omega_{i+1,k}(t))\; B_{i+1,k-1}(t) \tag{2b}$$

with

$$\omega_{i,k}(t) = \begin{cases} \dfrac{t - \tau_i}{\tau_{i+k} - \tau_i} & \text{if } \; \tau_{i+k} > \tau_i \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Each B-spline function is then a piecewise polynomial that switches of polynomial representation at each knot. Notice that, following the definition, the support of $B_{i,k}$ is $[\tau_i, \tau_{i+k+1})$. Outside of this domain, $B_{i,k}$ is null.

These B-spline functions define a partition of unity over the interval $[\tau_k, \tau_{n+1})$ (De Boor (1978), Piegl and Tiller (2012))

$$\begin{aligned} \forall i \in [\![0, n]\!] \quad & B_{i,k} \geqslant 0 \\ \forall t \in [\tau_k, \tau_{n+1}) \quad & \sum_{i=0}^{n} B_{i,k}(t) = 1 \end{aligned} \tag{4}$$

Figure 2 illustrates an example of B-spline functions of degree 2 with 8 knots. The thick line represents the sum of the basis functions and the light blue rectangle represents the domain over which the B-spline functions form a partition of unity.

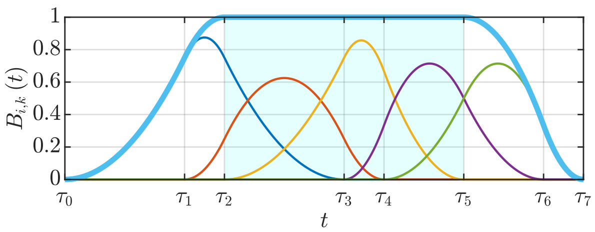

Figure 2: Example of B-spline functions

The B-spline functions can be used to build curves, as explained in the following paragraph.

3.2. B-spline curves

By introducing a set of $(n+1)$ control points $\{\mathbf{p}_i\}_{i \in [\![0,n]\!]}$ represented by

$$\begin{aligned} \forall i \in [\![0, n]\!] \quad \mathbf{p}_i &= \begin{pmatrix} p_i^x & p_i^y & p_i^z \end{pmatrix}^\top \in \mathbb{R}^{3\times 1} \\ \mathbf{P} &= \begin{pmatrix} \mathbf{p}_0 & \mathbf{p}_1 & \dots & \mathbf{p}_n \end{pmatrix} \in \mathbb{R}^{3\times(n+1)} \end{aligned} \tag{5}$$

a B-spline curve $\mathcal{B}$ can be defined as follows

$$\mathcal{B} = \sum_{i=0}^{n} B_{i,k}\, \mathbf{p}_i \tag{6}$$

Remark 1. *Though 3 dimensional control points are used in the following, as they will be used to parameterize the position of a quadrotor, their definition is not restricted to* $\mathbb{R}^3$ *in the general case.*

The evaluation of the curve at $t \in \mathbb{R}$ is then the combination of the control points weighted by the evaluation of the B-spline functions at t. Since these functions are piecewise polynomials, the B-spline curve $\mathcal{B}$ is a piecewise polynomial curve, switching between two polynomial representations at each knot.

Due to (4), in the interval $[\tau_k, \tau_{n+1})$, a B-spline curve verifies the so-called convex hull property (De Boor (1978), Piegl and Tiller (2012)):

Property 1. *For* $i \in [\![k, n]\!]$ *and* $t \in [\tau_i, \tau_{i+1})$

$$\mathcal{B}(t) \in \mathrm{Conv}\Big(\big\{ \mathbf{p}_j \mid j \in [\![i-k, i]\!] \big\} \Big) \tag{7}$$

As a consequence, the curve lies within the convex hull of the entire set of control points in $[\tau_k, \tau_{n+1})$. Though this is a weaker property than (7), it will be used in Section 5 to constrain B-spline curves into convex regions. It can then be convenient to separate the knot vector into:

- k external knots at the beginning of the knot vector and k external knots at the end;
- A starting knot τ_k and an ending knot τ_{n+1}, which define the domain over which the convex hull property holds;
- $n-k$ internal knots, which are the instants for which the B-spline curve switches between two polynomial representations, inside the interval $[\tau_k, \tau_{n+1})$

$$\Big(\underbrace{\tau_0 \;\dots\; \tau_{k-1}}_{k \text{ knots}} \;\; \underset{\substack{\text{starting}\\ \text{knot}}}{\underset{\downarrow}{\tau_k}} \;\; \underbrace{\tau_{k+1} \;\dots\; \tau_n}_{\substack{n-k \\ \text{internal knots}}} \;\; \underset{\substack{\text{ending}\\ \text{knot}}}{\underset{\downarrow}{\tau_{n+1}}} \;\; \underbrace{\tau_{n+2} \;\dots\; \tau_m}_{k \text{ knots}} \Big) \tag{8}$$

Figure 3 illustrates an example of a B-spline curve of degree 2, with the same knot vector as Figure 2. The thick black dots are the control points and the light blue polygon is the interior of their convex hull. The green line is the B-spline curve, evaluated inside (continuous) and outside (dashed) the domain $[\tau_k, \tau_{n+1})$.

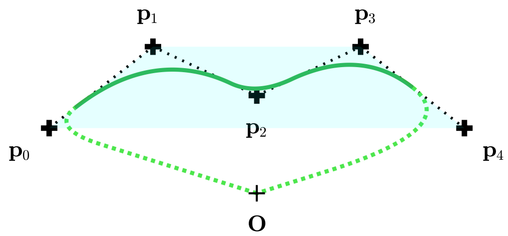

Figure 3: Convex hull of a B-spline of degree 2

The following paragraph details a specific case of B-spline curves, that is later used to parameterize trajectories.

3.3. Clamped B-spline

In this work the trajectory is parameterized as a clamped B-spline, i.e. a B-spline whose first k knots are equal to the starting knot τ_k, and last k knots are equal to the ending knot τ_{n+1}. The knot vector thus has the following form

$$\Big(\underbrace{\tau_k \;\dots\; \tau_k}_{k+1 \text{ equal knots}} \;\; \underbrace{\tau_{k+1} \;\dots\; \tau_n}_{n-k \text{ knots}} \;\; \underbrace{\tau_{n+1} \;\dots\; \tau_{n+1}}_{k+1 \text{ equal knots}} \Big) \tag{9}$$

The impact of this specificity is illustrated on Figure 4, with an example of B-spline functions of degree 2, with a knot vector of the same form as (9) and containing 2 internal knots (τ_3 and τ_4).

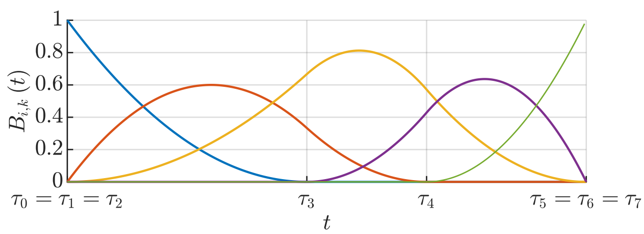

Figure 4: Example of clamped B-spline functions

The consequence of the particular form of the knot vector (9) is that, when evaluated over $[\tau_k, \tau_{n+1})$, the B-spline curve starts on the first control point and ends on the last one

$$\mathcal{B}(\tau_k) = \mathbf{p}_0, \quad \lim_{t \to \tau_{n+1}^-} \mathcal{B}(t) = \mathbf{p}_n \tag{10}$$

Remark 2. *To be more accurate, the curve tends to $\mathbf{p}_n$ when approaching τ_{n+1} by the left. The value of each B-spline function at τ_{n+1} being zero for a clamped B-spline curve, given (2), the evaluation of the curve then jumps to the origin when reaching τ_{n+1}.*

An example of a clamped B-spline curve of degree 2 is given on Figure 5. The knot vector is the same as the one on Figure 4 and the control points are the same as on Figure 3.

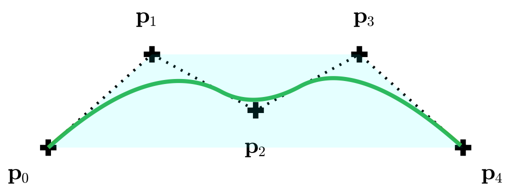

Figure 5: Example of clamped B-spline curve

Furthermore, given the form of this knot vector, it can be written differently. In this paper, a vector of knot steps is thus introduced, represented by the vector

$$\boldsymbol{\Delta\tau} = \begin{pmatrix} \Delta\tau_0 & \Delta\tau_1 & \dots & \Delta\tau_{n-k} \end{pmatrix} \in \mathbb{R}^{1\times(n-k+1)} \tag{11}$$

Given an initial knot τ_0, the knot vector can be reconstructed from the knot step vector as follows

$$\forall j \in [\![0, m]\!] \quad \tau_j = \begin{cases} \tau_0 & \text{if } j \leqslant k \\ \tau_0 + \displaystyle\sum_{i=0}^{j-k-1} \Delta\tau_i & \text{if } k < j \leqslant n \\ \tau_0 + \displaystyle\sum_{i=0}^{n-k} \Delta\tau_i & \text{otherwise} \end{cases} \tag{12}$$

In this work, the knot steps are imposed to be strictly positive, which imply that a B-spline curve of degree k is at least C^{k-1} continuous on $[\tau_k, \tau_{n+1})$. However these knot steps can differ from each other, unlike for *uniform* clamped B-splines, for which they are all equal.

3.4. Clamped B-spline derivatives

The derivatives of a clamped B-spline curve are also clamped B-spline curves, which share the same knot steps as the original curve. The control points of the B-spline curve derivatives are given by linear combination of the original ones. For the l-th derivative, $l \in [\![0, k]\!]$, each of these control points $\mathbf{p}_i^{(l)}$ can be obtained as follows (De Boor (1978), Piegl and Tiller (2012))

$$\mathbf{p}_i^{(l)} = \begin{cases} \mathbf{p}_i & \text{if } l = 0 \\ \dfrac{k-l+1}{\tau_{i+k+1} - \tau_{i+l}} \left(\mathbf{p}_{i+1}^{(l-1)} - \mathbf{p}_i^{(l-1)} \right) & \text{otherwise} \end{cases} \tag{13}$$

In this work, this expression is reformulated using the knot step vector introduced above. Define, for $l \in [\![1, k]\!]$ the matrix $\mathbf{D}_l(\boldsymbol{\Delta\tau}) \in \mathbb{R}^{(n+2-l)\times(n+1-l)}$ such that $\forall (i, j) \in [\![0, n+1-l]\!] \times [\![0, n-l]\!]$

$$D_{l_{i,j}}(\boldsymbol{\Delta\tau}) = \begin{cases} -\rho_j & \text{if } i = j \\ \rho_j & \text{if } i = j+1 \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

with

$$\rho_j = \frac{k-l+1}{\displaystyle\sum_{q=\max(0, j+l-k)}^{\min(n-k, j)} \Delta\tau_q} \tag{15}$$

given by injecting (12) into (13). Then (13) can be re-written

$$\mathbf{P}^{(l)} = \begin{cases} \mathbf{P} & \text{if } l = 0 \\ \mathbf{P}^{(l-1)}\, \mathbf{D}_l(\boldsymbol{\Delta\tau}) & \text{otherwise} \end{cases} \tag{16}$$

Defining, for $l \in [\![1, k]\!]$

$$\mathbf{D}_{0\to l}(\boldsymbol{\Delta\tau}) = \prod_{q=1}^{l} \mathbf{D}_q(\boldsymbol{\Delta\tau}) \tag{17}$$

where $\mathbf{D}_q$ is given by (14), it comes that

$$\mathbf{P}^{(l)} = \mathbf{P}\, \mathbf{D}_{0\to l}(\boldsymbol{\Delta\tau}) \tag{18}$$

This derivative has a polynomial degree $k - l$ and thus possesses $n + 1 - l$ control points. Since it is also a clamped B-spline, it verifies (10)

$$\lim_{t \to \tau_k^+} \frac{\mathrm{d}^l}{\mathrm{d}t^l} \mathcal{B}(t) = \mathbf{p}_0^{(l)}, \quad \lim_{x \to \tau_{n+1}^-} \frac{\mathrm{d}^l}{\mathrm{d}t^l} \mathcal{B}(t) = \mathbf{p}_{n-l}^{(l)} \tag{19}$$

In order to improve the paper readability, the rigorous limits symbols are dropped in the following. It should nonetheless be remembered that, due to their piecewise nature, some derivatives of a B-spline curve are only one-sided, or not defined at all.

3.5. Clamped B-spline integrals

In the same manner, for $l \in \mathbb{N}^*$, the matrix $\mathbf{D}_{-l}(\boldsymbol{\Delta\tau}) \in \mathbb{R}^{(n+l)\times(n+1+l)}$ can be defined such that $\forall (i, j) \in [\![0, n+l]\!] \times [\![0, n+1+l]\!]$

$$D_{-l_{i,j}}(\boldsymbol{\Delta\tau}) = \begin{cases} \sigma_i & \text{if } j \geqslant i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

with

$$\sigma_i = \frac{\displaystyle\sum_{q=\max(0, i-k-l)}^{\min(n-k, i-1)} \Delta\tau_q}{k+l} \tag{21}$$

The control points of the l-th integral ($l \geqslant 0$) can then be obtained as follows

$$\mathbf{P}^{(-l)} = \begin{cases} \mathbf{P} & \text{if } l = 0 \\ \mathbf{P}^{(1-l)}\,\mathbf{D}_{-l}(\boldsymbol{\Delta\tau}) + \mathbf{p}_0^{(-l)}\,\mathbf{1}_{1\times(n+1+l)} & \text{otherwise} \end{cases} \tag{22}$$

with $\mathbf{p}_0^{(-l)}$ the initial value of the l-th integral, at the starting knot, and $\mathbf{1}_{a\times b}$ a a-by-b matrix filled with ones. This integral has a polynomial degree $k + l$.

As a consequence, it is easy to constrain a B-spline curve and its derivatives (or integrals) into convex polytopes, thanks to the convex hull property.

In this section, clamped B-spline curves were presented along with some of their properties. In the following, they are used to parameterize trajectories, composed of several clamped B-spline curves laid end to end. Considering the specifications to be met by these trajectories (detailed in Section 2), a novel, compact way to represent them is further proposed.

4. Compact representation of the trajectory

4.1. Piecewise clamped B-spline trajectory

For a flight plan containing $N + 1$ waypoints, the trajectory is divided into N pieces, each one joining a pair of consecutive waypoints. For $i \in [\![1, N]\!]$, the i-th piece joins the waypoint $\mathbf{w}_{i-1}$ to the waypoint $\mathbf{w}_i$. Each i-th piece of the trajectory is parameterized by a clamped B-spline curve $\mathcal{B}_i$, defined by the knot steps $\boldsymbol{\Delta\tau}_i \in \mathbb{R}^{1\times(n-k+1)}$

$$\boldsymbol{\Delta\tau}_i = \begin{pmatrix} \Delta\tau_{i,0} & \Delta\tau_{i,1} & \dots & \Delta\tau_{i,n-k} \end{pmatrix} \tag{23}$$

and the control points $\mathbf{P}_i \in \mathbb{R}^{3\times(n+1)}$

$$\mathbf{P}_i = \begin{pmatrix} \mathbf{p}_{i,0} & \mathbf{p}_{i,1} & \dots & \mathbf{p}_{i,n} \end{pmatrix} \triangleq \begin{pmatrix} \mathbf{P}_i^x \\ \mathbf{P}_i^y \\ \mathbf{P}_i^z \end{pmatrix} \tag{24}$$

This choice to parameterize each piece of trajectory by a clamped B-spline is motivated by the two following reasons

- The extremities of each piece of trajectory correspond to the validation of a waypoint. The properties of clamped B-splines (19) and (10) are especially convenient to adjust the position and its time derivatives at the extremities of the pieces of trajectory, in order to meet the validation requirements of each waypoint.
- Each piece of trajectory should lie within its flight corridors and the magnitude of the time derivatives of the position should verify given upper bounds. The fact that the derivatives of a clamped B-spline are clamped B-spline too and the property (7) of these curves allows to efficiently formulate these constraints.

These trajectory pieces are connected so that the $L + 1$ first time derivatives (0-th derivative, i.e. the position, to the L-th time derivative) coincide at the connections, with $L < k$. The overall trajectory is then C^L continuous. Given an initial time t_0, these connections occur at the time instants $\{t_i\}_{i\in[\![1,N]\!]}$ such as

$$\forall i \in [\![1, N]\!] \quad t_i = t_{i-1} + \sum_{j=0}^{n-k} \Delta\tau_{i,j} \tag{25}$$

and correspond to the time instants where a waypoint is validated.

The evaluation of the overall trajectory $\boldsymbol{\zeta}$ at time t is given by

$$\boldsymbol{\zeta}(t) = \begin{cases} \mathbf{w}_0 & \text{if } t < t_0 \\ \mathcal{B}_i(t) & \text{if } t_{i-1} \leqslant t < t_i,\ i \in [\![1, N]\!] \\ \mathbf{w}_N & \text{if } t \geqslant t_N \end{cases} \tag{26}$$

with $\mathbf{w}_0, \mathbf{w}_N \in \mathbb{R}^{3\times1}$.

In order to validate the waypoints while remaining smooth, this trajectory has to satisfy several constraints at the connections between its different pieces, that are now detailed.

4.2. Trajectory derivatives at the connections

In order to get an overall trajectory that is C^L, $L < k$, and to respect the criteria of validation of each waypoint, boundary conditions on the time derivatives must be met at the beginning and at the end of each piece of trajectory. Depending on the type of waypoints, these derivatives can be imposed or set free, in which case they must coincide with the ones of the previous and the following pieces of trajectory. Considering that the first waypoint is either a *stop* or a *constrained* waypoint and that the last one is always a *stop* waypoint, these constraints on the time derivatives can be formulated as follows

$\forall i \in [\![1, N]\!]$

If $\mathbf{w}_{i-1}$ is a *stop* waypoint

$$\begin{cases} \mathcal{B}_i(t_{i-1}) = \mathbf{w}_{i-1} \\ \forall l \in [\![1, L]\!] \quad \dfrac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_i(t_{i-1}) = \mathbf{0} \end{cases} \tag{27a}$$

Else, if $\mathbf{w}_{i-1}$ is a *constrained* waypoint

$$\forall l \in [\![0, L]\!] \quad \frac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_i(t_{i-1}) = \mathbf{w}_{i-1}^{(l)} \tag{27b}$$

Else, if $\mathbf{w}_{i-1}$ is a *lock* waypoint

$$\begin{cases} \mathcal{B}_i(t_{i-1}) = \mathbf{w}_{i-1} \\ \forall l \in [\![1, L]\!] \quad \dfrac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_i(t_{i-1}) = \dfrac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_{i-1}(t_{i-1}) \end{cases} \tag{27c}$$

Else, if $\mathbf{w}_{i-1}$ is a *sphere* waypoint

$$\forall l \in [\![0, L]\!] \quad \frac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_i(t_{i-1}) = \frac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_{i-1}(t_{i-1}) \tag{27d}$$

If $\mathbf{w}_i$ is a *stop* waypoint

$$\forall l \in [\![0, L]\!] \quad \frac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_i(t_i) = \mathbf{0} \tag{27e}$$

Else, if $\mathbf{w}_i$ is a *lock* waypoint

$$\mathcal{B}_i(t_i) = \mathbf{w}_i \tag{27f}$$

Else, if $\mathbf{w}_i$ is a *sphere* waypoint

$$\|\mathcal{B}_i(t_i) - \mathbf{w}_i\|_2 \leqslant r_{\mathbf{w}_i} \tag{27g}$$

where $r_{\mathbf{w}_i}$ is the validation radius of the waypoint $\mathbf{w}_i$ and $\mathbf{w}_{i-1}^{(l)}$ is value of the l-th derivative of the trajectory imposed at the validation of the waypoint $\mathbf{w}_i$. Since clamped B-splines are considered, these constraints can directly be written as constraints on the first and the last control points of each piece of trajectory and their derivatives.

$$\frac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_i(t_{i-1}) = \mathbf{p}_{i,0}^{(l)}$$
$$\frac{\mathrm{d}^l}{\mathrm{d}t^l}\mathcal{B}_i(t_i) = \mathbf{p}_{i,n-l}^{(l)}$$

Whatever the type of waypoint, the $L+1$ first time derivatives (including the position) of a piece of trajectory are then constrained at its beginning. Some derivatives can also be imposed at the end of this piece, depending on the type of waypoint.

In the following, the time derivatives of the overall trajectory ζ at the time instant t_i, with $i \in [\![0, N]\!]$, are represented by the matrix $\mathbf{\Gamma}_{\mathbf{w}_i} \in \mathbb{R}^{3\times(L+1)}$

$$\mathbf{\Gamma}_{\mathbf{w}_i} = \begin{pmatrix} \zeta(t_i) & \frac{\mathrm{d}^1}{\mathrm{d}t^1}\zeta(t_i) & \dots & \frac{\mathrm{d}^L}{\mathrm{d}t^L}\zeta(t_i) \end{pmatrix} \tag{28}$$

These constraints allow reducing the number of parameters necessary for the trajectory generation, as explained in the next paragraph.

4.3. Reduced set of control points

As a consequence, the $L+1$ first control points of each piece of trajectory are imposed and can thus be removed from the trajectory generation problem. Furthermore, depending on the type of waypoint at the end of the trajectory piece, the $L+1$ last control points (*stop* waypoint) or the last control point (*lock* waypoint) can also be removed.

For each i-th piece of trajectory, a *reduced set of control points* is thus introduced, equal to the set of control points of the i-th piece minus the control points imposed by the continuity constraints. For $i \in [\![1, N]\!]$, the number of reduced control points of the i-th piece of trajectory is denoted by $\widetilde{n}_i \leqslant n - L$ such that

$$\widetilde{n}_i = \begin{cases} n - 2L - 1 & \text{if } \mathbf{w}_i \text{ is } \textit{stop} \\ n - L - 1 & \text{if } \mathbf{w}_i \text{ is } \textit{lock} \\ n - L & \text{if } \mathbf{w}_i \text{ is } \textit{sphere} \end{cases} \tag{29}$$

and the reduced set of control points is represented by the matrix $\widetilde{\mathbf{P}}_i \in \mathbb{R}^{3\times\widetilde{n}_i}$

$$\widetilde{\mathbf{P}}_i = \begin{pmatrix} \mathbf{p}_{L+1} & \mathbf{p}_{L+2} & \dots & \mathbf{p}_{\widetilde{n}_i+L} \end{pmatrix} \triangleq \begin{pmatrix} \widetilde{\mathbf{P}}_i^x \\ \widetilde{\mathbf{P}}_i^y \\ \widetilde{\mathbf{P}}_i^z \end{pmatrix} \tag{30}$$

The rest of this section details the reconstruction of the full set of control points from the reduced one, starting with the $L+1$ first control points.

4.4. Reconstruction of the $L+1$ first control points

For the i-th piece of trajectory, $i \in [\![1, N]\!]$, the $L+1$ first time derivatives at the beginning of the piece of trajectory, $\mathbf{\Gamma}_{\mathbf{w}_{i-1}}$, are either

- Explicitly given by the waypoint $\mathbf{w}_{i-1}$ (*stop* or *constrained* waypoint);
- Imposed equal to the $L+1$ first time derivatives at the end of the previous piece of trajectory (piece $i-1$, if $i > 1$) by the continuity constraints (*sphere* waypoint). These derivatives are given by the last $L+1$ control points of the $(i-1)$-th piece of trajectory;
- Given both by the waypoint $\mathbf{w}_{i-1}$ and the time derivatives at the end of the $(i-1)$-th piece of trajectory (*lock* waypoint).

The reconstruction of the $L+1$ first control points of the i-th piece of trajectory can then be performed in two steps, illustrated on Figure 6, for $L = 2$

- First, the time derivatives $\mathbf{\Gamma}_{\mathbf{w}_{i-1}}$ (green arrow on Figure 6) are determined from $\mathbf{w}_{i-1}$ and the last $L+1$ control points of the previous piece (piece $i-1$, if $i > 1$), using the derivative properties (16). The latter are represented by the matrix $\mathbf{P}_{i-1,\text{end}}$, equal to the last $L+1$ columns of $\mathbf{P}_{i-1}$ (control points in solid red on Figure 6);
- Then, the first $L+1$ control points of the piece i are reconstructed from $\mathbf{\Gamma}_{\mathbf{w}_{i-1}}$, using the integral properties (22). These control points are represented by the matrix $\mathbf{P}_{i,\text{start}}$ equal to the first $L+1$ columns of $\mathbf{P}_i$ (control points in solid blue on Figure 6).

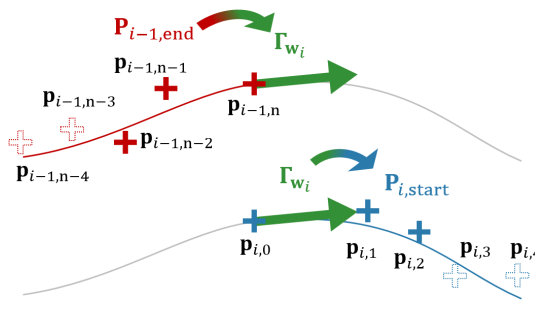

Figure 6: Reconstruction of the first $L+1$ control points

These two steps are now detailed, starting with the determination of $\mathbf{\Gamma}_{\mathbf{w}_{i-1}}$.

4.4.1. Derivatives at the end of the previous piece of trajectory

Since clamped B-splines are used to parameterize each piece of trajectory, the $L+1$ first time derivatives at the end of a piece are given by the last control point of each of its derivatives (19). These can be obtained from the knot steps and the last $L+1$ control points as follows.

For a given knot step vector $\boldsymbol{\Delta\tau}$, using (16), the matrix $\mathbf{O}_l(\boldsymbol{\Delta\tau}) \in \mathbb{R}^{(L+1)\times(L+1)}$ can be defined such that, for $l \in [\![1, L]\!]$ and $\forall(i, j) \in [\![0, L]\!]^2$

$$O_{l_{i,j}}(\boldsymbol{\Delta\tau}) = \begin{cases} 1 & \text{if } j < l \text{ and } i = j \\ -\gamma_j & \text{if } j \geqslant l \text{ and } i = j \\ \gamma_j & \text{if } j \geqslant l \text{ and } i = j-1 \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

where

$$\gamma_j = \frac{k - l + 1}{\sum_{q=n-k-j+l}^{n-k} \Delta\tau_q} \tag{32}$$

For $i \in [\![2, N]\!]$ the *control points to derivatives* matrix $\mathbf{M}_{\mathrm{p}\to\mathrm{d}}(\boldsymbol{\Delta\tau}) \in \mathbb{R}^{(L+1)\times(L+1)}$ giving the first $L+1$ derivatives at the end of the $(i-1)$-th piece of trajectory from its last $L+1$ control points can be defined

$$\mathbf{M}_{\mathrm{p}\to\mathrm{d}}(\boldsymbol{\Delta\tau}) = \mathbf{J}_{L+1} \prod_{l=1}^{L} \mathbf{O}_l(\boldsymbol{\Delta\tau}) \tag{33}$$

where $\mathbf{J}_a$ is the a-by-a anti-diagonal matrix with every anti-diagonal term equal to 1. The derivatives are then obtained as follows

$$\begin{aligned} &\begin{pmatrix} \mathbf{p}_{i-1,n} & \mathbf{p}^{(1)}_{i-1,n-1} & \cdots & \mathbf{p}^{(L)}_{i-1,n-L} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{p}_{i-1,n-L} & \mathbf{p}_{i-1,n-L+1} & \cdots & \mathbf{p}_{i-1,n} \end{pmatrix} \mathbf{M}_{\mathrm{p}\to\mathrm{d}}(\boldsymbol{\Delta\tau}_{i-1}) \end{aligned} \tag{34}$$

4.4.2. Derivatives of the trajectory on a waypoint

As explained before, some derivatives of the trajectory can be imposed on a waypoint, depending on its type. A generic expression of the first $L+1$ derivatives at the validation of a waypoint $\mathbf{w}_i$, $i \in [\![0, N]\!]$ is then

If $i = 0$

$$\boldsymbol{\Gamma}_{\mathbf{w}_i} = \begin{cases} \begin{pmatrix} \mathbf{w}_i & \mathbf{0}_{3\times L} \end{pmatrix} & \text{if } \mathbf{w}_i \text{ is } \textit{stop} \\ \begin{pmatrix} \mathbf{w}_i & \mathbf{w}_i^{(1)} & \cdots & \mathbf{w}_i^{(L)} \end{pmatrix} & \text{if } \mathbf{w}_i \text{ is } \textit{constrained} \end{cases} \tag{35a}$$

Else if $i = N$

$$\boldsymbol{\Gamma}_{\mathbf{w}_i} = \begin{pmatrix} \mathbf{w}_i & \mathbf{0}_{3\times L} \end{pmatrix} \tag{35b}$$

Else

$$\boldsymbol{\Gamma}_{\mathbf{w}_i}(\boldsymbol{\Delta\tau}_i, \mathbf{P}_{i,\mathrm{end}}) = \mathbf{P}_{i,\mathrm{end}}\, \mathbf{M}_{\mathrm{p}\to\mathrm{d}}(\boldsymbol{\Delta\tau}_i) \tag{35c}$$

where

$$\mathbf{P}_{i,\mathrm{end}} = \begin{cases} \begin{pmatrix} \mathbf{p}_{i,n-L} & \mathbf{p}_{i,n-L+1} & \cdots & \mathbf{p}_{i,n-1} & \mathbf{w}_i \end{pmatrix} & \text{if } \mathbf{w}_i \text{ is } \textit{lock} \\ \begin{pmatrix} \mathbf{p}_{i,n-L} & \mathbf{p}_{i,n-L+1} & \cdots & \mathbf{p}_{i,n} \end{pmatrix} & \text{if } \mathbf{w}_i \text{ is } \textit{sphere} \end{cases} \tag{36}$$

with the control points $\mathbf{p}_{i,j}$ in (36) being part of the reduced set $\widetilde{\mathbf{P}}_i$ (see (30)).

Based on $\boldsymbol{\Gamma}_{\mathbf{w}_{i-1}}$, the reconstruction of the first $L+1$ control points of the piece $i \in [\![1, N]\!]$ can now be performed, as detailed in the next paragraph.

4.4.3. Starting derivatives to control points

The $L+1$ first control points of a piece of trajectory can be obtained from the first $L+1$ time derivatives at its beginning. For a given knot step vector $\boldsymbol{\Delta\tau}$, using (22), the matrix $\mathbf{Q}_l(\boldsymbol{\Delta\tau}) \in \mathbb{R}^{(L+1)\times(L+1)}$ can be defined such that, for $l \in [\![1, L]\!]$ and $\forall(i, j) \in [\![0, L]\!]^2$

$$Q_{l_{i,j}}(\boldsymbol{\Delta\tau}) = \begin{cases} 1 & \text{if } j < l \text{ and } i = j \\ \nu_j & \text{if } j \geqslant l \text{ and } i = j \\ 1 & \text{if } j \geqslant l \text{ and } i = j-1 \\ 0 & \text{otherwise} \end{cases} \tag{37}$$

where

$$\nu_j = \frac{\sum_{q=0}^{l-1} \Delta\tau_q}{k + l - j} \tag{38}$$

For $i \in [\![1, N]\!]$ the *derivative to control points* matrix $\mathbf{M}_{\mathrm{d}\to\mathrm{p}}(\boldsymbol{\Delta\tau}) \in \mathbb{R}^{(L+1)\times(L+1)}$ giving the first $L+1$ control points of the i-th piece of trajectory from the first $L+1$ derivatives at the beginning of this piece of trajectory can be defined

$$\mathbf{M}_{\mathrm{d}\to\mathrm{p}}(\boldsymbol{\Delta\tau}) = \prod_{l=0}^{L-1} \mathbf{Q}_{L-l}(\boldsymbol{\Delta\tau}) \tag{39}$$

which gives the first $L+1$ control points

$$\mathbf{P}_{i,\mathrm{start}} = \begin{pmatrix} \mathbf{p}_{i,0} & \mathbf{p}_{i,1} & \cdots & \mathbf{p}_{i,L} \end{pmatrix} \tag{40}$$

from the starting derivatives. Indeed, for $i \in [\![1, N]\!]$, the following expressions hold

If $i = 1$

$$\mathbf{P}_{i,\mathrm{start}}(\boldsymbol{\Delta\tau}_i) = \boldsymbol{\Gamma}_{\mathbf{w}_{i-1}}\, \mathbf{M}_{\mathrm{d}\to\mathrm{p}}(\boldsymbol{\Delta\tau}_i) \tag{41a}$$

Else

$$\mathbf{P}_{i,\mathrm{start}}(\boldsymbol{\Delta\tau}_{i-1}, \boldsymbol{\Delta\tau}_i, \mathbf{P}_{i-1,\mathrm{end}}) = \boldsymbol{\Gamma}_{\mathbf{w}_{i-1}}(\boldsymbol{\Delta\tau}_{i-1}, \mathbf{P}_{i-1,\mathrm{end}})\, \mathbf{M}_{\mathrm{d}\to\mathrm{p}}(\boldsymbol{\Delta\tau}_i) \tag{41b}$$

The reconstruction of the remaining control points is now detailed.

4.5. Reconstruction of the full set of control points

In some cases, the last control points are also imposed. In the case where the next waypoint is a *stop* waypoint, the last $L+1$ control points are imposed equal to $\mathbf{w}_i$. Similarly, if $\mathbf{w}_i$ is a *lock* waypoint, the last control point is constrained on the waypoint. In the case of a *sphere* waypoint, there is no constraint on the last control points.

Finally, the remaining control points (i.e. other than the first and the last ones) do not require reconstruction, as they are directly equal to the reduced control points.

As a consequence, the overall reconstruction operation can be formulated as follows, for $i \in [\![1, N]\!]$

If $i = 1$

$$\mathbf{P}_i\left(\boldsymbol{\Delta\tau}_i, \widetilde{\mathbf{P}}_i\right) = \mathbf{w}_i\ \mathbf{T}_{\mathrm{end}i} + \widetilde{\mathbf{P}}_i\ \mathbf{T}_{\mathrm{mid}i} + \mathbf{P}_{i,\mathrm{start}}(\boldsymbol{\Delta\tau}_i)\ \mathbf{T}_{\mathrm{start}i} \tag{42a}$$

Else

$$\begin{aligned}\mathbf{P}_i\left(\boldsymbol{\Delta\tau}_{i-1}, \boldsymbol{\Delta\tau}_i, \mathbf{P}_{i-1,\mathrm{end}}, \widetilde{\mathbf{P}}_i\right) &= \mathbf{w}_i\ \mathbf{T}_{\mathrm{end}i} + \widetilde{\mathbf{P}}_i\ \mathbf{T}_{\mathrm{mid}i} \\ &\quad + \mathbf{P}_{i,\mathrm{start}}\left(\boldsymbol{\Delta\tau}_{i-1}, \boldsymbol{\Delta\tau}_i, \mathbf{P}_{i-1,\mathrm{end}}\right)\ \mathbf{T}_{\mathrm{start}i}\end{aligned} \tag{42b}$$

with

$$\begin{aligned}\mathbf{T}_{\mathrm{start}i} &= \begin{pmatrix}\mathbf{I}_{L+1} & \mathbf{0}_{(L+1)\times(n-L)}\end{pmatrix} \\ \mathbf{T}_{\mathrm{mid}i} &= \begin{pmatrix}\mathbf{0}_{\widetilde{n}_i\times(L+1)} & \mathbf{I}_{\widetilde{n}} & \mathbf{0}_{\widetilde{n}_i\times(n-\widetilde{n}_i-L)}\end{pmatrix} \\ \mathbf{T}_{\mathrm{end}i} &= \begin{cases}\begin{pmatrix}\mathbf{0}_{1\times(n-L)} & \mathbf{1}_{1\times(L+1)}\end{pmatrix} & \text{if } \mathbf{w}_i \text{ is } \textit{stop} \\ \begin{pmatrix}\mathbf{0}_{1\times n} & 1\end{pmatrix} & \text{if } \mathbf{w}_i \text{ is } \textit{lock} \\ \mathbf{0}_{1\times(n+1)} & \text{otherwise}\end{cases}\end{aligned} \tag{43}$$

Therefore, in the general case, the full set of control points of a given piece of trajectory

- Linearly depends on the reduced control points of this piece of trajectory;
- Nonlinearly depends on both the knot steps of this piece of trajectory and the knot steps and the reduced control points of the previous piece of trajectory.

The knot steps and the reduced control points thus constitute the degrees of freedom for adjusting the trajectory.

In this section, a novel, compact way to parameterize the entire trajectory $\boldsymbol{\zeta}$ as a piecewise clamped B-spline curve, by the mean of knot step vectors and reduced sets of control points was proposed. In the next section, this convenient compact representation is used to formulate the trajectory generation process as an optimization problem.

5. Minimum-time trajectory

One way to obtain a smooth trajectory meeting the requirements of Section 2 is to generate a minimum-time trajectory with constraints on the magnitude of the velocity and its derivatives. One contribution of this work is thus to formalize the minimum-time trajectory generation as an optimization problem on the duration of the trajectory, using the formalism of B-splines. The first paragraph details one general way to formulate this problem, using the trajectory parameterization introduced in Section 4. Then, the expression of the constraints is detailed in the following paragraphs. Notice that the formulation of the problem is specified for 3D trajectories, however it is easily adaptable to other dimensions and to other applications than quadrotor cinematography.

5.1. Optimization problem

The trajectory is parameterized as a piecewise clamped B-spline curve, with the compact representation presented in Section 4. It is then parameterized by

- N knot step vectors $\boldsymbol{\Delta\tau}_i$, each vector containing $n-k$ elements;
- N reduced sets of control points $\widetilde{\mathbf{P}}_i$, with the i-th set containing $\widetilde{n}_i$ control points, themselves given by 3 coordinates.

This gives a total of $n_\mathbf{x}$ parameters which are stored in the vector $\mathbf{x} \in \mathbb{R}^{n_\mathbf{x}}$, such that

$$\begin{aligned}\mathbf{x} = \Big(\boldsymbol{\Delta\tau}_1\ \ \boldsymbol{\Delta\tau}_2\ \ \ldots\ \ \boldsymbol{\Delta\tau}_N\ \ \widetilde{\mathbf{P}}_1^x\ \ \widetilde{\mathbf{P}}_1^y\ \ \widetilde{\mathbf{P}}_1^z \\ \widetilde{\mathbf{P}}_2^x\ \ \widetilde{\mathbf{P}}_2^y\ \ \widetilde{\mathbf{P}}_2^z\ \ \ldots\ \ \widetilde{\mathbf{P}}_N^x\ \ \widetilde{\mathbf{P}}_N^y\ \ \widetilde{\mathbf{P}}_N^z\Big)^\top\end{aligned} \tag{44}$$

containing all the information required to reconstruct the trajectory over its entire domain.

In order to complete the flight plan described in Section 2, a vector $\mathbf{x}$ minimizing the duration $T \in \mathbb{R}_+^*$ of the flight plan is chosen, under the following constraints

- The knot steps are strictly positive;
- The trajectory does not exit the corridors;
- The trajectory validates the *sphere* waypoints by passing within a neighborhood of specified radius around them;
- The magnitude of the trajectory derivatives does not exceed specified bounds.

The trajectory generation can then be formulated as the following nonlinear optimization problem

$$\begin{aligned}&\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathbb{R}^{n_\mathbf{x}}}\ T(\mathbf{x}) \\ &\text{s.t.} \begin{cases}x_i \geqslant \Delta\tau_{\min},\ i \in [\![0, (n-k+1)\,N-1]\!] \\ \mathbf{A}_{\mathrm{corridor}}\,\mathbf{x} - \mathbf{b}_{\mathrm{corridor}} \leqslant 0 \\ g_{\mathrm{corridor}}(\mathbf{x}) \leqslant 0 \\ g_{\mathrm{sphere}}(\mathbf{x}) \leqslant 0 \\ g_{\mathrm{derivatives}}(\mathbf{x}) \leqslant 0\end{cases}\end{aligned} \tag{45}$$

with the linear cost function

$$T(\mathbf{x}) = \sum_{i=1}^{N}\sum_{j=0}^{n-k}\Delta\tau_{i,j} \tag{46}$$

The expressions of the constraints are detailed in Section 5.2, Section 5.3 and Section 5.4.

Problem (45) constitutes a nonlinear programming problem (NLP) that can be solved using classical algorithms, such as a Sequential Quadratic Programming algorithm (SQP, Fletcher (2013)) for instance. This requires some care in the choice of the starting point of the algorithm. A series of rest-to-rest trajectories joining each pair of waypoints can be a candidate as initial guess for solving the problem. Though it can be quite suboptimal, there always exists a rest-to-rest trajectory joining two waypoints which satisfies the constraints of (45) and it is reasonably simple to compute one. An example of such an initial guess is proposed in Section 6.2.

The tuning parameters of the algorithm are the polynomial degree $k \geqslant 0$ and the number of control points $(n+1) > k$ of the clamped B-splines (thus setting their number of knot steps $n-k+1$). As mentioned in Section 3.3, a clamped B-spline with strictly positive knot steps is C^{k-1} continuous. In order to obtain a trajectory that is C^L, the polynomial degree must then verify $k > L$. The lowest value $k = L+1$ constitutes a good candidate as it keeps the polynomial degree as low as possible. Only the parameter n then remains to be set and constitutes a tuning parameter. However, the choice of n is not entirely free. As explained in Section 4, for a given vector of knot steps, each constraint on the time derivatives of the trajectory in (27) imposes the value of one control points (and this is used to reduce the number of optimized control points). For a C^L trajectory, there are at most $L+1$ constraints on the derivatives at one extremity of a piece of trajectory, when validating a waypoint. Since each piece of trajectory has 2 extremities, choosing $n \geqslant 2(L+1)$ guarantees that there are always enough control points to satisfy these constraints, whatever the type of waypoints.

5.2. *Corridor constraints*

The flight corridors are approximated by n_{corr}-prisms ($n_{\text{corr}} \geqslant 3$), leading to $n_{\text{corr}} + 2$ constraints for each control point

- 2 longitudinal constraints ensuring that the control point lies between the two waypoints of the corresponding piece of trajectory;
- n_{corr} lateral constraints ensuring that the control point respects the cross section of the prism (and thus of the cylinder), as illustrated in Figure 7.

Corridor longitudinal constraints. Define the unit vector $\mathbf{u}_i$, for $i \in [\![1, N]\!]$

$$\mathbf{u}_i = \frac{\mathbf{w}_i - \mathbf{w}_{i-1}}{\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2} \tag{47}$$

where $\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2$ is always strictly positive, see Section 2.2. Then, the longitudinal constraints are given by

$$\begin{aligned} \mathbf{u}_i{}^\top \left(\mathbf{p}_{i,j} - \mathbf{w}_{i-1}\right) &\geqslant -\varepsilon \\ \mathbf{u}_i{}^\top \left(\mathbf{p}_{i,j} - \mathbf{w}_{i-1}\right) &\leqslant \|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 + \varepsilon \end{aligned} \tag{48}$$

with $\varepsilon \geqslant 0$ a given tolerance that can be set to 0. These constraints can be rewritten as

$$\begin{aligned} -\mathbf{u}_i{}^\top \mathbf{p}_{i,j} + \mathbf{u}_i{}^\top \mathbf{w}_{i-1} - \varepsilon &\leqslant 0 \\ \mathbf{u}_i{}^\top \mathbf{p}_{i,j} - \mathbf{u}_i{}^\top \mathbf{w}_{i-1} - \|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 - \varepsilon &\leqslant 0 \end{aligned} \tag{49}$$

Corridor lateral constraints. Regarding the lateral constraints, they can be formulated by considering the 2D vector which corresponds to the projection of $\left(\mathbf{p}_{i,j} - \mathbf{w}_{i-1}\right)$ onto the cross section of the prism. To this aim, a 2D orthonormal basis $\left\{\tilde{\mathbf{e}}_{x_i}, \tilde{\mathbf{e}}_{y_i}\right\}$ of this cross section is constructed, in which the equation of the boundaries of the prism can be easily expressed. The vector among $\left\{\mathbf{u}_i^\times \mathbf{e}_x,\ \mathbf{u}_i^\times \mathbf{e}_y,\ \mathbf{u}_i^\times \mathbf{e}_z\right\}$, with the highest norm is normalized and chosen as the first vector of this basis, $\tilde{\mathbf{e}}_{x_i}$. The second vector is then $\tilde{\mathbf{e}}_{y_i} = \mathbf{u}_i^\times \tilde{\mathbf{e}}_{x_i}$. On Figure 7, the corridor between the waypoints $\mathbf{w}_{i-1}$ and $\mathbf{w}_i$ is represented in light blue and the unit vector $\mathbf{u}_i$ is represented in red. The cross section of a 5-prism is represented in dark blue, with its basis $\left\{\tilde{\mathbf{e}}_{x_i}, \tilde{\mathbf{e}}_{y_i}\right\}$ in green. The control point $\mathbf{p}_{i,j}$ (in orange) lies within the section of the prism and it is between the two waypoints, which guarantees its satisfaction of the corridor constraint.

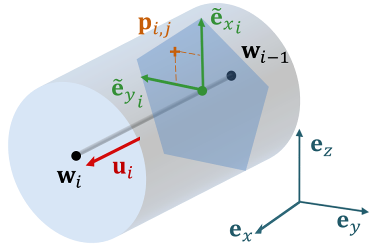

Figure 7: Approximation of the cylindrical flight corridor by a prism

The n_{corr} lateral constraints on each control point can be expressed as follows, $\forall i \in [\![1, N]\!]$, $\forall j \in [\![0, n]\!]$, $\forall q \in [\![1, n_{\text{corr}}]\!]$

$$\begin{aligned} a_{i,q}\, \tilde{\mathbf{e}}_{x_i}^\top &\left(\mathbf{u}_i^\times \left(\mathbf{p}_{i,j} - \mathbf{w}_{i-1}\right)\right) \\ &+ b_{i,q}\, \tilde{\mathbf{e}}_{y_i}^\top \left(\mathbf{u}_i^\times \left(\mathbf{p}_{i,j} - \mathbf{w}_{i-1}\right)\right) + c_{i,q} \leqslant 0 \end{aligned} \tag{50}$$

where the following coefficients are defined

$$\begin{aligned} a_{i,q} &= \cos(q\delta) - \cos((q+1)\delta) \\ b_{i,q} &= \sin((q+1)\delta) - \sin(q\delta) \\ c_{i,q} &= -r_{\text{corr}\,i} \sin(\delta) \end{aligned} \tag{51}$$

with $\delta = \frac{2\pi}{n_{\text{corr}}}$. Regrouping the terms related to the control points and to the waypoints, the expression (50) can be rewritten as follows, $\forall i \in [\![1, N]\!]$, $\forall j \in [\![0, n]\!]$, $\forall q \in [\![1, n_{\text{corr}}]\!]$

$$\begin{aligned} &\left(a_{i,q}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,q}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times\ \mathbf{p}_{i,j} \\ &\qquad - \left(a_{i,q}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,q}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times\ \mathbf{w}_{i-1} + c_{i,q} \leqslant 0 \end{aligned} \tag{52}$$

Corridor inequality constraints. The corridor constraints (49) and (52) for a given control point can be written as follows

$$\boldsymbol{\Lambda}_i\, \mathbf{p}_{i,j} - \boldsymbol{\lambda}_i \leqslant 0 \tag{53}$$

with $\mathbf{p}_{i,j}$ given by (42) and

$$\begin{aligned} \boldsymbol{\Lambda}_i &= \begin{pmatrix} -\mathbf{u}_i{}^\top \\ \mathbf{u}_i{}^\top \\ \left(a_{i,1}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,1}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times \\ \left(a_{i,2}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,2}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times \\ \vdots \\ \left(a_{i,n_{\text{corr}}}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,n_{\text{corr}}}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times \end{pmatrix} \\ \boldsymbol{\lambda}_i &= \begin{pmatrix} -\mathbf{u}_i{}^\top \mathbf{w}_{i-1} + \varepsilon \\ \mathbf{u}_i{}^\top \mathbf{w}_{i-1} + \|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 + \varepsilon \\ \left(a_{i,1}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,1}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times\ \mathbf{w}_{i-1} - c_{i,1} \\ \left(a_{i,2}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,2}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times\ \mathbf{w}_{i-1} - c_{i,2} \\ \vdots \\ \left(a_{i,n_{\text{corr}}}\, \tilde{\mathbf{e}}_{x_i}^\top + b_{i,n_{\text{corr}}}\, \tilde{\mathbf{e}}_{y_i}^\top\right) \mathbf{u}_i^\times\ \mathbf{w}_{i-1} - c_{i,n_{\text{corr}}} \end{pmatrix} \end{aligned} \tag{54}$$

Since the reduced control points are part of the optimization vector $\mathbf{x}$ defined in (44), this constraint is linear relatively to $\mathbf{x}$ if $\mathbf{p}_{i,j}$ belongs to the reduced set of control points. However, the other control points of the full set must be reconstructed, using (42), which is a nonlinear operation on the optimization vector. As a consequence, this constraint is nonlinear for the control points needing reconstruction.

The corridor constraints which are linear in $\mathbf{x}$ can be gathered and expressed as follows

$$\mathbf{A}_{\text{corridor}}\,\mathbf{x} - \mathbf{b}_{\text{corridor}} \leqslant 0$$

while the nonlinear ones can be written as a nonlinear function of $\mathbf{x}$

$$g_{\text{corridor}}(\mathbf{x}) \leqslant \mathbf{0}$$

Since they are independent of the decision variables, the $\boldsymbol{\Lambda}_i$ and $\boldsymbol{\lambda}_i$ are computed only once for each piece of trajectory (during the initialization phase of the optimization). Furthermore, these constraints can be removed for the control points that are imposed equal to a waypoint, i.e. on *stop* or *lock* waypoints, since the waypoints always lie in their corridors.

This formulation of the corridors constraints, by maintaining the control points inside the flight corridors, provides a sufficient condition for guaranteeing that the trajectory lies inside the corridors. As explained in Mercy et al. (2017), a certain conservatism can be expected from such a formulation. However, this conservatism can be reduced by increasing the number of control points, which allows the curve to stick closer to the control polygon. The counterpart is that, trough a finite, reasonable amount of constraints, the *entire* piece of trajectory is guaranteed to satisfy the constraints. This is a significant advantage over methods such as gridding for instance, for which the satisfaction of the constraints is only guaranteed on some points of the trajectory.

5.3. Sphere waypoints validation

When a waypoint $\mathbf{w}_i$ ($i \in [\![1, N-1]\!]$) is a *sphere* waypoint, the i-th piece of trajectory must end within a sphere of radius $r_{\mathbf{w}_i}$, centered on $\mathbf{w}_i$. This can be ensured by constraining its last control point inside this sphere, since clamped B-spline curves are used. Then, due to the continuity of the trajectory, the next piece starts inside the sphere and validates $\mathbf{w}_i$ as well.

Each *sphere* waypoint validation can be formulated by one nonlinear constraint

$$\left\|\mathbf{p}_{i,n} - \mathbf{w}_i\right\|_2^2 - r_{\mathbf{w}_i}^2 \leqslant 0 \tag{55}$$

which can be rewritten

$$\mathbf{p}_{i,n}{}^{\top}\mathbf{p}_{i,n} - 2\,\mathbf{w}_i{}^{\top}\mathbf{p}_{i,n} + \mathbf{w}_i{}^{\top}\mathbf{w}_i - r_{\mathbf{w}_i}^2 \leqslant 0 \tag{56}$$

All theses quadratic constraints can be regrouped as a nonlinear function of $\mathbf{x}$

$$g_{\text{sphere}}(\mathbf{x}) \leqslant 0 \tag{57}$$

These constraints can also be approximated by a set of linear constraints on $\mathbf{p}_{i,n}$, by replacing the sphere by a convex polyhedron (e.g. a platonic or archimedean solid). Since $\mathbf{p}_{i,n}$ is part of the reduced control points for a *sphere* waypoint and does not need reconstruction, these linear constraints in $\mathbf{p}_{i,n}$ are also linear in $\mathbf{x}$.

5.4. Time derivatives bounds

For each piece of trajectory, bounds on the magnitude of the time derivatives can be imposed. The convex hull property can be used to this aim, as for the corridor constraints. For $l \in [\![1, L+1]\!]$, the l-th time derivative of the trajectory is also a clamped B-spline, of control points given by (18). Therefore, constraining these control points into a sphere ensures that the magnitude of this derivative will never exceed the value of the radius of this sphere

$$\forall (i,j) \in [\![1, N]\!] \times [\![0, n-l]\!] \quad \left\|\mathbf{p}_{i,j}^{(l)}\right\|_2^2 - {d_{i,l}}_{\max}^2 \leqslant 0 \tag{58}$$

with $d_{i,l_{\max}}$ the bound on the l-th time derivative for the i-th piece of trajectory and the control points $\mathbf{p}_{i,j}^{(l)}$ given by (18) and (42). Since equation (18) is nonlinear relatively to the knot steps, the constraint (58) is a nonlinear constraint in $\mathbf{x}$

$$g_{\text{derivatives}}(\mathbf{x}) \leqslant \mathbf{0} \tag{59}$$

5.5. Receding waypoint horizon

The size of the optimization problem (45) is mostly determined by the amount of pieces of trajectory to generate, directly given by the number of waypoints in the flight plan. As this number can be arbitrarily large (the user should be free to use as many waypoints in a flight plan as desired), there is no limit on the size of the optimization problem to solve, which can be an issue in terms of memory usage and computation load. In order to bound the size of the optimization problem (45) without introducing an upper limit on the size of the flight plan, the same receding waypoint horizon as in Rousseau et al. (2018) can be adapted to the trajectory generation algorithm presented in this work. If the flight plan contains more waypoints than a given limit, the trajectory is not computed over the entire flight plan at once but in several steps, over truncated pieces of the overall flight plan.

For a horizon N_H and a flight plan containing at least $N_H + 2$ waypoints, the trajectory is first computed between the first and the $(N_H + 1)$-th waypoints (i.e. $\mathbf{w}_0$ to $\mathbf{w}_{N_H}$). This results in a piecewise trajectory $\widetilde{\boldsymbol{\zeta}}_1$ containing N_H pieces. The first piece of this trajectory corresponds to the trajectory joining $\mathbf{w}_0$ to $\mathbf{w}_1$. As the trajectory does not necessarily pass on the waypoint $\mathbf{w}_1$ (for *sphere* waypoints), the end point of this first piece of trajectory is denoted by κ_1 (see Figure 8) and the l-th derivatives of the trajectory on this point ($l \in [\![1, L]\!]$) are denoted by $\kappa_1^{(l)}$. Only this first piece of trajectory is kept and will constitute the first piece of the overall, final trajectory $\boldsymbol{\zeta}$. According to the receding horizon strategy, the horizon is then shifted by one waypoint and a new trajectory $\widetilde{\boldsymbol{\zeta}}_2$ is computed, having κ_1 as starting waypoint $\mathbf{w}_1$. This new starting waypoint is of special type *constrained*, meaning that the first $L+1$ derivatives (including the position) are imposed, in this case equal to κ_1, $\kappa_1^{(1)}$, ..., $\kappa_1^{(L)}$. Again, only the first piece of this new trajectory $\widetilde{\boldsymbol{\zeta}}_2$ is kept and it constitutes the second piece of the final trajectory $\boldsymbol{\zeta}$. The continuity at the connection between the two trajectory pieces is ensured by the constraints on the position and its derivatives on κ_1. The process is repeated until the last waypoint of the flight plan is reached. In order to ensure the feasibility of the problem, the last waypoint of the horizon is always imposed to be of type *stop*, even though it was not in the initial flight plan. The next step is then initialized with the previous solution, plus a feasible rest-to-rest trajectory for the new piece of trajectory (there always exists one).

Example 1. *A flight plan containing 5 waypoints* $\{\mathbf{w}_0, \ldots, \mathbf{w}_4\}$ *and a horizon* $N_H = 3$ *is illustrated in Figure 8. The trajectory is computed in 2 steps: between the waypoints* $\mathbf{w}_0$ *and* $\mathbf{w}_3$ *first, and between* $\mathbf{w}_1$ *and* $\mathbf{w}_4$ *next.*

Step 1: A trajectory is generated between $\mathbf{w}_0$ *and* $\mathbf{w}_3$*, with* $\mathbf{w}_3$ *replaced by a stop waypoint at the same position.*

Step 2: A trajectory is generated between the waypoints $\mathbf{w}_1$ *and* $\mathbf{w}_4$*, with the waypoint* $\mathbf{w}_3$ *of type sphere, as it was originally, and the waypoint* $\mathbf{w}_1$ *replaced by a constrained waypoint at the position* κ_1*, ensuring the continuity with the trajectory computed at Step 1.*

Since the new trajectory reaches the final waypoint $\mathbf{w}_4$*, there is no need to repeat the process.*

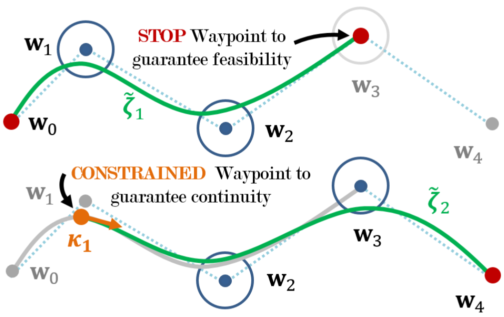

Figure 8: Receding waypoint horizon strategy used for trajectory generation

In order to illustrate the behaviour of the overall trajectory generation strategy, it is applied to the case of aerial cinematography with a quadrotor, both in simulation and on an outdoor flight experiment.

6. Application to a quadrotor

In this section, the strategy described above is applied to perform a cinematographic flight plan (cf. Section 2) with a quadrotor. The goal of this work is to generate a smooth and natural trajectory respecting the speed limitation requested by the user. To this aim, a minimum-time trajectory with speed, acceleration and jerk constraints is generated. The reason for limiting the magnitude of the jerk and the acceleration of the trajectory is twofold. Firstly, they are both a way to quantify the smoothness of the trajectory. Secondly, this allows increasing the feasibility of the trajectory as the acceleration (respectively the jerk) of the drone is strongly linked to its angle relatively to the ground (respectively its rotation speed). Candidates for the admissible bounds on these quantities can be found in Hehn and D'Andrea (2015). From a slow but feasible suboptimal trajectory, the knot steps of each piece of trajectory are reduced and their control points optimized until the constraints on the derivatives are active. It can be noticed that ensuring feasibility through bounds on the derivative can introduce some conservatism in the trajectory as a quadrotor dynamics is more complex than a multiple integrator. However, in the context of video making, smoothness and video quality can be considered more important than having the most time-optimal trajectory. For a more generic solution, not specific to video making, this strategy could be improved by including the dynamics of the quadrotor, in a similar fashion as in Nguyen et al. (2018) (where flatness properties are used jointly with uniform B-splines).

6.1. Tuning parameters for trajectory generation

In order to ensure a continuous angular velocity of the quadrotor as in Boeuf et al. (2014) or Mueller et al. (2015), a C^3 trajectory, continuous up to the jerk is chosen. The degree of the trajectory is chosen as the lowest one allowing a C^3 trajectory, i.e. since the clamped B-splines are C^{k-1} continuous (see Section 3.3), $k = 4$. Finally, in order to use the initialization strategy described below, each piece of trajectory contains $n + 1 = 11$ control points and thus 7 knot steps.

For each piece of trajectory, the speed must not exceed the reference and the norm of the acceleration and of the jerk are respectively limited to $a_{\max}$ and $j_{\max}$. The snap is also limited in order to prevent the optimal trajectory to present short periods of time with an excessively high snap, leading to an almost discontinuous behaviour of the jerk.

6.2. Initial feasible trajectory

As an initial feasible guess for the optimization process, a rest-to-rest trajectory with a bang-off-bang snap profile is used, divided into three phases: acceleration, cruising and braking. Two cases are distinguished: first, the case in which the speed $\widetilde{v}_i$ is reached and there is a cruising phase (60a), and, second, the case where the distance is too short for this (60b). For $i \in [\![1, N]\!]$

If $4\widetilde{v}_i \sqrt[3]{\frac{\widetilde{v}_i}{2s}} < \|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2$

$$\begin{aligned}
\Delta\tau_{\text{acc}} &= \sqrt[3]{\frac{\widetilde{v}_i}{2s}}, \quad \Delta\tau_{\text{cruise}} = \frac{\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2}{\widetilde{v}_i} - 4\Delta\tau_{\text{acc}} \\
\boldsymbol{\Delta\tau}_{\text{init}} &= (\Delta\tau_{\text{acc}}\ 2\Delta\tau_{\text{acc}}\ \Delta\tau_{\text{acc}}\ \Delta\tau_{\text{cruise}}\ \Delta\tau_{\text{acc}}\ 2\Delta\tau_{\text{acc}}\ \Delta\tau_{\text{acc}}) \\
\mathbf{P}^{(4)}_{\text{init}} &= \begin{pmatrix} s\,\mathbf{u}_i & -s\,\mathbf{u}_i & s\,\mathbf{u}_i & \mathbf{0} & -s\,\mathbf{u}_i & s\,\mathbf{u}_i & -s\,\mathbf{u}_i \end{pmatrix}
\end{aligned} \tag{60a}$$

Else

$$\begin{aligned}
\Delta\tau_{\text{acc}} &= \sqrt[4]{\frac{\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2}{8s}} \\
\boldsymbol{\Delta\tau}_{\text{init}} &= \left(\Delta\tau_{\text{acc}}\ 2\Delta\tau_{\text{acc}}\ \frac{1}{2}\Delta\tau_{\text{acc}}\ \frac{1}{2}\Delta\tau_{\text{acc}}\ \Delta\tau_{\text{acc}}\ 2\Delta\tau_{\text{acc}}\ \Delta\tau_{\text{acc}}\right) \\
\mathbf{P}^{(4)}_{\text{init}} &= \begin{pmatrix} s\,\mathbf{u}_i & -s\,\mathbf{u}_i & s\,\mathbf{u}_i & s\,\mathbf{u}_i & -s\,\mathbf{u}_i & s\,\mathbf{u}_i & -s\,\mathbf{u}_i \end{pmatrix}
\end{aligned} \tag{60b}$$

with

$$\widetilde{v}_i = \min\left(v_i, \frac{8\,a_{\max}^2}{9\,j_{\max}}\right), \quad s = \frac{3\,j_{\max}^2}{2\,a_{\max}} \tag{61}$$

and v_i the speed reference of the i-th piece of trajectory. The initial control points for this piece of trajectory can then be deduced using (22). A proof of feasibility of this trajectory, regarding the constraints of the problem (45), is given in the Appendix.

An example of such a trajectory is represented on Figure 9, for which $v_i = \frac{8\,a_{\max}^2}{9\,j_{\max}}$. The blue lines represent the evaluation of the velocity, acceleration, jerk and snap of a rest-to-rest trajectory, projected along the vector $\mathbf{u}_i$ (the trajectory is a straight line of direction $\mathbf{u}_i$), while the red lines represent the bounds.

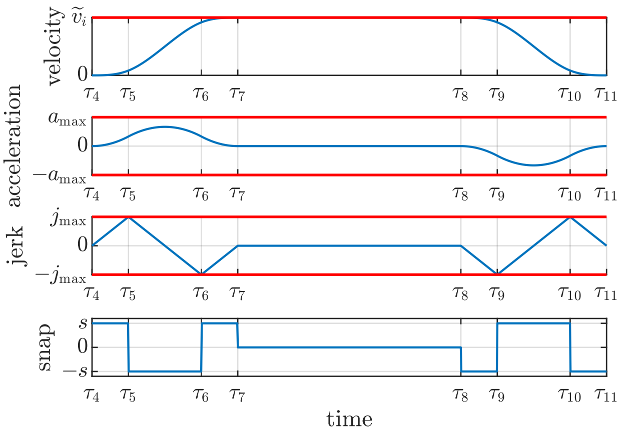

Figure 9: Initial rest-to-rest trajectory profile

6.3. Simulation

The above strategy is applied to a flight plan containing $N = 4$ pieces of trajectory. The first and last waypoints, $\mathbf{w}_0$ and $\mathbf{w}_4$, are *stop* waypoints. The second waypoint $\mathbf{w}_1$ is a *lock* waypoint while the third and fourth ones, $\mathbf{w}_2$ and $\mathbf{w}_3$, are *sphere* waypoints with a radius of 3 m. Each piece of trajectory is contained in a flight corridor with a radius of 3 m. The speed references from the first to the last pieces of trajectory are respectively 2 m.s^{-1}, 10 m.s^{-1}, 3 m.s^{-1} and 7 m.s^{-1}. Finally, the maximum acceleration and jerk are respectively 2 m.s^{-2} and 0.5 m.s^{-3}, and the maximum snap is chosen to be the one used for the initial guess (60). The waypoint horizon is $N_H = 3$, so that the overall trajectory is computed in 2 steps. The time-optimal trajectory and its control points are visible on Figure 10, while the norm of its derivatives on Figure 11.

All the constraints are verified, the trajectory is smooth and natural, while getting as close as possible to the speed references, without exceeding them.

6.4. Comparison with uniform clamped B-splines

Often, *uniform* clamped B-splines are used, which means that the knot steps are all equal. This is not the case in this work, where they are optimized to fully exploit the piecewise polynomial nature of B-splines.

Using uniform B-spline makes the formulation easier and the computation faster, as most of the calculations can be performed once, during the initialization phase of the resolution of

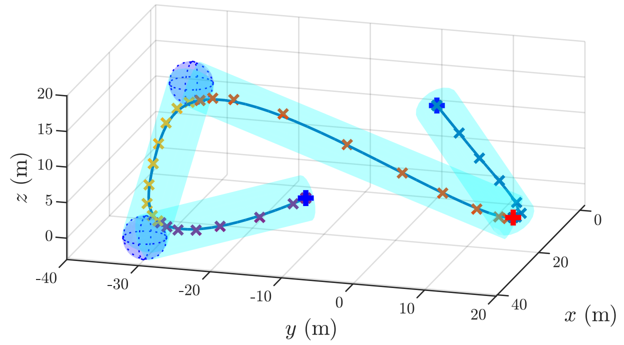

Figure 10: Example of optimal trajectory

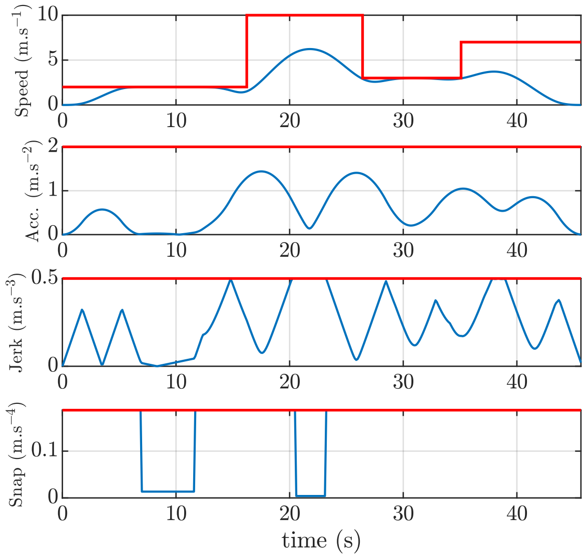

Figure 11: Norm of the derivatives of the trajectory on Figure 10 (blue) vs. their maximum admissible value on each piece of trajectory (red)

the optimization problem, or even be done off-line. This is especially convenient for onboard computation. However, there are only two ways to act on the derivatives of such a spline. They can either be scaled in magnitude by scaling the knot steps (which does not change the shape of the overall curve), or be changed locally by moving the control points (13), which changes the shape of the trajectory. The latter solution is limited in presence of flight corridors.

The limitations of this kind of splines are illustrated on Figure 12. In this example, a trajectory is generated for a simple flight plan of only 2 waypoints separated by 100 m, to join with a reference speed of 1 m.s^{-1}, using the strategy described in this paper with the same parameters as in Section 6.1. In the first case, a clamped B-spline is used (top), while in a second case, an uniform B-spline is used (middle). The speed profile of each trajectory is presented on Figure 12, with each knot marked by a vertical grey line. Notice that since the two considered waypoints are *stop*, only 3 control points are optimized, the other 8 being imposed on the waypoints.

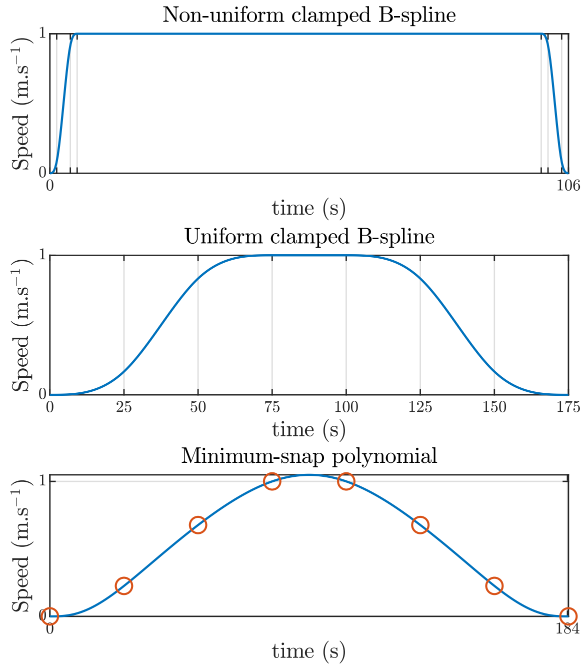

Figure 12: Comparison of a minimum-time clamped B-spline (top), a minimum-time uniform clamped B-spline (middle) and a minimum-snap polynomial (bottom) for the same flight plan; internal knots marked by vertical grey lines

Both solutions verify all the constraints: speed, acceleration, jerk, snap and corridor constraints, though the satisfaction of the corridor constraints is trivial as the trajectory is a straight line in this example. The uniform B-spline trajectory is much slower however, due to long acceleration and braking phases. Indeed, for a B-spline curve of degree k, the k-th derivative is constant between 2 knots, i.e. for a duration equal to a knot step (see Figure 9). For a uniform clamped B-spline, the knot steps being all equal, these durations are directly given by the length of the interval $[\tau_k, \tau_{n+1})$ and by the number of knot steps, $n - k + 1$, fixed by the number of control points and the polynomial degree. The constant knot steps of a uniform B-spline hence prevents the possibility to have arbitrarily long (or short) acceleration, deceleration and cruising phases.

In order to deal with this issue, one could introduce more control points, but considering the wide variety of possible flight plans, it can be delicate to choose an adequate number of control points. In Stoican et al. (2017), control points are added or removed on-the-flight, during the optimization process, which results in a Mixed Integer Problem (MIP) that can significantly impact the computation time.

Freeing the knot steps by using non-uniform clamped B-splines, as it is proposed in this paper, allows the derivatives to be more independent relatively to the control points and the shape of the curve. This is an elegant and robust way to solve the issue, but, as for the MIP, it comes at a certain cost on the computation time.

Finally, Figure 12 also presents the speed profile obtained with a minimum-snap polynomial trajectory, as proposed by Mellinger and Kumar (2011), Richter et al. (2013), for comparison with the algorithm presented in this paper. The overall duration of this minimum-snap trajectory is chosen as the smallest duration guaranteeing the bounds on the speed, acceleration, jerk and snap given above , with a similar algorithm as in Rousseau et al. (2018). Without the convex hull property of B-splines, these bounds on the derivatives are enforced through a *gridding* method, i.e. enforced on a finite set of evaluation points (in red on Figure 12). This does not guarantee the respect of the bounds between these evaluation points and it can be seen on Figure 12 that the speed reference is actually exceeded in this example. The polynomial degree of the minimum-snap trajectory is 11 (more than twice the polynomial degree of the clamped B-spline used in this example) and the result is still quite suboptimal (duration 74% longer than the non-uniform clamped B-spline).

6.5. *Experiment on a Parrot ANAFI quadrotor*

In order to evaluate the cinematographic quality and the feasibility of the trajectories obtained with the method proposed above, a trajectory was generated *off-line* and then performed by a real quadrotor. This test consists in the performance of a cinematographic flight plan containing 6 waypoints. A minimum-time B-spline trajectory is generated to join the waypoints while respecting corridors and bounds on the derivatives. The trajectory is generated off-line, using a waypoint horizon $N_H = 3$, which means that the trajectory is computed in 3 steps. An undersampled linear predictive controller such as described in Rousseau et al. (2018) is used to track the camera angles references smoothly, without introducing too much framing error thanks to the anticipative aspect of the controller and the knowledge of the overall reference trajectory. The flight plan is performed by a Parrot ANAFI quadrotor (see Figure 13, Parrot Drones (2018)), under an average wind of approximately $5.3\,\mathrm{m.s^{-1}}$ (19 km/h), with gusts up to $9.0\,\mathrm{m.s^{-1}}$ (32 km/h).

Figure 13: Parrot ANAFI quadrotor

A video of the outdoor flight is available at `https://youtu.be/A0oYx268sis`. Figure 14 presents the flight plan performed by the quadcopter and the corresponding optimal trajectory, as well as the position tracking error during the flight. The latter is computed using satellite navigation and the internal sensors of the drone.

The trajectory satisfies the specifications of Section 2. Again, the trajectory is smooth and it is difficult to guess the position of the waypoints only by looking at the trajectory or the video, which comforts the idea that the trajectory is natural. The position tracking error remains low for an outdoor flight (< 0.4 m despite the wind gusts), which gives confidence in the feasibility of the trajectory.

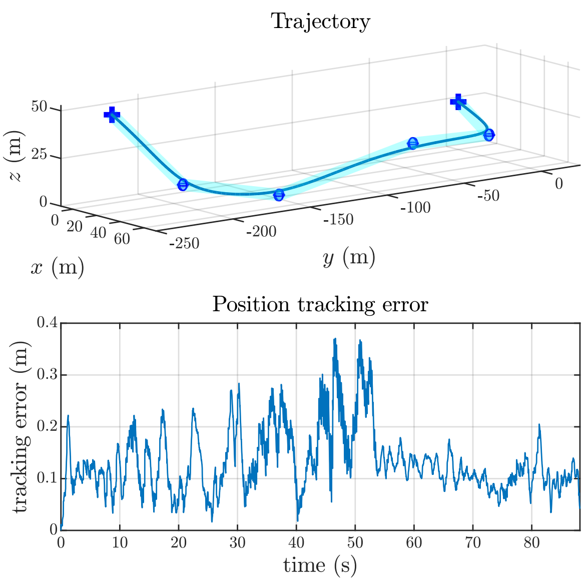

Figure 14: Trajectory for the flight experiment (top) and position tracking error during its performance (bottom)

7. Conclusion

This paper presented a trajectory generation algorithm suited for a wide variety of flight plans, guaranteeing the corridors and the bounds on the derivatives. To this aim, the piecewise nature of clamped B-splines is exploited, by optimizing their knots and control points, in order to obtain a time-optimal trajectory. Confronted to a simulation and an outdoor flight, the method responded to the cinematographic requirements, returning smooth and feasible trajectories, with a suitable speed profile.

A next step will be to implement the trajectory generation algorithm onboard to evaluate its potential use in real-time. To this aim, the implementation of the algorithm used for this paper will be optimized, which will also allow comparing the computation load with other methods. Finally, obstacle avoidance seems like a necessity for autonomous flight. Such a feature could be added to the presented work through the use of local replanning methods such as Usenko et al. (2017).

8. Acknowledgement

This work was supported by Parrot Drones.

9. Declaration of interest

Conflict of interest - none declared.

10. Appendix

This section provides one way to initialize the NLP (45) with a C^3, feasible, clamped B-spline trajectory, for the case studied in Section 6. This initial guess is a rest-to-rest trajectory with a *bang-off-bang* snap profile and is divided into three phases: accelerating, cruising, braking. A feasibility proof of this initial guess is given.

For a piece of trajectory i, joining two waypoints $\mathbf{w}_{i-1}$ and $\mathbf{w}_i$ such that $\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 > 0$, with a speed reference $v_i > 0$, a maximum admissible acceleration $a_{\max} > 0$ and a maximum admissible jerk $j_{\max} > 0$, the two following quantities are first defined

$$\widetilde{v}_i = \min\left(v_i, \frac{8\,a_{\max}^2}{9\,j_{\max}}\right) \tag{62a}$$

$$s = \frac{3\,j_{\max}^2}{2\,a_{\max}} \tag{62b}$$

Case with a cruising phase

It is first considered the case

$$4\,\widetilde{v}_i\sqrt[3]{\frac{\widetilde{v}_i}{2s}} < \|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 \tag{63}$$

for which it can be defined

$$\Delta\tau_{\text{acc}} = \sqrt[3]{\frac{\widetilde{v}_i}{2s}}, \quad \Delta\tau_{\text{cruise}} = \frac{\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2}{\widetilde{v}_i} - 4\Delta\tau_{\text{acc}} \tag{64}$$

The clamped B-spline defined by the following knot steps and control points is chosen as initial guess

$$\boldsymbol{\Delta\tau}_{\text{init}} = (\Delta\tau_{\text{acc}}\ \ 2\Delta\tau_{\text{acc}}\ \ \Delta\tau_{\text{acc}}\ \ \Delta\tau_{\text{cruise}}\ \ \Delta\tau_{\text{acc}}\ \ 2\Delta\tau_{\text{acc}}\ \ \Delta\tau_{\text{acc}}) \tag{65a}$$

$$\mathbf{P}_{\text{init}}^{(4)} = \mathbf{u}_i{}^{\top}(s\quad -s\quad s\quad 0\quad -s\quad s\quad -s) \tag{65b}$$

First, since $v_i > 0$, $a_{\max} > 0$ and $j_{\max} > 0$, it comes that $\Delta\tau_{\text{acc}} > 0$. Furthermore, using the assumption (63) and given that $\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 > 0$, it follows that $\Delta\tau_{\text{cruise}} > 0$.

Using (22), the control points of the jerk are deduced from (65b), with a null jerk as initial condition

$$\mathbf{P}_{\text{init}}^{(3)} = \mathbf{u}_i{}^{\top}\,(0\ \ s\Delta\tau_{\text{acc}}\ \ -s\Delta\tau_{\text{acc}}\ \ 0\ \ 0\ \ -s\Delta\tau_{\text{acc}}\ \ s\Delta\tau_{\text{acc}}\ \ 0) \tag{66}$$

From (62a) it can be deduced that $\widetilde{v}_i \leqslant \frac{8\,a_{\max}^2}{9\,j_{\max}}$, which, injected into (64), leads to

$$\Delta\tau_{\text{acc}} \leqslant \frac{2\,a_{\max}}{3\,j_{\max}} \tag{67}$$

Equations (62b) and (67) lead to $s\Delta\tau_{\text{acc}} \leqslant j_{\max}$. The norm of each control point of $\mathbf{P}_{\text{init}}^{(3)}$ is bounded by $s\Delta\tau_{\text{acc}}$ and thus bounded by $j_{\max}$.

Similarly, the control points of the acceleration are

$$\mathbf{P}_{\text{init}}^{(2)} = \mathbf{u}_i{}^{\top}\left(0\ \ 0\ \ \frac{3}{2}s\,\Delta\tau_{\text{acc}}^2\ \ 0\ \ 0\ \ 0\ \ -\frac{3}{2}s\,\Delta\tau_{\text{acc}}^2\ \ 0\ \ 0\right) \tag{68}$$

Using (62b) and (67), it follows that $\frac{3}{2}s\,\Delta\tau_{\text{acc}}^2 \leqslant a_{\max}$. The norm of each control point of $\mathbf{P}_{\text{init}}^{(2)}$ is thus bounded by $a_{\max}$.

The control points of the velocity are

$$\mathbf{P}_{\text{init}}^{(1)} = \mathbf{u}_i{}^{\top}\left(0\ \ 0\ \ 0\ \ 2s\Delta\tau_{\text{acc}}^3\ \ 2s\Delta\tau_{\text{acc}}^3\ \ 2s\Delta\tau_{\text{acc}}^3\ \ 2s\Delta\tau_{\text{acc}}^3\ \ 0\ \ 0\ \ 0\right) \tag{69}$$

Equation (64) directly gives $\Delta\tau_{\text{acc}}^3 = \frac{\widetilde{v}_i}{2s}$. The bound on the norm of the control points of $\mathbf{P}_{\text{init}}^{(1)}$ is then $\widetilde{v}_i \leqslant v_i$.

Finally, one last integration step with the initial condition $\mathbf{p}_{\text{init}_0} = \mathbf{w}_{i-1}$ gives the control points of the position

$$\mathbf{P}_{\text{init}} = \mathbf{u}_i{}^\top \left(0 \;\; 0 \;\; 0 \;\; 0 \;\; 2\alpha \;\; 4\alpha \;\; 6\alpha \;\; 8\alpha \;\; 8\alpha \;\; 8\alpha \;\; 8\alpha\right) + \mathbf{w}_{i-1} \tag{70}$$

with $\alpha = s\Delta\tau_{\text{acc}}^3\left(\Delta\tau_{\text{acc}} + \frac{1}{4}\Delta\tau_{\text{cruise}}\right)$. Using (64) and $\Delta\tau_{\text{acc}}^3 = \frac{\widetilde{v}_i}{2s}$ leads to $\alpha = \frac{1}{8}\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2$. The last control point of the position is thus $\mathbf{w}_i$.

Case without a cruising phase

If (63) is not satisfied, i.e.

$$4\widetilde{v}_i \sqrt[3]{\frac{\widetilde{v}_i}{2s}} \geqslant \|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2 \tag{71}$$

it can be defined

$$\Delta\tau_{\text{acc}} = \sqrt[4]{\frac{\|\mathbf{w}_i - \mathbf{w}_{i-1}\|_2}{8s}} \tag{72}$$

Injecting (71) into (72) gives the same bound on $\Delta\tau_{\text{acc}}$ as in the case with a cruising phase

$$\Delta\tau_{\text{acc}} \leqslant \sqrt[3]{\frac{\widetilde{v}_i}{2s}} \leqslant \frac{2\,a_{\max}}{3\,j_{\max}} \tag{73}$$

The clamped B-spline defined by the following knot steps and control points is chosen as initial guess

$$\boldsymbol{\Delta\tau}_{\text{init}} = \left(\Delta\tau_{\text{acc}} \;\; 2\Delta\tau_{\text{acc}} \;\; \frac{1}{2}\Delta\tau_{\text{acc}} \;\; \frac{1}{2}\Delta\tau_{\text{acc}} \;\; \Delta\tau_{\text{acc}} \;\; 2\Delta\tau_{\text{acc}} \;\; \Delta\tau_{\text{acc}}\right) \tag{74a}$$

$$\mathbf{P}_{\text{init}}^{(4)} = \mathbf{u}_i{}^\top \left(s \;\; -s \;\; s \;\; s \;\; -s \;\; s \;\; -s\right) \tag{74b}$$

The control points of the jerk are

$$\mathbf{P}_{\text{init}}^{(3)} = \mathbf{u}_i{}^\top \left(0 \;\; s\Delta\tau_{\text{acc}} \;\; -s\Delta\tau_{\text{acc}} \;\; -\frac{1}{2}s\Delta\tau_{\text{acc}} \;\; 0 \;\; -s\Delta\tau_{\text{acc}} \;\; s\Delta\tau_{\text{acc}} \;\; 0\right) \tag{75}$$

Therefore, given (62b) and (73), their norm is bounded by $j_{\max}$.

The control points of the acceleration are

$$\mathbf{P}_{\text{init}}^{(2)} = \mathbf{u}_i{}^\top \left(0 \;\; 0 \;\; \frac{3}{2}s\,\Delta\tau_{\text{acc}}^2 \;\; \frac{1}{4}s\,\Delta\tau_{\text{acc}}^2 \;\; 0 \;\; 0 \;\; -\frac{3}{2}s\,\Delta\tau_{\text{acc}}^2 \;\; 0 \;\; 0\right) \tag{76}$$

Thus, given (62b) and (73), their norm is bounded by $a_{\max}$.

The control points of the velocity are

$$\mathbf{P}_{\text{init}}^{(1)} = \mathbf{u}_i{}^\top \left(0 \;\; 0 \;\; 0 \;\; \frac{7}{4}s\Delta\tau_{\text{acc}}^3 \;\; 2s\Delta\tau_{\text{acc}}^3 \;\; 2s\Delta\tau_{\text{acc}}^3 \;\; 2s\Delta\tau_{\text{acc}}^3 \;\; 0 \;\; 0 \;\; 0\right) \tag{77}$$

Equation (73) gives $\Delta\tau_{\text{acc}}^3 \leqslant \frac{\widetilde{v}_i}{2s}$, hence, the norm of each control points of the velocity is bounded by $\widetilde{v}_i \leqslant v_i$.

Finally, the control points of the position are

$$\mathbf{P}_{\text{init}} = \mathbf{u}_i{}^\top \left(0 \;\; 0 \;\; 0 \;\; 0 \;\; \frac{7}{4}\beta \;\; \frac{15}{4}\beta \;\; \frac{23}{4}\beta \;\; 8\beta \;\; 8\beta \;\; 8\beta \;\; 8\beta\right) + \mathbf{w}_{i-1} \tag{78}$$

with $\beta = s\Delta\tau_{\text{acc}}^4$. Given (72), the last control point of the position is thus $\mathbf{w}_i$.

Feasibility

In both cases the position trajectory contains $n + 1 = 11$ control points and $n - k + 1 = 7$ knot steps. Its polynomial degree is thus $k = 4$, which imply that the position trajectory is C^3.

The fist control point of the position is $\mathbf{w}_{i-1}$ and the last one is $\mathbf{w}_i$. Since clamped B-splines are used, the trajectory then starts on $\mathbf{w}_{i-1}$ and ends on $\mathbf{w}_i$. The trajectory is a straight line, thus the lateral corridor constraints are satisfied. The position control points are contained between $\mathbf{w}_{i-1}$ and $\mathbf{w}_i$ thus the convex hull property ensures that the longitudinal corridor constraints are satisfied too. The bounds on the control points of the time derivatives of the position are respected. Finally, the first and last control point of the velocity, the acceleration and the jerk are $\mathbf{0}$. Laying end to end several of these feasible rest-to-rest trajectories thus leads to an overall trajectory that is C^3.

All the constraints are satisfied, the initial guess is feasible.

References

Abdolhosseini, M., Zhang, Y., Rabbath, C.A., 2013. An efficient model predictive control scheme for an unmanned quadrotor helicopter. Journal of Intelligent & Robotic Systems 70, 27–38.

Bangura, M., Mahony, R., 2014. Real-time model predictive control for quadrotors. IFAC Proceedings Volumes 47, 11773–11780.

Boeuf, A., Cortés, J., Alami, R., Siméon, T., 2014. Planning agile motions for quadrotors in constrained environments, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Chicago, Illinois, USA. pp. 218–223.

Bouffard, P., 2012. On-board model predictive control of a quadrotor helicopter: Design, implementation, and experiments. Technical Report. Department of Electrical Engineering and Computer Sciences, University of California, Berkeley.

De Boor, C., 1978. A practical guide to splines. volume 27 of *Mathematics of Computation*.

Engelhardt, T., Konrad, T., Schäfer, B., Abel, D., 2016. Flatness-based control for a quadrotor camera helicopter using model predictive control trajectory generation, in: Proceedings of the 24th Mediterranean Conference on Control and Automation, Athens, Greece. pp. 852–859.

Fletcher, R., 2013. Practical methods of optimization. John Wiley & Sons.

Fleureau, J., Galvane, Q., Tariolle, F.L., Guillotel, P., 2016. Generic drone control platform for autonomous capture of cinema scenes, in: Proceedings of the 2nd Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use, Singapore. pp. 35–40.

Galvane, Q., Christie, M., Ronfard, R., Lim, C.K., Cani, M.P., 2013. Steering behaviors for autonomous cameras, in: Proceedings of Conference on Motion in Games, Dublin, Ireland. pp. 93–102.

Galvane, Q., Fleureau, J., Tariolle, F.L., Guillotel, P., 2017. Automated cinematography with unmanned aerial vehicles, in: Proceedings of the Eurographics Workshop on Intelligent Cinematography and Editing, Lisbonne, Portugal.

Galvane, Q., Lino, C., Christie, M., Fleureau, J., Servant, F., Tariolle, F., Guillotel, P., et al., 2018. Directing cinematographic drones. ACM Transactions on Graphics 37, 34.

Gebhardt, C., Hepp, B., Nägeli, T., Stevšić, S., Hilliges, O., 2016. Airways: Optimization-based planning of quadrotor trajectories according to high-level user goals, in: Proceedings of the Conference on Human Factors in Computing Systems, San Jose, California, USA. pp. 2508–2519.

Hehn, M., D'Andrea, R., 2015. Real-time trajectory generation for quadrocopters. IEEE Transactions on Robotics 31, 877–892.

Joubert, N., 2017. Tools to facilitate autonomous quadrotor cinematography. Technical Report. Stanford University.

Joubert, N., Jane, L.E., Goldman, D.B., Berthouzoz, F., Roberts, M., Landay, J.A., Hanrahan, P., 2016. Towards a drone cinematographer: Guiding quadrotor cameras using visual composition principles. arXiv preprint arXiv:1610.01691 .

Joubert, N., Roberts, M., Truong, A., Berthouzoz, F., Hanrahan, P., 2015. An interactive tool for designing quadrotor camera shots. ACM Transactions on Graphics 34, 238–238.

Kamel, M., Alexis, K., Achtelik, M., Siegwart, R., 2015. Fast nonlinear model predictive control for multicopter attitude tracking on so (3), in: Proceedings of the IEEE Conference on Control Applications, Sydney, Australia. pp. 1160–1166.

Mellinger, D., Kumar, V., 2011. Minimum snap trajectory generation and control for quadrotors, in: Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China. pp. 2520–2525.

Mercy, T., Van Parys, R., Pipeleers, G., 2017. Spline-based motion planning for autonomous guided vehicles in a dynamic environment. IEEE Transactions on Control Systems Technology 26, 1–8.

Mueller, M.W., Hehn, M., D'Andrea, R., 2015. A computationally efficient motion primitive for quadrocopter trajectory generation. IEEE Transactions on Robotics 31, 1294–1310.

Nägeli, T., Alonso-Mora, J., Domahidi, A., Rus, D., Hilliges, O., 2017a. Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. IEEE Robotics and Automation Letters 2, 1696–1703.

Nägeli, T., Meier, L., Domahidi, A., Alonso-Mora, J., Hilliges, O., 2017b. Real-time planning for automated multi-view drone cinematography. ACM Transactions on Graphics 36, 1–10.

Nguyen, N.T., Prodan, I., Lefèvre, L., 2018. Flat trajectory design and tracking with saturation guarantees: a nano-drone application. International Journal of Control , 1–14.

Parrot Drones, 2018. ANAFI. URL: `https://www.parrot.com/fr/drones/anafi`.

Piegl, L., Tiller, W., 2012. The NURBS book. Springer Science & Business Media.

Richter, C., Bry, A., Roy, N., 2013. Polynomial trajectory planning for quadrotor flight, in: Proceedings of the IEEE International Conference on Robotics and Automation, Karlsruhe, Germany. pp. 1–8.

Richter, C., Bry, A., Roy, N., 2016. Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments, in: Robotics Research: The 16th International Symposium of Robotics Research, pp. 649–666.

Roberts, M., Hanrahan, P., 2016. Generating dynamically feasible trajectories for quadrotor cameras. ACM Transactions on Graphics 35, 1–11.

Rousseau, G., Stoica Maniu, C., Tebbani, S., Babel, M., Martin, N., 2018. Quadcopter-performed cinematographic flight plans using minimum jerk trajectories and predictive camera control, in: Proceedings of the European Control Conference, Limassol, Cyprus. pp. 2897–2903.

Schöllig, A., Hehn, M., Lupashin, S., D'Andrea, R., 2011. Feasiblity of motion primitives for choreographed quadrocopter flight, in: Proceedings of the American Control Conference, San Francisco, California, USA. pp. 3843–3849.

Stoican, F., Prodan, I., Popescu, D., Ichim, L., 2017. Constrained trajectory generation for UAV systems using a B-spline parametrization, in: Proceedings of the 25th Mediterranean Conference on Control and Automation, Valletta, Malta. pp. 613–618.

Usenko, V., von Stumberg, L., Pangercic, A., Cremers, D., 2017. Real-time trajectory replanning for MAVs using uniform B-splines and a 3D circular buffer, in: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Vancouver, Canada. pp. 215–222.

Van Loock, W., Pipeleers, G., Swevers, J., 2013. Time-optimal quadrotor flight, in: Proceedings of the European Control Conference, Zurich, Switzerland. pp. 1788–1792.

Van Loock, W., Pipeleers, G., Swevers, J., 2015. B-spline parameterized optimal motion trajectories for robotic systems with guaranteed constraint satisfaction. Mechanical Sciences 6, 163–171.

Van Parys, R., Pipeleers, G., 2017. Spline-based motion planning in an obstructed 3D environment. IFAC-PapersOnLine 50, 8668–8673.