AIAA SciTech Forum
5-9 January 2015, Kissimmee, Florida
AIAA Guidance, Navigation, and Control Conference

Trajectory Generation using Spatial Pythagorean Hodograph Bézier Curves*

Ronald Choe†, Javier Puig-Navarro‡, Venanzio Cichella§, Enric Xargay¶, Naira Hovakimyan‖
University of Illinois at Urbana-Champaign, Urbana, IL 61801

This paper presents a cooperative trajectory-generation framework that efficiently computes a set of spatial trajectories for a team of cooperating vehicles, and in particular for unmanned aerial vehicles. The desired and feasible trajectories are generated using Pythagorean Hodograph Bézier curves and satisfy stringent spatial and temporal constraints, do not violate the dynamic constraints of the vehicles, and guarantee spatial separation between the paths for safe operation. A simulation scenario is presented to show the efficacy of the proposed cooperative trajectory-generation framework.

I. Introduction

The advent of powerful embedded systems, sensors, and communication networks has drawn widespread interest in the use of unmanned systems (UxSs) to execute missions with limited involvement of human operators. In recent years, UxSs have been playing an increasingly important role in military reconnaissance and strike operations, border patrol missions, aerobiological sampling, forest fire detection, police surveillance, and recovery operations, to name but a few. In simple applications, a single vehicle can be managed by a crew using a ground station provided by the vehicle manufacturer. However, the growing complexity of the envisioned mission scenarios poses several new challenges to the design and integration of UxSs, especially in terms of *autonomy* and *cooperation*.

In their Flight Plan 2009–2047 [1], the USAF anticipates that future operations will require teams of heterogeneous systems working in cooperation to achieve common objectives, while being able to safely operate and execute coordinated tasks in highly uncertain areas. The USAF vision includes all-weather and self-separation operational concepts, and aims at the integration of mixed-capability aircraft into a common airspace, including both manned and unmanned aerial systems. The plan also places special emphasis on the development of *flexible* and *reliable* solutions that can adapt to changing mission objectives, fleet size, available autonomy-capabilities, and stealth requirements. For teams of aerial systems with high levels of autonomy, a paradigm shift is also foreseen in the role of the human operator, which will become less one of direct control of the vehicles and more one of mission management and monitoring. Implementation of this plan will enable the execution of mission profiles that go far beyond today's spectrum of fairly stationary ISR operations, and will develop into cooperating systems that can engage in highly complex operations.

A key enabling element for the successful realization of this vision is the availability of efficient cooperative planning strategies that can be implemented onboard the vehicles. A path-planning algorithm has to work within a complex set of constraints and (near) real-time generation of trajectories is desired to allow the vehicles to re-plan their trajectories, if necessary. More than often, the vehicles carry low power-consumption processors with limited memory in order to save weight for maximum payload capabilities. Therefore, the path-planning algorithm has to be computationally efficient. Several approaches in cooperative path and trajectory planning have been reported in the literature [2–6]. Of special interest is the work that use *pseudospectral optimal control theory*, which has become the modern method of choice for solving trajectory

*Research is supported by NASA.
†Doctoral Student, Dept. of Aerospace Engineering, Student Member AIAA; choe19@illinois.edu.
‡Doctoral Student, Dept. of Aerospace Engineering, Student Member AIAA; puignav2@illinois.edu.
§Doctoral Student, Dept. of Mechanical Science and Engineering, Student Member AIAA; cichell2@illinois.edu.
¶Postdoctoral Research Associate, Dept. of Mechanical Science and Engineering, Member AIAA; xargay@illinois.edu.
‖Professor, Dept. of Mechanical Science and Engineering, Associate Fellow AIAA; nhovakim@illinois.edu.

and maneuver optimization problems; see, for example, [7–15] and references therein. Most of the available literature in this field addresses the optimization of trajectories involving a single vehicle. To the best of our knowledge, only the work in [14] presents a solution to the problem of cooperative trajectory generation for two vehicles that must execute time-critical, collision-free maneuvers. The results reported in the paper demonstrate that pseudospectral optimal control theory can be used to efficiently generate trajectories that maintain a minimal separation between the two cooperating vehicles, while at the same time satisfying their dynamic constraints. We note that, for this particular problem, the authors of [14] impose trajectory deconfliction through *temporal separation*. However, the method seems to be less suitable to generate trajectories that are spatially separated, that is, the minimum distance between any two points on the paths is greater or equal than the minimum spatial clearance that is required.

This paper builds on the work that was presented in [16, 17], where planar feasible trajectories were generated using Pythagorean Hodograph Bézier curves. It was shown that Bézier curves can reduce the computational load, by exploiting their favorable geometric properties. Bézier curves are widely used in computer graphics, animations and type fonts such as postscript fonts and true type fonts. The curves were invented by Dr. Pierre Bézier in the early 1960s to aid the engineers and artists in the shape design of the cars at the Renault company in France. A Bézier curve is completely determined by a set of *control points*. The curve starts at the first and ends at the last control point, while the curve lies completely in the *convex hull* containing the set of control points. This latter property is extremely useful when path planning for aircraft systems is considered, as the airspace in which an aircraft is allowed to maneuver is often allocated and limited by strict boundaries. An example of spatially deconflicted paths generated for a bounded airspace using Bézier curves can be found in [18]. The control points also allow for intuitive (re)design of the shape of the Bézier curves. The authors of [19] present an algorithm where collision avoidance is achieved through proper relocation of the control points of the Bézier curves.

In this paper, we extend the results of [16, 17] to three-dimensional trajectories. We introduce a different approach in satisfying the dynamic constraints, that does not limit the set of feasible curves that satisfy the spatial and temporal specifications. Moreover we also use a different timing law than the one proposed in [16, 17], which allows us to ensure *temporal separation* of the trajectories, in addition to *spatial separation*. Therefore, our approach offers more flexibility in generating collision-free trajectories. An illustrative example is presented where multiple vehicles are executing a time-critical cooperative mission. The desired trajectories are generated a priori and a simulation with simplified kinematic and dynamic models of the vehicles shows that the vehicles are able to converge and closely track the generated trajectories.

The paper is organized as follows: in Section II we describe the general trajectory-generation problem, where we introduce the new approach in formulating the dynamic constraints and the new timing law. Section III gives a brief overview of Pythagorean Hodograph Bézier Curves, Section IV formulates the trajectory-generation framework in terms of these Pythagorean Hodograph Bézier Curves, and Section V presents simulation results illustrating the efficacy and efficiency of the algorithm. Finally, conclusions are drawn in Section VI.

II. General Trajectory Generation Framework

In this section, we formulate the problem of trajectory generation for multiple vehicles in a centralized setting and introduce the *timing law* that decomposes the geometric element from the temporal element of a trajectory. Subsequently, mathematical definitions will be given of the set of constraints that has to be satisfied in order for the desired trajectories to be *feasible*. This set of constraints consists of mission-specific constraints, dynamic constraints of the vehicles, and inter-vehicular safety distance requirements.

A. Problem Formulation

In the 3-dimensional trajectory-generation problem, the objective is to generate N feasible trajectories $\mathbf{p}_{d,i}(t_d)$

$$\mathbf{p}_{d,i} : [0, t^{\mathrm{f}}_{d,i}] \to \mathbb{R}^3 \qquad\qquad i = 1, 2, \ldots, N\,, \tag{1}$$

where N is the number of vehicles, $t^{\mathrm{f}}_{d,i} \in \mathbb{R}^+$ are the individual final mission times of the vehicles obtained during the planning phase, and $t_d \in [0, T_d\,]$, with $T_d := \max\{t^{\mathrm{f}}_{d,1}, \ldots, t^{\mathrm{f}}_{d,N}\}$, is the time variable (used during the trajectory-generation phase and distinct from the actual mission time t that evolves as the mission unfolds). Note that the mission for vehicle i has terminated for all $t_d > t^{\mathrm{f}}_{d,i}$. The feasible trajectories

$\mathbf{p}_{d,i}(t_d)$ together minimize a global cost function $J(\cdot)$ and satisfy boundary conditions, spatial constraints, and temporal constraints. Spatial and temporal constraints can be mission-specific, such as simultaneous arrival of the vehicles, but can also be related to the dynamics of the vehicles, for example adhering to the maximum allowable turn rates of the vehicles. Trajectories that satisfy the dynamic constraints are called *flyable* trajectories. The trajectories are *feasible* when they are also deconflicted in space by satisfying additional spatial or temporal separation constraints. For computational efficiency, these trajectories $\mathbf{p}_{d,i}(\cdot)$ are generally described by real polynomials.

At this point, it is important to distinguish clearly between a (spatial) *path* and a *trajectory*. A spatial path is a curve in space that is parameterized by a (non-temporal) variable defined on an arbitrary interval. A spatial path specifies the locus of the vehicle, without imposing any *temporal* specifications. A trajectory is a spatial path with a designated temporal assignment. A trajectory is thus defined as a function of *time* (or of time variable t_d as in Equation (1)) and, hence, describes the (desired) position of the vehicle at any point in time. It is clear that, besides the spatial path, a trajectory also contains information about the (desired) speed profile $v_{d,i}(\cdot)$ with which the vehicle moves along this path.

Therefore, instead of generating the trajectories explicitly as a function of time, we first decompose the trajectory into a *spatial path*, a geometric element with no temporal specifications, and a *timing law* associated with this path, that captures the temporal assignments of the trajectory. To this purpose, we introduce a dimensionless parameter ζ_i. For convenience, we let $\zeta_i \in [0,1]$. As it will become clear later, this is a natural choice since we will be working with Bézier curves that are defined on the interval $[0,1]$. Then, the following map defines the spatial path $\mathbf{p}_{d,i}(\zeta_i)$:

$$\mathbf{p}_{d,i} : [0,1] \to \mathbb{R}^3 \qquad \zeta_i \in [0,1]\,, \quad i = 1,2,\ldots,N. \tag{2}$$

Next, the timing law dictates how the variable ζ_i for the ith vehicle evolves with the time variable t_d and, as such, affects the desired rate at which the vehicle moves along the path. Hence, the timing law offers a means to meet the temporal requirements of the mission. Let the timing law $\theta_i(\cdot)$ be defined through a dynamic relation of the form

$$\theta_i(\cdot) = \frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d}\,,$$

where $\theta_i(\cdot)$ is a positive function, smooth in its arguments. This function will be defined in the subsequent section; however, it is important to note that the timing law $\theta_i(\cdot)$ will be chosen such that an analytical expression for the function $\zeta_i(t_d)$ exists. This is highly desirable, as the map $\zeta_i(t_d)$ will allow us to relate the time variable t_d to the parameter ζ_i, with which the desired position $\mathbf{p}_{d,i}(\cdot)$ of the ith vehicle at time t_d can be found through the map in Equation (2).

Now, the cooperative trajectory-generation problem can be defined as follows in terms of the 3-dimensional spatial paths $\mathbf{p}_{d,i}(\cdot)$ and the corresponding timing laws $\theta_i(\cdot)$:

Definition 1 (Cooperative Trajectory-Generation Problem) *Find N pairs of*

i) 3-dimensional spatial paths $\mathbf{p}_{d,i}(\zeta_i)$, conveniently parameterized by a dimensionless variable $\zeta_i \in [0,1]$, and

ii) corresponding timing laws $\theta_i(\cdot)$, appropriately defined such that the functions $\zeta_i(t_d)$ can be expressed analytically,

that together minimize a cost function $J(\cdot)$, satisfy desired boundary conditions, do not violate the dynamic constraints of each vehicle, ensure that the vehicles maintain a predefined spatial clearance, and satisfy prespecified mission-specific constraints.

Given the problem formulation above, the trajectory-generation framework can be cast into a constrained optimization problem where a set of desired trajectories are obtained by minimizing the cost function $J(\cdot)$. The optimization problem can be formulated as follows:

$$\begin{aligned} &\min_{\substack{\mathbf{p}_{d,i}\in\mathcal{P}\\ \theta_i\in\Theta\\ i=1,\ldots,N}} \; J(\cdot) \\ &\text{subject to} \quad \begin{array}{l} \text{boundary conditions,} \\ \text{dynamic constraints of the vehicles,} \\ \text{mission-specific constraints,} \\ \text{minimum separation constraints,} \end{array} \end{aligned} \tag{3}$$

where the set $\mathcal{P}$ is the set of degree n polynomial curves, the set Θ denotes the set of degree n_θ polynomial curves, and $J(\cdot)$ is a given cost function and may include terms related to mission-specific goals. The main challenge is to solve this multi-vehicle optimization problem in (near) real-time, with a possibility that the curse of dimensionality phenomena arises as the number of vehicles increases. We present an approach to reduce the computation time of the trajectory-generation algorithm by exploiting the mathematical and geometric properties of a particular class of curves, known in the literature as Pythagorean Hodograph Bézier curves.

In the remainder of this section, we will discuss the timing law and the set of constraints that together define the trajectory-generation framework.

B. Definition of the Timing Law

The timing law allows us to independently adjust the speeds of the vehicles, without altering the spatial paths along which these vehicles travel. In other words, we can assign different speed profiles $v_{d,i}(\cdot)$ to a given spatial path $\mathbf{p}_{d,i}(\zeta_i)$ by changing the parameters of the timing law. This can be easily illustrated by the expression of the speed profile $v_{d,i}(\cdot)$, which is given by:

$$v_{d,i}(\zeta_i, \theta_i(\cdot)) = \left\| \frac{\mathrm{d}\mathbf{p}_{d,i}(\zeta_i)}{\mathrm{d}t_d} \right\|_2 = \left\| \frac{\mathrm{d}\mathbf{p}_{d,i}(\zeta_i)}{\mathrm{d}\zeta_i} \frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d} \right\|_2 = \theta_i(\cdot) \left\| \mathbf{p}'_{d,i}(\zeta_i) \right\|_2 , \qquad \zeta_i \in [0,1], \tag{4}$$

where the first parametric derivative of the spatial path $\mathbf{p}_{d,i}(\zeta_i)$ is defined as $\mathbf{p}'_{d,i}(\zeta_i) = \mathrm{d}\mathbf{p}_{d,i}(\zeta_i)/\mathrm{d}\zeta_i$. The last equality in (4) is obtained by noting that the timing law is a positive function, that is, $\theta_i(\cdot) > 0$. From Equation (4) it is clear that, with a timing law other than $\theta_i(\cdot) \equiv 1$, the speed profile can be chosen independently from the spatial path $\mathbf{p}_{d,i}(\zeta_i)$ so as to meet the temporal specifications. The concept of introducing a timing law in order to adjust the spatial path and the speed profile independently was first described in [20], and later applied in [21] and [22]. In [20] a separate reference function was used for the speed profile, whereas in [22] the speed profile was obtained by integrating the acceleration equation using a predetermined thrust history.

In our approach we seek to find a suitable structure and parametrization for the timing law, such that the temporal constraints are met and an analytical expression for the function $\zeta_i(t_d)$ exists. Taking into consideration that the spatial paths $\mathbf{p}_{d,i}(\zeta_i)$ are described by polynomials for computational efficiency, from Equation (4) it can be seen that the timing laws $\theta_i(\cdot)$ have to be polynomial functions as well in order to maintain the polynomial structure for the speed profiles. However, there are two feasible approaches to parametrize the timing laws.

First, let the timing law $\theta_i(\cdot)$ be a smooth positive polynomial function of the parameter ζ_i, that is

$$\theta_i(\zeta_i) = \frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d} , \qquad \zeta_i \in [0,1] . \tag{5}$$

The functions $t_{d,i}(\zeta_i)$ that map the parameter ζ_i to the time variable t_d can be found through integration of (5):

$$t_{d,i}(\zeta_i) = \int_0^{\zeta_i} \frac{1}{\theta_i(\tau)} \mathrm{d}\tau. \tag{6}$$

In general, the timing laws $\theta_i(\zeta_i)$ are distinct and, hence, the time *functions* $t_{d,i}(\zeta_i)$ are different for each vehicle as well. However, the time *variable* t_d is a common and available parameter to *all* vehicles. Therefore, it is desirable to derive an analytical expression for the map $\zeta_i(t_d)$, since the spatial paths are parameterized by the variable ζ_i. It can be shown that, with a proper choice of the timing law, an analytical solution exists for the integral in Equation (6). From the definition of $\theta_i(\zeta_i)$, it follows that $t_{d,i}(\zeta_i)$ is a strictly monotonically increasing function, and thus its inverse function $t_{d,i}^{-1} = \zeta_i(t_d)$ exists. Another advantage is that the temporal constraints are explicitly parametrized by ζ_i. For example, the desired speed profile, given by Equation (4), will be a function of the parameter ζ_i only.

A preliminary *planar* trajectory-generation framework is presented in [16, 17], where the timing law θ_i was chosen as a second-degree polynomial in ζ_i, so that Equation (6) allowed for a solution in closed-form *and* the inverse function $\zeta_i(t_d)$ could be derived analytically as well. Following this approach, in combination with Bézier curves to describe the paths $\mathbf{p}_{d,i}(\zeta_i)$ and the timing laws $\theta_i(\zeta_i)$, the results in [16, 17] indicate

that the developed framework is able to generate efficiently a set of collision-free trajectories that satisfy all boundary conditions and dynamic constraints of the vehicles. However, there are two major drawbacks in this particular choice of structure for the timing law:

1. While it is true that the time integral (6) admits a closed-form solution $t_{d,i}(\zeta_i)$ for a polynomial function $\theta_i(\zeta_i)$ of any degree $n_\theta \in \mathbb{N}$, for $n_\theta > 2$ it becomes extremely intractable or even impossible to obtain an analytical expression for the inverse function $\zeta_i(t_d)$. Therefore, the only practically feasible choice for the timing law is a second-degree polynomial. Unfortunately, this leaves the framework with no room for increasing n_θ in scenarios where more free variables are necessary in order to satisfy additional (temporal) constraints.

2. In the case where $n_\theta = 2$, the solutions to the time integral (6) are either trigonometric or hyperbolic functions [16, 17] and, hence, the inverse functions $\zeta_i(t_d)$ are not polynomials. This poses a problem when spatial deconfliction has to be ensured through *temporal* separation, that is the minimum distance between two points on the paths of the ith and jth vehicle, at time t_d, is greater or equal than the minimum spatial clearance E:

$$\min_{\substack{i,j=1,\dots,N \\ i\neq j}} \|\mathbf{p}_{d,i}(\zeta_i(t_d)) - \mathbf{p}_{d,j}(\zeta_j(t_d))\|_2^2 \geq E^2\,, \quad \forall t_d \in [0, t^{\mathrm{f}}_{d_{ij}}],$$

where $t^{\mathrm{f}}_{d_{ij}} := \min\{t^{\mathrm{f}}_{d,i}, t^{\mathrm{f}}_{d,j}\}$. Recalling the fact that the spatial paths $\mathbf{p}_{d,i}(\zeta_i)$ are polynomials for computational efficiency, the above equation becomes impractical to evaluate if the functions $\zeta_i(t_d)$ are not of polynomial form as well.

To overcome these two drawbacks, the timing law $\theta_i(\cdot)$ will be defined differently in the following. Let the timing law $\theta_i(\cdot)$ be a smooth positive polynomial function of the parameter t_d, that is

$$\theta_i(t_d) = \frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d}\,, \qquad \text{for} \quad t_d \in [0, t^{\mathrm{f}}_{d,i}]\,. \tag{7}$$

With the timing law $\theta_i(t_d)$ defined as in Equation (7), the map $\zeta_i(t_d)$ is given by the integral

$$\zeta_i(t_d) = \int_0^{t_d} \theta_i(\tau)\mathrm{d}\tau\,, \tag{8}$$

where, from the definition of ζ_i, the functions $\zeta_i(t_d)$ must satisfy $\zeta_i(0) = 0$ and $\zeta_i(t^{\mathrm{f}}_{d,i}) = 1$. If the timing laws $\theta_i(t_d)$ are polynomials of degree n_θ, then integrating Equation (8) will yield functions $\zeta_i(t_d)$ that are polynomials of degree $(n_\theta + 1)$. It is clear that, in this case, we are not limited to $n_\theta = 2$ in order to express the functions $\zeta_i(t_d)$ analytically and, hence, we can choose n_θ according to the set of constraints that needs to be satisfied. Moreover, the temporal separation constraint given above reduces to a polynomial since the composite functions $(\mathbf{p}_{d,i} \circ \zeta_i)(t_d)$ and $(\mathbf{p}_{d,j} \circ \zeta_j)(t_d)$ are polynomials of the same degree. Nevertheless, the temporal separation constraint results in a high-degree polynomial and determining its roots is not straightforward. We shall see that using Bézier curves to describe the spatial paths and the timing laws significantly simplifies constructing the composite functions and reduces the computational load while evaluating the temporal separation constraint. A similar observation can be made for the temporal constraints, which are equations in two variables, namely in t_d (through the timing law $\theta_i(t_d)$) and ζ_i. We can convert the temporal constraints to (high-degree) polynomials of one single variable t_d by substituting Equation (8).

Considering the favorable properties of describing the timing law by Equation (7) instead of Equation (5), the second approach will be adopted in the subsequent development of the 3-dimensional trajectory-generation framework, in contrary to the work described in [16, 17]. Nonetheless, the results reported in [16, 17] are valid and as such the planar trajectory-generation framework presented therein serves as a feasibility study and baseline for the spatial framework described hereafter.

C. Boundary Conditions and Constraints

In the problem of trajectory generation for autonomous vehicles, we typically deal with missions for which the initial and final conditions on the trajectory, i.e. the boundary conditions, are prespecified. Therefore, in

the following development of the trajectory-generation framework, it is assumed that the positions, speeds, flight-path and heading angles of each individual vehicle at the endpoints are prescribed, that is

$$\begin{aligned} \mathbf{p}_{d,i}(\zeta_i = 0) &= \mathbf{p}_i^{\mathrm{i}}\,, \quad \gamma_i(\zeta_i = 0) = \gamma_i^{\mathrm{i}}\,, \quad \psi_i(\zeta_i = 0) = \psi_i^{\mathrm{i}}\,, \qquad v_{d,i}(t_d = 0) = v_i^{\mathrm{i}}\,, \\ \mathbf{p}_{d,i}(\zeta_i = 1) &= \mathbf{p}_i^{\mathrm{f}}\,, \quad \gamma_i(\zeta_i = 1) = \gamma_i^{\mathrm{f}}\,, \quad \psi_i(\zeta_i = 1) = \psi_i^{\mathrm{f}}\,, \quad v_{d,i}(t_d = t_{d,i}^{\mathrm{f}}) = v_i^{\mathrm{f}}\,, \end{aligned} \tag{9}$$

for $i = 1, \ldots, N$. Here, $\mathbf{p}_i^{\mathrm{i}}$, γ_i^{i}, ψ_i^{i}, and v_i^{i} are the initial position, flight-path angle, heading angle, and speed, respectively, while $\mathbf{p}_i^{\mathrm{f}}$, γ_i^{f}, ψ_i^{f}, and v_i^{f} are the specified quantities at the final endpoint of the trajectory. Equation (9) gives the boundary conditions that need to be satisfied by the generated trajectory for the ith vehicle. In fact, constructing a smooth curve with given endpoints and derivatives, such as the boundary conditions given above, is defined as the first-order Hermite interpolation problem [23].

Additional constraints that a set of generated trajectories has to satisfy can be mission-specific, for example simultaneous arrival at predefined destinations or specified lengths of each individual path. When simultaneous time-of-arrival of the vehicles is required, and the absolute time-of-arrival is not specified, then an additional constraint is

$$t_{d,i}^{\mathrm{f}} - t_{d,j}^{\mathrm{f}} = 0 \qquad \text{for} \quad i \neq j, \quad i, j = 1, 2, \ldots, N. \tag{10}$$

Another mission-specific requirement could be, for example, ensuring continuity of the turn-rate $\dot{\psi}$ at the final point of the trajectory, if the vehicle has to smoothly transition from the path onto a circular orbit.

As mentioned before, our objective is not solely to satisfy the mission requirements, but, equally importantly, to generate trajectories that the vehicles are able to follow and, at the same time, are collision-free. Therefore, the trajectories have to satisfy additional constraints, namely a set of dynamic constraints, and also requirements regarding the minimum spatial clearance between the paths or the vehicles.

1. *Flyable trajectories: dynamic constraints of the vehicles*

Flyable trajectories are trajectories that satisfy the dynamic constraints of the vehicles and, hence, trajectories that the vehicles should be able to follow closely if they are equipped with autopilots that enable accurate tracking of the control commands. Trajectories that are not flyable will inevitably result in path-following errors and jeopardize the successful completion of the mission or, in the worst case, lead to a loss of the vehicle. In the case of fixed-wing aircraft, violating the stall speed $v_{\min}$ is a good example of the latter.

In [16,17,24,25] the rotational dynamic constraints are given in terms of intrinsic geometrical properties. For planar curves, the *curvature* κ is bounded in order not to violate the maximum turn rate $\dot{\psi}_{\max}$ of the vehicle. This approach works well for planar trajectories, since there is a direct relation between the maximum turn rate $\dot{\psi}_{\max}$ and the maximum allowable curvature $\kappa_{\max}$:

$$\kappa_{\max} = \frac{1}{R_{\min}} = \frac{\dot{\psi}_{\max}}{v_{\max}}\,,$$

where $R_{\min}$ is the minimum turn radius, and $v_{\max}$ is the maximum speed of the vehicle. An observation that one can make is that bounding the curvature is in fact a conservative approach, since a turn at $\dot{\psi}_{\max}$ with a turn radius smaller than $R_{\min}$ is perfectly acceptable, if the speed $v < v_{\max}$. Nevertheless, the results in [16,17] for planar trajectories justify meeting the dynamic constraints by means of bounding the maximum curvature of the path.

The authors of [24, 25] use a similar notion to extend their framework to 3-dimensional trajectories, by introducing and bounding the *torsion* τ, in addition to the bound on the curvature. While similar arguments hold regarding the conservatism, for spatial curves there is, however, an added complexity to this approach: it is not straightforward to translate dynamic bounds on the yaw, pitch, and roll dynamics of the vehicle into limits on curvature κ and torsion τ. For example, the yaw motion is no longer associated only with the curvature, but determines both the curvature and torsion of the spatial path.

However, the main drawback of using geometrical properties to satisfy the dynamic constraints becomes evident from the definitions of both the curvature and the torsion of a curve $\mathbf{r}(\zeta)$:

$$\kappa(\zeta) = \frac{\|\mathbf{r}'(\zeta) \times \mathbf{r}''(\zeta)\|_2}{\|\mathbf{r}'(\zeta)\|_2^3}\,, \qquad \tau(\zeta) = \frac{(\mathbf{r}'(\zeta) \times \mathbf{r}''(\zeta)) \cdot \mathbf{r}'''(\zeta)}{\|\mathbf{r}'(\zeta) \times \mathbf{r}''(\zeta)\|_2^2}\,,$$

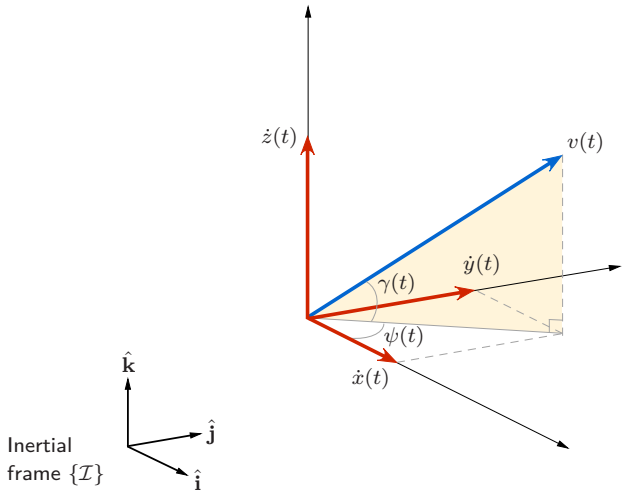


Figure 1: Definition of the flight-path angle $\gamma(t)$ and the heading angle $\psi(t)$.

where $\mathbf{r}'(\zeta)$, $\mathbf{r}''(\zeta)$, and $\mathbf{r}'''(\zeta)$ are the first, second, and third derivatives of $\mathbf{r}(\zeta)$ with respect to ζ, respectively. It can be observed that the torsion τ is not well-behaved and, in fact, has a singularity for curves where the curvature κ is (momentarily) zero. These include curves exhibiting inflection points, but also straight paths (or segments). Therefore, bounding the torsion τ is very restrictive and highly undesirable, since that excludes, among others, the simplest family of curves, namely that of straight lines.

In our framework, we proceed from the kinematics of the vehicles and derive bounds that are directly related to the dynamics and do not suffer from the shortcomings of the geometrical approach. First, let the inertial reference frame $\{\mathcal{I}\}$ be characterized by the orthonormal vectors $\{\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}\}$ (see Figure 1). The unit vectors $\hat{\mathbf{i}}$ and $\hat{\mathbf{j}}$ lie in the horizontal plane, while the unit vector $\hat{\mathbf{k}}$ points up vertically in the opposite direction of the gravity vector. Then, the (translational) equations of a vehicle, governing the kinematics in this inertial reference frame $\{\mathcal{I}\}$, are given by:

$$\begin{aligned} \dot{x} &= v \cos\gamma \cos\psi \,, \\ \dot{y} &= v \cos\gamma \sin\psi \,, \\ \dot{z} &= v \sin\gamma \,, \end{aligned}$$

subject to

$$\begin{aligned} &v_{\min} \leq v(t) \leq v_{\max} \,, \quad |a(t)| \leq a_{\max} \,, \\ &\gamma_{\min} \leq \gamma(t) \leq \gamma_{\max} \,, \quad |\dot{\gamma}(t)| \leq \dot{\gamma}_{\max} \,, \quad |\dot{\psi}(t)| \leq \dot{\psi}_{\max} \,, \qquad \forall t \geq 0 \,, \end{aligned} \tag{11}$$

where $[x(t), y(t), z(t)]^\top$, $v(t)$, and $a(t)$ are the coordinates, speed, and acceleration, respectively, of the UAV given in the inertial reference frame $\{\mathcal{I}\}$, $\gamma(t)$ denotes the flight-path angle, and $\psi(t)$ represents the heading angle of the vehicle (see Figure 1). The bounds in Equation (11) are vehicle-specific and can be derived from the dynamics of the vehicles. The conditions in (11) prescribe the physical limitations of the vehicle and, therefore, the generated desired trajectories shall not demand maneuvering along the path that exceeds these dynamic constraints. Hence, we need to derive the expressions for the desired speed and acceleration profiles as well as for the flight-path angle, rate of change in the flight-path angle, and the turn rate along the generated trajectory of the ith vehicle, in terms of the spatial paths $\mathbf{p}_{d,i}(\zeta_i)$ and the timing laws $\theta_i(t_d)$. The flight-path angle γ_i is a pure *geometric* property of the spatial path and, hence, it is natural to have it parameterized by ζ_i:

$$\gamma_i(\zeta_i) = \arcsin\left(\frac{\mathbf{p}'_{d,i}(\zeta_i) \cdot \hat{\mathbf{k}}}{\left\| \mathbf{p}'_{d,i}(\zeta_i) \right\|_2} \right) . \tag{12}$$

The other dynamic properties are of *temporal* nature and, therefore, we characterized them by the time

variable t_d:

$$v_{d,i}(t_d) = \left\|\mathbf{p}'_{d,i}(\zeta_i(t_d))\right\|_2 \theta_i(t_d)\,, \tag{13a}$$

$$a_{d,i}(t_d) = \left\|\mathbf{p}'_{d,i}(\zeta_i(t_d))\dot{\theta}_i(t_d) + \mathbf{p}''_{d,i}(\zeta_i(t_d))\theta_i^2(t_d)\right\|_2\,, \tag{13b}$$

$$\dot{\gamma}_i(t_d) = \frac{\left\|\mathbf{p}'_{d,i}(\zeta_i(t_d))\right\|_2 \left(\mathbf{p}''_{d,i}(\zeta_i(t_d))\cdot\hat{\mathbf{k}}\right) - \sigma'_i(\zeta_i(t_d))\left(\mathbf{p}'_{d,i}(\zeta_i(t_d))\cdot\hat{\mathbf{k}}\right)}{\left\|\mathbf{p}'_{d,i}(\zeta_i(t_d))\right\|_2 \left(\left(\mathbf{p}'_{d,i}(\zeta_i(t_d))\cdot\hat{\mathbf{i}}\right)^2 + \left(\mathbf{p}'_{d,i}(\zeta_i(t_d))\cdot\hat{\mathbf{j}}\right)^2\right)^{\frac{1}{2}}}\theta_i(t_d)\,, \tag{13c}$$

$$\dot{\psi}_i(t_d) = \frac{\left(\mathbf{p}'_{d,i}(\zeta_i(t_d)) \times \mathbf{p}''_{d,i}(\zeta_i(t_d))\right)\cdot\hat{\mathbf{k}}}{\left(\mathbf{p}'_{d,i}(\zeta_i(t_d))\cdot\hat{\mathbf{i}}\right)^2 + \left(\mathbf{p}'_{d,i}(\zeta_i(t_d))\cdot\hat{\mathbf{j}}\right)^2}\theta_i(t_d)\,, \tag{13d}$$

where

$$\sigma'_i(\zeta_i) = \frac{\mathrm{d}}{\mathrm{d}\zeta_i}\left\|\mathbf{p}'_{d,i}(\zeta_i)\right\|_2 = \frac{\left(\mathbf{p}'_{d,i}(\zeta_i)\cdot\mathbf{p}''_{d,i}(\zeta_i)\right)}{\left\|\mathbf{p}'_{d,i}(\zeta_i)\right\|_2}\,,$$

and $\mathbf{p}'_{d,i}(\zeta_i)$ and $\mathbf{p}''_{d,i}(\zeta_i)$ are the first and second derivatives of $\mathbf{p}_{d,i}(\zeta_i)$ with respect to ζ_i, respectively. The time derivatives of $\theta_i(t_d)$, $\gamma_i(t_d)$, and $\psi_i(t_d)$ in Equation (13) are taken with respect to t_d. Also notice, that the functions in Equation (13) are written as functions of a single parameter t_d, by substituting Equation (8) for the variable ζ_i. A trajectory is said to be *flyable* if the equations in (12) and (13) meet similar bounds as in Equation (11), which implies that the vehicle is able to follow the trajectory without exceeding its dynamic constraints, such as the minimum (stall) and maximum speeds, and the maximum turn rate. Hence, Equations (12) and (13) have to satisfy

$$\begin{aligned} v_{d,\min} \le v_{d,i}(t_d) \le v_{d,\max}\,, \quad |a_{d,i}(t_d)| \le a_{d,\max}\,, \\ \gamma_{\min} \le \gamma_i(\zeta_i) \le \gamma_{\max}\,, \quad |\dot{\gamma}_i(t_d)| \le \dot{\gamma}_{\max}\,, \quad |\dot{\psi}_i(t_d)| \le \dot{\psi}_{\max}\,, \qquad \forall\,(\zeta_i, t_d) \in [0,1]\times[0, t^{\mathrm{f}}_{d,i}]\,, \end{aligned} \tag{14}$$

where $0 < v_{\min} \le v_{d,\min} < v_{d,\max} \le v_{\max}$ and $a_{d,\max} \le a_{\max}$, for the trajectories $\mathbf{p}_{d,i}(\zeta_i(t_d))$ to be flyable. Needless to say that the expressions in Equations (12) and (13) can be very complex and result in high-degree polynomials and, because of this, evaluating Equation (14) is not trivial and computationally expensive.

Remark 1 *Note that, in general, $v_{d,i}(t_d)$ is the total commanded speed to the autopilot and $v_{d,min} = v_{min}$, $v_{d,max} = v_{max}$, and $a_{d,max} = a_{max}$. However, for certain specific missions, such as time-coordinated missions [3, 26–30] or, in particular circumstances, for example in the presence of uncooperative vehicles [17], the speeds of vehicles are adjusted from $v_{d,i}(t_d)$ during the execution of the mission, as to achieve coordination, or to avert an imminent collision, respectively. In these cases, $v_{d,i}(t_d)$ is no longer the total commanded speed and, hence, the more conservative bounds given in Equation (14) are adopted during the trajectory-generation phase, in order to prevent premature saturation of the total speed command during mission execution.*

2. *Feasible trajectories: spatial and temporal separation*

Trajectories of different vehicles have to be spatially deconflicted a priori in order to avoid collision and ensure safe simultaneous operation in the airspace. Trajectories that are flyable and safe to fly are called *feasible* trajectories. Deconfliction between trajectories can be guaranteed through *spatial* separation or *temporal* separation. Spatial separation is guaranteed if the minimum distance between any two points on the paths of the ith and jth vehicle is greater or equal than the minimum spatial clearance E, i.e.

$$\min_{\substack{i,j=1,\ldots,N \\ i\neq j}} \left\|\mathbf{p}_{d,i}(\zeta_i) - \mathbf{p}_{d,j}(\zeta_j)\right\|_2^2 \ge E^2\,, \qquad \forall \zeta_i, \zeta_j \in [0,1]. \tag{15}$$

On the other hand, temporal separation is ensured if the minimum distance between two points on the paths of the ith and jth vehicle, at a time t_d, is greater or equal than the minimum spatial clearance E, that is

$$\min_{\substack{i,j=1,\ldots,N \\ i\neq j}} \left\|\mathbf{p}_{d,i}(\zeta_i(t_d)) - \mathbf{p}_{d,j}(\zeta_j(t_d))\right\|_2^2 \ge E^2\,, \quad \forall t_d \in [0, t^{\mathrm{f}}_{d_{ij}}], \tag{16}$$

where $t^{\mathrm{f}}_{d_{ij}} := \min\{t^{\mathrm{f}}_{d,i}, t^{\mathrm{f}}_{d,j}\}$. In (16) the paths are allowed to intersect, but the vehicles are time separated and, hence, do not arrive at the intersection point simultaneously. The generated trajectories are spatially deconflicted and form a set of feasible trajectories if and only if either Equation (15) or (16) is satisfied.

Spatial and temporal separation ensure collision-free trajectories at all times, and the selection of E is based on the performance of an underlying path-following controller onboard the vehicles. In addition, spatial deconfliction through temporal separation is also dependent on the performance of a necessary coordination controller, and on the quality and robustness of the communications network over which the vehicles exchange information with each other. Since the trajectories are allowed to intersect, a collision may potentially occur when the communications network is faulty or jammed. In contrary, spatial separation results in a more conservative set of trajectories that requires a larger (air)space, but guarantees minimum spatial clearance at all times, even if the communications network is temporarily, or even permanently, unavailable.

The type of separation that is ultimately required depends on a variety of factors and considerations, such as the type of mission, the environment, the quality of the communications network, security requirements on data transmission, etc, and has to be carefully decided on by the mission planner. Nevertheless, the developed framework offers the mission planner the flexibility to generate trajectories that are either spatially or temporally deconflicted.

III. Pythagorean Hodograph Bézier Curves

The cooperative trajectory-generation framework was formulated and set up mathematically in Section II. The necessary operations involved include differentiation, integration, and root finding of, generally, high-degree polynomials and rational functions of polynomials, which may be computationally expensive to perform. Additionally, multiplication and composition of two polynomial functions may not be trivial and, more than often, the resulting high-degree polynomial exhibits undesirable behavior.

Hence, trajectory planning is not a trivial issue. A trajectory-planning algorithm has to work within a complex set of constraints (Section II.C), and (near) real-time generation of trajectories is desired to allow the UAVs to re-plan their trajectories onboard, in cases where for example communication is lost between the ground control station and the UAV. Often, the UAVs carry low power-consumption processors with limited memory in order to save weight to maximize payload capacity. Therefore, the trajectory-planning algorithm has to be computationally efficient. One approach to relieve the computational load is to seek a class of curves with geometric properties that can be exploited in satisfying the set of imposed constraints as described in Section II.C. One such family of curves, possessing attractive geometric properties, is formed by the *Bézier curves*. A Bézier curve is completely determined by a set of *control points*. Since Bézier curves are polynomials, the set of Bézier curves is closed under the operations of addition/subtraction, multiplication, and scaling, and is also closed under differentiation and integration. Computationally efficient algorithms are available to perform these operations on the control points [23, 31].

Within the class of polynomial curves, there exists a sub-class of curves of which the components of the hodographs satisfy a Pythagorean quartuple of polynomials. These are called *Pythagorean Hodograph* (PH) curves. The hodograph of a curve is the locus described by the first parametric derivative of that curve [32,33]. PH curves have the attractive property that the speed is a polynomial function of its parameter and, therefore, admit an exact measurement of the arc length by simply evaluating a polynomial. This, of course, reduces the computational load considerably compared to obtaining the arc lengths of the curves through numerical integration.

In this section we introduce the *Pythagorean Hodograph Bézier curves* that will be used to mathematically describe the trajectories. These curves are Bézier curves of which the hodographs satisfy the Pythagorean polynomial quartuple. However, in this section we give a treatment of these curves individually: first, we give an overview of the properties and construction of PH curves, followed by a similar discussion for Bézier curves. In the next section we will describe in more detail how PH Bézier curves can be constructed using a *quaternion representation*.

A. Pythagorean Hodograph Curves

One of the questions that comes to one's mind when dealing with curves, and especially with trajectories, is what is the exact arc length of the curve? It seems like an innocent question. However, from a differential geometer's point of view, the answer is not trivial at all. Let $s(\zeta)$ be the arc length of a differentiable spatial

curve $\mathbf{r}(\zeta) = [x(\zeta), y(\zeta), z(\zeta)]^\top$, given by:

$$s(\zeta) = \int_0^\zeta \|\mathbf{r}'(\tau)\|_2 \, d\tau = \int_0^\zeta \sqrt{x'^2(\tau) + y'^2(\tau) + z'^2(\tau)} \, d\tau \,, \tag{17}$$

where $\mathbf{r}'(\zeta) = \mathrm{d}\mathbf{r}(\zeta)/\mathrm{d}\zeta = [x'(\zeta), y'(\zeta), z'(\zeta)]^\top$ denotes the hodograph of $\mathbf{r}(\zeta)$ and thus $\|\mathbf{r}'(\tau)\|_2$ represents the *parametric speed* of $\mathbf{r}(\zeta)$. Note that the parametric speed $\|\mathbf{r}'(\zeta)\|_2$ is *not* the desired speed profile $v_{d,i}(t_d)$ from Equation (13a). In general, Equation (17) does not allow a solution in closed-form since the argument of the square root in the integrand is a polynomial function and, hence, the integral has to be approximated using numerical methods.

In the ideal case, an analytical expression for Equation (17) can be found if $\|\mathbf{r}'(\zeta)\|_2 \equiv 1$, i.e. the "unit speed" parametrization and, therefore, the curve is conveniently parameterized by its arc length since $s(\zeta) = \zeta$. Unfortunately, as shown by Theorem 16.1 in [23], such a parametrization is not possible for any planar (and spatial) curve, other than for straight lines. However, even if a unit speed parametrization does not exist, there is still an elegant way to simplify Equation (17) significantly. For that, we need to incorporate *a priori* a Pythagorean structure in the hodograph of $\mathbf{r}(\zeta)$. To this end, let

$$x'^2(\zeta) + y'^2(\zeta) + z'^2(\zeta) = \sigma^2(\zeta) \,, \tag{18}$$

for some polynomial $\sigma(\zeta)$. In other words, we ensure that $x'^2(\zeta) + y'^2(\zeta) + z'^2(\zeta)$ is a perfect square of the polynomial $\sigma(\zeta)$ (see Figure 2). Curves with hodographs satisfying Equation (18) are called *Pythagorean Hodograph* curves [23]. Substituting Equation (18) into (17) results in

$$s(\zeta) = \int_0^\zeta \sigma(\tau) d\tau,$$

and the arc length $s(\zeta)$ can be found as a closed-form function through a simple integration of polynomial $\sigma(\zeta)$ of any degree. Theorem 21.1 in [23] states that the Pythagorean condition (18) is satisfied if and only if $x'(\zeta)$, $y'(\zeta)$, $z'(\zeta)$, and $\sigma(\zeta)$ can be expressed in terms of other real polynomials $u(\zeta)$, $v(\zeta)$, $p(\zeta)$ and $q(\zeta)$ in the form

$$x'(\zeta) = u^2(\zeta) + v^2(\zeta) - p^2(\zeta) - q^2(\zeta) \,, \tag{19a}$$
$$y'(\zeta) = 2\left[u(\zeta)q(\zeta) + v(\zeta)p(\zeta)\right] \,, \tag{19b}$$
$$z'(\zeta) = 2\left[v(\zeta)q(\zeta) - u(\zeta)p(\zeta)\right] \,, \tag{19c}$$
$$\sigma(\zeta) = u^2(\zeta) + v^2(\zeta) + p^2(\zeta) + q^2(\zeta) \,. \tag{19d}$$

Although Equation (19) is a sufficient and necessary condition, constructing PH spatial curves using Equations (19a)-(19d) in this form results in a system of nine coupled quadratic equations for twelve real unknowns. A more compact system is described in [23,34–36], based on the quaternion representation of Equation (19) that was first introduced in [37].

A quaternion $\mathcal{A} = a + a_x\hat{\mathbf{i}} + a_y\hat{\mathbf{j}} + a_z\hat{\mathbf{k}}$ comprises of a scalar part a and a vector part $\mathbf{a} = a_x\hat{\mathbf{i}} + a_y\hat{\mathbf{j}} + a_z\hat{\mathbf{k}}$, and its conjugate $\mathcal{A}^*$ is defined as $\mathcal{A}^* = a - a_x\hat{\mathbf{i}} - a_y\hat{\mathbf{j}} - a_z\hat{\mathbf{k}}$. We can also write the quaternion $\mathcal{A}$ as $\mathcal{A} = |\mathcal{A}|\mathcal{U}$, where $|\mathcal{A}|^2 = \mathcal{A}\mathcal{A}^* = a^2 + |\mathbf{a}|^2$ is the square of the magnitude of the quaternion $\mathcal{A}$, and $\mathcal{U}$ is the *unit* quaternion with $|\mathcal{U}| = 1$, defined as $\mathcal{U} = \cos\frac{1}{2}\delta + \sin\frac{1}{2}\delta\,\mathbf{n}$ for some angle δ and unit vector $\mathbf{n}$. Then, given a pure vector quaternion $\mathbf{v}$ and a unit quaternion $\mathcal{U}$, the quaternion product $\mathcal{U}\,\mathbf{v}\,\mathcal{U}^*$, represents a rotation of $\mathbf{v}$ through an angle δ about the unit vector $\mathbf{n}$, and results in a pure vector quaternion.

The first-order Hermite interpolation problem, concerned with the construction of a smooth curve matching given endpoints and derivatives, often yields equations of the following form when formulated in the quaternion representation:

$$\mathcal{A}\,\hat{\mathbf{i}}\,\mathcal{A}^* = \mathbf{c} \,, \tag{20}$$

with $\mathbf{c} = c_x\hat{\mathbf{i}} + c_y\hat{\mathbf{j}} + c_z\hat{\mathbf{k}}$. The *one-parameter family of general solutions* to (20) is given by:

$$\mathcal{A}(\phi) = \sqrt{\frac{1}{2}(1+\lambda)|\mathbf{c}|}\left(-\sin\phi + \cos\phi\,\hat{\mathbf{i}} + \frac{\mu\cos\phi + \nu\sin\phi}{1+\lambda}\hat{\mathbf{j}} + \frac{\nu\cos\phi - \mu\sin\phi}{1+\lambda}\hat{\mathbf{k}}\right) \,, \tag{21}$$

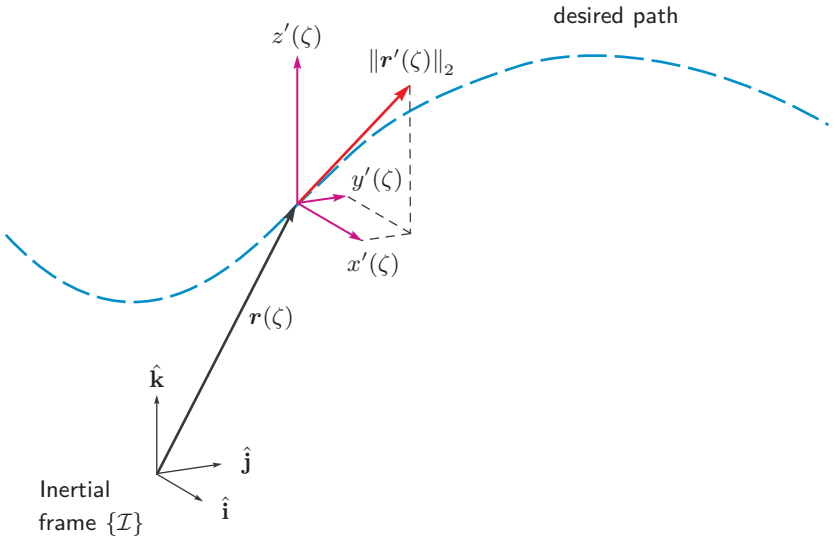


Figure 2: PH curves have the property that the parametric speed $\|\boldsymbol{r}'(\zeta)\|_2 = \sigma(\zeta)$ is a polynomial function.

where (λ, μ, ν) and $|\mathbf{c}| \in \mathbb{R}^+$ are respectively the direction-cosines and the magnitude of the vector $\mathbf{c}$. Next, consider a quaternion *polynomial*

$$\mathcal{A}(\zeta) = u(\zeta) + v(\zeta)\hat{\mathbf{i}} + p(\zeta)\hat{\mathbf{j}} + q(\zeta)\hat{\mathbf{k}}\,. \tag{22}$$

Then the following product $\mathcal{A}(\zeta)\,\hat{\mathbf{i}}\,\mathcal{A}^*(\zeta)$ results in:

$$\mathcal{A}(\zeta)\,\hat{\mathbf{i}}\,\mathcal{A}^*(\zeta) = \left[u^2(\zeta) + v^2(\zeta) - p^2(\zeta) - q^2(\zeta)\right]\hat{\mathbf{i}} + 2\left[u(\zeta)q(\zeta) + v(\zeta)p(\zeta)\right]\hat{\mathbf{j}} + 2\left[v(\zeta)q(\zeta) - u(\zeta)p(\zeta)\right]\hat{\mathbf{k}}\,. \tag{23}$$

Comparing this result with Equation (19), we observe that the hodograph $\mathbf{r}'(\zeta) = \mathcal{A}(\zeta)\,\hat{\mathbf{i}}\,\mathcal{A}^*(\zeta)$ satisfies the conditions to be the hodograph of a PH spatial curve $\mathbf{r}(\zeta)$. Rewriting the spatial hodograph $\mathbf{r}'(\zeta)$ in terms of the unit quaternion $\mathcal{U}(\zeta)$ yields $\mathbf{r}'(\zeta) = |\mathcal{A}(\zeta)|^2\,\mathcal{U}(\zeta)\,\hat{\mathbf{i}}\,\mathcal{U}^*(\zeta)$. Hence, we can interpret Equation (23) geometrically as generating a spatial hodograph through a continuous family of spatial rotations and scalings of the basis vector $\hat{\mathbf{i}}$.

With the above formulation of spatial PH curves in terms of quaternions, and using Bézier curves to describe the polynomials $u(\zeta)$, $v(\zeta)$, $p(\zeta)$, and $q(\zeta)$, we shall see in the next section that the construction of *Pythagorean Hodograph Bézier curves* simplifies considerably both in number of equations and unknowns. PH Bézier curves also provide a closed-form solution for the energy integral, and exact representations of offset curves at a distance d from the curve for all ζ as rational Bézier curves. The former property allows the designer to obtain the optimal shape of the curves by minimizing the energy, while the latter results in offset curves, that are rational Bézier curves themselves and, hence, inherit the nice geometric properties of the family of Bézier curves.

The efforts in finding an exact measurement of the arc length is not only to satisfy the differential geometer's curiosity. In the context of time-critical cooperative control of multiple vehicles, more than often, simultaneous time-of-arrival of the vehicles is required. This is practically only feasible if the intersection of the *arrival-time windows* of the vehicles is non-empty, where the arrival-time window for the ith vehicle is defined as:

$$\delta t^{\mathrm{f}}_{d,i} := \left[t^{\mathrm{f}}_{d_{\min},i}, t^{\mathrm{f}}_{d_{\max},i}\right]\,,$$

$$t^{\mathrm{f}}_{d_{\min},i} := \frac{s_i}{v_{\max}}\,, \qquad t^{\mathrm{f}}_{d_{\max},i} := \frac{s_i}{v_{\min}}\,,$$

where s_i is the total arc length of the spatial path. Clearly, in order to determine the arrival-time windows, one needs to compute the exact total arc length of the curve. In [27] the *arrival margin* is defined from

arrival-time windows, and it is argued that this arrival margin plays a key role in providing robustness to the mission operation at the coordination level. It is obvious that this class of curves also offers an appealing solution in cases where the total arc lengths of the paths have to be minimized.

B. Bézier Curves

Next, an overview is given of Bézier curves that are used to mathematically describe the trajectories, i.e. the spatial paths $\mathbf{p}_{d,i}(\zeta_i)$ and the timing laws $\theta_i(t_d)$. Bézier curves are polynomials over the finite interval $[0,1]$ expressed in the *Bernstein* basis, consisting of Bernstein polynomials:

$$b_k^n(\zeta) := \binom{n}{k}(1-\zeta)^{n-k}\zeta^k\,, \qquad \zeta \in [0,1]\,, \tag{24}$$

where n is the degree of the polynomial.

In this work, we explore the suitability and feasibility of using Bézier curves, and in particular PH Bézier curves, to describe the flight trajectories mathematically. Expressing the desired flight paths in terms of Bézier polynomials is attractive, due to numerous favorable geometric properties that these curves exhibit. A degree n Bézier curve is given by:

$$\mathbf{r}(\zeta) = \sum_{k=0}^{n} \bar{\mathbf{r}}_k\, b_k^n(\zeta)\,, \qquad \zeta \in [0,1]\,, \tag{25}$$

where ζ is a dimensionless parameter, the points $\bar{\mathbf{r}}_0, \ldots, \bar{\mathbf{r}}_n \in \mathbb{R}^3$ are the *control points* of the Bézier curve, and $b_k^n(\zeta)$ are the Bernstein polynomials given in (24). The set of Bézier polynomials is closed under the arithmetic operations of addition, subtraction, and multiplication, and under differentiation, integration, and composition. Besides being the coefficients that completely determine the Bézier polynomial, the control points are also key elements that define the attractive geometric properties of Bézier curves. One important property that is exploited extensively is the fact that a Bézier curve lies completely in the convex hull of these control points $\bar{\mathbf{r}}_0, \ldots, \bar{\mathbf{r}}_n$. Hence, the convex hull is a polyhedron (or a polygon for planar curves) with a subset of the control points as its vertices. Figure 3 shows an example of a *quintic* planar Bézier curve. The curve here lies completely in the green dash-dotted polygon. Additionally, a Bézier curve starts at the first control point $\bar{\mathbf{r}}_0$ and ends at the last control point $\bar{\mathbf{r}}_n$, and it lies tangent to the *control polygon* (not to be confused with the *convex hull*) at these two end control points. Many operations performed on Bézier curves are computationally efficient, such as arithmetic operations, differentiation, and integration, which in general reduce to recursive algorithms in terms of the control points. Comprehensive lists of algorithms are given in [23, 31].

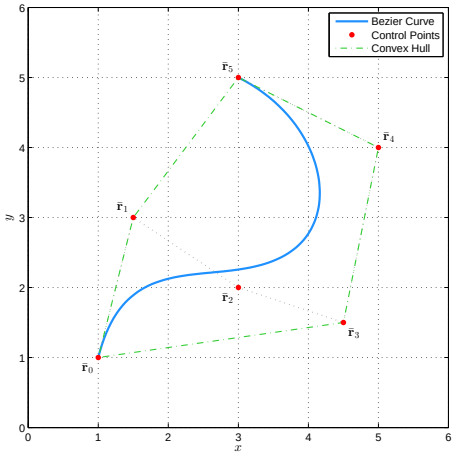


Figure 3: Planar quintic Bézier curve.

One computational procedure that is often associated with Bézier curves is the *de Casteljau* algorithm. For a degree n Bézier curve and given $\zeta^0 \in (0,1)$, this recursive algorithm computes the point $\mathbf{r}(\zeta^0)$ on the

curve that subdivides the curve into two subsegments, namely, one that is defined on $\zeta \in [0, \zeta^0]$ and the other one on $\zeta \in [\zeta^0, 1]$, and provides the control points of each of the two (sub)Bézier curves that are also of degree n. Moreover, the control points of the two newly formed Bézier curves converge towards the curve.

The *de Casteljau* algorithm is fundamental to many computational procedures applied to Bézier curves, such as determining the minimum distance between two Bézier curves [38, 39]. The authors in [38] propose an iterative algorithm to compute the minimum distance between two Bézier curves, where the convex hull property and the *de Casteljau* algorithm are cleverly and extensively exploited in combination with the Gilbert-Johnson-Keerthi (GJK) distance algorithm [40–43]. The latter is an algorithm that is widely used in the computer graphics and animation world, to compute the minimum distance between two *convex shapes*. This GJK algorithm works flawlessly for Bézier curves, since these are always contained in their *convex hulls*. The *de Casteljau* algorithm continuously subdivides the curves, as to eventually converge to the points of minimum distance between the curves. The complete algorithm works extremely efficiently and computes the minimum distance between the two Bézier curves to within a desired tolerance and, therefore, is not an approximation method.

In the course of the work, the minimum-distance algorithm is modified to efficiently compute the extremes of scalar Bézier polynomials. Recall that the constraints in (14) require computing the extremes of the kinematic expressions given in Equations (12) and (13). In Section IV.C, these expressions will be rewritten in terms of Bézier curves. Hence, the modified minimum distance algorithm allows us to efficiently compute the 'exact' value of the extremes of these dynamic expressions.

Lastly, besides (polynomial) Bézier curves that are described by Equation (25), there are also *rational* Bézier curves. A degree n rational Bézier curve is given by

$$\mathbf{r}(\zeta) = \frac{\sum_{k=0}^{n} w_k \bar{\mathbf{r}}_k \, b_k^n(\zeta)}{\sum_{k=0}^{n} w_k \, b_k^n(\zeta)}, \qquad \zeta \in [0, 1],$$

where $\bar{\mathbf{r}}_k \in \mathbb{R}^3$ are the control points, $b_k^n(\zeta)$ are the Bernstein polynomials, and $w_k \in \mathbb{R}$ are the weights of the rational Bézier curve. These weights provide an extra degree-of-freedom for modifying the shape of the curve. In fact, rational Bézier curves are the only way to describe conic sections, other than the parabola. Many properties of (polynomial) Bézier curves carry over to rational Bézier curves. Likewise, computational procedures designed for Bézier curves can be extended to rational Bézier curves, such as the minimum distance and extreme-finding algorithm. This is extremely helpful since, as we shall see in Section IV.C, many of the expressions in Equations (12) and (13) can be expressed as rational Bézier curves.

Remark 2 *The convex hull property, on which practically many algorithms and properties hinge, also carries over to rational Bézier curves* if *all the weights w_k are positive. Hence, the weights have to be chosen carefully when possible; otherwise, this condition has to be verified and satisfied first, before proceeding with algorithms that rely on the convex hull property.*

IV. Cooperative Trajectory Generation using PH Bézier Curves

The framework for trajectory generation that is presented hereafter combines the favorable geometric properties of Bézier curves with the Pythagorean property of PH curves. The resulting PH Bézier curves retain their computational efficiency while also allowing closed-form functions for the arc length $s_i(\zeta_i)$. Bézier curves were used in [18] to generate paths for multiple UAVs in a bounded airspace, while PH Bézier curves were generated for simultaneous arrival of multiple UAVs in [25]. In the latter work, the trajectories were generated such that the paths were of equal lengths while the desired speeds were constant and equal for all UAVs. As discussed earlier, in our trajectory-generation framework, the arc lengths and the speed profiles are not restricted to such constraints and hence offer greater flexibility. Quintic PH Bézier curves are most commonly used to generate paths in several works on trajectory generation [25, 44, 45]. In fact, quintic PH Bézier curves are the simplest PH Bézier curves that allow first-order Hermite interpolation, i.e. constructing a smooth curve with given endpoints and derivatives. The properties and behavior of the quintic PH Bézier curves are also extensively studied in [23] and [32, 33]. Hence, in our approach we seek to generate *quintic* PH Bézier curves to describe the spatial paths and *quadratic* Bézier polynomials to represent

the timing laws, that is

$$\mathbf{p}_{d,i}(\zeta_i) = \sum_{k=0}^{5} \bar{\mathbf{p}}_{i,k}\, b_k^5(\zeta_i)\,,$$
$$\theta_i(t_d) = \sum_{k=0}^{2} \bar{\theta}_{i,k}\, b_k^2(t_d)\,, \qquad \text{for} \quad i = 1,\ldots,N\,, \quad (\zeta_i, t_d) \in [0,1] \times [0, t_{d,i}^{\mathrm{f}}]\,,$$

where $\bar{\mathbf{p}}_{i,k} \in \mathbb{R}^3$ and $\bar{\theta}_{i,k} \in \mathbb{R}$ are the control points of the spatial path $\mathbf{p}_{d,i}(\zeta_i)$ and the timing law $\theta_i(t_d)$, respectively. Increasing the degree of the spatial PH Bézier curves will offer more flexibility in satisfying the constraints, but will potentially result in an increased computational cost. Also, PH Bézier curves of higher degree may exhibit unknown undesirable behavior and characteristics, since they have not been extensively explored yet. The motivation behind the choice of quadratic Bézier polynomials for the timing laws is elaborated in Section IV.B.

A. Quaternion Representation of Spatial PH Bézier Curves

In our approach we seek to generate spatial quintic PH Bézier curves to describe the paths $\mathbf{p}_{d,i}(\zeta_i)$ mathematically. Recall that Equation (23) represents the hodograph of the spatial PH path

$$\mathbf{p}'_{d,i}(\zeta_i) = \mathcal{A}_i(\zeta_i)\,\hat{\mathbf{i}}\,\mathcal{A}_i^*(\zeta_i) \tag{26}$$

and, hence, the hodograph $\mathbf{p}'_{d,i}(\zeta_i)$ must be of degree 4. To this end, we let $\mathcal{A}_i(\zeta_i)$ be a quadratic Bézier (quaternion) polynomial:

$$\mathcal{A}_i(\zeta_i) = \sum_{k=0}^{2} \bar{\mathcal{A}}_{i,k} b_k^2(\zeta_i)\,,$$

where the control points $\bar{\mathcal{A}}_{i,k}$ are quaternions and defined as

$$\bar{\mathcal{A}}_{i,k} = \bar{u}_{i,k} + \bar{v}_{i,k}\hat{\mathbf{i}} + \bar{p}_{i,k}\hat{\mathbf{j}} + \bar{q}_{i,k}\hat{\mathbf{k}}\,, \qquad \text{for} \quad k = 0,1,2\,.$$

With the above definitions, the quaternion polynomial of Equation (22) is recovered, where now $u_i(\zeta_i)$, $v_i(\zeta_i)$, $p_i(\zeta_i)$, and $q_i(\zeta_i)$ are quadratic Bézier polynomials:

$$u_i(\zeta_i) = \sum_{k=0}^{2} \bar{u}_{i,k} b_k^2(\zeta_i)\,, \quad v_i(\zeta_i) = \sum_{k=0}^{2} \bar{v}_{i,k} b_k^2(\zeta_i)\,, \quad p_i(\zeta_i) = \sum_{k=0}^{2} \bar{p}_{i,k} b_k^2(\zeta_i)\,, \quad q_i(\zeta_i) = \sum_{k=0}^{2} \bar{q}_{i,k} b_k^2(\zeta_i)\,.$$

The control points of the spatial path $\mathbf{p}_{d,i}(\zeta_i)$, in the form $\bar{\mathbf{p}}_{i,k} = \bar{\mathrm{p}}_{x_{i,k}}\hat{\mathbf{i}} + \bar{\mathrm{p}}_{y_{i,k}}\hat{\mathbf{j}} + \bar{\mathrm{p}}_{z_{i,k}}\hat{\mathbf{k}}$, can be obtained by integrating Equation (26):

$$\begin{aligned}
\bar{\mathbf{p}}_{i,1} &= \bar{\mathbf{p}}_{i,0} + \frac{1}{5}\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^*\,,\\
\bar{\mathbf{p}}_{i,2} &= \bar{\mathbf{p}}_{i,1} + \frac{1}{10}\left(\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,1}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^*\right)\,,\\
\bar{\mathbf{p}}_{i,3} &= \bar{\mathbf{p}}_{i,2} + \frac{1}{30}\left(\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^* + 4\bar{\mathcal{A}}_{i,1}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^*\right)\,,\\
\bar{\mathbf{p}}_{i,4} &= \bar{\mathbf{p}}_{i,3} + \frac{1}{10}\left(\bar{\mathcal{A}}_{i,1}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^* + \bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,1}^*\right)\,,\\
\bar{\mathbf{p}}_{i,5} &= \bar{\mathbf{p}}_{i,4} + \frac{1}{5}\bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^*\,,
\end{aligned} \tag{27}$$

where $\bar{\mathbf{p}}_{i,0} = \mathbf{p}_i^{\mathrm{i}}$ and $\bar{\mathbf{p}}_{i,5} = \mathbf{p}_i^{\mathrm{f}}$. It is clear that the quintic PH Bézier curve $\mathbf{p}_{d,i}(\zeta_i)$ is completely determined by the three unknowns $\bar{\mathcal{A}}_{i,k}$ for $k = 0,1,2$. The first-order Hermite interpolation conditions at the endpoints provide three equations that can be used to solve for the three unknowns $\bar{\mathcal{A}}_{i,0}$, $\bar{\mathcal{A}}_{i,1}$, and $\bar{\mathcal{A}}_{i,2}$. First, let the derivatives at the initial and final point be given by the vectors $\mathbf{d}_i^{\mathrm{i}}$ and $\mathbf{d}_i^{\mathrm{f}}$, respectively. Then the

direction-cosines $(\lambda_i, \mu_i, \nu_i)$ can be expressed in terms of the flight-path angle γ_i and heading angle ψ_i at the endpoints:

$$\begin{aligned} \lambda_i^{\mathrm{i}} &= \cos\gamma_i^{\mathrm{i}} \cos\psi_i^{\mathrm{i}}\,, & \lambda_i^{\mathrm{f}} &= \cos\gamma_i^{\mathrm{f}} \cos\psi_i^{\mathrm{f}}\,, \\ \mu_i^{\mathrm{i}} &= \cos\gamma_i^{\mathrm{i}} \sin\psi_i^{\mathrm{i}}\,, & \mu_i^{\mathrm{f}} &= \cos\gamma_i^{\mathrm{f}} \sin\psi_i^{\mathrm{f}}\,, \\ \nu_i^{\mathrm{i}} &= \sin\gamma_i^{\mathrm{i}}\,, & \nu_i^{\mathrm{f}} &= \sin\gamma_i^{\mathrm{f}}\,. \end{aligned}$$

The magnitude of the vectors $\mathbf{d}_i^{\mathrm{i}}$ and $\mathbf{d}_i^{\mathrm{f}}$ are denoted by $|\mathbf{d}_i^{\mathrm{i}}| \in \mathbb{R}^+$, and $|\mathbf{d}_i^{\mathrm{f}}| \in \mathbb{R}^+$, respectively. Note, that these are *not* equal to the given initial and final speed v_i^{i}, and v_i^{f}, due to the decoupling through the timing law $\theta_i(t_d)$. With these definitions, interpolation of the end-derivatives yields the following two equations:

$$\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^{*} = \mathbf{d}_i^{\mathrm{i}} \qquad \text{and} \qquad \bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^{*} = \mathbf{d}_i^{\mathrm{f}}\,.$$

These two equations are in the form of Equation (20) and, hence, the solutions for $\bar{\mathcal{A}}_{i,0}$ and $\bar{\mathcal{A}}_{i,2}$ are given by Equation (21):

$$\begin{aligned} \bar{\mathcal{A}}_{i,0}(|\mathbf{d}_i^{\mathrm{i}}|, \phi_{i,0}) =& \sqrt{\frac{1}{2}(1+\lambda_i^{\mathrm{i}})|\mathbf{d}_i^{\mathrm{i}}|} \\ & \times \left(-\sin\phi_{i,0} + \cos\phi_{i,0}\,\hat{\mathbf{i}} + \frac{\mu_i^{\mathrm{i}}\cos\phi_{i,0} + \nu_i^{\mathrm{i}}\sin\phi_{i,0}}{1+\lambda_i^{\mathrm{i}}}\hat{\mathbf{j}} + \frac{\nu_i^{\mathrm{i}}\cos\phi_{i,0} - \mu_i^{\mathrm{i}}\sin\phi_{i,0}}{1+\lambda_i^{\mathrm{i}}}\hat{\mathbf{k}} \right), \end{aligned} \tag{28}$$

$$\begin{aligned} \bar{\mathcal{A}}_{i,2}(|\mathbf{d}_i^{\mathrm{f}}|, \phi_{i,2}) =& \sqrt{\frac{1}{2}(1+\lambda_i^{\mathrm{f}})|\mathbf{d}_i^{\mathrm{f}}|} \\ & \times \left(-\sin\phi_{i,2} + \cos\phi_{i,2}\,\hat{\mathbf{i}} + \frac{\mu_i^{\mathrm{f}}\cos\phi_{i,2} + \nu_i^{\mathrm{f}}\sin\phi_{i,2}}{1+\lambda_i^{\mathrm{f}}}\hat{\mathbf{j}} + \frac{\nu_i^{\mathrm{f}}\cos\phi_{i,2} - \mu_i^{\mathrm{f}}\sin\phi_{i,2}}{1+\lambda_i^{\mathrm{f}}}\hat{\mathbf{k}} \right), \end{aligned} \tag{29}$$

where $\phi_{i,0}, \phi_{i,2} \in [-\frac{1}{2}\pi, \frac{1}{2}\pi]$ are free angular parameters. In contrary to the general solution given by Equation (21), which has only angle ϕ as a single free parameter, $\bar{\mathcal{A}}_{i,0}$ and $\bar{\mathcal{A}}_{i,2}$ are also dependent on $|\mathbf{d}_i^{\mathrm{i}}|$, and $|\mathbf{d}_i^{\mathrm{f}}|$. This extra degree-of-freedom is a result of the fact that we introduced the timing law in order to capture the temporal specifications, such as the specified initial and final speeds, separately.

The last unknown $\bar{\mathcal{A}}_{i,1}$ can be determined from the endpoint conditions:

$$\begin{aligned} \int_0^1 \mathcal{A}_i(\zeta_i)\,\hat{\mathbf{i}}\,\mathcal{A}_i^{*}(\zeta_i)\mathrm{d}\zeta_i =& \quad \mathbf{p}_i^{\mathrm{f}} - \mathbf{p}_i^{\mathrm{i}} \\ =& \quad \frac{1}{5}\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^{*} + \frac{1}{10}\left(\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,1}^{*} + \bar{\mathcal{A}}_{i,1}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^{*}\right) \\ & + \frac{1}{30}\left(\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^{*} + 4\bar{\mathcal{A}}_{i,1}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,1}^{*} + \bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^{*}\right) \\ & + \frac{1}{10}\left(\bar{\mathcal{A}}_{i,1}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^{*} + \bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,1}^{*}\right) + \frac{1}{5}\bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^{*}\,. \end{aligned} \tag{30}$$

Equation (30) can be rewritten as

$$\begin{aligned} & \left(3\bar{\mathcal{A}}_{i,0} + 4\bar{\mathcal{A}}_{i,1} + 3\bar{\mathcal{A}}_{i,2}\right)\,\hat{\mathbf{i}}\,\left(3\bar{\mathcal{A}}_{i,0} + 4\bar{\mathcal{A}}_{i,1} + 3\bar{\mathcal{A}}_{i,2}\right)^{*} \\ & \quad = 120\left(\mathbf{p}_i^{\mathrm{f}} - \mathbf{p}_i^{\mathrm{i}}\right) - 15\left(\mathbf{d}_i^{\mathrm{f}} + \mathbf{d}_i^{\mathrm{i}}\right) + 5\left(\bar{\mathcal{A}}_{i,0}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,2}^{*} + \bar{\mathcal{A}}_{i,2}\,\hat{\mathbf{i}}\,\bar{\mathcal{A}}_{i,0}^{*}\right) \\ & \quad = \mathbf{c}\,, \end{aligned} \tag{31}$$

where $\mathbf{c} = c_x\hat{\mathbf{i}} + c_y\hat{\mathbf{j}} + c_z\hat{\mathbf{k}}$, since the right hand side is a pure vector. Again, the solution to Equation (31) is given by the general solution (21), and using Equations (28) and (29), $\bar{\mathcal{A}}_{i,1}$ can be found as

$$\begin{aligned} \bar{\mathcal{A}}_{i,1}(\phi_{i,1}) =& -\frac{3}{4}\left(\bar{\mathcal{A}}_{i,0} + \bar{\mathcal{A}}_{i,2}\right) + \frac{\sqrt{\frac{1}{2}(1+\lambda_i^{\mathrm{c}})|\mathbf{c}|}}{4} \\ & \times \left(-\sin\phi_{i,1} + \cos\phi_{i,1}\,\hat{\mathbf{i}} + \frac{\mu_i^{\mathrm{c}}\cos\phi_{i,1} + \nu_i^{\mathrm{c}}\sin\phi_{i,1}}{1+\lambda_i^{\mathrm{c}}}\hat{\mathbf{j}} + \frac{\nu_i^{\mathrm{c}}\cos\phi_{i,1} - \mu_i^{\mathrm{c}}\sin\phi_{i,1}}{1+\lambda_i^{\mathrm{c}}}\hat{\mathbf{k}} \right), \end{aligned} \tag{32}$$

where $(\lambda_i^{\mathrm{c}}, \mu_i^{\mathrm{c}}, \nu_i^{\mathrm{c}})$ and $|\mathbf{c}| \in \mathbb{R}^+$ are respectively the direction-cosines and the magnitude of the vector $\mathbf{c}$, and $\phi_{i,1} \in [-\frac{1}{2}\pi, \frac{1}{2}\pi]$ is another free angular parameter. However, in [23,34,36] it is shown that the shape of the curves are only dependent on *differences* of the angles $\phi_{i,0}$, $\phi_{i,1}$, and $\phi_{i,2}$. Hence, without loss of generality we assume that $\phi_{i,1} = -\frac{1}{2}\pi$, and Equation (32) reduces to

$$\bar{\mathcal{A}}_{i,1} = -\frac{3}{4}\left(\bar{\mathcal{A}}_{i,0} + \bar{\mathcal{A}}_{i,2}\right) + \frac{\sqrt{\frac{1}{2}(1+\lambda_i^{\mathrm{c}})|\mathbf{c}|}}{4}\left(1 - \frac{\nu_i^{\mathrm{c}}}{1+\lambda_i^{\mathrm{c}}}\hat{\mathbf{j}} + \frac{\mu_i^{\mathrm{c}}}{1+\lambda_i^{\mathrm{c}}}\hat{\mathbf{k}}\right). \tag{33}$$

It is clear that the unknowns $\bar{\mathcal{A}}_{i,0}$, $\bar{\mathcal{A}}_{i,1}$, and $\bar{\mathcal{A}}_{i,2}$ are completely determined by four parameters $|\mathbf{d}_i^{\mathrm{i}}|$, $|\mathbf{d}_i^{\mathrm{f}}|$, $\phi_{i,0}$, and $\phi_{i,2}$. Therefore, a four-parameter family of solutions characterizes the spatial quintic PH Bézier curve that satisfies the first-order Hermite interpolation. The timing law provides additional degrees-of-freedom to meet the dynamic and mission-specific constraints as well as the minimum spatial clearance between the vehicles.

B. Timing Law as a Bézier Polynomial

In Section II.B we defined the timing law $\theta_i(t_d)$ as (see Equation (7)):

$$\theta_i(t_d) = \frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d}, \qquad \text{for} \quad t_d \in [0, t_{d,i}^{\mathrm{f}}].$$

To preserve the favorable properties of Bézier curves, it is intuitive to describe the timing law as a Bézier polynomial. An immediate observation is that the parameter t_d for vehicle i is defined on $[0, t_{d,i}^{\mathrm{f}}]$, whereas the dimensionless variable parameterizing Bézier polynomials is defined on $[0, 1]$. Hence, we introduce the *dimensionless* time variable

$$\hat{t}_{d,i} := \frac{t_d}{t_{d,i}^{\mathrm{f}}}, \qquad \mathrm{d}\hat{t}_{d,i} = \frac{1}{t_{d,i}^{\mathrm{f}}}\mathrm{d}t_d, \tag{34}$$

and re-define the timing law as follows:

$$\theta_i(\hat{t}_{d,i}) = \frac{\mathrm{d}\zeta_i}{\mathrm{d}\hat{t}_{d,i}}, \qquad \hat{t}_{d,i} \in [0, 1].$$

The function $\zeta_i(\hat{t}_{d,i})$ is represented by the integral

$$\zeta_i(\hat{t}_{d,i}) = \int_0^{\hat{t}_{d,i}} \theta_i(\tau)\mathrm{d}\tau$$

and, if we assume that $\theta_i(\hat{t}_{d,i})$ is a Bézier polynomial of degree n_θ, then $\zeta_i(\hat{t}_{d,i})$ will be a degree $(n_\theta + 1)$ Bézier polynomial. Using the integration property of Bézier polynomials [23,31], the control points $\bar{\zeta}_{i,k}$ can be expressed in terms of the control points $\bar{\theta}_{i,k}$:

$$\begin{aligned} \bar{\zeta}_{i,0} &= 0, \\ \bar{\zeta}_{i,k} &= \frac{1}{n_\theta + 1}\sum_{l=0}^{k-1}\bar{\theta}_{i,l}, \qquad \text{for} \quad k = 1, \ldots, (n_\theta + 1). \end{aligned}$$

The first control point $\bar{\zeta}_{i,0} = 0$ follows from the fact that $\zeta_i(0) = 0$. Similarly, since $\zeta_i(1) = 1$ and, therefore, $\bar{\zeta}_{i,n_\theta+1} = 1$, we can derive an equality constraint in terms of the control points $\bar{\theta}_{i,k}$ that has to be satisfied:

$$\frac{1}{n_\theta + 1}\sum_{k=0}^{n_\theta}\bar{\theta}_{i,k} = 1. \tag{35}$$

Next, we observe that the dynamic relation of the parameter ζ_i with respect to the time variable t_d is now given by

$$\frac{\mathrm{d}\zeta_i}{\mathrm{d}t_d} = \frac{1}{t_{d,i}^{\mathrm{f}}}\theta_i(\hat{t}_{d,i}).$$

Then the desired speed profile becomes

$$v_{d,i}(\hat{t}_{d,i}) = \frac{1}{t^{\mathrm{f}}_{d,i}} \left\| \mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i})) \right\|_2 \theta_i(\hat{t}_{d,i}) \,, \qquad \hat{t}_{d,i} \in [0,1] \,,$$

and, by using the differentiation property of Bézier polynomials [23,31], the initial and final speed conditions can be written as:

$$v^{\mathrm{i}}_i = \frac{5}{t^{\mathrm{f}}_{d,i}} \left\| \bar{\mathbf{p}}_{i,1} - \bar{\mathbf{p}}_{i,0} \right\|_2 \bar{\theta}_{i,0} \,, \qquad v^{\mathrm{f}}_i = \frac{5}{t^{\mathrm{f}}_{d,i}} \left\| \bar{\mathbf{p}}_{i,5} - \bar{\mathbf{p}}_{i,4} \right\|_2 \bar{\theta}_{i,n_\theta} \,, \tag{36}$$

where $\bar{\theta}_{i,0}$ and $\bar{\theta}_{i,n_\theta}$ are the first and last control points, respectively, of the timing law Bézier polynomial. It is obvious that, given the three equalities in Equations (35) and (36), at least *three* free parameters (including $t^{\mathrm{f}}_{d,i}$) are required to solve this system of equations. Therefore, it is necessary that $n_\theta \geq 2$ such that there remains at least one free parameter for meeting the temporal specifications. In this framework we decide to work with *quadratic* Bézier polynomials to describe the timing laws $\theta_i(\hat{t}_{d,i})$, that is

$$\theta_i(\hat{t}_{d,i}) = \sum_{k=0}^{2} \bar{\theta}_{i,k}\, b^2_k(\hat{t}_{d,i}) \,, \qquad \text{for} \quad i = 1, \ldots, N \,, \quad \hat{t}_{d,i} \in [0,1] \,. \tag{37}$$

Note that the timing laws are Bézier polynomials that do not have Pythagorean hodographs. With the timing law defined as in Equation (37), it is easy to verify that the function $\zeta_i(\hat{t}_{d,i})$ can be found as:

$$\zeta_i(\hat{t}_{d,i}) = \sum_{k=0}^{3} \bar{\zeta}_{i,k} b^3_k(\hat{t}_{d,i}) \,,$$
$$\bar{\zeta}_{i,0} = 0 \,, \quad \bar{\zeta}_{i,1} = \frac{1}{3}\bar{\theta}_{i,0} \,, \quad \bar{\zeta}_{i,2} = \frac{1}{3}\left(\bar{\theta}_{i,0} + \bar{\theta}_{i,1}\right) \,, \quad \bar{\zeta}_{i,3} = 1 \,.$$

Also, with the timing law defined as a quadratic Bézier polynomial, Equations (35) and (36) form a system of three equations with four unknowns. For convenience, we let $t^{\mathrm{f}}_{d,i}$ be the free parameter here. Then, combining the free parameters characterizing the spatial PH Bézier curve $\mathbf{p}_{d,i}(\zeta_i)$, we can define for the ith vehicle, a vector of free parameters as $\boldsymbol{\Xi}_i = \left[\, |\mathbf{d}^{\mathrm{i}}_i|, |\mathbf{d}^{\mathrm{f}}_i|, \phi_{i,0}, \phi_{i,2}, t^{\mathrm{f}}_{d,i} \,\right]^\top$.

C. Set of Constraints in Bézier Form

In this section, we re-formulate the set of constraints that was presented in Section II.C, in terms of the (PH) Bézier polynomials. In our proposed trajectory-generation framework, the spatial paths $\mathbf{p}_{d,i}(\zeta_i)$ are described by quintic PH Bézier curves, while the timing laws $\theta_i(\hat{t}_{d,i})$ are represented by quadratic Bézier polynomials. In the previous sections we derived that, if for the ith vehicle the boundary conditions

$$\mathbf{p}_{d,i}(\zeta_i = 0) = \mathbf{p}^{\mathrm{i}}_i \,, \quad \gamma_i(\zeta_i = 0) = \gamma^{\mathrm{i}}_i \,, \quad \psi_i(\zeta_i = 0) = \psi^{\mathrm{i}}_i \,, \quad v_{d,i}(\hat{t}_{d,i} = 0) = v^{\mathrm{i}}_i \,,$$
$$\mathbf{p}_{d,i}(\zeta_i = 1) = \mathbf{p}^{\mathrm{f}}_i \,, \quad \gamma_i(\zeta_i = 1) = \gamma^{\mathrm{f}}_i \,, \quad \psi_i(\zeta_i = 1) = \psi^{\mathrm{f}}_i \,, \quad v_{d,i}(\hat{t}_{d,i} = 1) = v^{\mathrm{f}}_i \,,$$

and the vector of free parameters $\boldsymbol{\Xi}_i = \left[\, |\mathbf{d}^{\mathrm{i}}_i|, |\mathbf{d}^{\mathrm{f}}_i|, \phi_{i,0}, \phi_{i,2}, t^{\mathrm{f}}_{d,i} \,\right]^\top$ are given, then a quintic PH Bézier curve of the form

$$\mathbf{p}_{d,i}(\zeta_i) = \sum_{k=0}^{5} \bar{\mathbf{p}}_{i,k} b^5_k(\zeta_i)$$

can be constructed that interpolates these endpoint conditions. The control points $\bar{\mathbf{p}}_{i,k}$ are defined by Equation (27) where $\bar{\mathcal{A}}_{i,0}$, $\bar{\mathcal{A}}_{i,1}$, and $\bar{\mathcal{A}}_{i,2}$ can be determined from Equations (28), (33), and (29), respectively. Moreover, if the control points for the timing law

$$\theta_i(\hat{t}_{d,i}) = \sum_{k=0}^{2} \bar{\theta}_{i,k}\, b^2_k(\hat{t}_{d,i})$$

are chosen as follows:

$$\bar{\theta}_{i,0} = \frac{t^{\mathrm{f}}_{d,i} v^{\mathrm{i}}_i}{|\mathbf{d}^{\mathrm{i}}_i|} \,, \qquad \bar{\theta}_{i,2} = \frac{t^{\mathrm{f}}_{d,i} v^{\mathrm{f}}_i}{|\mathbf{d}^{\mathrm{f}}_i|} \,, \qquad \bar{\theta}_{i,1} = 3 - \bar{\theta}_{i,0} - \bar{\theta}_{i,2} \,,$$

then the function

$$\zeta_i(\hat{t}_{d,i}) = \sum_{k=0}^{3} \bar{\zeta}_{i,k} b_k^3(\hat{t}_{d,i}) \,,$$

with control points

$$\bar{\zeta}_{i,0} = 0 \,, \quad \bar{\zeta}_{i,1} = \frac{1}{3}\bar{\theta}_{i,0} \,, \quad \bar{\zeta}_{i,2} = \frac{1}{3}\left(\bar{\theta}_{i,0} + \bar{\theta}_{i,1}\right) \,, \quad \bar{\zeta}_{i,3} = 1 \,,$$

satisfies its boundary conditions $\zeta_i(0) = 0$ and $\zeta_i(1) = 1$ as well.

Hence, for any vector $\boldsymbol{\Xi}_i$, we can construct a pair of quintic spatial PH Bézier curve $\mathbf{p}_{d,i}(\zeta_i)$ and timing law $\theta_i(\hat{t}_{d,i})$ in quadratic Bézier polynomial form, such that the boundary conditions for $\mathbf{p}_{d,i}(\zeta_i)$ and $\zeta_i(\hat{t}_{d,i})$ are satisfied. As discussed in Section II, we seek to generate feasible trajectories that are collision-free and meet the dynamic constraints of the vehicles. The kinematics along the trajectory are given by Equations (12) and (13) and their bounds in (14). In general, it is possible to preserve the Bézier structure in these equations and take full advantage of the favorable properties of Bézier curves. For those cases where this is not immediately possible, we aim to rewrite and transform the equations into Bézier (or rational Bézier) form. Evaluating these dynamic constraints involves finding the extremes of high-degree polynomials, and we have seen in Section III.B that finding the minimum and maximum of a Bézier (and rational Bézier) curve is extremely computationally efficient. To begin with, we derive from the above-given spatial path $\mathbf{p}_{d,i}(\zeta_i)$, the first and second derivative with respect to ζ_i:

$$\mathbf{p}'_{d,i}(\zeta_i) = 5\sum_{k=0}^{4} \left(\bar{\mathbf{p}}_{i,k+1} - \bar{\mathbf{p}}_{i,k}\right) b_k^4(\zeta_i) \,, \tag{38a}$$

$$\mathbf{p}''_{d,i}(\zeta_i) = 20\sum_{k=0}^{3} \left(\bar{\mathbf{p}}_{i,k+2} - 2\bar{\mathbf{p}}_{i,k+1} + \bar{\mathbf{p}}_{i,k}\right) b_k^3(\zeta_i) \,. \tag{38b}$$

Subsequently, as an inherent property of PH Bézier curves, the parametric speed $\sigma_i(\zeta_i)$ and its first derivative $\sigma'_i(\zeta_i)$ with respect to ζ_i are Bézier polynomials as well

$$\sigma_i(\zeta_i) = \sum_{k=0}^{4} \bar{\sigma}_{i,k} b_k^4(\zeta_i) \,, \tag{39a}$$

$$\sigma'_i(\zeta_i) = 4\sum_{k=0}^{3} \left(\bar{\sigma}_{i,k+1} - \bar{\sigma}_{i,k}\right) b_k^3(\zeta_i) \,, \tag{39b}$$

where the control points $\bar{\sigma}_{i,k}$ are functions of $\bar{\mathcal{A}}_{i,0}$, $\bar{\mathcal{A}}_{i,1}$, and $\bar{\mathcal{A}}_{i,2}$:

$$\bar{\sigma}_{i,0} = |\bar{\mathcal{A}}_{i,0}|^2 \,, \qquad \bar{\sigma}_{i,1} = \frac{1}{2}\left(\bar{\mathcal{A}}_{i,0}\bar{\mathcal{A}}^*_{i,1} + \bar{\mathcal{A}}_{i,1}\bar{\mathcal{A}}^*_{i,0}\right) \,,$$
$$\bar{\sigma}_{i,2} = \frac{1}{6}\left(\bar{\mathcal{A}}_{i,0}\bar{\mathcal{A}}^*_{i,2} + 4|\bar{\mathcal{A}}_{i,1}|^2 + \bar{\mathcal{A}}_{i,2}\bar{\mathcal{A}}^*_{i,0}\right) \,,$$
$$\bar{\sigma}_{i,3} = \frac{1}{2}\left(\bar{\mathcal{A}}_{i,1}\bar{\mathcal{A}}^*_{i,2} + \bar{\mathcal{A}}_{i,2}\bar{\mathcal{A}}^*_{i,1}\right) \,, \qquad \bar{\sigma}_{i,4} = |\bar{\mathcal{A}}_{i,2}|^2 \,.$$

Now, we can use Equations (38), (39), and the expression for the timing law $\theta_i(\hat{t}_{d,i})$ to develop the kinematic equations. Equation (12) shows that the flight-path angle $\gamma_i(\zeta_i)$ involves an arcsin(·) function, which clearly is not of the Bézier form. Also, given the fact that the bounds for the flight-path angle are asymmetric and that $\gamma_i(\zeta_i)$ can take negative values, there is a need for a signed quantity with which an equivalent constraint can be constructed and preferably expressed in Bézier form. Fortunately, this is achieved by evaluating $\sin\left(\gamma_i(\zeta_i)\right)$ instead of $\gamma_i(\zeta_i)$, since firstly, the argument of the arcsin(·) function in Equation (12) is a rational function, which can be rewritten as a rational Bézier curve, and secondly, the function sin(·) is a strictly monotonically increasing function with the appropriate sign convention on the interval $[-\frac{1}{2}\pi, \frac{1}{2}\pi]$. Here, the assumption is made that $-\frac{1}{2}\pi < \gamma_{\min} < \gamma_{\max} < \frac{1}{2}\pi$, which is a valid assumption, considering the fact that constraints on the flight-path angles are not required when $|\gamma_i| \geq \frac{1}{2}\pi$ are allowed. Hence, Equation (12) can

be transformed into a standard rational Bézier curve:

$$\begin{aligned}\sin\left(\gamma_i(\zeta_i)\right) &= \frac{\mathbf{p}'_{d,i}(\zeta_i)\cdot\hat{\mathbf{k}}}{\sigma_i(\zeta_i)}\\ &= \frac{\sum_{k=0}^{4} w_{\gamma_{si},k}\bar{\gamma}_{s_i,k}b_k^4(\zeta_i)}{\sum_{k=0}^{4} w_{\gamma_{si},k}b_k^4(\zeta_i)}\,,\end{aligned} \tag{40}$$

where $\bar{\gamma}_{s_i,k}$ and $w_{\gamma_{si},k}$ are the control points and weights of the rational Bézier curve, respectively. The equivalent constraint for the flight-path angle $\gamma_i(\zeta_i)$ is given in Equation (45). The desired speed profile $v_{d,i}$ in Equation (13a) can be easily expressed as a Bézier polynomial, by first finding the composition of $(\mathbf{p}'_{d,i}\circ\zeta_i)(\hat{t}_{d,i})$ as outlined in [31, 46] and, subsequently, using the multiplication property of Bernstein polynomials to determine the control points of $v_{d,i}(\hat{t}_{d,i})$:

$$\begin{aligned}v_{d,i}(\hat{t}_{d,i}) &= \frac{1}{t^{\mathrm{f}}_{d,i}}\left\|\mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i}))\right\|_2\theta_i(\hat{t}_{d,i})\\ &= \sum_{k=0}^{14}\bar{v}_{d_i,k}b_k^{14}(\hat{t}_{d,i})\,.\end{aligned} \tag{41}$$

The acceleration profile $a_{d,i}(\hat{t}_{d,i})$ is then defined as the first derivative of $v_{d,i}(\hat{t}_{d,i})$ with respect to t_d. Given the relation between $\hat{t}_{d,i}$ and t_d in (34), we obtain the following Bézier polynomial for the acceleration profile:

$$\begin{aligned}a_{d,i}(\hat{t}_{d,i}) &= \frac{14}{t^{\mathrm{f}}_{d,i}}\sum_{k=0}^{13}\left(\bar{v}_{d_i,k+1}-\bar{v}_{d_i,k}\right)b_k^{13}(\hat{t}_{d,i})\\ &= \sum_{k=0}^{13}\bar{a}_{d_i,k}b_k^{13}(\hat{t}_{d,i})\,.\end{aligned} \tag{42}$$

Note that Equation (42) represents the acceleration *along the path*, and not the *total* acceleration as given by Equation (13b), since limiting the acceleration along the path is more relevant to the simulation example that will be shown in the next section. The equations for $\dot{\gamma}_i$ and $\dot{\psi}_i$ in (13c) and (13d), respectively, can be transformed into rational Bézier curves as well. This can be performed straightforwardly for the turn rate $\dot{\psi}_i$, since Equation (13d) is already a rational function. To evaluate the bounds on the rate of change in the flight-path angle, we derive the expression for $\dot{\gamma}_i^2$ in terms of Bézier polynomials, instead of the rate of change in the flight-path angle $\dot{\gamma}_i$, due the square root in the denominator of Equation (13c). Equivalence of the constraint is achieved, if we assume that the bounds on $\dot{\gamma}_i$ are of symmetric nature. We obtain the following expression for $\dot{\gamma}_i^2$:

$$\begin{aligned}\dot{\gamma}_i^2(\hat{t}_{d,i}) &= \left[\frac{\sigma_i(\zeta_i(\hat{t}_{d,i}))\left(\mathbf{p}''_{d,i}(\zeta_i(\hat{t}_{d,i}))\cdot\hat{\mathbf{k}}\right)-\sigma'_i(\zeta_i(\hat{t}_{d,i}))\left(\mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i}))\cdot\hat{\mathbf{k}}\right)}{\sigma_i(\zeta_i(\hat{t}_{d,i}))\left(\left(\mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i}))\cdot\hat{\mathbf{i}}\right)^2+\left(\mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i}))\cdot\hat{\mathbf{j}}\right)^2\right)^{\frac{1}{2}}}\frac{\theta_i(\hat{t}_{d,i})}{t^{\mathrm{f}}_{d,i}}\right]^2\\ &= \frac{\sum_{k=0}^{48} w_{\gamma_{di},k}\bar{\gamma}_{d_i,k}b_k^{48}(\hat{t}_{d,i})}{\sum_{k=0}^{48} w_{\gamma_{di},k}b_k^{48}(\hat{t}_{d,i})}\,,\end{aligned} \tag{43}$$

while $\dot{\psi}_i(\hat{t}_{d,i})$ is given by

$$\begin{aligned}\dot{\psi}_i(\hat{t}_{d,i}) &= \frac{\left(\mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i}))\times\mathbf{p}''_{d,i}(\zeta_i(\hat{t}_{d,i}))\right)\cdot\hat{\mathbf{k}}}{\left(\mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i}))\cdot\hat{\mathbf{i}}\right)^2+\left(\mathbf{p}'_{d,i}(\zeta_i(\hat{t}_{d,i}))\cdot\hat{\mathbf{j}}\right)^2}\frac{\theta_i(\hat{t}_{d,i})}{t^{\mathrm{f}}_{d,i}}\\ &= \frac{\sum_{k=0}^{24} w_{\psi_{di},k}\bar{\psi}_{d_i,k}b_k^{24}(\hat{t}_{d,i})}{\sum_{k=0}^{24} w_{\psi_{di},k}b_k^{24}(\hat{t}_{d,i})}\,.\end{aligned} \tag{44}$$

Hence, in order for the generated trajectories to meet the dynamic constraints of the vehicles, Equations (40)-(44) are required to satisfy the following bounds that are *equivalent* to the bounds given in (14):

$$\begin{aligned} v_{d,\min} \leq v_{d,i}(\hat{t}_{d,i}) \leq v_{d,\max}\,, \quad |a_{d,i}(\hat{t}_{d,i})| \leq a_{d,\max}\,, \\ \sin(\gamma_{\min}) \leq \sin(\gamma_i(\zeta_i)) \leq \sin(\gamma_{\max})\,, \quad \dot{\gamma}_i^2(\hat{t}_{d,i}) \leq \dot{\gamma}_{\max}^2\,, \quad |\dot{\psi}_i(\hat{t}_{d,i})| \leq \dot{\psi}_{\max}\,, \qquad \forall \zeta_i, \hat{t}_{d,i} \in [0,1]\,. \end{aligned} \tag{45}$$

Lastly, a minimum spatial clearance E has to be ensured for the generated trajectories to be collision-free. In Section II.C.2, two strategies were presented to guarantee spatial deconfliction between the trajectories. Spatial separation is guaranteed if and only if the inequality in Equation (15) is satisfied. Since the spatial paths are PH Bézier curves, this inequality can be expressed as:

$$\min_{\substack{i,j=1,\ldots,N \\ i\neq j}} \left\| \sum_{k=0}^{5} \bar{\mathbf{p}}_{i,k} b_k^5(\zeta_i) - \sum_{k=0}^{5} \bar{\mathbf{p}}_{j,k} b_k^5(\zeta_j) \right\|_2^2 \geq E^2\,, \qquad \forall \zeta_i, \zeta_j \in [0,1]. \tag{46}$$

In the case when temporal separation is preferred over spatial separation, the path $\mathbf{p}_{d,i}(\zeta_i)$ has to be re-parameterized by the time variable $\hat{t}_{d,i}$. We have seen earlier that this can be performed efficiently since we are working with Bézier polynomials, and the composed function is again a Bézier polynomial on its own. It can be easily verified that $\mathbf{p}_{d,i}(\hat{t}_{d,i})$ is a degree 15 spatial PH Bézier curve. Hence, let

$$\mathbf{p}_{d,i}(\hat{t}_{d,i}) = \sum_{k=0}^{15} \tilde{\mathbf{p}}_{i,k} b_k^{15}(\hat{t}_{d,i})\,, \qquad \forall \hat{t}_{d,i} \in [0,1]\,,$$

where $\tilde{\mathbf{p}}_{i,k}$ are the control points, which can be determined using the recursive algorithm for computing the control points of the composition of two Bernstein polynomials. At this point, we assume that $t_{d,i}^{\mathrm{f}} = t_{d,j}^{\mathrm{f}} = T_d$ for $i,j = 1,\ldots,N$, i.e. missions where the vehicles arrive simultaneously at their desired destination and, hence, the temporal separation between the ith and jth vehicle is defined as

$$\min_{\substack{i,j=1,\ldots,N \\ i\neq j}} \left\| \sum_{k=0}^{15} \left(\tilde{\mathbf{p}}_{i,k} - \tilde{\mathbf{p}}_{j,k}\right) b_k^{15}(\hat{t}_d) \right\|_2^2 \geq E^2\,, \quad \forall \hat{t}_d \in [0,1]\,, \tag{47}$$

where now $\hat{t}_d = t_d/T_d$. One of the main motivations to use PH Bézier curves to describe the trajectories lies in the fact that evaluating Equations (46) and (47) is extremely computationally efficient as discussed in Section III.B.

Remark 3 *In the general case, when $t_{d,i}^{f} \neq t_{d,j}^{f}$ for $i,j = 1,\ldots,N$ and $i \neq j$, the issue arises that $\hat{t}_{d,i} \neq \hat{t}_{d,j}$ at time t_d, since $\hat{t}_{d,i}$ and $\hat{t}_{d,j}$ are variables normalized by $t_{d,i}^{f}$ and $t_{d,j}^{f}$, respectively. In [47] it is shown that this can be easily solved by subdividing one of the trajectories using the* de Casteljau *algorithm.*

D. Trajectory Generation: Constrained Optimization

The constrained optimization problem presented in Equation (3) can now be re-formulated for the framework that specifically uses quintic PH Bézier curves to represent the spatial paths, and quadratic Bézier polynomials for the timing laws:

$$\begin{aligned} &\min_{\boldsymbol{\Xi}_i \times \ldots \times \boldsymbol{\Xi}_N} J(\cdot) \\ &\text{subject to} \quad \begin{array}{l} \text{dynamic constraints of the vehicles (45),} \\ \text{Equation (46) or (47) for spatial deconfliction,} \\ \text{mission-specific constraints,} \end{array} \end{aligned}$$

where $\boldsymbol{\Xi}_i = \left[\, |\mathbf{d}_i^{\mathrm{i}}|, |\mathbf{d}_i^{\mathrm{f}}|, \phi_{i,0}, \phi_{i,2}, t_{d,i}^{\mathrm{f}} \,\right]^\top$ represents the vector of optimization parameters for the ith vehicle, and $J(\cdot)$ is a given cost function. The optimization problem is non-convex and can be solved using nonlinear programming. Note again that the boundary conditions in (9) and the boundary conditions on the function $\zeta_i(\hat{t}_{d,i})$ are automatically satisfied and, hence, do not impose extra constraints. Lastly, if simultaneous time-of-arrival is required, then the additional constraints would be given by Equation (10). However, this can be implemented simpler, by substituting for all $t_{d,i}^{\mathrm{f}}$ with T_d in the equations of Sections IV.B and IV.C. This method reduces the number of optimization parameters by $(n-1)$, but does not result in $\frac{1}{2}n(n-1)$ additional constraints.

V. Simulation Example

In this section a simulation scenario will be presented where three fixed-wing UAVs are tasked to converge to and follow three spatially deconflicted paths and arrive at their final destinations at the same time. Representative examples of such missions are simultaneous monitoring of multiple targets located at different positions, and time efficient retrieval of assets after a survey mission. Note that these missions impose only *relative* temporal constraints on the arrival of the UAVs. In particular, the mission scenario presented in this section is designed to be executed by small tactical UAVs equipped with an autopilot providing angular-rate and speed tracking capabilities, such as the one shown in Figure 4. For the execution of the mission, we will rely on the cooperative path-following framework presented in [27], which ensures that the UAVs follow the desired paths and arrive simultaneously at the desired final locations.

Figure 4: SIG Rascal 110, a small research aircraft with a wingspan of 110 in.

The simulation scenario is depicted in Figure 5. Three fixed-wing UAVs are tasked to arrive at their respective final positions simultaneously. The initial and final flight conditions (see Table 1) are chosen such that if the trajectory generation was performed individually rather than cooperatively, the paths would intersect at a single point where the UAVs will collide with each other. First we demonstrate that the proposed cooperative trajectory-generation framework generates collision-free trajectories by enforcing spatial deconfliction through spatial or temporal separation of the vehicles. Subsequently, we present simulation results of the execution of the mission, where the vehicles are temporally separated during the planning phase. A time-critical cooperative path-following algorithm enables the UAVs to track these desired paths closely, while meeting the stringent temporal specifications that ensure simultaneous time-of-arrival and temporal separation between the vehicles at all times [27, 28].

A. Generated Trajectories

In the planning phase we generate a desired trajectory for each UAV that cooperatively execute the mission shown in Figure 5. The numerical values for the boundary conditions are given in Table 1, along with the dynamic constraints for each individual aircraft. The cooperative trajectory-generation framework as outlined in Section IV is implemented in Matlab® on a desktop computer with Intel® Core™ i5-3470 CPU 3.20GHz, 8GB of RAM and running 64-bit Windows 7. Standard Matlab® routines are used to solve the constrained nonlinear programming. To show the efficacy of the framework in generating desired deconflicted trajectories, three different cases are considered. In Case I, no minimum separation constraints are imposed; Case II guarantees a minimum separation between the vehicles via spatial separation; while Case III ensures collision-free trajectories through temporal separation. In all three cases, the generated desired trajectories are satisfying the dynamic constraints of the vehicles and also the temporal specifications of the mission.

Figure 6 shows the flight paths for Case I. The separation between the vehicles is presented in Figure 6b and separations between the paths are given in Figures 6c-6e. Since no deconfliction constraints are imposed here, by construction of the example, the three generated paths intersect at one point as expected, and the UAVs arrive at the intersection simultaneously. The collision occurs around $t_d = 150$ s (see Figure 6b). Hence, although the generated trajectories are *flyable*, they are under no circumstances *feasible*.

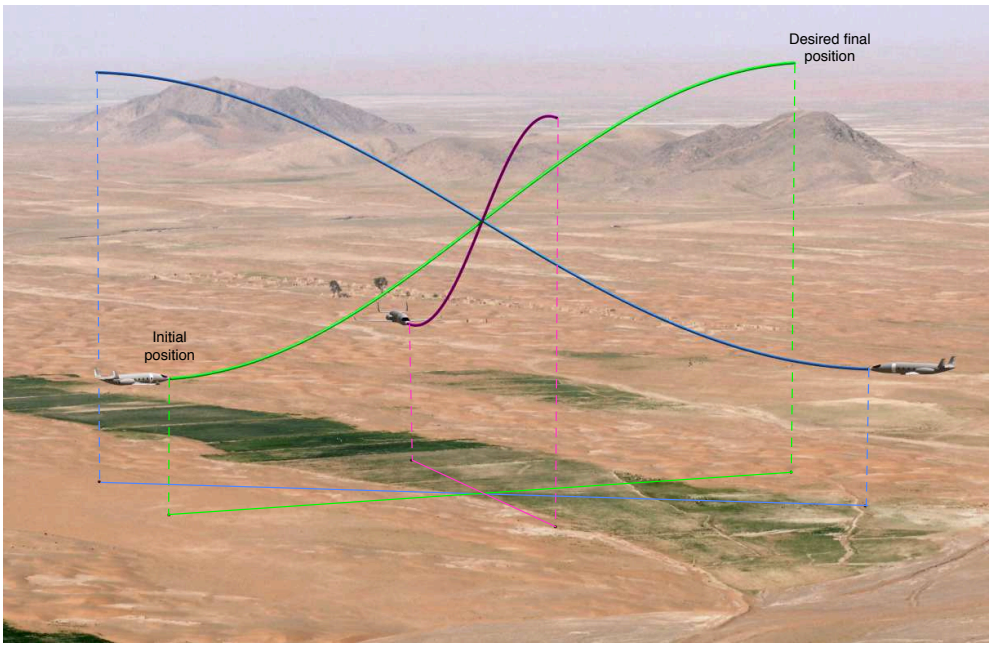


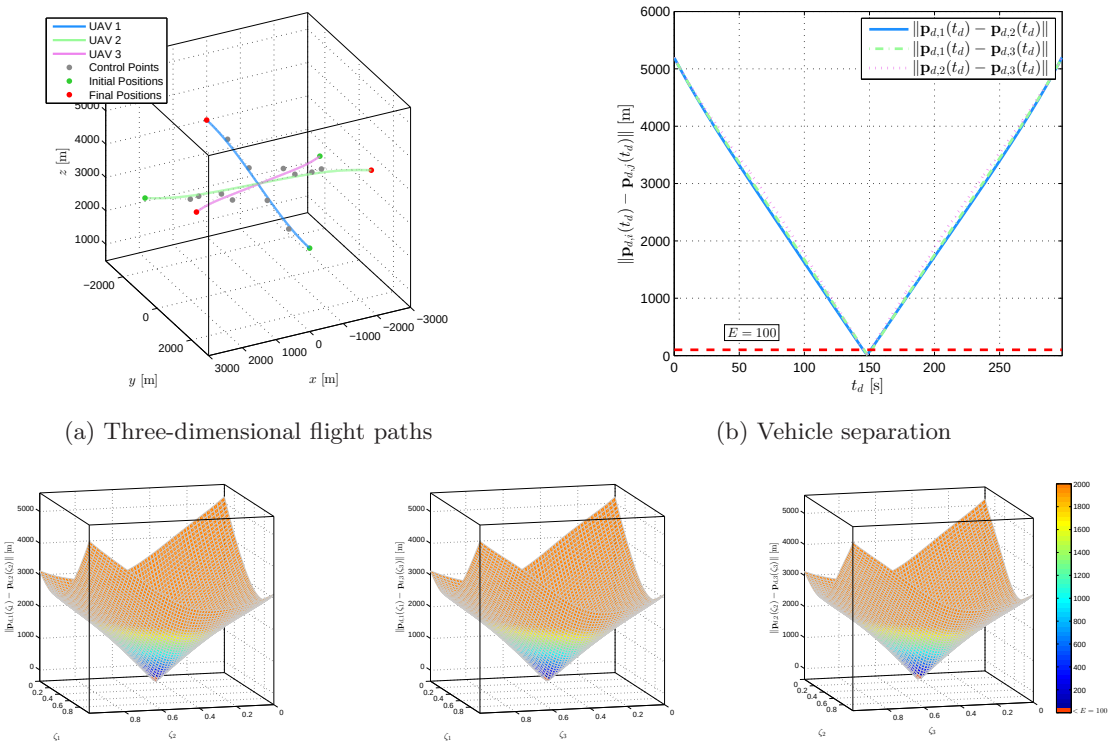
Figure 5: Artists' representation of the simulation scenario.

Table 1: Flight conditions and dynamic constraints of the UAVs.

		UAV 1	UAV 2	UAV 3
$\mathbf{p}_d^{\mathrm{i}}$	[km]	(0 , 3.00 , 3.00)	(2.60 , −1.50 , 3.00)	(−2.60 , −1.50 , 3.00)
v_d^{i}	[m/s]	25	25	25
γ^{i}	[deg]	0	0	0
ψ^{i}	[deg]	−90	150	30
$\mathbf{p}_d^{\mathrm{f}}$	[km]	(0 , −3.00 , 4.00)	(−2.60 , 1.50 , 4.00)	(2.60 , 1.50 , 4.00)
v_d^{f}	[m/s]	25	25	25
γ^{f}	[deg]	0	0	0
ψ^{f}	[deg]	−90	150	30
$v_{d,\max}$	[m/s]	32	32	32
$v_{d,\min}$	[m/s]	18	18	18
$a_{d,\max}$	[m/s^2]	10	10	10
$\gamma_{\min}$	[deg]	−20	−20	−20
$\gamma_{\max}$	[deg]	30	30	30
$\dot{\gamma}_{\max}$	[deg/s]	11.46	11.46	11.46
$\dot{\psi}_{\max}$	[deg/s]	11.46	11.46	11.46

Case II demonstrates the generation of collision-free trajectories through spatial separation, where the governing constraints are given in Equation (46). The results are presented in Figure 7, where it can be observed that the three trajectories are indeed separated both in space as well as in time, since the temporal separation is always greater than or equal to the spatial separation.

In the third and final case, spatial deconfliction is ensured via temporal separation (Equation (47)). The generated trajectories for this case are given in Figures 8 and 9. Comparison of Figure 8 with Figure 6 shows that the generated spatial paths in both cases do not differ much. However, deconfliction through temporal separation between the aircraft is ensured in Case III by adjusting the speed profiles of the UAVs along their respective paths. This is an example where the benefit of introducing the timing law comes to its full right. Here, independent adjustment of the speed profiles from the spatial paths is achieved through the decoupling of the temporal from the spatial elements of the trajectory by the timing law. Lastly, from Figure 9 it is



(a) Three-dimensional flight paths

(b) Vehicle separation

(c) Path separation UAV 1 and 2

(d) Path separation UAV 1 and 3

(e) Path separation UAV 2 and 3

Figure 6: Three-dimensional trajectories for a team of three cooperating UAVs. Case I: no spatial deconfliction is imposed. Computation time is 5.6 s.

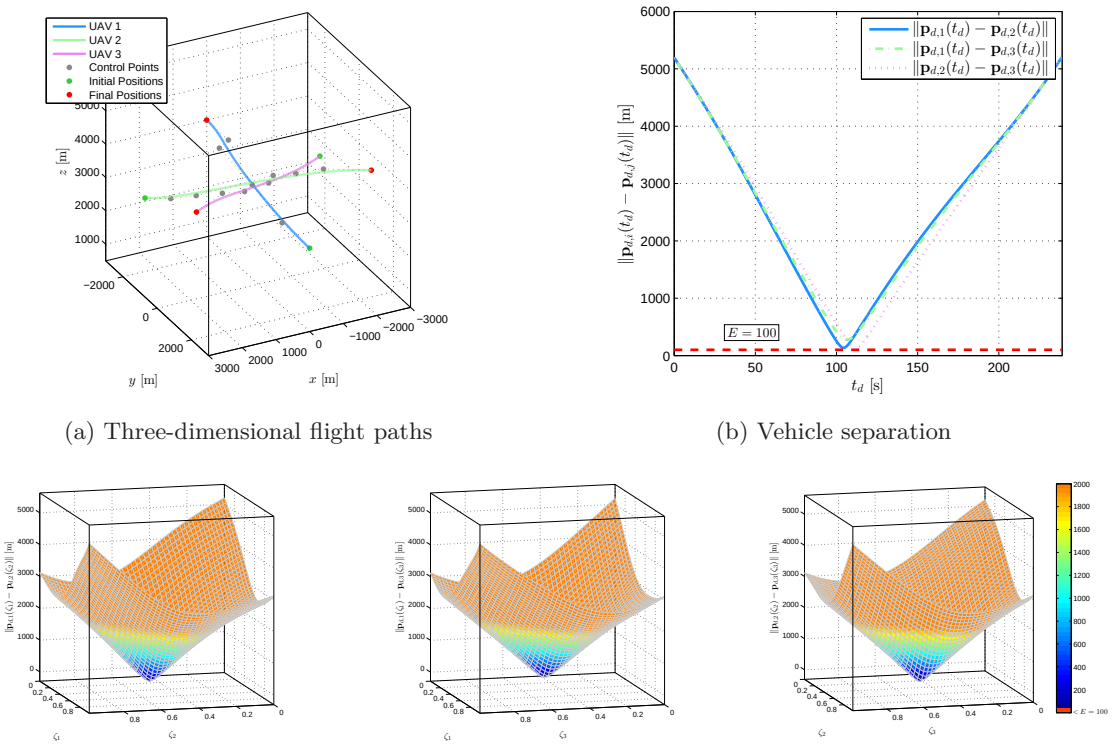
clear that the generated desired trajectories are flyable, as all dynamic constraints are satisfied.

Next, we show simulation results of the execution of the mission under Case III. Although the desired trajectories are separated in time, a novel time-critical cooperative path-following algorithm is still required to ensure that the desired vehicle separations are obeyed at all times during the execution phase. Moreover, the cooperative algorithm also ensures that the stringent time specifications of the mission, such as simultaneous time-of-arrival, are satisfied.

B. Mission Execution

To execute the multi-vehicle mission described above, we take advantage of the cooperative motion-control algorithms proposed in [27]. A nonlinear path-following algorithm ensures that each UAV converges to and follows its corresponding desired path, while a coordination control law adjusts the speed profiles of the vehicles to meet the temporal assignments of the mission. The coordination algorithm relies on the exchange of information among the vehicles over a supporting communications network. We note that the simulation results presented in this section are based on a kinematic model of a fixed-wing UAV, along with a simplified, decoupled linear model describing the roll, pitch, yaw, and speed dynamics of the closed-loop UAV with its autopilot; details about this simplified model can be found in [27].

Figures 10 and 11 illustrate the evolution of the UAVs (blue lines) moving along the paths (green lines). The UAVs start the mission with an initial offset in both position and attitude with respect to the beginning of the framed paths. As can be seen in Figure 10, the path-following algorithm eliminates this initial offset and steers the UAVs along the corresponding paths, while the coordination algorithm ensures simultaneous arrival at the end of the paths at $t = 249.2$ s. Note that the actual final mission time is slightly greater than the planned time, which was $T_d = 245.2$ s. However, the mission objective is met, since we are only concerned about meeting the relative temporal constraints of the mission, which ensure both vehicle deconfliction and simultaneous arrival of the UAVs at the desired final locations. Should one want to achieve simultaneous



(a) Three-dimensional flight paths

(b) Vehicle separation

(c) Path separation UAV 1 and 2

(d) Path separation UAV 1 and 3

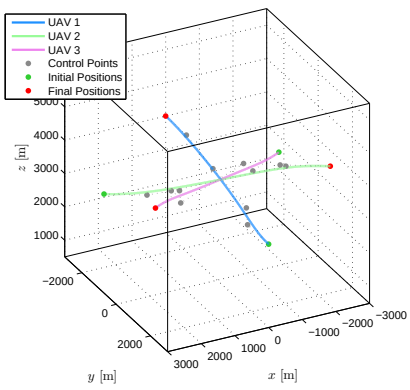
(e) Path separation UAV 2 and 3

Figure 7: Three-dimensional trajectories for a team of three cooperating UAVs. Case II: spatial deconfliction is ensured through spatial separation. Computation time is 16.2 s.

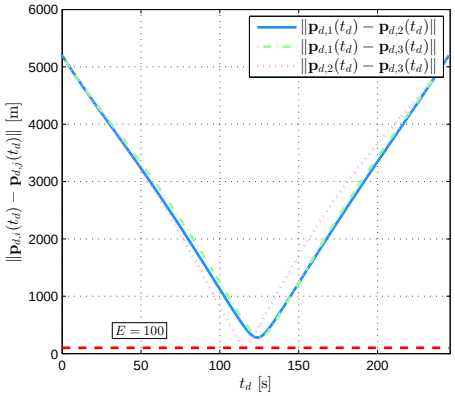
arrival within a prespecified temporal window, the work in [48] presents a modification of the coordination control law described in [27] that is capable of enforcing absolute temporal constraints.

Details about the performance of the path-following algorithm are shown in Figure 12. As described in [27], the key idea of the path-following algorithm is to use the attitude control effectors of the UAV to follow a *virtual target* vehicle running along the path. To this effect, a moving frame is attached to this virtual target and a generalized error vector is defined that characterizes the position and attitude errors between this moving coordinate system and a frame attached to the actual UAV. As illustrated in Figures 12a and 12c, the path-following position and attitude errors, denoted by $\boldsymbol{p_{F,i}}(t)$ and $\Psi(\tilde{\boldsymbol{R}}_{\boldsymbol{i}}(t))$, converge to a neighborhood of zero within 40 s and 10 s, respectively. Figures 12b and 12d present the angular-rate commands, $q_{c,i}(t)$ and $r_{c,i}(t)$, the actual angular rates, $q_i(t)$ and $r_i(t)$, as well as the rate of progression $\dot{\ell}_i(t)$ of the virtual targets along the path.

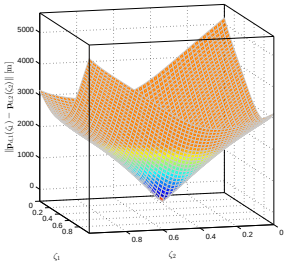
To enforce the temporal constraints that must be met to coordinate the entire fleet of UAVs, the speed profile of each vehicle is adjusted based on coordination information exchanged among the vehicles over the supporting communications network. As explained in [27], the time-coordination problem is formulated as a consensus problem, in which the objective of the fleet of vehicles is to reach agreement on some distributed variables of interest, denoted by ξ_i, $i = 1, 2, 3$. In particular, the temporal specifications of the mission are met if, first, $\xi_i = \xi_j$ for any two vehicles i and j and, second, $\dot{\xi}_i = 1$. The former ensures that the vehicles maintain desired relative position along their paths, while the latter implies that each vehicle travels at the desired speed $v_{d,i}$. The time-coordination algorithm relies on a distributed leader-follower control law with a proportional-integral structure that requires only the exchange of the coordination states ξ_i among the UAVs. The evolution of both the coordination errors $(\xi_i(t) - \xi_j(t))$ and the rate of change of the coordination states $\dot{\xi}_i(t)$ is illustrated in Figure 13, along with the resulting UAV speeds and the integral states implemented on the follower vehicles. The figure shows that the coordination errors converge to a neighborhood of zero, while the rate of change of the coordination states converges to the desired rate of 1 s/s. In particular, Figure 13b illustrates how the vehicles adjust their speeds (with respect to the desired



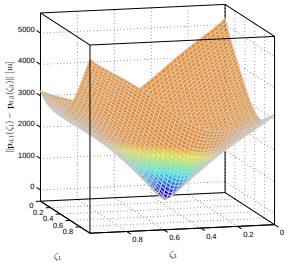
(a) Three-dimensional flight paths



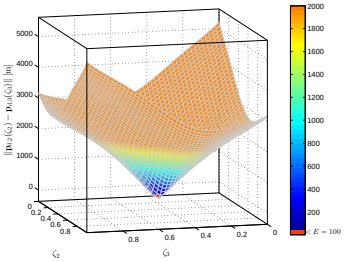
(b) Vehicle separation



(c) Path separation UAV 1 and 2



(d) Path separation UAV 1 and 3



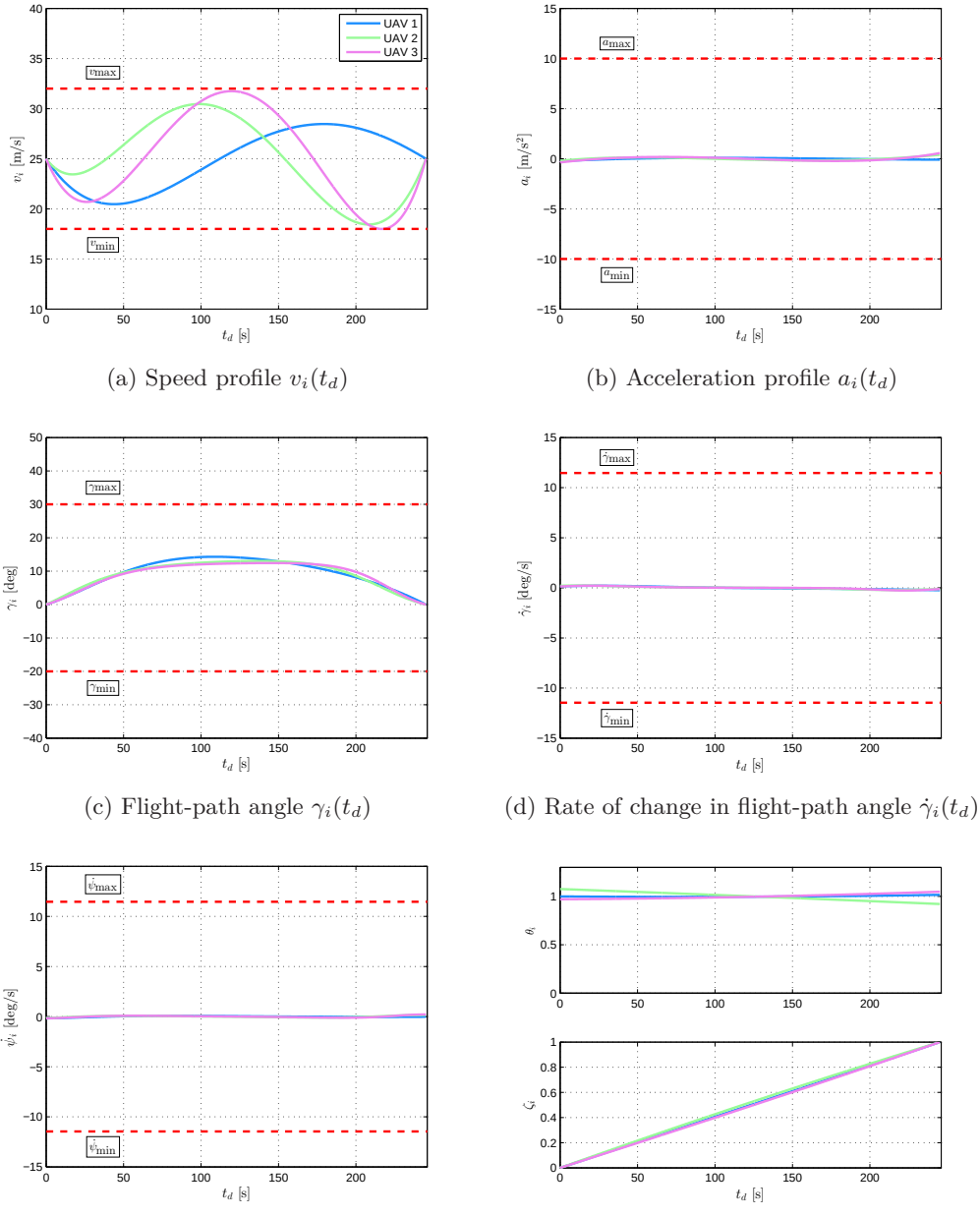
(e) Path separation UAV 2 and 3

Figure 8: Three-dimensional trajectories for a team of three cooperating UAVs. Case III: spatial deconfliction is ensured through temporal separation. Computation time is 14.5 s.

speed profiles) to achieve coordination. The figure also shows that, as a result of the switching nature of the network topology, the speed commands of the three vehicles are discontinuous. Finally, Figure 14 shows that the vehicles always maintain a minimal inter-vehicle distance of 100 m, as specified in the formulation of the trajectory-generation problem.

VI. Conclusions

A cooperative trajectory-generation framework is presented for generating a set of spatial trajectories for a team of cooperating vehicles that execute time-critical missions. In comparison with our previous work, a new timing law and a new approach of incorporating the dynamic constraints are introduced. It is shown that the framework is able to generate collision-free trajectories that are either spatially or temporally separated, and that are satisfying the dynamic constraints of the vehicles. The framework also has the potential to compute these trajectories in (near) real-time, as computationally efficient algorithms are used to evaluate the complex set of constraints. This is achieved by using Pythagorean Hodograph Bézier curves to describe the trajectories. The simulation results show the seamless integration of the cooperative trajectory-generation framework with a novel time-critical cooperative path-following algorithm, that together ensure that the vehicles closely track collision-free trajectories, while meeting the stringent temporal specifications of the mission.



(a) Speed profile $v_i(t_d)$

(b) Acceleration profile $a_i(t_d)$

(c) Flight-path angle $\gamma_i(t_d)$

(d) Rate of change in flight-path angle $\dot{\gamma}_i(t_d)$

(e) Turn rate $\dot{\psi}_i(t_d)$

(f) Timing law $\theta_i(t_d)$ and the function $\zeta_i(t_d)$

Figure 9: Dynamic constraints and timing laws for a team of three cooperating UAVs. Case III: spatial deconfliction is ensured through temporal separation.

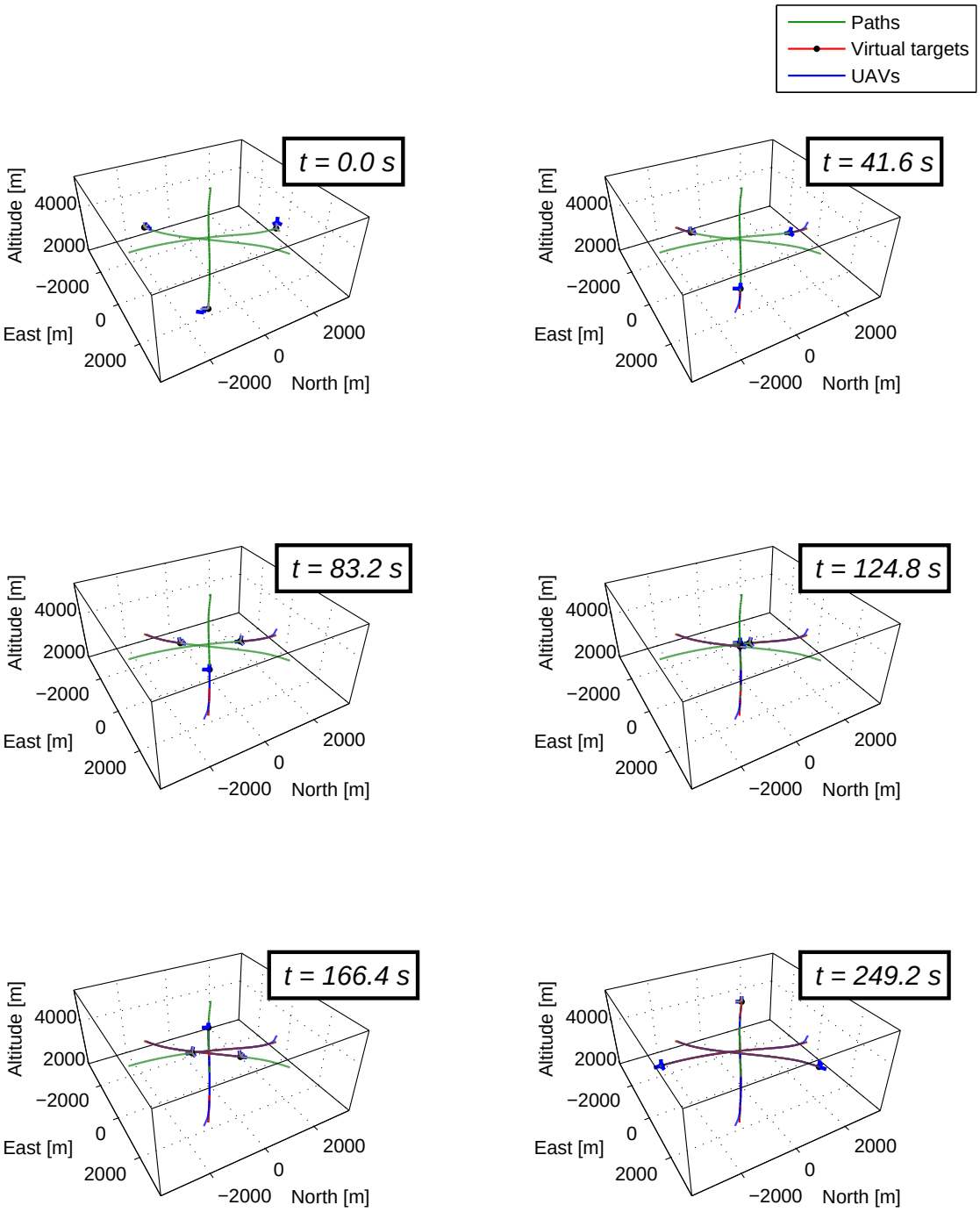


Figure 10: The three UAVs follow their paths, avoid collision, and achieve simultaneous arrival at their final destinations at $t = 249.2$ s.

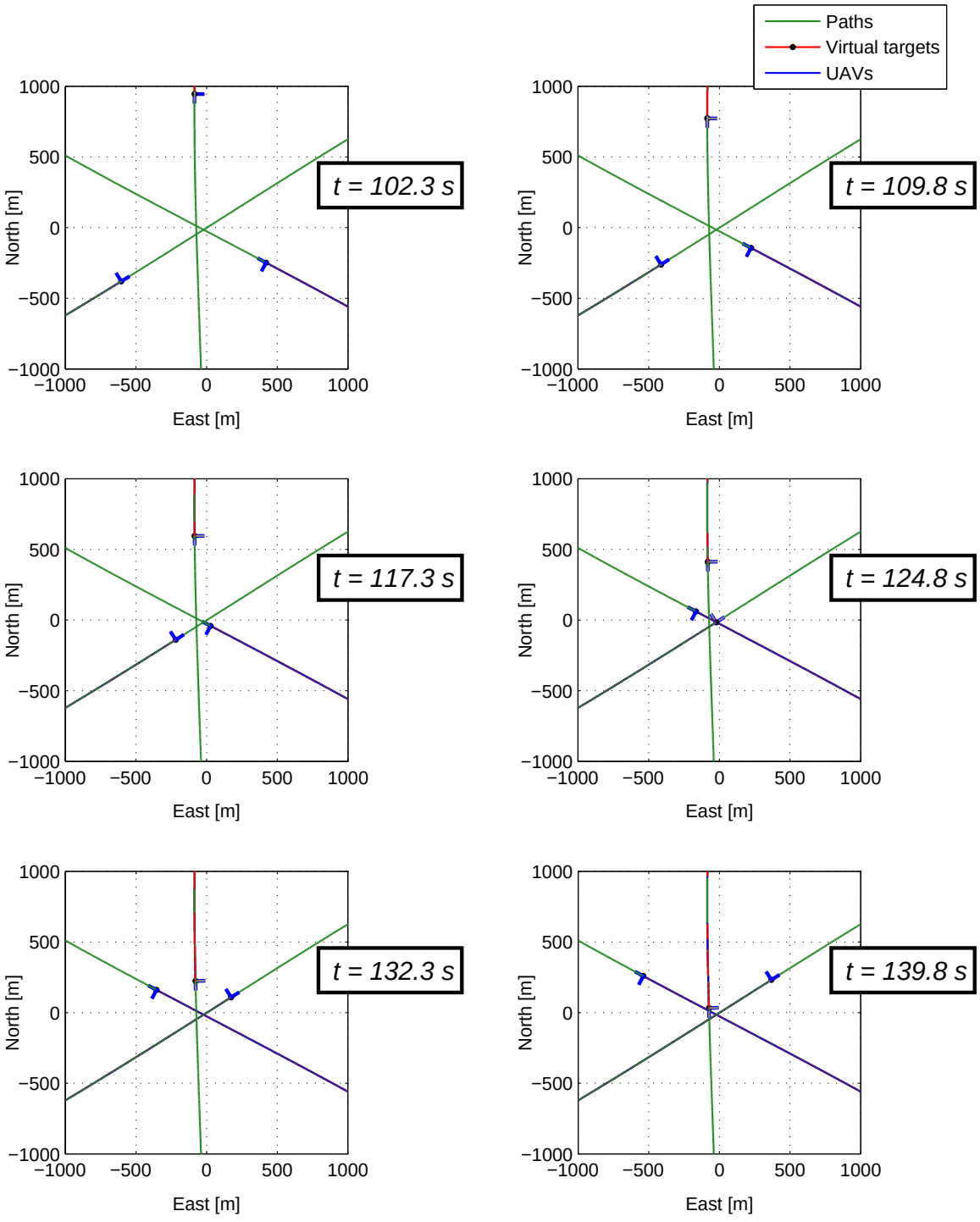
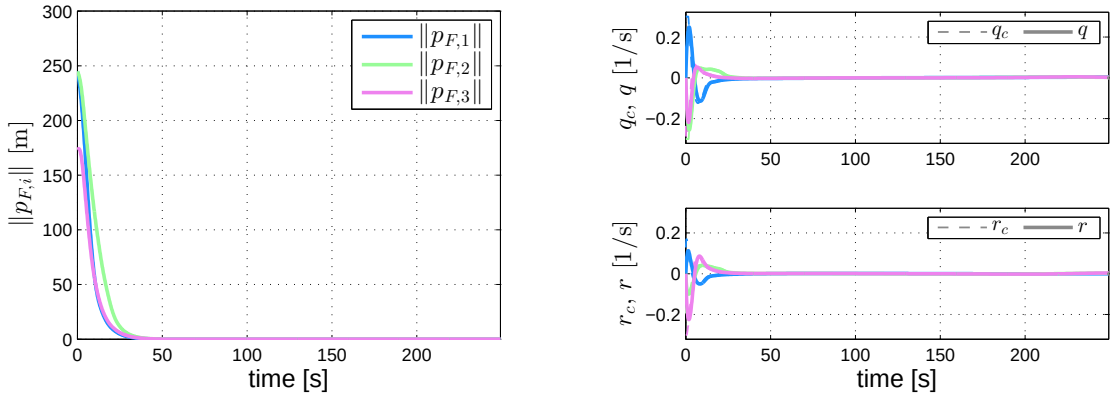
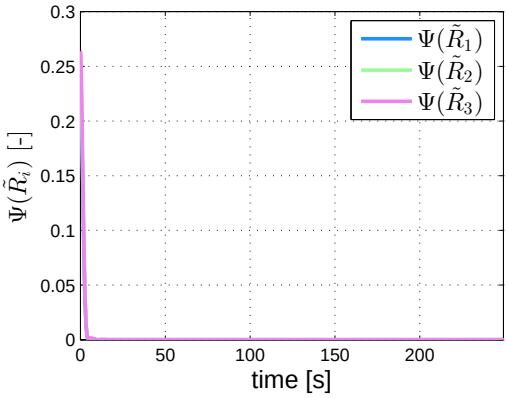


Figure 11: The three UAVs avoid collision and maintain a minimal spatial separation of approximately 100 m.

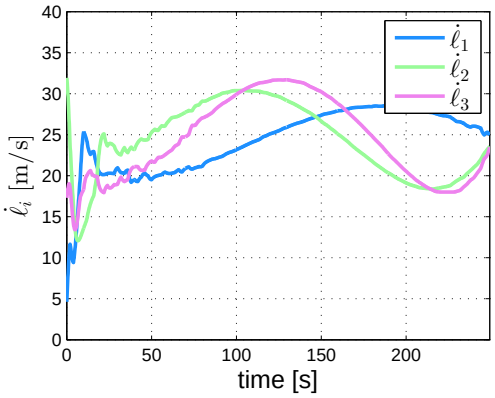


(a) Position error

(b) Angular rates

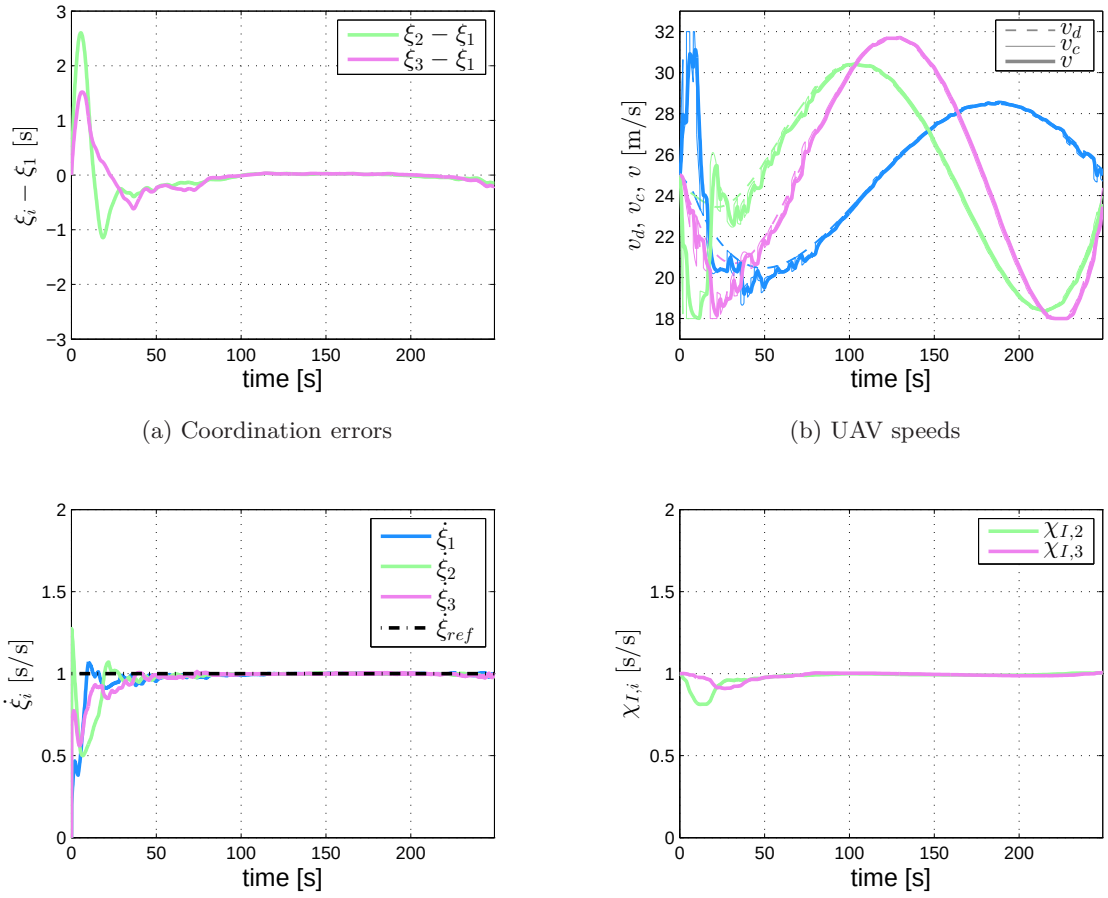


(c) Attitude error



(d) Progression of the virtual targets

Figure 12: The path-following algorithm drives the path-following position and attitude errors to a neighborhood of zero.



(a) Coordination errors

(b) UAV speeds

(c) Coordination-state rates

(d) Integral states

Figure 13: The coordination control law ensures that the coordination errors converge to a neighborhood of zero and also that the rate of change of the coordination states evolves at about the desired rate of 1 s/s.

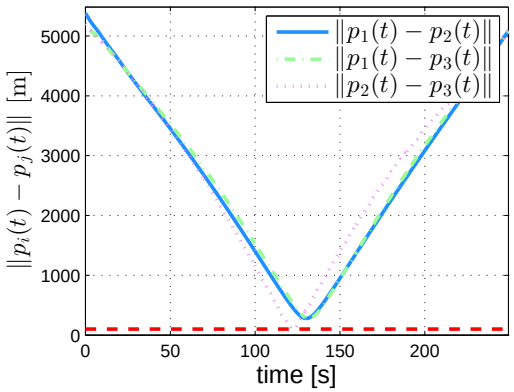


Figure 14: The inter-vehicle distance between all of the UAVs is always greater than 100 m.

References

[1]USAF unmanned aircraft systems flight plan 2009–2047. Technical report, United States Air Force, Washington, DC, May 2009. Available online: http://www.fas.org/irp/program/collect/uas_2009.pdf.

[2]Tom Schouwenaars, Jonathan P. How, and Eric Feron. Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees. In *AIAA Guidance, Navigation and Control Conference*, Providence, RI, August 2004. AIAA 2004-5141.

[3]Timothy W. McLain and Randal W. Beard. Coordination variables, coordination functions, and cooperative timing missions. *Journal of Guidance, Control and Dynamics*, 28(1):150–161, January–February 2005.

[4]Jarurat Ousingsawat and Mark E. Campbell. Optimal cooperative reconnaissance using multiple vehicles. *Journal of Guidance, Control and Dynamics*, 30(1):122–132, January–February 2007.

[5]Eelco Scholte and Mark E. Campbell. Robust nonlinear model predictive control with partial state information. *IEEE Transactions on Control System Technology*, 16(4):636–651, July 2008.

[6]Yoshiaki Kuwata and Jonathan P. How. Cooperative distributed robust trajectory optimization using receding horizon MILP. *IEEE Transactions on Control System Technology*, 19(2):423–431, March 2011.

[7]Jeremy Rea. Launch vehicle trajectory optimization using a Legendre pseudospectral method. In *AIAA Guidance, Navigation and Control Conference*, Austin, TX, August 2003. AIAA 2003-5640.

[8]Fariba Fahroo and I. Michael Ross. On discrete-time optimality conditions for pseudospectral methods. In *AIAA/AAS Astrodynamics Specialist Conference*, Keystone, CO, August 2006. AIAA 2006-6304.

[9]John P. Riehl, Stephen W. Paris, and Waldy K. Sjauw. Comparison of implicit integration methods for solving aerospace trajectory optimization problems. In *AIAA/AAS Astrodynamics Specialist Conference*, Keystone, CO, August 2006. AIAA 2006-6033.

[10]Alisa M. Hawkins, Thomas J. Fill, Ronald J. Proulx, and Eric M. Feron. Constrained trajectory optimization for lunar landing. In *AIAA/AAS Astrodynamics Specialist Conference*, Tampa, FL, January 2006. AAS 06-153.

[11]Kevin P. Bollino, L. Ryan Lewis, Pooya Sekhavat, and I. Michael Ross. Pseudospectral optimal control: A clear road for autonomous intelligent path planning. In *AIAA Infotech@Aerospace*, Rohnert Park, CA, May 2007. AIAA 2007-2831.

[12]Qi Gong, L. Ryan Lewis, and I. Michael Ross. Pseudospectral motion planning for autonomous vehicles. *Journal of Guidance, Control and Dynamics*, 32(3):1039–1045, May–June 2009.

[13]Nazareth S. Bedrossian, Sagar Bhatt, Wei Kang, and I. Michael Ross. Zero-propellant maneuver guidance. *IEEE Control Systems Magazine*, 29(5):53–73, October 2009.

[14]Kevin P. Bollino and L. Ryan Lewis. Collision-free multi-UAV optimal path planning and cooperative control for tactical applications. In *AIAA Guidance, Navigation and Control Conference*, Honolulu, HI, August 2008. AIAA 2008-7134.

[15]Nazareth S. Bedrossian, Mark Karpenko, and Sagar Bhatt. Overclock my satellite: Sophisticated algorithms boost satellite performance on the cheap. *IEEE Spectrum*, 49(11):54–62, November 2012.

[16]Ronald Choe, Venanzio Cichella, Enric Xargay, Naira Hovakimyan, Anna C. Trujillo, and Isaac Kaminer. A trajectory-generation framework for time-critical cooperative missions. In *AIAA Infotech@Aerospace Conference*, Boston, MA, August 2013. AIAA 2013-4582.

[17]Venanzio Cichella, Ronald Choe, Bilal Syed Mehdi, Enric Xargay, Naira Hovakimyan, Vladimir Dobrokhodov, and Isaac Kaminer. Trajectory generation and collision avoidance for safe operation of cooperating UAVs. In *AIAA Guidance, Navigation and Control Conference*, National Harbor, MD, January 2014. AIAA-2014-0972.

[18]Mariano I. Lizarraga and Gabriel H. Elkaim. Spatially deconflicted path generation for multiple UAVs in a bounded airspace. In *IEEE/ION Position, Location and Navigation Symposium*, pages 1213–1218, Monterey, CA, May 2008.

[19]Syed Bilal Mehdi, Ronald Choe, Venanzio Cichella, and Naira Hovakimyan. Collision avoidance through path replanning using Bézier curves. In *AIAA Guidance, Navigation and Control Conference*, Kissimmee, FL, January 2015.

[20]Vasily T. Taranenko. *Experience on Application of Ritz's, Poincare's, and Lyapunov's Methods for Solving Flight Dyanmics Problems*. Air Force Engineering Academy Press, Moscow, 1968. (in Russian).

[21]Andreas J. Häusler, Reza Ghabcheloo, António M. Pascoal, A. Pedro Aguiar, Isaac Kaminer, and Vladimir N. Dobrokhodov. Temporally and spatially deconflicted path planning for multiple autonomous marine vehicles. In *8th Conference on Manoeuvring and Control of Marine Craft*, Guarujá (SP), Brazil, September 2009.

[22]Oleg A. Yakimenko. Direct method for rapid prototyping of near-optimal aircraft trajectories. *Journal of Guidance, Control and Dynamics*, 23(5):865–875, September–October 2000.

[23]Rida T. Farouki. *Pythagorean-Hodograph Curves*. Springer-Verlag, Berlin Heidelberg, 2008.

[24]Antonios Tsourdos, Brian White, and Madhavan Shanmugavel. *Cooperative Path Planning of Unmanned Aerial Vehicles*. American Institute of Aeronautics and Astronautics, Inc, Reston, VA, 2011.

[25]Madhavan Shanmugavel, Antonios Tsourdos, Rafał Żbikowski, Brian A. White, Camille-Alain Rabbath, and Nicolas Léchevin. A solution to simultaneous arrival of multiple UAVs using Pythagorean Hodograph curves. In *American Control Conference*, pages 2813–2818, Mineapolis, MN, June 2006.

[26]Reza Ghabcheloo, A. Pedro Aguiar, António M. Pascoal, Carlos Silvestre, Isaac Kaminer, and João P. Hespanha. Coordinated path-following in the presence of communication losses and delays. *SIAM Journal on Control and Optimization*, 48(1):234–265, 2009.

[27]Enric Xargay. *Time-Critical Cooperative Path-Following Control of Multiple Unmanned Aerial Vehicles*. PhD thesis, University of Illinois at Urbana-Champaign, Urbana, IL, United States, May 2013.

[28]Enric Xargay, Isaac Kaminer, António Pascoal, Naira Hovakimyan, Vladimir Dobrokhodov, Venanzio Cichella, A. Pedro Aguiar, and Reza Ghabcheloo. Time-critical cooperative path following of multiple unmanned aerial vehicles over time-varying networks. *Journal of Guidance, Control and Dynamics*, 36(2):499–516, March-April 2013.

[29]Enric Xargay, Vladimir Dobrokhodov, Isaac Kaminer, António M. Pascoal, Naira Hovakimyan, and Chengyu Cao. Time-critical cooperative control for multiple autonomous systems. *IEEE Control Systems Magazine*, 32(5):49–73, October 2012.

[30]Venanzio Cichella, Isaac Kaminer, Enric Xargay, Vladimir Dobrokhodov, Naira Hovakimyan, A. Pedro Aguiar, and António M. Pascoal. A Lyapunov-based approach for time-coordinated 3D path-following of multiple quadrotors. In *IEEE Conference on Decision and Control*, pages 1776–1781, Maui, HI, December 2012.

[31]Rida T. Farouki. The Bernstein polynomial basis: A centennial retrospective. *Computer Aided Geometric Design*, 29(6):379–419, August 2012.

[32]Rida T. Farouki and Takis Sakkalis. Pythagorean hodographs. *IBM Journal of Research and Development*, 34(5):736–752, September 1990.

[33]Rida T. Farouki. The elastic bending energy of Pythagorean-hodograph curves. *Computer Aided Geometric Design*, 13:227–241, 1996.

[34]Rida T. Farouki, Carlotta Giannelli, Carla Manni, and Alessandra Sestini. Identification of spatial PH quintic Hermite interpolants with near-optimal shape measures. *Computer Aided Geometric Design*, 25(4–5):274–297, May–June 2008.

[35]Rida T. Farouki and Chang Yong Han. Algorithms for spatial Pythagorean-Hodograph curves. In *Geometric Properties for Incomplete Data*.

[36]Rida T. Farouki, Mohammad al Kandari, and Takis Sakkalis. Hermite interpolation by rotation-invariant spatial Pythagorean-Hodograph curves. *Advances in Computational Mathematics*, 17(4):369–383, November 2002.

[37]Hyeong In Choi, Doo Seok Lee, and Hwan Pyo Moon. Clifford algebra, spin representation, and rational parameterization of curves and surfaces. *Advances in Computational Mathematics*, 17(1–2):5–48, July 2002.

[38]Jung-Woo Chang, Yi-King Choi, Myung-Soo Kim, and Wenping Wang. Computation of the minimum distance between two Bézier curves/surfaces. *Computers & Graphics*, 35(3):677–684, June 2011.

[39]Xiao-Diao Chen, Linqiang Chen, Yigang Wang, Gang Xu, and Jun-Hai Yong. Computing the minimum distance between two Bézier curves. *Journal of Computational and Applied Mathematics*, 229(1):294–301, July 2009.

[40]Elmer G. Gilbert, Daniel W. Johnson, and S. Sathiya Keerthi. A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation*, 4(2):193–203, April 1988.

[41]Gino van den Bergen. A fast and robust GJK implementation for collision detection of convex objects. *Journal of Graphics Tools*, 4(2):7–25, March 1999.

[42]Patrick Lindemann. The Gilbert-Johnson-Keerthi distance algorithm. In *Media Informatics Proseminar on Algorithms in Media Informatics*, 2009.

[43]Christer Ericson. The Gilbert-Johnson-Keerthi algorithm. SIGGRAPH Presentation, 2004. Sony Computer Entertainment America.

[44]Mohammed Afzal Shah and Nabil Aouf. 3D cooperative Pythagorean Hodograph path planning and obstacle avoidance for multiple UAVs. In *IEEE International Conference on Cybernetic Intelligence Systems*, Reading, UK, September 2010.

[45]Subchan Subchan, Brian A. White, Antonios Tsourdos, Madhavan Shanmugavel, and Rafał Żbikowski. Pythagorean Hodograph (PH) path planning for tracking airborne contaminant using sensor swarm. In *IEEE International Instrumentation and Measurement Technology Conference*, Victoria, Vancouver Island, Canada, May 2008.

[46]Tony D. DeRose. Composing Bézier simplexes. *ACM Transactions on Graphics*, 7(3):198–221, July 1988.

[47]Ronald Choe. Distributed cooperative trajectory generation for multiple autonomous vehicles using Pythagorean Hodograph Bézier curves. Preliminary Examination, University of Illinois at Urbana-Champaign, Urbana, IL, United States, July 2014.

[48]Javier Puig-Navarro, Enric Xargay, Ronald Choe, and Naira Hovakimyan. Time-critical coordination of multiple UAVs with absolute temporal constraints. In *AIAA Guidance, Navigation and Control Conference*, Kissimmee, FL, January 2015.