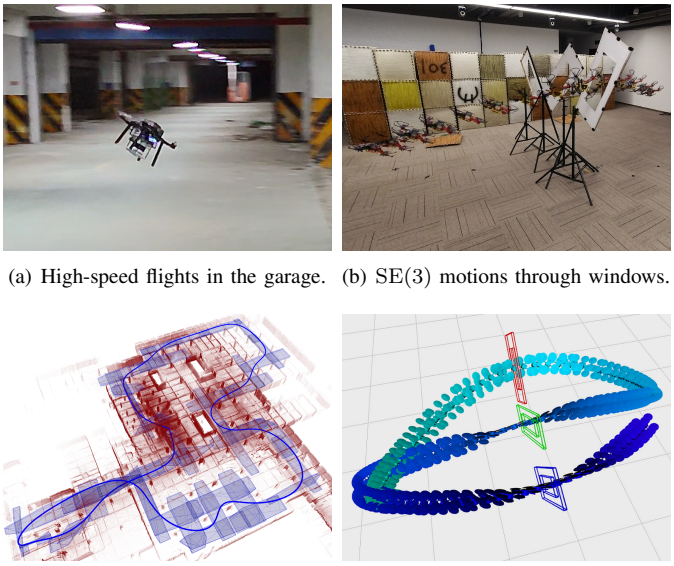
Geometrically Constrained Trajectory Optimization for Multicopters

Zhepei Wang, Xin Zhou, Chao Xu, and Fei Gao

***Abstract*—In this article, we present an optimization-based framework for multicopter trajectory planning subject to geometrical configuration constraints and user-defined dynamic constraints. The basis of the framework is a novel trajectory representation built upon our novel optimality conditions for unconstrained control effort minimization. We design linear-complexity operations on this representation to conduct spatial-temporal deformation under various planning requirements. Smooth maps are utilized to exactly eliminate geometrical constraints in a lightweight fashion. A variety of state-input constraints are supported by the decoupling of dense constraint evaluation from sparse parameterization, and backward differentiation of flatness map. As a result, this framework transforms a generally constrained multicopter planning problem into an unconstrained optimization that can be solved reliably and efficiently. Our framework bridges the gaps among solution quality, planning efficiency, and constraint fidelity for a multicopter with limited resources and maneuvering capability. Its generality and robustness are both demonstrated by applications to different flight tasks. Extensive simulations and benchmarks are also conducted to show its capability of generating high-quality solutions while retaining the computation speed against other specialized methods by orders of magnitude. The source code of our framework is available at: https://github.com/ZJU-FAST-Lab/GCOPTER.**

***Index Terms*—Aerial Systems: Applications, Motion and Path Planning, Autonomous Vehicle Navigation, Collision Avoidance.**



(a) High-speed flights in the garage. (b) SE(3) motions through windows.

(c) Global trajectory planning results. (d) SE(3) trajectory planning results.

Fig. 1. Experimental validation of our framework through extreme flights. The left figures show long-distance high-speed flights in an underground garage. A $343.57m$ trajectory is computed in $0.29s$. The right figures show aggressive yet robust SE(3) maneuvers through narrow windows repeated for 12 times. The flight speed and tilt angle in our experiments reach $12.0m/s$ and 90°, respectively. Details can be found in the attached multimedia.

I. Introduction

MULTICOPTERS rely on robust and efficient trajectory planning for safe yet agile autonomous navigation in complex environments [1]–[6]. For robotics, precisely incorporating dynamics, smoothness, and safety is essential to generate high-quality motions. Moreover, lightweight robots, such as multicopters under SWaP (size, weight, and power) constraints, put further hard requirements on the real-time computing using limited onboard resources. Despite that various successful tools in general-purpose kinodynamic planning or optimal control have been presented, few of them guarantee efficient online planning while also considering general constraints on dynamics for multicopters. Consequently, existing applications often use oversimplified requirements on trajectories for better computation efficiency, thus limiting the full exploitation of vehicle's capability.

The high-performance planning mentioned above possesses four major algorithmic challenges. First, ensuring safety often involves frequent interactions with a large volume of highly discretized environment data. Second, the nonlinearity of vehicle dynamics brings difficulties to directly enforcing physically acceptable states and inputs when the multicopter is flying at the limit of its capabilities. Third, high-quality motions conventionally need fine discretization of the dynamic process, where requirements for task resources tend to be unrealistic. Fourth, methods that use sparse representation for trajectories lack an effective way to optimize the temporal profile while satisfying continuous-time constraints.

In this article, we overcome these challenges by designing a lightweight and flexible optimization framework to meet user-defined requirements based on a novel trajectory class.

As the theoretical foundation of our framework, we present necessary and sufficient optimality conditions to multistage control effort minimization for the concerned linear dynamics, which are given for the first time to the best of our knowledge. The conditions are easy to use in that the unique optimal solution can be directly constructed with linear complexity in both time and space aspects. More importantly, the existence and uniqueness of conditions further provide crucial information on smoothness of the problem parameter sensitivity.

All authors are with the College of Control Science and Engineering, Zhejiang University, Hangzhou, 310027, China, and also with the Huzhou Institute of Zhejiang University, Huzhou, 313000, China. {wangzhepei, iszhouxin, cxu, fgaoaa}@zju.edu.cn This work was supported by the National Natural Science Foundation of China under Grant 62003299 and Grant 62088101. *(Corresponding authors: Fei Gao, Chao Xu.)*

To ease computation burden without sacrificing trajectory quality, it is essential to use sparse parameterization while keeping the flexibility to suit multicopter dynamics. Therefore, we design a novel trajectory class based on our optimality conditions. Any element in this class is by default an unconstrained control effort minimizer, thus we name it as MINCO (Minimum Control). MINCO differs from conventional splines that majorly focus on the smoothness of the geometrical shape, such as B-Splines and Bézier curves. Its sparse parameters are designed to directly control both the spatial and temporal profile of a trajectory, which are of equal importance for dynamic feasibility. Besides, a spatial-temporal deformation scheme is also designed such that MINCO can be optimized under any user-defined objective.

Our framework utilizes the geometrical approximation of low-dimensional free space based on results of sampling-based or search-based global methods. The safety is ensured via configuration constraints formed by the union of obstacle-free convex primitives. Constraint elimination schemes are proposed such that MINCO can be freely deformed through unconstrained optimization. The schemes exactly eliminate constraints that are directly defined on decision variables without introducing extra local minima.

Reliable motion planning requires admissible states and inputs, while most existing flatness-based methods only support differential constraints. To ensure high-fidelity feasibility, we propose a systematic way to enforce user-defined state-input constraints for our sparse parameterization without resorting to a fine discretization of trajectories. We exploit the backward differentiation of flatness map such that the constraint violation can be reflected in their gradient w.r.t. sparse parameters. Besides, a differentiable penalty functional is also proposed to enforce general continuous-time constraints.

Our framework focuses on computationally efficient yet high-quality trajectory planning for multicopters where there are complex constraints for safety, critical limits on dynamics, and task-specified requirements. To validate its effectiveness, we conduct extensive benchmarks against various cutting-edge multicopter trajectory planning methods. Results show that our method exceeds existing methods for orders of magnitude in efficiency, and retains comparable solution quality against general-purpose optimal-control solvers. We also conduct versatile simulations and extreme real-world flights to show the practical performance of our approach.

The contributions of this article are as follows.

- Optimality conditions in a general form on multi-stage control effort minimization are proposed with a proof of both the necessity and sufficiency for the first time.
- A novel trajectory class is designed to meet user-defined objectives while retaining local smoothness by spatial-temporal deformation via linear-complexity operations.
- A flexible trajectory planning framework that leverages both constraint elimination and constraint transcription is proposed for multicopter systems with user-defined state-input constraints.
- A set of simulations and experiments that validate our method significantly outperforms state-of-the-art works in efficiency, optimality, robustness, and generality.

II. Related Work

Despite various planning approaches in existing literature, there has yet to emerge a complete framework to accomplish time-critical large-scale trajectory planning for multicopters while incorporating user-defined continuous-time constraints on state and control. Our framework bridges this gap by exploring and exploiting different capabilities from both optimal control and motion planning.

A. Differentially Flat Multicopters

The concept of differential flatness has been introduced by Fliess et al. [7] and drawn great attentions in robotics trajectory planning [8]–[10]. The property makes it possible to recover the full state and input of a flat system from finite derivatives of its flat outputs. Mellinger and Kumar [11] validate the flatness of quadcopters with aligned propellers, which takes the thrust and three-dimensional torque as inputs. Watterson and Kumar [12] use Hopf fibration to decompose the quadcopter rotation, thus achieving the minimum singularity number in flatness maps. Ferrin et al. [13] show the flatness of a hexacopter whose inputs are desired orientation and thrust. They utilize the flatness to compute the nominal state where a Linear Quadratic Regulator (LQR) is applied. Faessler et al. [14] further consider linear drags that produce extra linear and angular accelerations. They show the flatness of parallel-rotor multicopters subject to the drag effect. Moreover, Mu and Chirarattananon [15] investigate underactuated multicopters with tilted propellers. They prove that the flatness holds for a wide range of tricopters, quadcopters, and hexacopters as long as the input rank condition is satisfied.

The flatness of a multicopter, if holds, benefits trajectory generation and tracking control in obtaining the reference state and input without integrating differential equations. Literature mentioned above uses flatness to avoid confronting system dynamics during planning. However, dynamics of a real physical system are only valid for reasonable state and admissible input. Although our framework also utilizes the flatness property, it differs from previous works in that a general form of state-input constraints is formally supported.

B. Sampling-Based Motion Planning

Sampling-based motion planners focus on global solutions of problems by exploration and exploitation, where the complexity mainly originates from the configuration space. The Probabilistic Roadmap (PRM) [16] and the Rapidly-exploring Random Tree (RRT) [17] are both probabilistically complete since their probability of failure decays to zero exponentially as the sample number goes to infinity [18]. Karaman and Frazzoli [19] propose asymptotically optimal variants of PRM and RRT, known as PRM* and RRT*, which ensure the convergence to globally optimal solutions as the sample number goes to infinity. There are also algorithms [20]–[22] that further improve the efficiency or applicability of randomized motion planning. Our method exploits sampling-based planners to overcome the complexity from environments. It accomplishes the optimization of a dynamically feasible trajectory that is

homotopic to a given low-dimensional collision-free path. It is designed to flexibly incorporate system state-input constraints, which is not the strength of sampling-based methods. In this way, the complexity from both the environments and dynamics are divided and conquered.

C. Optimization-Based Motion Planning

Optimization-based planners focus on local solutions by using high-order information of the problems. They depend on specific environment pre-processing methods such that the obstacle information is encoded into the optimization.

Trajectory optimization has long been studied for general systems in the control community [23]. Many general-purpose methods are designed for high-quality solutions, such as the collocation-based method GPOPS-II [24], and the shooting-based one ACADO [25]. They transcribe the original problem into a Nonlinear Programming (NLP) using a lot of equalities and variables, then resort to well-established NLP solvers such as SNOPT [26] or IPOPT [27]. However, trajectory planning in robotics may impose hard-to-formulate constraints, non-smoothness, and integer variables. Besides, general-purpose methods often take a long computation time, making them inappropriate for time-critical tasks. For example, Bry et al. [28] report that Direct Collocation (DC) with SNOPT takes several minutes to optimize a $4.5m$ trajectory for a 12-state airplane flying among cylindrical obstacles [29]. Therefore, specialized methods are on calling to overcome these difficulties.

For differentially flat multicopters, motion planning can be transformed into optimization of low-dimensional trajectories of flat outputs. Mellinger and Kumar [11] use fixed-duration splines to represent quadcopter trajectories. A Quadratic Programming (QP) is formulated by the quadratic cost of snap and linear constraints of safety. However, perturbation problems need to be solved in finite difference to estimate the gradient for time allocation. Its actuator constraints are also oversimplified. Bry et al. [28] propose a closed-form solution for this QP without safety constraints. They heuristically add waypoints from a collision-free path of RRT* to recompute the solution until the safety is satisfied. This method is admittedly efficient but cannot guarantee high-quality solutions in obstacle-rich environments. Besides, it involves the inverse of a matrix whose non-singularity is never discussed. Deits and Tedrake [30] approximate the free space using polytopes. The safety of each piece of trajectory is equivalent to a Sum-of-Square (SOS) condition if it entirely lies inside a polytope. They solve interval assignment using Mixed Integer Second Order Conic Programming (MISOCP). It generates globally optimal trajectories while the computation time is unacceptable. Gao et al. [31] also use the polyhedron-shaped free space representation. They alternately optimize the geometrical and temporal profile of a trajectory. The safety is enforced by the convex-hull property of Bézier curves, and the dynamic profile is optimized via Time-Optimal Path Parameterization (TOPP) [32]. There are also variants that propose improvements over the above methods. For example, Tordesillas et al. [33] improve the efficiency of [30] by substituting SOS conditions on polynomials with linear constraints on Bézier curves at the cost of conservatism [34]. Sun et al. [35] avoid integer variables by optimizing time allocation instead, where the sensitivity of a bilevel optimization is exploited.

These specialized methods utilize the continuous-time trajectory parameterization to avoid the computation burden from the fine discretization. However, they do not support flexibly optimizing its time allocation, decoupling temporal resolutions of constraints from decision variable dimensions, or enforcing high-fidelity constraints except for restrictions on derivative norms. In this article, our framework supports all these features by introducing unified techniques for a novel sparse parameterization. Moreover, the solution quality is comparable with that of the general-purpose optimal-control solvers.

III. Preliminaries

A. Differential Flatness

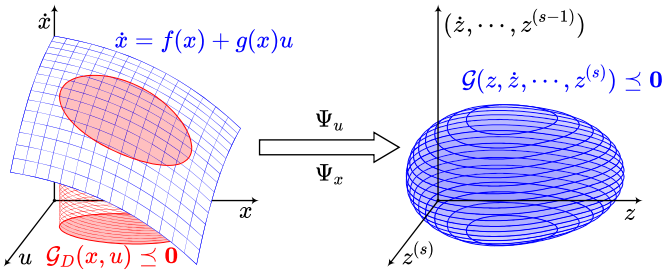


Fig. 2. Transform Ψ_u and Ψ_x of a flat system eliminate differential constraints (blue surface) from dynamics in the state-input space (left coordinate). The original state-input constraint $\mathcal{G}_\mathcal{D}$ (red area) is also transformed into a new constraint $\mathcal{G}$ (blue volume) in the flat-output space (right coordinate).

Consider a dynamical system of the following type

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

with $f : \mathbb{R}^n \mapsto \mathbb{R}^n$, $g : \mathbb{R}^n \mapsto \mathbb{R}^{n\times m}$, state $x \in \mathbb{R}^n$, and input $u \in \mathbb{R}^m$. The map g is assumed to have rank m. The system is said to be *differentially flat* [7], if there exists a *flat output* $z \in \mathbb{R}^m$ determined by x and finite derivatives of u, such that x and u can both be parameterized by finite derivatives of z:

$$x = \Psi_x(z, \dot{z}, \ldots, z^{(s-1)}), \tag{2}$$

$$u = \Psi_u(z, \dot{z}, \ldots, z^{(s)}), \tag{3}$$

where $\Psi_x : (\mathbb{R}^m)^s \mapsto \mathbb{R}^n$ and $\Psi_u : (\mathbb{R}^m)^{s+1} \mapsto \mathbb{R}^m$ are both induced by f and g. Intuitively, the state and control can be determined from z without explicit integration of the system dynamics (1).

Leveraging the flatness of a system, the trajectory generation is convenient when there are only differential constraints in (1). If we introduce a new control variable $v = z^{(s)}$ and denote $z^{[s-1]} \in \mathbb{R}^{ms}$ as

$$z^{[s-1]} = (z^{\mathrm{T}}, \dot{z}^{\mathrm{T}}, \ldots, z^{(s-1)^{\mathrm{T}}})^{\mathrm{T}}, \tag{4}$$

the input $u = \Psi_u(z^{[s-1]}, v)$ then exactly linearizes the original flat system into m decoupled chains of s-integrators. Let z_i denote the i-th entry in z, v_i the i-th entry in v and $z_i^{[s-1]} = (z_i, \dot{z}_i, \ldots, z_i^{(s-1)})^{\mathrm{T}}$. The i-th integrator chain is

$$\dot{z}_i^{[s-1]} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_{s-1} \\ 0 & \mathbf{0}^{\mathrm{T}} \end{pmatrix} z_i^{[s-1]} + \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} v_i, \tag{5}$$

where $\mathbf{0}$ and $\mathbf{I}$ are a zero matrix and an identity matrix with appropriate sizes, respectively. Given an initial state and a goal state, boundary values of each integrator chain (5) can be algebraically computed. Thus any trajectory integrated from these m integrator chains can be transformed into a feasible trajectory [8] for the original flat system via (2) and (3).

For dynamics with a small m, the flatness maps Ψ_x and Ψ_u further reduce the trajectory dimension and eliminate the differential constraints (1), which is illustrated in Fig. 2. As a side effect, nonlinearity coming from both Ψ_x and Ψ_u brings additional difficulties in trajectory generation for z when there are additional state-input constraints for (1). However, such an effect is relieved if the flat-output space coincides with the configuration space of the relevant planning problem.

B. Direct Optimization in Flat-Output Space

Fortunately, the differential flatness of multicopters has been well studied and shown to have physically meaningful flat-output space which overlaps with the configuration space. Explicit forms of Ψ_x and Ψ_u are available in [11]–[14] for a variety of underactuated multicopters. More importantly, their flat outputs share the same form in general:

$$z = (p_x, p_y, p_z, \psi)^{\mathrm{T}} \tag{6}$$

where $(p_x, p_y, p_z)^{\mathrm{T}}$ is the translation of the Center of Gravity (CoG) and ψ the yaw angle of the vehicle. The flat output z, especially its translation, provides a lot of convenience for the multicopter motion planning with complex spatial constraints.

To generate feasible motions for a multicopter, we first optimize the trajectory $z(t) : [0, T] \mapsto \mathbb{R}^m$ in its flat-output space such that most of the spatial constraints are directly enforced. Then, the flatness maps Ψ_x and Ψ_u are applied to transform $z(t)$ into the state-input trajectory $x(t)$ and $u(t)$.

For motion smoothness, the quadratic control effort [36] with time regularization is adopted as a cost functional of $z(t)$. General constraints on multicopters can be classified into configuration constraints and user-defined dynamic constraints. Normally, a collision-free motion implies

$$z(t) \in \mathcal{F},\ \forall t \in [0, T], \tag{7}$$

where $\mathcal{F}$ is the concerned *obstacle-free* region in configuration space. Besides, user-defined state-input constraints such as actuator limits or task-specific constraints are denoted by

$$\mathcal{G}_{\mathcal{D}}(x(t), u(t)) \preceq \mathbf{0},\ \forall t \in [0, T]. \tag{8}$$

Exploiting Ψ_x and Ψ_u, the corresponding constraints on $z(t)$ are computed as

$$\mathcal{G}_{\mathcal{D}}(\Psi_x(z^{[s-1]}(t)), \Psi_u(z^{[s]}(t))) \preceq \mathbf{0},\ \forall t \in [0, T]. \tag{9}$$

Apparently, via the flatness, a constraint on x and u has its equivalent form on the finite derivatives of $z(t)$. For simplicity, we denote (9) hereafter by

$$\mathcal{G}(z(t), \dot{z}(t), \ldots, z^{(s)}(t)) \preceq \mathbf{0},\ \forall t \in [0, T], \tag{10}$$

where $\mathcal{G}$ consists of n_g equivalent constraints.

It is worth noting that we do not make further assumptions on the multicopter dynamics and flatness maps. In other words, the proposed framework supports a wide range of multicopters, including, but not limited to the ones in [11]–[15].

C. Problem Formulation

Concluding above descriptions gives the following problem:

$$\min_{z(t),T} \int_0^T v(t)^{\mathrm{T}} \mathbf{W} v(t) \mathrm{d}t + \rho(T), \tag{11a}$$

$$s.t.\ \ z^{(s)}(t) = v(t),\ \forall t \in [0, T], \tag{11b}$$

$$\mathcal{G}(z(t), \ldots, z^{(s)}(t)) \preceq \mathbf{0},\ \forall t \in [0, T], \tag{11c}$$

$$z(t) \in \mathcal{F},\ \forall t \in [0, T], \tag{11d}$$

$$z^{[s-1]}(0) = \bar{z}_o,\ z^{[s-1]}(T) = \bar{z}_f, \tag{11e}$$

where $\mathbf{W} \in \mathbb{R}^{m \times m}$ is a positive diagonal matrix, $\rho : [0, \infty) \mapsto [0, \infty]$ the time regularization, $\bar{z}_o \in \mathbb{R}^{ms}$ the initial condition and $\bar{z}_f \in \mathbb{R}^{ms}$ the terminal condition. The control input v is allowed to be discontinuous in a finite number of time instants, as is commonly assumed in existing literature [36].

The trajectory optimization (11) is nontrivial because of the continuous-time constraints $\mathcal{G}$ and the nonconvex set $\mathcal{F}$. We further specify some reasonable conditions to make it a well defined problem. As for time regularization ρ, it trades off between the control effort and the expectation of total time,

$$\rho_s(T) = \sum_{i=0}^{M_T} b_i T^i, \tag{12}$$

where b_{M_T} is positive. Common choices are $\rho_s(T) = k_\rho T$ and $\rho_s(T) = k_\rho (T - T_\Sigma)^2$ with an expected time T_Σ. Besides, ρ can also be defined to strictly fix the total time:

$$\rho_f(T) = \begin{cases} 0 & if\ T = T_\Sigma, \\ \infty & if\ T \neq T_\Sigma. \end{cases} \tag{13}$$

As for nonlinear constraints $\mathcal{G}$, they are required to be C^2, i.e., twice continuously differentiable. As for the feasible region $\mathcal{F}$ in the configuration space, we approximate it geometrically by the union of $M_{\mathcal{P}}$ closed convex sets as

$$\mathcal{F} \simeq \tilde{\mathcal{F}} = \bigcup_{i=1}^{M_{\mathcal{P}}} \mathcal{P}_i. \tag{14}$$

For simplicity, locally sequential connection is assumed on these convex sets:

$$\begin{cases} \mathcal{P}_i \cap \mathcal{P}_j = \varnothing & if\ |i - j| = 2, \\ \mathrm{Int}\left(\mathcal{P}_i \cap \mathcal{P}_j\right) \neq \varnothing & if\ |i - j| \leq 1, \end{cases} \tag{15}$$

where $\mathrm{Int}(\cdot)$ means the interior of a set. The translation of $\bar{z}_o$ and $\bar{z}_f$ is inscribed in $\mathcal{P}_1$ and $\mathcal{P}_{M_{\mathcal{P}}}$, respectively. As for $\tilde{\mathcal{F}}$, we consider the case that each $\mathcal{P}_i$ is a closed m-dimensional ball:

$$\mathcal{P}_i^{\mathcal{B}} = \left\{ x \in \mathbb{R}^m \;\middle|\; \|x - o_i\|_2 \leq r_i \right\}, \tag{16}$$

or, more generally, a bounded convex polytope described by its $\mathcal{H}$-representation [37] with potentially redundant constraints:

$$\mathcal{P}_i^{\mathcal{H}} = \left\{ x \in \mathbb{R}^m \;\middle|\; \mathbf{A}_i x \preceq b_i \right\}. \tag{17}$$

For the optimization in (11), we aim to construct a computationally efficient solver while retaining the flexibility to handle different task-specific constraints $\mathcal{G}_{\mathcal{D}}$ in (8).

IV. Multi-Stage Control Effort Minimization

In this section, we analyze the multi-stage control effort minimization without functional constraints. For this problem, we propose easy-to-use optimality conditions for general cases, which are proved to be necessary and sufficient. Leveraging our conditions, the optimal trajectory is directly constructed in linear complexity of time and space, without evaluating the cost functional explicitly or implicitly. Base on them, a novel trajectory class along with linear-complexity spatial-temporal deformation is designed to meet user-defined objectives in various trajectory planning scenarios.

A. Unconstrained Control Effort Minimization

When constraint $\mathcal{F}$ exists, adjusting the waypoints [28] or control points [33] of a trajectory helps to ensure safety. When constraint $\mathcal{G}$ exists, adjusting the time allocation also helps to enforce physical limits [38]. Therefore, spatial and temporal parameters are both vital to a flexible trajectory representation. A natural problem is to generate a smooth trajectory subject to these parameters. We solve *Linear Quadratic Minimum-Time* (LQMT) problems to generate trajectories from spatial-temporal parameters. Although the LQMT problems have extensive studies and applications, only single-stage problems are considered in the literature [39]–[41]. We study the multi-stage problems where intermediate points and time vector are fixed in advance for multi-piece trajectories. Consider an M-stage control effort minimization without $\mathcal{F}$ and $\mathcal{G}$,

$$\min_{z(t)} \int_{t_0}^{t_M} v(t)^{\mathrm{T}}\mathbf{W}v(t)\mathrm{d}t, \tag{18a}$$

$$s.t.\ \ z^{(s)}(t) = v(t),\ \forall t \in [t_0, t_M], \tag{18b}$$

$$z^{[s-1]}(t_0) = \bar{z}_o,\ z^{[s-1]}(t_M) = \bar{z}_f, \tag{18c}$$

$$z^{[d_i-1]}(t_i) = \bar{z}_i,\ 1 \le i < M, \tag{18d}$$

$$t_{i-1} < t_i,\ 1 \le i \le M. \tag{18e}$$

The time interval $[t_0, t_M]$ is split into M stages by $M+1$ fixed timestamps, with constant boundary conditions $\bar{z}_o, \bar{z}_f \in \mathbb{R}^{ms}$. Intermediate conditions $\bar{z}_i \in \mathbb{R}^{md_i}$ with $d_i \le s$ specify the value of $z(t_i), \dot{z}(t_i), \ldots, z^{(d_i-1)}(t_i)$, where d_i is number of derivatives fixed at t_i. For example, if $z(t)$ is only required to pass a given position at t_i, then $d_i = 1$ because $\bar{z}_i$ contains the 0-order derivative and nothing else.

Existing works focus on the necessary conditions for special cases of (18). In aerial robotics area, the QP formulation [11] and the closed-form one [28] implicitly or explicitly optimize unknown knot derivatives, taking parameterization as *a priori*. This extra computation actually makes them less efficient. In control area, a special case where $d_i = 1$ is also studied in [42] and [43] via controllability Gramian. The result is for general linear systems with possibly nonpolynomial solutions while it is less intuitive considering the computational aspect. These necessary conditions can cause potential degeneracy in trajectory representation and sensitivity, if further parametric optimization on spatial-temporal parameters is needed.

B. Optimality Conditions

We propose necessary and sufficient optimality conditions for (18) with all possible settings of d_i, $\bar{z}_i$, and t_i. Thus, an optimal trajectory can be directly constructed from spatial-temporal parameters. Furthermore, the existence and uniqueness of the optimal trajectory are always guaranteed.

We transform (18) into the Mayer form [23] in which a new state $y \in \mathbb{R}^{ms+1}$ augmented by $\tilde{y} \in \mathbb{R}$ is defined as

$$y = \begin{pmatrix} z^{[s-1]} \\ \tilde{y} \end{pmatrix}. \tag{19}$$

The augmented system $\dot{y} = \hat{f}(y, v)$ has the structure

$$\dot{y} = \begin{pmatrix} \bar{\mathbf{A}} & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & 0 \end{pmatrix} y + \begin{pmatrix} \mathbf{0} \\ v \\ v^{\mathrm{T}}\mathbf{W}v \end{pmatrix}, \tag{20}$$

where

$$\bar{\mathbf{A}} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_{m(s-1)} \\ \mathbf{0}_{m\times m} & \mathbf{0}^{\mathrm{T}} \end{pmatrix} \in \mathbb{R}^{ms\times ms}. \tag{21}$$

We design a running process for the augmented system in M stages, of which the i-th is $\Delta_i = [t_{i-1}, t_i]$. It is worth noting that state switching occurs in this running process. Strictly speaking, the state switching only occurs on $\tilde{y}$ at the beginning of each stage. Denote by $y_{[i]} : \Delta_i \mapsto \mathbb{R}^{ms+1}$ the augmented state trajectory in the i-th stage, which consists of two parts, $z_{[i]}^{[s-1]}$ and $\tilde{y}_{[i]}$. At each timestamp t_i, the state transfers from $y_{[i]}$ to $y_{[i+1]}$, and the part $\tilde{y}$ is reset as

$$\tilde{y}_{[i+1]}(t_i) = 0,\ 0 \le i < M, \tag{22}$$

thus switching the partial state from $\tilde{y}_{[i]}(t_i)$ to 0. The $z^{[s-1]}$ part remains continuous between stages, which means

$$z_{[i]}^{[s-1]}(t_i) = z_{[i+1]}^{[s-1]}(t_i),\ 1 \le i < M. \tag{23}$$

The conditions in (18c) and (18d) are still satisfied, i.e.,

$$z_{[1]}^{[s-1]}(t_0) = \bar{z}_o,\ z_{[M]}^{[s-1]}(t_M) = \bar{z}_f, \tag{24}$$

$$z_{[i]}^{[d_i-1]}(t_i) = \bar{z}_i,\ 1 \le i < M. \tag{25}$$

In this process, the cost functional in (18) is converted into the sum of terminal cost of each stage for the augmented system, i.e., $\sum_{i=1}^{M} \tilde{y}_{[i]}(t_i)$. Therefore, the optimal trajectories for the augmented system and the original one are identical in $z^{[s-1]}$.

We utilize the *Hybrid Maximum Principle* [44] to derive necessary conditions for the optimal solution.

Theorem 1 (Hybrid Maximum Principle). *Let $t_0 < \cdots < t_M$ be real numbers and $\Delta_k = [t_{k-1}, t_k]$. For any collection of absolute continuous functions $x_k : \Delta_k \mapsto \mathbb{R}^{n_k}$, define a vector, $x_\Sigma \in \mathbb{R}^{\bar{n}}$ where $\bar{n} = 2\sum_{k=1}^{M} n_k$, as*

$$x_\Sigma = \left(x_1^{\mathrm{T}}(t_0), x_1^{\mathrm{T}}(t_1), \ldots, x_M^{\mathrm{T}}(t_{M-1}), x_M^{\mathrm{T}}(t_M)\right)^{\mathrm{T}}. \tag{26}$$

On the time interval $\Delta = [t_0, t_M]$ consider the problem

$$\min_{u_k, x_k} J(x_\Sigma), \tag{27a}$$

$$s.t.\ \ \dot{x}_k(t) = f_k(x_k(t), u_k(t)), \tag{27b}$$

$$u_k(t) \in U_k \subseteq \mathbb{R}^{r_k}, \tag{27c}$$

$$\forall t \in \Delta_k,\ k = 1, \ldots, M, \tag{27d}$$

$$\eta(x_\Sigma) = \mathbf{0}, \tag{27e}$$

where $f_k : \mathbb{R}^{n_k} \times \mathbb{R}^{r_k} \mapsto \mathbb{R}^{n_k}$, $J : \mathbb{R}^{\bar{n}} \mapsto \mathbb{R}$ and $\eta : \mathbb{R}^{\bar{n}} \mapsto \mathbb{R}^q$ are continuously differentiable, $u_k : \mathbb{R} \mapsto \mathbb{R}^{r_k}$ are measurable and bounded on the corresponding Δ_k.

Denote an optimal process for (27) by $(x^*(t), u^*(t))$. Then, there exists a collection $(\alpha, \gamma, \psi_1, \ldots, \psi_M)$, where $\alpha \geq 0$, $\gamma \in \mathbb{R}^q$ and $\psi_k : \Delta_k \mapsto \mathbb{R}^{n_k}$ are Lipschitz continuous. It generates M Pontryagin functions

$$H_k(\psi_k, x_k, u_k) = \psi_k^{\mathrm{T}} f_k(x_k, u_k),\ t \in \Delta_k, \tag{28}$$

and a Lagrange function $L(x_\Sigma) = \alpha J(x_\Sigma) + \gamma^{\mathrm{T}} \eta(x_\Sigma)$. The following conditions are satisfied for all $k = 1, \ldots, M$.

- *Nontriviality condition:*

$$(\alpha, \gamma^{\mathrm{T}}) \neq \mathbf{0}; \tag{29}$$

- *Adjoint equations: for almost all $t \in \Delta_k$,*

$$\dot{\psi}_k(t) = -\frac{\partial H_k}{\partial x_k}(\psi_k(t), x_k^*(t), u_k^*(t)); \tag{30}$$

- *Transversality conditions:*

$$\begin{cases} \psi_k(t_{k-1}) = L_{x_k(t_{k-1})}(x_\Sigma^*), \\ \psi_k(t_k) = -L_{x_k(t_k)}(x_\Sigma^*); \end{cases} \tag{31}$$

- *Maximality conditions: for all $t \in \Delta_k$,*

$$\begin{aligned} & H_k(\psi_k(t), x_k^*(t), u_k^*(t)) \\ = & \sup_{u_k \in U_k} H_k(\psi_k(t), x_k^*(t), u_k) \\ = & 0. \end{aligned} \tag{32}$$

Proof. The proof can be directly adapted from Theorem 4 by Dmitruk and Kaganovich [44]. Here we only consider each system f_k to be time-invariant and all intervals Δ_k to be fixed. Besides, no inequality constraints are specified on x_Σ. □

According to Theorem 1, the costate $\psi_{[i]} : \Delta_i \mapsto \mathbb{R}^{ms+1}$ in the i-th stage is defined as

$$\psi_{[i]} = \begin{pmatrix} \lambda_{[i]} \\ \mu_{[i]} \end{pmatrix} = \left(\lambda_{[i]_1}, \lambda_{[i]_2}, \ldots, \lambda_{[i]_s}, \mu_{[i]}\right)^{\mathrm{T}}, \tag{33}$$

where $\mu_{[i]} : \Delta_i \mapsto \mathbb{R}$. $\lambda_{[i]_j} : \Delta_i \mapsto \mathbb{R}^m$ is the j-th map in $\lambda_{[i]} : \Delta_i \mapsto \mathbb{R}^{ms}$. The i-th Pontryagin function of (20) is

$$\begin{aligned} & H_i(\psi_{[i]}, y_{[i]}, v_{[i]}) = \psi_{[i]}^{\mathrm{T}} \hat{f}(y_{[i]}, v_{[i]}) \\ & = \lambda_{[i]}^{\mathrm{T}} \bar{\mathbf{A}} z_{[i]}^{[s-1]} + \lambda_{[i]_s}^{\mathrm{T}} v_{[i]} + \mu_{[i]} v_{[i]}^{\mathrm{T}} \mathbf{W} v_{[i]}. \end{aligned} \tag{34}$$

By applying the adjoint equation (30) for $\mu_{[i]}$, we have $\dot{\mu}_{[i]} = 0$, which means $\mu_{[i]}(t) = \bar{\mu}_i \in \mathbb{R}$ is a constant in Δ_i. Therefore, H_i is always a quadratic function of $v_{[i]}$,

$$H_i(\psi_{[i]}, y_{[i]}, v_{[i]}) = \lambda_{[i]}^{\mathrm{T}} \bar{\mathbf{A}} z_{[i]}^{[s-1]} + \lambda_{[i]_s}^{\mathrm{T}} v_{[i]} + \bar{\mu}_i v_{[i]}^{\mathrm{T}} \mathbf{W} v_{[i]}. \tag{35}$$

By applying the adjoint equation for $\lambda_{[i]}$, we obtain

$$\dot{\lambda}_{[i]} = -\bar{\mathbf{A}}^{\mathrm{T}} \lambda_{[i]}, \tag{36}$$

which is expanded as

$$\dot{\lambda}_{[i]_j} = \begin{cases} \mathbf{0} & \text{if } j = 1, \\ -\lambda_{[i]_{j-1}} & \text{if } 2 \leq j \leq s. \end{cases} \tag{37}$$

It is obvious that $\lambda_{[i]_s}(t)$ is an $s-1$ degree polynomial.

According to maximality conditions (32), the supremum of H_i is always 0 in Δ_i. Thus the positive definiteness of $\mathbf{W}$ implies $\bar{\mu}_i \leq 0$. If $\bar{\mu}_i = 0$, then (35) becomes a linear function of $v_{[i]}$. The zero supremum means that $\lambda_{[i]_s}(t) = \mathbf{0}$ in Δ_i. As the result of (36), $\psi_{[i]}(t) = \mathbf{0}$ holds for all t in Δ_i. In such a case, a contradiction occurs that the nontriviality condition (29) and the transversality conditions (31) cannot be satisfied at the same time. Therefore, $\bar{\mu}_i < 0$ always holds in the whole Δ_i. The optimal control $v_{[i]}^*$ can be obtained from

$$\frac{\partial H_i}{\partial v_{[i]}}(\psi_{[i]}, y_{[i]}^*, v_{[i]}^*) = \lambda_{[i]_s} + 2\bar{\mu}_i \mathbf{W} v_{[i]}^* = \mathbf{0}, \tag{38}$$

i.e.,

$$v_{[i]}^*(t) = -\frac{1}{2\bar{\mu}_i} \mathbf{W}^{-1} \lambda_{[i]_s}(t),\ \forall t \in \Delta_i. \tag{39}$$

Then, $z_{[i]}^*$ produced by a chain of s-integrators from $\lambda_{[i]_s}(t)$, is a $2s-1$ degree polynomial.

To further explore structures of the solution, we generate the Lagrange function using the cost of augmented system along with all constraints in (23), (24) and (25). We have

$$\begin{aligned} L(y_\Sigma) = & \alpha \sum_{i=1}^{M} \tilde{y}_{[i]}(t_i) + \sum_{i=0}^{M-1} \gamma_i \tilde{y}_{[i+1]}(t_i) \\ & + \sum_{i=1}^{M-1} (\zeta_i^{\mathrm{T}}, \sigma_i^{\mathrm{T}})(z_{[i]}^{[s-1]}(t_i) - z_{[i+1]}^{[s-1]}(t_i)) \\ & + \theta_o^{\mathrm{T}}(z_{[1]}^{[s-1]}(t_0) - \bar{z}_o) + \theta_f^{\mathrm{T}}(z_{[M]}^{[s-1]}(t_M) - \bar{z}_f) \\ & + \sum_{i=1}^{M-1} \theta_i^{\mathrm{T}}(z_{[i]}^{[d_i-1]}(t_i) - \bar{z}_i), \end{aligned} \tag{40}$$

where $\gamma_i \in \mathbb{R}$, $\zeta_i \in \mathbb{R}^{md_i}$, $\sigma_i \in \mathbb{R}^{m(s-d_i)}$, $\theta_o \in \mathbb{R}^{ms}$, $\theta_f \in \mathbb{R}^{ms}$ and $\theta_i \in \mathbb{R}^{md_i}$ are all constant coefficients of corresponding constraints, y_Σ is defined as in (26). Following transversality conditions (31), taking the derivative of L w.r.t. y_Σ gives the boundary values of costates $\psi_{[i]}$ and $\psi_{[i+1]}$, i.e.,

$$\lambda_{[i]}(t_i) = -\begin{pmatrix} \zeta_i + \theta_i \\ \sigma_i \end{pmatrix},\ \lambda_{[i+1]}(t_i) = -\begin{pmatrix} \zeta_i \\ \sigma_i \end{pmatrix}, \tag{41}$$

$$\mu_{[i]}(t_i) = \mu_{[i+1]}(t_{i+1}) = -\alpha. \tag{42}$$

Because $\mu_{[i+1]}(t) = \bar{\mu}_{i+1}$ in Δ_{i+1}, we have

$$\bar{\mu}_i = -\alpha,\ 1 \leq i \leq M. \tag{43}$$

Finally, by substituting (36), (41) and (43) into (39), we obtain that the optimal controls of two consecutive stages satisfy

$$v_{[i]}^{*(j)}(t_i) = v_{[i+1]}^{*(j)}(t_i),\ 0 \leq j < (s - d_i). \tag{44}$$

We finally know that the optimal control of the problem (18) is actually $s - d_i - 1$ times continuously differentiable at the timestamp t_i. Accordingly, the optimal state trajectory, consisting of M polynomials with $2s-1$ degree, is indeed $2s - d_i - 1$ times continuously differentiable at t_i.

Now we conclude the conditions derived from both (39) and (44) in the following theorem, which are proved to be necessary and sufficient optimality conditions of (18).

Theorem 2 (Optimality Conditions). *A trajectory, denoted by $z^*(t) : [t_0, t_M] \mapsto \mathbb{R}^m$, is optimal for the problem (18), if and only if the following conditions are satisfied:*

- *The map $z^*(t) : [t_{i-1}, t_i] \mapsto \mathbb{R}^m$ is parameterized as a $2s-1$ degree polynomial for any $1 \leq i \leq M$;*
- *The boundary conditions in (18c);*
- *The intermediate conditions in (18d);*
- *$z^*(t)$ is $\bar{d}_i - 1$ times continuously differentiable at t_i for any $1 \leq i < M$ where $\bar{d}_i = 2s - d_i$.*

Moreover, a unique trajectory exists for these conditions.

Proof. The proof of necessity is evident in the analyses from (33) to (44) that are directly derived from Theorem 1. The proof of sufficiency is sketched below: (a) The first and fourth conditions always determine a linear spline space of dimension $2s + \sum_{i=1}^{M-1} d_i$ for any sequence of d_i; (b) The second and third conditions are shown to form a square coefficient matrix on a basis of the spline space; (c) The matrix is proved to be nonsingular since $t_{i-1} < t_i$ for each i, implying the existence and uniqueness of solution; (d) The existence and uniqueness for the necessary conditions yield their sufficiency. This proof of sufficiency is detailed in Appendix A. □

To further explain the optimality conditions, we take the multi-stage jerk minimization as an example. In this example, the position, velocity and acceleration are states of the jerk-controlled system ($s = 3$). There are intermediate points ($d_i = 1$) that the trajectory should pass through at certain timestamps. The continuity of state only requires the continuity up to acceleration of the minimum-jerk trajectory, while jerk and snap of the optimal trajectory are also continuous everywhere. Accordingly, if we enforce all these continuity conditions, then Theorem 2 guarantees that only one trajectory exists, which is exactly the optimal one.

C. Minimization Without Cost Functional

Theorem 2 provides a direct way to construct the unique optimal trajectory. The computation enjoys linear complexity in time and space. It does not even require explicit or implicit evaluation of the cost functional or its gradient.

Consider an m-dimensional trajectory whose i-th piece is denoted by an $N = 2s-1$ degree polynomial:

$$p_i(t) = \mathbf{c}_i^{\mathrm{T}} \beta(t - t_{i-1}),\ t \in [t_{i-1}, t_i], \tag{45}$$

where $\beta(x) = (1, x, \ldots, x^N)^{\mathrm{T}}$ is the basis and $\mathbf{c}_i \in \mathbb{R}^{2s \times m}$ the coefficients. For simplicity, we use the timeline relative to $t_0 = 0$. The trajectory is described by a coefficient matrix $\mathbf{c} \in \mathbb{R}^{2Ms \times m}$ and a time vector $\mathbf{T} \in \mathbb{R}^M_{>0}$ defined by

$$\mathbf{c} = \left(\mathbf{c}_1^{\mathrm{T}}, \ldots, \mathbf{c}_M^{\mathrm{T}}\right)^{\mathrm{T}},\ \mathbf{T} = (T_1, \ldots, T_M)^{\mathrm{T}}, \tag{46}$$

where T_i means the duration of the i-th piece. Then we have the timestamp $t_i = \sum_{j=1}^{i} T_j$ and the total duration $T = \|\mathbf{T}\|_1$. The M-piece trajectory $p : [0, T] \mapsto \mathbb{R}^m$ is defined by

$$p(t) = p_i(t),\ \forall t \in [t_{i-1}, t_i),\ \forall i \in \{1, \ldots, M\}. \tag{47}$$

To compute the unique solution for (18), we directly enforce optimality conditions on the coefficient matrix $\mathbf{c}$. Denote by $\mathbf{D}_0, \mathbf{D}_M \in \mathbb{R}^{s \times m}$ and $\mathbf{D}_i \in \mathbb{R}^{d_i \times m}$ the specified derivatives at boundaries and intermediate timestamp t_i, respectively. Each column of $\mathbf{D}_i$ is related to one dimension. Then, conditions at t_i are formulated by $\mathbf{E}_i, \mathbf{F}_i \in \mathbb{R}^{2s \times 2s}$:

$$\begin{pmatrix} \mathbf{E}_i & \mathbf{F}_i \end{pmatrix} \begin{pmatrix} \mathbf{c}_i \\ \mathbf{c}_{i+1} \end{pmatrix} = \begin{pmatrix} \mathbf{D}_i \\ \mathbf{0}_{\bar{d}_i \times m} \end{pmatrix}, \tag{48}$$

$$\begin{aligned} \mathbf{E}_i = (&\beta(T_i), \ldots, \beta^{(d_i-1)}(T_i), \\ &\beta(T_i), \ldots, \beta^{(\bar{d}_i-1)}(T_i))^{\mathrm{T}}, \end{aligned} \tag{49}$$

$$\mathbf{F}_i = (\mathbf{0}, -\beta(0), \ldots, -\beta^{(\bar{d}_i-1)}(0))^{\mathrm{T}}. \tag{50}$$

Specially, define $\mathbf{F}_0, \mathbf{E}_M \in \mathbb{R}^{s \times 2s}$ as

$$\mathbf{F}_0 = (\beta(0), \ldots, \beta^{(s-1)}(0))^{\mathrm{T}}, \tag{51}$$

$$\mathbf{E}_M = (\beta(T_M), \ldots, \beta^{(s-1)}(T_M))^{\mathrm{T}}. \tag{52}$$

The linear system for the optimal coefficient matrix is

$$\mathbf{M}\mathbf{c} = \mathbf{b} \tag{53}$$

where $\mathbf{M} \in \mathbb{R}^{2Ms \times 2Ms}$ and $\mathbf{b} \in \mathbb{R}^{2Ms \times m}$ are

$$\mathbf{M} = \begin{pmatrix} \mathbf{F}_0 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{E}_1 & \mathbf{F}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_2 & \mathbf{F}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{F}_{M-1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{E}_M \end{pmatrix}, \tag{54}$$

$$\mathbf{b} = (\mathbf{D}_0^{\mathrm{T}}, \mathbf{D}_1^{\mathrm{T}}, \mathbf{0}_{m \times \bar{d}_1}, \ldots, \mathbf{D}_{M-1}^{\mathrm{T}}, \mathbf{0}_{m \times \bar{d}_{M-1}}, \mathbf{D}_M^{\mathrm{T}})^{\mathrm{T}}. \tag{55}$$

It is essential that the uniqueness in Theorem 2 ensures the nonsingularity of $\mathbf{M}$ for any time vector $\mathbf{T} \succ \mathbf{0}$. Consequently, the unique solution $\mathbf{c}$ can be obtained via linear equation system (53) with a banded matrix $\mathbf{M}$, i.e., a *banded system*. As for a nonsingular banded system, its *Banded PLU Factorization* always exists [45], which can be employed to compute the solution with $O(M)$ time and space complexity [46]. Therefore, without the need of cost functional, the unique solution of problem (18) is obtained in the lowest complexity, by directly applying our optimality conditions.

D. MINCO Trajectories With Spatial-Temporal Deformation

For multicopters, there are often task-specific requirements apart from feasibility, such as the perception quality in active SLAM [47] or the occlusion rate in aerial videography [48]. These user-defined requirements majorly need to flexibly and adaptively deform both the spatial and temporal profile of a trajectory. Therefore, we select the intermediate points and the time vector as two salient parameters in (18). Fortunately, the existence and uniqueness of solution guarantee the smoothness of sensitivity for them. An iterative procedure is then designed to conduct the spatial-temporal deformation with the lowest computation complexity per iteration.

We denote the intermediate points by $\mathbf{q} = (q_1, \ldots, q_{M-1})$ where $q_i \in \mathbb{R}^m$ is a specified 0-order derivative at t_i. Denote by $\mathbf{T} = (T_1, \ldots, T_M)^{\mathrm{T}}$ the time vector where $T_i \in \mathbb{R}_{>0}$. For any pair of $\mathbf{q}$ and $\mathbf{T}$, Theorem 2 naturally determines a trajectory belonging to a class of control effort minimizers, named

MINCO hereafter. The MINCO trajectory class, denoted by $\mathfrak{T}_{\rm MINCO}$, is defined as

$$\mathfrak{T}_{\rm MINCO}=\Big\{p(t):[0,T]\mapsto\mathbb{R}^m\ \Big|\ \mathbf{c}=\mathbf{c}(\mathbf{q},\mathbf{T})\text{ determined by Theorem 2, }\forall\ \mathbf{q}\in\mathbb{R}^{m\times(M-1)},\ \mathbf{T}\in\mathbb{R}^M_{>0}\Big\}.$$

The dimension m, the system order s, initial and terminal conditions are omitted here for brevity. Intuitively, all trajectories in $\mathfrak{T}_{\rm MINCO}$ are compactly parameterized by only $\mathbf{q}$ and $\mathbf{T}$. Evaluating an element in $\mathfrak{T}_{\rm MINCO}$ directly follows our linear-complexity formulation.

We denote any user-defined objective (or constraint) on a trajectory by a C^2 function $\mathcal{K}(\mathbf{c},\mathbf{T})$ with available gradient. This objective on $\mathfrak{T}_{\rm MINCO}$ can be computed as

$$\mathcal{W}(\mathbf{q},\mathbf{T})=\mathcal{K}(\mathbf{c}(\mathbf{q},\mathbf{T}),\mathbf{T}).\tag{56}$$

To accomplish deformation of $\mathfrak{T}_{\rm MINCO}$, the function $\mathcal{W}$ together with its gradient $\partial\mathcal{W}/\partial\mathbf{q}$ and $\partial\mathcal{W}/\partial\mathbf{T}$ are needed for a high-level optimizer to optimize the objective. Obviously, evaluating $\mathcal{W}$ shares the same complexity as evaluating any trajectory in $\mathfrak{T}_{\rm MINCO}$. The key procedure is to compute the gradient. Now we give a linear-complexity scheme to compute $\partial\mathcal{W}/\partial\mathbf{q}$ and $\partial\mathcal{W}/\partial\mathbf{T}$ from the given $\partial\mathcal{K}/\partial\mathbf{c}$ and $\partial\mathcal{K}/\partial\mathbf{T}$. We first rewrite the linear equation system (53) as

$$\mathbf{M}(\mathbf{T})\mathbf{c}(\mathbf{q},\mathbf{T})=\mathbf{b}(\mathbf{q}).\tag{57}$$

Without causing ambiguity, we omit parameters in $\mathbf{M},\mathbf{b},\mathbf{c},\mathcal{K}$ and $\mathcal{W}$ temporarily for simplicity. Any notation involving $\mathbf{c}$ is interpreted as $\mathbf{c}(\mathbf{q},\mathbf{T})$. Denote by $q_{i,j}$ the j-th entry in q_i.

As for the gradient w.r.t. $\mathbf{q}$, we first differentiate both sides of (57) w.r.t. $q_{i,j}$, which gives

$$\frac{\partial\mathbf{c}}{\partial q_{i,j}}=\mathbf{M}^{-1}\frac{\partial\mathbf{b}}{\partial q_{i,j}}.\tag{58}$$

Then,

$$\begin{aligned}\frac{\partial\mathcal{W}}{\partial q_{i,j}}&=\mathrm{Tr}\left\{\left(\frac{\partial\mathbf{c}}{\partial q_{i,j}}\right)^{\mathrm{T}}\frac{\partial\mathcal{K}}{\partial\mathbf{c}}\right\}\\&=\mathrm{Tr}\left\{\left(\mathbf{M}^{-1}\frac{\partial\mathbf{b}}{\partial q_{i,j}}\right)^{\mathrm{T}}\frac{\partial\mathcal{K}}{\partial\mathbf{c}}\right\}\\&=\mathrm{Tr}\left\{\left(\frac{\partial\mathbf{b}}{\partial q_{i,j}}\right)^{\mathrm{T}}\left(\mathbf{M}^{-\mathrm{T}}\frac{\partial\mathcal{K}}{\partial\mathbf{c}}\right)\right\},\end{aligned}\tag{59}$$

where $\mathrm{Tr}\,(\cdot)$ is the trace operation. The definition of $\mathbf{b}(\mathbf{q})$ in (55) implies that $\partial\mathbf{b}/\partial q_{i,j}$ only has a single nonzero entry 1 at its $(2i-1)s+1$ row and j column. Thus, stacking all the resultant scalars gives

$$\frac{\partial\mathcal{W}}{\partial q_i}=\left(\mathbf{M}^{-\mathrm{T}}\frac{\partial\mathcal{K}}{\partial\mathbf{c}}\right)^{\mathrm{T}}e_{(2i-1)s+1},\tag{60}$$

where e_j is the j-th column of $\mathbf{I}_{2Ms}$. Now that we have already conducted the banded PLU factorization for $\mathbf{M}$ when we compute $\mathbf{c}$. We can reuse the factorization to avoid inverting $\mathbf{M}^{\mathrm{T}}$. Define a matrix $\mathbf{G}\in\mathbb{R}^{2Ms\times m}$ as

$$\mathbf{M}^{\mathrm{T}}\mathbf{G}=\frac{\partial\mathcal{K}}{\partial\mathbf{c}}.\tag{61}$$

We only need to compute $\mathbf{G}$ once to obtain $\partial\mathcal{W}/\partial q_i$ for all $1\le i<M$. Denote the factorization of $\mathbf{M}$ as $\mathbf{M}=\mathbf{PLU}$. Specifically, $\mathbf{L}$ is a banded matrix with zero upper bandwidth and all-ones diagonal entries. $\mathbf{U}$ is a banded matrix with zero lower bandwidth and nonzero diagonal entries because of the nonsingularity of $\mathbf{M}$. The pivoting matrix $\mathbf{P}$ simply changes the row order of the operand, satisfying $\mathbf{P}^{\mathrm{T}}\mathbf{P}=\mathbf{I}$. Consequently, the transpose also has a *Banded LUP Factorization* [45]. Specifically, $\mathbf{M}^{\mathrm{T}}=\bar{\mathbf{L}}\bar{\mathbf{U}}\mathbf{P}^{\mathrm{T}}$, where

$$\bar{\mathbf{L}}=\mathbf{U}^{\mathrm{T}}\left(\mathbf{U}\circ\mathbf{I}\right)^{-1},\ \bar{\mathbf{U}}=\left(\mathbf{I}\circ\mathbf{U}\right)\mathbf{L}^{\mathrm{T}},\tag{62}$$

where the inverse is only done for a diagonal matrix and $\circ$ the Hadamard product. Then, $\mathbf{G}$ can be also computed in linear complexity through such a factorization. For convenience, we partition $\mathbf{G}$ into

$$\mathbf{G}=\left(\mathbf{G}_0^{\mathrm{T}},\mathbf{G}_1^{\mathrm{T}},\ldots,\mathbf{G}_{M-1}^{\mathrm{T}},\mathbf{G}_M^{\mathrm{T}}\right)^{\mathrm{T}}\tag{63}$$

such that $\mathbf{G}_0,\mathbf{G}_M\in\mathbb{R}^{s\times m}$ and $\mathbf{G}_i\in\mathbb{R}^{2s\times m}$ for $1\le i<M$. After that, the gradient of $\mathcal{W}$ w.r.t. $\mathbf{q}$ is determined as

$$\frac{\partial\mathcal{W}}{\partial\mathbf{q}}=\left(\mathbf{G}_1^{\mathrm{T}}e_1,\ldots,\mathbf{G}_{M-1}^{\mathrm{T}}e_1\right),\tag{64}$$

where e_1 is the first column of $\mathbf{I}_{2s}$. This operation takes out $M-1$ specific columns in $\mathbf{G}^{\mathrm{T}}$.

As for the gradient w.r.t. $\mathbf{T}$, differentiating both sides of (57) w.r.t. T_i gives

$$\frac{\partial\mathbf{M}}{\partial T_i}\mathbf{c}+\mathbf{M}\frac{\partial\mathbf{c}}{\partial T_i}=\mathbf{0}.\tag{65}$$

Thus,

$$\begin{aligned}\frac{\partial\mathcal{W}}{\partial T_i}&=\frac{\partial\mathcal{K}}{\partial T_i}+\mathrm{Tr}\left\{\left(\frac{\partial\mathbf{c}}{\partial T_i}\right)^{\mathrm{T}}\frac{\partial\mathcal{K}}{\partial\mathbf{c}}\right\}\\&=\frac{\partial\mathcal{K}}{\partial T_i}-\mathrm{Tr}\left\{\left(\frac{\partial\mathbf{M}}{\partial T_i}\mathbf{c}\right)^{\mathrm{T}}\mathbf{M}^{-\mathrm{T}}\frac{\partial\mathcal{K}}{\partial\mathbf{c}}\right\}\\&=\frac{\partial\mathcal{K}}{\partial T_i}-\mathrm{Tr}\left\{\mathbf{G}^{\mathrm{T}}\frac{\partial\mathbf{M}}{\partial T_i}\mathbf{c}\right\}\end{aligned}\tag{66}$$

The banded structure of $\mathbf{M}$ implies that

$$\mathbf{G}^{\mathrm{T}}\frac{\partial\mathbf{M}}{\partial T_i}\mathbf{c}=\mathbf{G}_i^{\mathrm{T}}\frac{\partial\mathbf{E}_i}{\partial T_i}\mathbf{c}_i.\tag{67}$$

Then we obtain the gradient w.r.t. T_i computed as

$$\frac{\partial\mathcal{W}}{\partial T_i}=\frac{\partial\mathcal{K}}{\partial T_i}-\mathrm{Tr}\left\{\mathbf{G}_i^{\mathrm{T}}\frac{\partial\mathbf{E}_i}{\partial T_i}\mathbf{c}_i\right\},\tag{68}$$

where $\partial\mathbf{E}_i/\partial T_i$ can be analytically derived from (49). Computing (68) for every $1\le i\le M$ gives $\partial\mathcal{W}/\partial\mathbf{T}$.

Finally, we finish the computation of $\partial\mathcal{W}/\partial\mathbf{q}$ and $\partial\mathcal{W}/\partial\mathbf{T}$. It can be verified from both (64) and (68) that the gradient propagation is also done in $O(M)$ complexity. As for $\mathcal{K}$, we make no assumption on its concrete form. Actually, the smoothness of $\mathcal{K}$ is not even needed if only the resultant $\mathcal{W}$ is C^2. In other word, the linear-complexity gradient propagation enjoys both efficiency and flexibility. By incorporating it into common optimizers, we can accomplish the spatial-temporal deformation of $\mathfrak{T}_{\rm MINCO}$ for a wide range of planning purposes while maintaining the local smoothness of trajectories.

V. Geometrically Constrained Flight Trajectory Optimization

In this section, we provide a unified framework for flight trajectory optimization with different time regularization $\rho(T)$, spatial constraints $\tilde{\mathcal{F}}$ and continuous-time constraints $\mathcal{G}$. This framework indeed relaxes the original problem into $\mathfrak{T}_{\text{MINCO}}$. The spatial-temporal deformation is utilized to meet various feasibility requirements. Lightweight schemes are specially designed to eliminate geometrical constraints such that the trajectory can be freely deformed. For continuous-time constraints, a time integral penalty functional is proposed to ensure the feasibility without sacrificing the scalability. Finally, our framework transforms the constrained trajectory optimization into a sparse unconstrained one which can be reliably solved.

A. Temporal Constraint Elimination

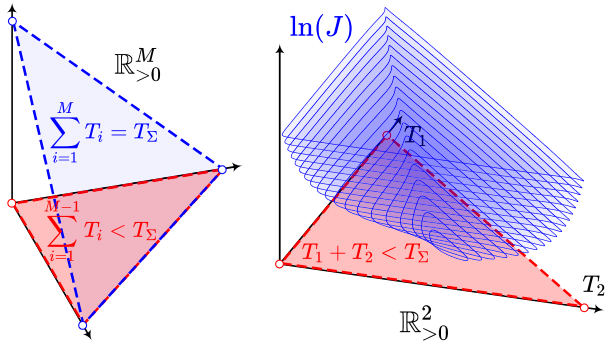


Fig. 3. Left: Domain of J on an M-piece trajectory with total time fixed as T_Σ. The domain is indeed the relative interior of an $(M-1)$-simplex in $\mathbb{R}^M_{>0}$. Right: Contour of $\ln J$ with $M=3$. The function goes to infinity as the time vector approaches the boundary of the open domain in $\mathbb{R}^2_{>0}$.

Deforming MINCO needs standard optimizers that are often designed for Euclidean spaces. However, the trajectory definition and cost functional (11) both restrict the domain of $\mathbf{T}$ to simple manifolds, on which frequent retractions are needed during optimization. We give explicit diffeomorphisms for $\mathbf{T}$ such that surrogate variables are in Euclidean spaces. Thus, common efficient optimizers can be conveniently applied.

For polynomial splines, the control effort in (11) is a function of $\mathbf{c}$ and $\mathbf{T}$, denoted by $J_c(\mathbf{c},\mathbf{T})$. Analytical expressions of J_c, $\partial J_c/\partial \mathbf{c}$, and $\partial J_c/\partial \mathbf{T}$ are available in [28]. Now that $\mathfrak{T}_{\text{MINCO}}$ are polynomial splines with coefficients determined by $\mathbf{c}(\mathbf{q},\mathbf{T})$, the cost functional of (11) can be written as

$$J(\mathbf{q},\mathbf{T}) = J_q(\mathbf{q},\mathbf{T}) + \rho(\|\mathbf{T}\|_1), \tag{69}$$

where J_q is defined as $J_q(\mathbf{q},\mathbf{T}) = J_c(\mathbf{c}(\mathbf{q},\mathbf{T}),\mathbf{T})$. Obviously, computing J_q, $\partial J_q/\partial \mathbf{q}$, and $\partial J_q/\partial \mathbf{T}$ from any provided J_c, $\partial J_c/\partial \mathbf{c}$, and $\partial J_c/\partial \mathbf{T}$ can be done in $O(M)$ complexity, as already shown in deformation of $\mathfrak{T}_{\text{MINCO}}$.

It is natural to optimize $\mathbf{T}$ via $\partial J/\partial \mathbf{T}$. However, $J_q(\mathbf{q},\mathbf{T})$ has its definition over $\mathbf{T}\in\mathbb{R}^M_{>0}$. It becomes unbounded when any T_i approaches zero and no consecutively repeating points appear in $\mathbf{q}$. Besides, ρ_f defined in (13) further restricts the domain of J to $\sum_{i=1}^{M-1} T_i < T_\Sigma$, as shown in Fig. 3.

We use diffeomorphisms to eliminate constraints for ρ_f and ρ_s. Consider the domain of ρ_f in (13),

$$\mathcal{T}_f = \left\{ \mathbf{T}\in\mathbb{R}^M \;\middle|\; \|\mathbf{T}\|_1 = T_\Sigma,\ \mathbf{T}\succ\mathbf{0} \right\}. \tag{70}$$

It is clear that $J(\mathbf{q},\cdot)$ is finite for a nontrivial $\mathbf{q}$ if and only if $\mathbf{T}\in\mathrm{RelInt}(\mathcal{T}_f)$, i.e., the relative interior of $\mathcal{T}_f$.

Proposition 1. *$\mathcal{T}_f$ defined by (70) is diffeomorphic to $\mathbb{R}^{M-1}$. Denote by $\boldsymbol{\tau}=(\tau_1,\ldots,\tau_{M-1})$ an element in $\mathbb{R}^{M-1}$. A C^∞ diffeomorphism is given by the map below for $1\le i<M$:*

$$T_i = \frac{e^{\tau_i}}{1+\sum_{j=1}^{M-1} e^{\tau_j}} T_\Sigma, \quad T_M = T_\Sigma - \sum_{j=1}^{M-1} T_j. \tag{71}$$

By exploiting the explicit diffeomorphism (71), we directly minimize the cost function J over $\mathbb{R}^{M-1}$ via $\boldsymbol{\tau}$, because the domain constraints are satisfied by default.

Optimizing $\boldsymbol{\tau}$ requires gradient propagation. We partition the gradient as $\partial J_q/\partial\mathbf{T} = (g_a^{\mathrm{T}}, g_b)^{\mathrm{T}}$, where $g_a\in\mathbb{R}^{M-1}$ and $g_b\in\mathbb{R}$. Differentiating the layer in (71) yields the gradient of J w.r.t. $\boldsymbol{\tau}$,

$$\frac{\partial J}{\partial \boldsymbol{\tau}} = \frac{(g_a - g_b\mathbf{1})\circ e^{[\boldsymbol{\tau}]}}{1+\|e^{[\boldsymbol{\tau}]}\|_1} - \frac{\left(g_a^{\mathrm{T}} e^{[\boldsymbol{\tau}]} - g_b\|e^{[\boldsymbol{\tau}]}\|_1\right) e^{[\boldsymbol{\tau}]}}{\left(1+\|e^{[\boldsymbol{\tau}]}\|_1\right)^2}, \tag{72}$$

where $e^{[\cdot]}$ is the entry-wise exponential map, and $\mathbf{1}$ an all-ones vector. If an initial guess $\mathbf{T}$ is specified, the corresponding $\boldsymbol{\tau}$ can be computed via the inverse map of the diffeomorphism, given by $\tau_i = \ln(T_i/T_M)$ for $1\le i<M$. As for ρ_s in (12), only $\mathbf{T}\succ\mathbf{0}$ needs to be ensured. It suffices to use $\mathbf{T}=e^{[\boldsymbol{\tau}]}$ as the diffeomorphism between $\mathbb{R}^M$ and $\mathbb{R}^M_{>0}$.

For either ρ_f or ρ_s, we denote the diffeomorphism by $\mathbf{T}(\boldsymbol{\tau})$. Unconstrained optimization on $\boldsymbol{\tau}$ can be directly conducted to minimize $J(\mathbf{q},\mathbf{T}(\boldsymbol{\tau}))$. Although $\mathbf{T}(\boldsymbol{\tau})$ does not preserve convexity, the original cost $J(\mathbf{q},\mathbf{T})$ is already nonconvex as given in (57). Thus, the only concern is whether $\mathbf{T}(\boldsymbol{\tau})$ brings new local minima in the space of $\boldsymbol{\tau}$ or eliminates local minima in the space of $\mathbf{T}$.

Proposition 2. *Denote by $F:\mathbb{D}_{\mathrm{F}}\mapsto\mathbb{R}$ any C^2 function with a convex open domain $\mathbb{D}_{\mathrm{F}}\in\mathbb{R}^N$. Given any C^2 diffeomorphism $\mathbf{G}:\mathbb{R}^N\mapsto\mathbb{D}_{\mathrm{F}}$, define $H:\mathbb{R}^N\mapsto\mathbb{R}$ as $H(y)=F(\mathbf{G}(y))$ for $y\in\mathbb{R}^N$. For any $x\in\mathbb{D}_{\mathrm{F}}$ and $y\in\mathbb{R}^N$ satisfying $x=\mathbf{G}(y)$ or equivalently $y=\mathbf{G}^{-1}(x)$, the following statements always hold:*

- *$\nabla F(x)=\mathbf{0}$ if and only if $\nabla H(y)=\mathbf{0}$;*
- *$\nabla^2 F(x)$ is positive-definite (or positive-semidefinite) at $\nabla F(x)=\mathbf{0}$, if and only if $\nabla^2 H(y)$ is positive-definite (or positive-semidefinite) at $\nabla H(y)=\mathbf{0}$.*

Proof. See Appendix B. □

Proposition 2 confirms that $\mathbf{T}(\boldsymbol{\tau})$ preserves the first/second-order necessary optimality conditions and second-order sufficient optimality conditions [49]. It is also applicable to substitute the exponential map in this subsection with any C^2 diffeomorphism from $\mathbb{R}$ to $\mathbb{R}_{>0}$ for a better numerical condition. In the sense of commonly-used optimality conditions, our constraint elimination does not produce extra spurious local minima or cancel any existing one.

B. Spherical Spatial Constraint Elimination

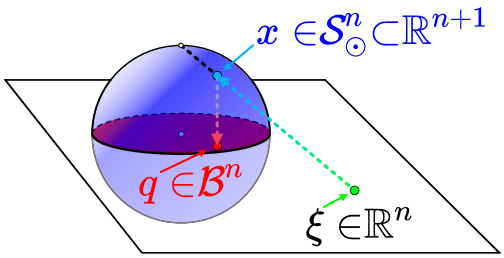


Fig. 4. Inverse stereographic projection f_s maps the Euclidean space $\mathbb{R}^n$ onto a sphere without north pole $\mathcal{S}^n_\odot$ in an $(n+1)$-dimensional space. The orthographic projection f_o maps $\mathcal{S}^n_\odot$ onto an n-dimensional ball $\mathcal{B}^n$. The variable ξ moves freely in $\mathbb{R}^n$ while the transformed variable q stays in $\mathcal{B}^n$. Optimization on ξ becomes unconstrained when q is constrained by a ball.

We enforce motion safety by confining trajectories into the feasible region $\tilde{\mathcal{F}}$. Although $\tilde{\mathcal{F}}$ is nonconvex, it is a union of convex primitives that are sequentially connected. If all pieces have been assigned into these primitives, the safety constraint on each piece becomes convex and thus can be conveniently encoded in $\mathcal{G}$. Owing to the feature of MINCO, the traverse time for every primitive can be directly optimized. Thus, we fix the piece assignment before optimization, rather than resorting to integer variables during optimization [33]. Consequently, intermediate points should be contained by the overlap between primitives, forming inequalities. For Inequality Constrained Problems (ICPs), general methods successively approximate the constraints via additional parameters. However, we aim to apply the constraints directly and efficiently. Therefore, we propose spatial constraint elimination to enforce them exactly, leveraging their geometrical properties.

Consider the constraint $q \in \mathcal{P} \subset \mathbb{R}^n$ where $\mathcal{P}$ is a closed ball. Its dimension satisfies $n \leq m$ since a low-dimensional constraint also exists in $\mathbb{R}^m$. If $\mathcal{P}$ is a closed ball $\mathcal{P}^{\mathcal{B}}$ centered at point o with radius r,

$$\mathcal{P}^{\mathcal{B}} = \left\{ x \in \mathbb{R}^n \;\middle|\; \|x - o\|_2 \leq r \right\}, \tag{73}$$

We utilize a smooth surjection to map $\mathbb{R}^n$ to $\mathcal{P}^{\mathcal{B}}$ such that optimization over $\mathbb{R}^n$ implicitly satisfies the constraint $\mathcal{P}^{\mathcal{B}}$. As illustrated in Fig. 4, the map is a composition of the inverse stereographic projection and the orthographic projection. First, we utilize the inverse stereographic projection to map $\mathbb{R}^n$ to $\mathcal{S}^n_\odot$, where $\mathcal{S}^n_\odot$ is a unit sphere without north pole, i.e.,

$$\mathcal{S}^n_\odot = \left\{ x \in \mathbb{R}^{n+1} \;\middle|\; \|x\|_2 = 1,\ x_{n+1} < 1 \right\}. \tag{74}$$

The inverse stereographic projection f_s is define as

$$f_s(x) = \frac{(2x^{\mathrm{T}}, x^{\mathrm{T}}x - 1)^{\mathrm{T}}}{x^{\mathrm{T}}x + 1} \in \mathcal{S}^n_\odot,\ \forall x \in \mathbb{R}^n. \tag{75}$$

Note that f_s is a diffeomorphism between $\mathbb{R}^n$ and $\mathcal{S}^n_\odot$ [50]. We then project $\mathcal{S}^n_\odot$ from $\mathbb{R}^{n+1}$ back in $\mathbb{R}^n$ to obtain

$$\mathcal{B}^n = \left\{ x \in \mathbb{R}^n \;\middle|\; \|x\|_2 \leq 1 \right\}. \tag{76}$$

The map is described by

$$f_o(x) = (x_1, \ldots, x_n)^{\mathrm{T}} \in \mathcal{B}^n,\ \forall x \in \mathcal{S}^n_\odot, \tag{77}$$

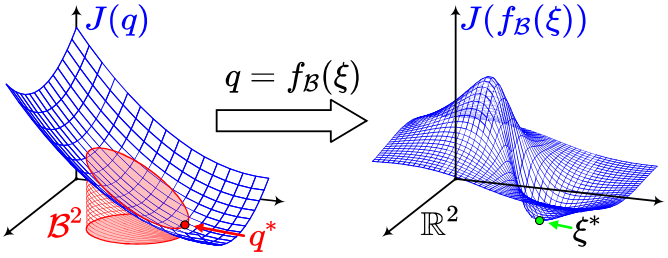


Fig. 5. Constrained minimum q^* of a convex function $J(q)$ within a 2-D ball. Transformed by $f_{\mathcal{B}}$, the resultant function $J(f_{\mathcal{B}}(\xi))$ becomes nonconvex but it preserves the local minimum ξ^* satisfying $q^* = f_{\mathcal{B}}(\xi^*)$ with no additional local minimum introduced.

which is indeed a smooth surjection onto $\mathcal{B}^n$. Each point in $\mathcal{B}^n$, except the center, is paired with two points in $\mathcal{S}^n_\odot$. The composition of f_s, f_o, and a linear transformation, is a smooth surjection:

$$f_{\mathcal{B}}(x) = o + \frac{2rx}{x^{\mathrm{T}}x + 1} \in \mathcal{P}^{\mathcal{B}},\ \forall x \in \mathbb{R}^n. \tag{78}$$

The map $f_{\mathcal{B}}$ introduces a new coordinate, denoted by ξ, such that optimizing ξ over $\mathbb{R}^n$ always satisfies the constraint on q described by $\mathcal{P}^{\mathcal{B}}$. For the i-th intermediate point q_i, denote by ξ_i the corresponding new coordinate. Accordingly, denote by $\boldsymbol{\xi}$ the new coordinate for $\mathbf{q}$. Optimizing $\boldsymbol{\xi}$ requires gradient propagation for $\partial J/\partial \mathbf{q}$. Denote by g_i the i-th entry $\partial J/\partial q_i$ in $\partial J/\partial \mathbf{q}$. Differentiating the layer $f_{\mathcal{B}}$ gives the gradient

$$\frac{\partial J}{\partial \xi_i} = \frac{2r_i g_i}{\xi_i^{\mathrm{T}}\xi_i + 1} - \frac{4r_i(\xi_i^{\mathrm{T}} g_i)\xi_i}{(\xi_i^{\mathrm{T}}\xi_i + 1)^2}. \tag{79}$$

If the optimization needs to start from an initial guess $\mathbf{q}$, the backward evaluation of $\boldsymbol{\xi}$ can be done by using a local inverse of $f_{\mathcal{B}}$, given by ξ_i for $1 \leq i < M$:

$$\xi_i = \frac{r_i - \sqrt{r_i^2 - \|q_i - o_i\|_2^2}}{\|q_i - o_i\|_2^2}(q_i - o_i). \tag{80}$$

Similarly, we analyze influences that the smooth surjection $f_{\mathcal{B}}$ imposes on the constrained local minima in $\mathcal{P}^{\mathcal{B}}$. Although $f_{\mathcal{B}}$ lacks the one-to-one correspondence as diffeomorphisms possess, its components are all well-formed. Firstly, f_o only takes the first n entries of a point. This operation preserves at least the first-order necessary conditions for local minima in either $\mathcal{B}^n$ or $\mathcal{S}^n_\odot$. Secondly, f_s is a diffeomorphism between $\mathcal{S}^n_\odot$ and $\mathbb{R}^n$, thus satisfying Proposition 2. Therefore, we can also confirm that $f_{\mathcal{B}}$ does not produce extra spurious local minima or cancel any existing one. As shown in Fig. 5, the constrained minimum within a 2-D ball is transformed into an unconstrained minimum.

C. Polyhedral Spatial Constraint Elimination

Now we consider the elimination of polyhedral constraints. Specifically, $\mathcal{P}$ is a closed convex polytope $\mathcal{P}^{\mathcal{H}}$ defined by

$$\mathcal{P}^{\mathcal{H}} = \left\{ x \in \mathbb{R}^n \;\middle|\; \mathbf{A}x \preceq b \right\}. \tag{81}$$

where $\mathrm{Int}(\mathcal{P}^{\mathcal{H}}) \neq \varnothing$ according to (15). Common optimization algorithms use the $\mathcal{H}$-representation of $\mathcal{P}^{\mathcal{H}}$ as linear inequality

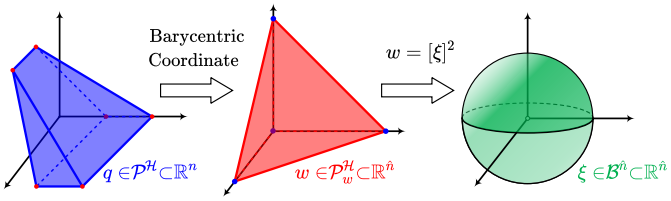


Fig. 6. Transformations on a convex polytope. A convex polytope $\mathcal{P}^{\mathcal{H}}$ with $\hat{n}+1$ vertices is indeed a standard $\hat{n}$-simplex in the barycentric coordinate. The simplex $\mathcal{P}_w^{\mathcal{H}}$ is then the image of an entry-wise square map $[\cdot]^2$ with ball-shaped domain, which can be eliminated as in Fig. 4.

constraints. In our framework, we exploit their geometrical property to eliminate these constraints so that $\mathfrak{T}_{\text{MINCO}}$ can be freely deformed. To achieve this, we use the $\mathcal{V}$-representation of $\mathcal{P}^{\mathcal{H}}$ instead, where any $q \in \mathcal{P}^{\mathcal{H}}$ has a (general) barycentric coordinate, i.e., a convex combination of vertices. To obtain the vertices, we apply the efficient convex hull algorithm [51] to the dual of $\mathcal{P}^{\mathcal{H}}$ based on an interior point calculated by Seidel's algorithm [52]. Note that this procedure produces negligible overhead in our case ($n \leq 4$).

The procedure to eliminate a polytope constraint is illustrated in the Fig. 6. We denote all $\hat{n}+1$ vertices of $\mathcal{P}^{\mathcal{H}}$ by $(v_0, \dots, v_{\hat{n}})$, where $v_i \in \mathbb{R}^n$ for each i. The barycentric coordinate of a point $q \in \mathcal{P}^{\mathcal{H}}$ consists of the weights for these vertices. To obtain a more compact form, define $\hat{v}_i = v_i - v_0$ and $\hat{\mathbf{V}} = (\hat{v}_1, \dots, \hat{v}_{\hat{n}})$, then the position can be calculated as

$$q = v_0 + \hat{\mathbf{V}} w, \tag{82}$$

where $w = (w_1, \dots, w_{\hat{n}})^{\mathrm{T}} \in \mathbb{R}^{\hat{n}}$ is the last $\hat{n}$ entries in the barycentric coordinate. The set of coordinates in convex combinations can also be written as

$$\mathcal{P}_w^{\mathcal{H}} = \left\{ w \in \mathbb{R}^{\hat{n}} \;\middle|\; w \succeq \mathbf{0},\ \|w\|_1 \leq 1 \right\}. \tag{83}$$

The *Main Theorem of Polytope Theory* in [37] confirms the equivalence between $\mathcal{P}_w^{\mathcal{H}}$ and $\mathcal{P}^{\mathcal{H}}$ under (82). The polytope is exactly converted into a standard $(\hat{n}+1)$-simplex by simply adding auxiliary variables and applying a linear map to q. This process does not produce additional nonlinearity in the optimization problem except that the dimension of decision variables is increased. Therefore, we only consider the decision variables on q as the corresponding w hereafter.

The simplex (83) can be eliminated by nonlinear transformations. We first use an entry-wise square map $[\cdot]^2 : \mathbb{R}^{\hat{n}} \mapsto \mathbb{R}^{\hat{n}}$ proposed in [53] to eliminate nonnegativity constraints using $w = [x]^2$. Then, the constraint $\mathcal{P}_w^{\mathcal{H}}$ on w is transformed into a closed unit ball $\mathcal{B}^{\hat{n}}$ on x,

$$\mathcal{B}^{\hat{n}} = \left\{ x \in \mathbb{R}^{\hat{n}} \;\middle|\; \|x\|_2 \leq 1 \right\}. \tag{84}$$

Consequently, we can utilize the smooth surjection $f_{\mathcal{B}}$ in (78) again. The composition of (82), $[\cdot]^2$, and $f_{\mathcal{B}}$ yields a smooth surjection $f_{\mathcal{H}}$ from $\mathbb{R}^{\hat{n}}$ onto $\mathcal{P}^{\mathcal{H}}$:

$$f_{\mathcal{H}}(x) = v_0 + \frac{4\hat{\mathbf{V}}[x]^2}{(x^{\mathrm{T}}x+1)^2} \in \mathcal{P}^{\mathcal{H}},\ \forall x \in \mathbb{R}^{\hat{n}}. \tag{85}$$

A new coordinate ξ is introduced by $f_{\mathcal{H}}$, where any $\xi \in \mathbb{R}^{\hat{n}}$ ensures $q \in \mathcal{P}^{\mathcal{H}}$. The boundary of $\mathcal{P}^{\mathcal{H}}$ is also attainable. Similarly, $\boldsymbol{\xi}$ is the new coordinate for $\mathbf{q}$. Optimizing $\boldsymbol{\xi}$ requires gradient propagation. Denote by g_i the i-th gradient $\partial J/\partial q_i$ in $\partial J/\partial \mathbf{q}$, then differentiating the layer $f_{\mathcal{H}}$ gives

$$\frac{\partial J}{\partial \xi_i} = \frac{8\xi_i \circ \hat{\mathbf{V}}^{\mathrm{T}} g_i}{(\xi_i^{\mathrm{T}}\xi_i+1)^2} - \frac{16 g_i^{\mathrm{T}}\hat{\mathbf{V}}[\xi_i]^2}{(\xi_i^{\mathrm{T}}\xi_i+1)^3}\xi_i. \tag{86}$$

If an initial guess $\mathbf{q}$ is specified, the corresponding $\boldsymbol{\xi}$ can be computed via the local inverse of $f_{\mathcal{H}}$. The barycentric coordinate of each q_i can be obtained using the analytic approach by Warren et al. [54]. After that the analytic local inverses of $[\cdot]^2$ and $f_{\mathcal{B}}(\cdot)$ give us the desired ξ_i. Another flexible way is to directly minimize the squared distance between $f_{\mathcal{H}}(\xi)$ and the given q_i. Both approaches have negligible time consumption but promising results.

The map $[\cdot]^2$ in $f_{\mathcal{H}}$ presents additional nonlinearity into optimization. Fortunately, variable transformation via $[\cdot]^2$ is a special case of the inequality-to-equality conversion [55]. Concretely, the inequality constraints are $-w \preceq \mathbf{0}$. By introducing additional variables x, the equivalent equality constraints are $-w + [x]^2 = \mathbf{0}$, yielding $w = [x]^2$. Such type of constraint conversion is proved to preserve first/second-order necessary conditions and second-order sufficient conditions for ICPs by Bertsekas as provided in Section 4.3 of [55]. We confirm that the additional nonlinearity in $f_{\mathcal{H}}$ does not exclude the desired minimum or produce undesired minimum practically.

Direct constraints on $\mathbf{q}$ are eliminated for either $\mathcal{P}^{\mathcal{B}}$ or $\mathcal{P}^{\mathcal{H}}$ using a smooth surjection $\mathbf{q}(\boldsymbol{\xi})$. We can conduct unconstrained optimization on $\boldsymbol{\xi}$ to minimize $J(\mathbf{q}(\boldsymbol{\xi}), \mathbf{T}(\boldsymbol{\tau}))$ hereafter.

D. Time Integral Penalty Functional

After eliminating direct constraints, $\mathfrak{T}_{\text{MINCO}}$ can be freely deformed to meet the continuous-time constraints $\mathcal{G}$. However, enforcing $\mathcal{G}$ over the entire trajectory involves infinitely many inequalities that cannot be solved by constrained optimization. It further needs temporal discretization that usually produces a large number of decision variables. To preserve the sparsity of trajectory parameterization, we decouple the resolution of constraint evaluation from the number of decision variables. Inspired by the constraint transcription [56], we transform $\mathcal{G}$ into finite constraints by integral of constraint violations.

For a trajectory $p : [0, T] \mapsto \mathbb{R}^m$, we define

$$I_{\mathcal{G}}^k[p] = \int_0^T \max\left[\mathcal{G}(p(t), \dots, p^{(s)}(t)), \mathbf{0}\right]^k \,\mathrm{d}t, \tag{87}$$

where $k \in \mathbb{R}_{>0}$ and $\max[\cdot, \mathbf{0}]^k$ is the composition of the entry-wise maximum and an entry-wise power function. Specifically, smoothing is needed if $k \leq 1$. The functional-type constraint is then equivalent to equality constraints $I_{\mathcal{G}}^k[p] = \mathbf{0}$. Actually, $I_{\mathcal{G}}^k[p]$ is a function of trajectory parameters, which we adopt as penalty terms. If $k = 1$, it forms a nonsmooth but exact penalty. If $k > 1$, it forms a differentiable strictly convex penalty. Thus either $I_{\mathcal{G}}^3[p]$ or a smoothing approximation of $I_{\mathcal{G}}^1[p]$ can be adopted. For simplicity, we utilize $I_{\mathcal{G}}^3[p]$ hereafter unless otherwise specified. There are two reasons for choosing a penalty function method. Firstly, the integral in (87) can only be evaluated numerically, making the constraint approximation

inevitable. Secondly, penalty methods have no requirement on a feasible initial guess which is nontrivial to construct.

We define the time integral penalty functional for $p(t)$ as

$$I_{\mathcal{G}}[p] = \chi^{\mathrm{T}} I_{\mathcal{G}}^{k}[p]. \tag{88}$$

where $\chi \in \mathbb{R}^{n_g}_{\geq 0}$ is a weight vector. Normally, χ should contain large constants. If no constraint is violated, $I_{\mathcal{G}}[p]$ remains zero. Otherwise, if any part on $p(t)$ violates any constraint in $\mathcal{G}$, the penalty functional $I_{\mathcal{G}}[p]$ grows rapidly. By incorporating $I_{\mathcal{G}}[p]$ into the cost functional, continuous-time constraints are enforced within an acceptable tolerance.

Practically, $I_{\mathcal{G}}[p]$ can only be evaluated by quadrature. To conduct the quadrature, we first define a sampled function $\mathcal{G}_\tau : \mathbb{R}^{2s\times m} \times \mathbb{R}_{>0} \times [0,1] \mapsto \mathbb{R}^{n_g}$ as

$$\mathcal{G}_\tau(\mathbf{c}_i, T_i, \tau) = \mathcal{G}\left(\mathbf{c}_i^{\mathrm{T}}\beta(T_i\cdot\tau), \ldots, \mathbf{c}_i^{\mathrm{T}}\beta^{(s)}(T_i\cdot\tau)\right), \tag{89}$$

where $\tau \in [0,1]$ is a normalized stamp. Then the quadrature for $I_{\mathcal{G}}[p]$, denoted by $I : \mathbb{R}^{2Ms\times m}\times\mathbb{R}^{M}_{>0} \mapsto \mathbb{R}_{>0}$, is computed as a weighted sum of the sampled penalty,

$$I(\mathbf{c},\mathbf{T}) = \sum_{i=1}^{M}\frac{T_i}{\kappa_i}\sum_{j=0}^{\kappa_i}\bar{\omega}_j\chi^{\mathrm{T}}\max\left[\mathcal{G}_\tau(\mathbf{c}_i,T_i,\frac{j}{\kappa_i}),\mathbf{0}\right]^k, \tag{90}$$

where κ_i controls the resolution. We choose the trapezoidal rule $(\bar{\omega}_0,\bar{\omega}_1,\ldots,\bar{\omega}_{\kappa_i-1},\bar{\omega}_{\kappa_i}) = (1/2,1,\ldots,1,1/2)$ because of its reliable performance for ill-shaped C^2 integrands in our practice. Intuitively, $I(\mathbf{c},\mathbf{T})$ is a differentiable approximation to $I_{\mathcal{G}}[p]$, whose precision is adjustable through κ_i. The value and gradient at most timestamps can be parallelly computed then directly combined as one.

E. Trajectory Optimization via Unconstrained NLP

Due to $\mathcal{G}$ and $\mathcal{F}$ in (11), the optimal trajectory parameterization is generally hard to know. Unlike traditional methods approximating solutions via a large number of variables [23], we propose to solve a lightweight relaxed optimization via unconstrained NLP, where the spatial-temporal deformation of $\mathfrak{T}_{\mathrm{MINCO}}$ is applied. The relaxation to (11) is defined as

$$\min_{\boldsymbol{\xi},\boldsymbol{\tau}}\ J(\mathbf{q}(\boldsymbol{\xi}),\mathbf{T}(\boldsymbol{\tau})) + I(\mathbf{c}(\mathbf{q}(\boldsymbol{\xi}),\mathbf{T}(\boldsymbol{\tau})),\mathbf{T}(\boldsymbol{\tau})), \tag{91}$$

where J is the time-regularized control effort (69) for $\mathfrak{T}_{\mathrm{MINCO}}$ and I is the quadrature of penalty functional (90). Note that any task-specific requirement, either objectives or constraints, can be combined in (91) without affecting its structure.

To generate trajectories for a flat multicopter, we first parameterize its flat-output trajectory as $\mathfrak{T}_{\mathrm{MINCO}}$. After assigning a fixed number of pieces into each $\mathcal{P}_i$, variable transformations are applied to eliminate all direct constraints. User-defined $\mathcal{G}_{\mathcal{D}}$ are also transformed into $\mathcal{G}$ via Ψ_x and Ψ_u. Finally, we obtain the cost function (91). Apparently, the gradient propagation is derived for all layers except Ψ_x and Ψ_u. One can either apply *Automatic Differentiation* (AD) [57] to Ψ_x and Ψ_u or derive the gradient propagation analytically by following the reverse-mode AD. The efficiency is the same as the flatness map as ensured by *Baur-Strassen Theorem* [58]. The differentiation is only needed for the given flat dynamics once and for all. With available gradient, the relaxation (91) is then solved by the L-BFGS algorithm [59].

VI. Applications

A. Large-Scale Unconstrained Control Effort Minimization

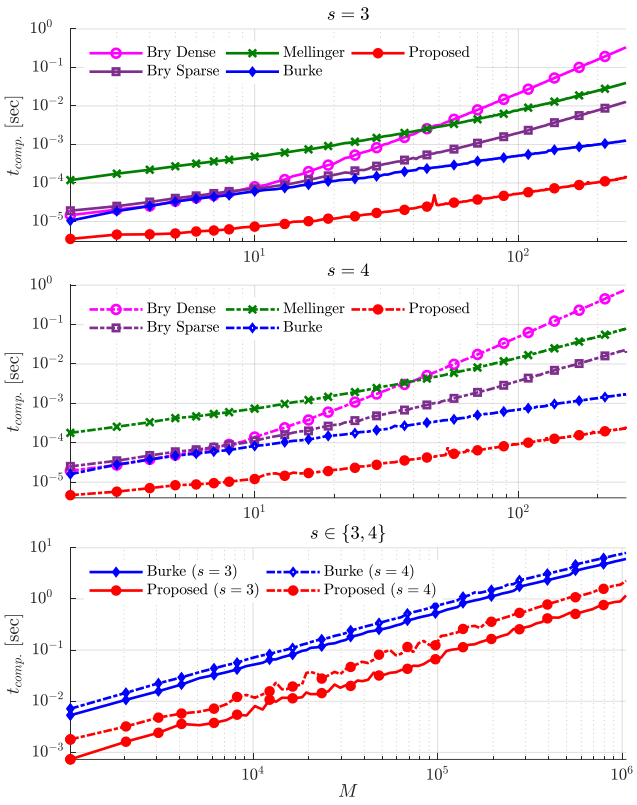


Fig. 7. Computation time $t_{comp.}$ under different piece numbers M. The top and middle figures give the performance for jerk minimization ($s = 3$) and snap minimization ($s = 4$), respectively. The bottom figure shows the efficiency of two linear-complexity schemes for very large-scale problems.

We benchmark several existing schemes over problem (18), including the QP formulation by Mellinger and Kumar [11], the closed-form solution by Bry et al. [28], and the linear-complexity scheme by Burke et al. [60]. We implement all these schemes in C++11 without any explicit hardware acceleration. Mellinger's scheme is implemented using OSQP [61]. Bry's solution is evaluated by both a dense solver and a sparse one [62]. Burke's scheme is re-implemented here for fairness, which is faster than the original one [60]. The benchmark is conducted on an Intel Core i7-8700 CPU under Linux.

The performance is reported in Fig. 7. Both jerk $s = 3$ and snap $s = 4$ are minimized as defined in (18). Mellinger's scheme [11] only performs better than the dense evaluation of Bry's closed-form solution [28] on middle-scale problems ($10^1 < M < 10^3$). Burke's scheme [60] benefits from its linear complexity, thus it can solve large-scale problems ($10^4 < M < 10^6$). Our scheme improves the computation speed by orders of magnitude against the others at any problem scale while retaining $O(M)$ complexity.

In conclusion, our optimality conditions provide a practical way to directly construct the solution of problem (18), which possesses simplicity, efficiency, stability and scalability. The trajectory class $\mathfrak{T}_{\mathrm{MINCO}}$ can serve as a reliable submodule of our optimization framework.

B. Trajectory Generation Within Safe Flight Corridors

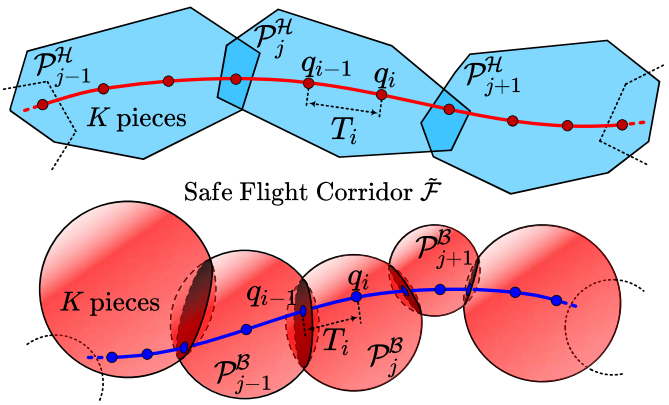


Fig. 8. Piece assignment for a trajectory within different kinds of safe flight corridors in $\mathbb{R}^n$. Each geometrical primitive is assigned with K pieces. An intermediate point q_i is assigned to $\mathcal{P}_{\lceil i/K \rceil} \cap \mathcal{P}_{\lceil (i+1)/K \rceil}$. For ball-shaped corridors, a point is further anchored to an $(n-1)$-dimensional disk if it is assigned to the intersection of two n-dimensional balls.

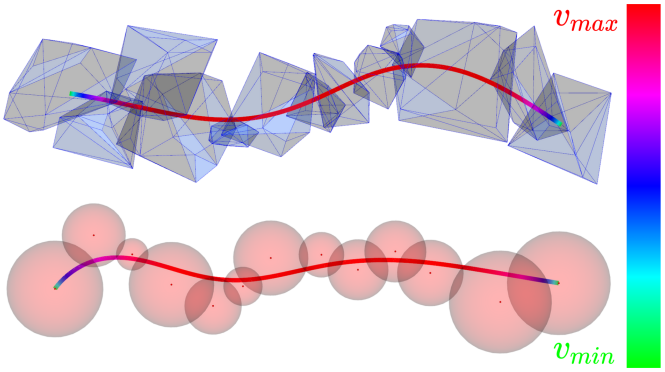


Fig. 9. Optimized trajectories within different kinds of 3-D SFCs. The speed profile is colored according to its magnitude. The proposed method generates smooth trajectories within randomly generated SFCs. The speed persistently attains the maximum even if SFCs are narrow and twisted.

As a special case of problem (11), trajectory generation within 3-D Safe Flight Corridors (SFCs) has been widely adopted in real-world applications such as [31], [63], and [64]. The SFCs are usually generated by the *front end* of a trajectory planning framework as an abstraction of the concerned configuration space, such as the Parallel Convex Cluster Inflation (PCCI) [31], the Regional Inflation by Line Search (RILS) [38], the Safe-Region RRT* Expansion [64], or the Iterative Regional Inflation by Semidefinite programming (IRIS) [65]. We assume that an SFC, either polyhedron-shaped or ball-shaped, is already obtained here as in (14) and (15). Optimizing dynamically feasible trajectories within SFCs is usually taken as a *back end* of such kind of frameworks.

As is illustrated in Fig. 8, we consider two kinds of SFCs. Each convex primitive is assigned with K trajectory pieces, thus $M = M_{\mathcal{P}} K$. The i-th trajectory piece $p_i(t) : [0, T_i] \mapsto \mathbb{R}^3$ is assigned to $\mathcal{P}_{\lceil i/K \rceil}$. The intermediate point assignment of $\mathfrak{T}_{\text{MINCO}}$ is also determined. Applying the constraint elimination, direct constraints on $\mathbf{T}$ and $\mathbf{q}$ are automatically satisfied, such as $\mathbf{T} \in \mathbb{R}_{>0}$ for $\rho_s(T) = k_\rho T$, $\|\mathbf{T}\|_1 < T_\Sigma$ for ρ_f, as

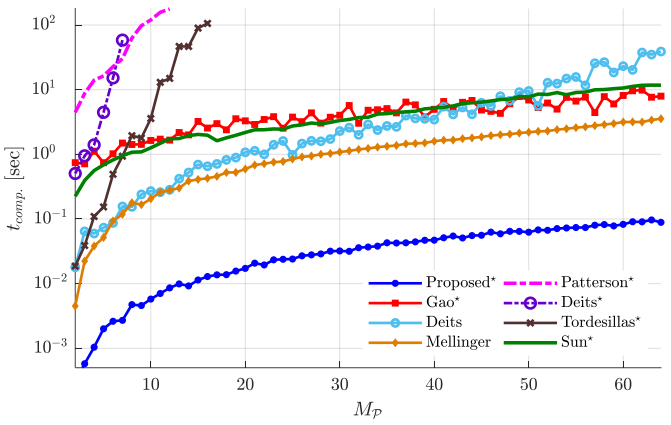


Fig. 10. Benchmark on computation efficiency. The Proposed⋆ one outperforms other methods by orders of magnitudes. Methods from Tordesillas⋆ and Deits⋆ suffer from combinatorial explosion, but they are faster than Patterson⋆ on small-scale problems. Methods not supporting time or interval optimization consume less computation time at the sacrifice of quality.

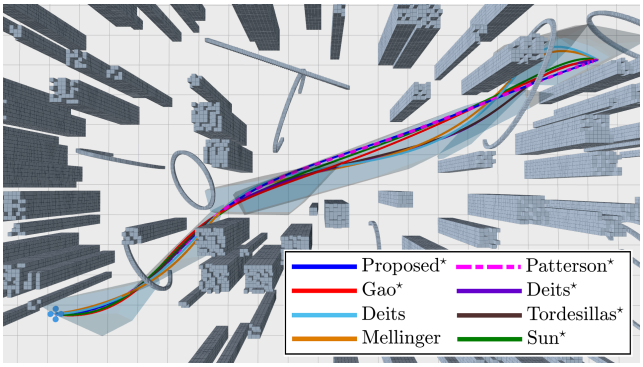


Fig. 11. Geometrical profiles of trajectories generated by different methods in a random environment. The trajectory from the Proposed⋆ one is closer to the ground truth from Patterson⋆ than all other specialized ones.

well as $q_i \in \mathcal{P}_{\lceil i/K \rceil} \cap \mathcal{P}_{\lceil (i+1)/K \rceil}$ for all i. Constraints $\mathcal{G}$ are specified as follows to ensure both safety and dynamic limits:

$$\begin{cases} p_i(t) \in \mathcal{P}_{\lceil i/K \rceil}, & \forall t \in [0, T_i],\ \forall 1 \le i \le M, \\ \|p_i^{(1)}(t)\|^2 \le v_{max}^2, & \forall t \in [0, T_i],\ \forall 1 \le i \le M, \\ \|p_i^{(2)}(t)\|^2 \le a_{max}^2, & \forall t \in [0, T_i],\ \forall 1 \le i \le M, \end{cases} \tag{92}$$

where v_{max} and a_{max} are dynamic limits. Then, the trajectory generation in $\tilde{\mathcal{F}}$ can be accomplished by solving the unconstrained NLP in (91). We show some optimization results in Fig. 9 for randomly generated SFCs. Both the polyhedron-shaped and ball-shaped SFCs are handled.

To further evaluate the performance of our method, we benchmark several existing methods over polyhedron-shaped SFCs. Technical details for all methods are listed as here:

- Proposed⋆: Jerk energy minimization is conducted with either linear time regularization or fixed total time. Constraints in (92) are enforced.
- Patterson⋆ [24]: The LQMT problem of a jerk-controlled system is solved using Gauss pseudospectral method. Each trajectory phase is confined within one polytope. Dynamic limits are enforced through path constraints.
- Gao⋆ [31]: A geometrical curve is optimized via QP formed by jerk energy cost and linear safety constraints

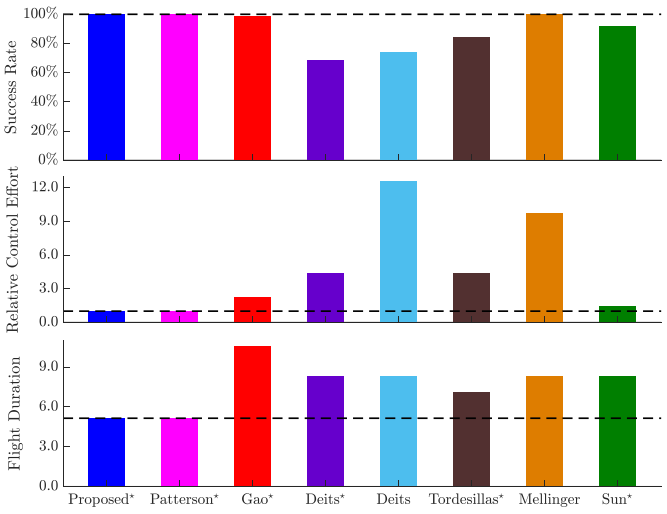
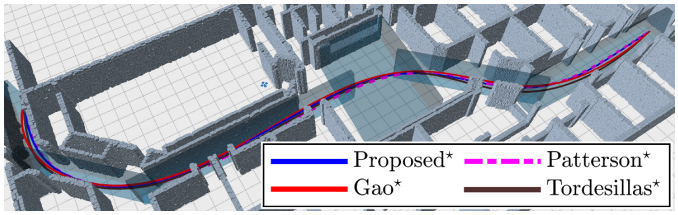
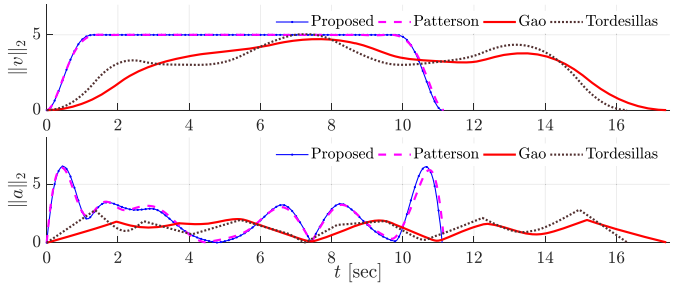


Fig. 12. Benchmark on success rates, relative control effort, and flight durations. Methods from Deits⋆ and Tordesillas⋆ have relatively low success rates because they optimize interval allocation which involves integer variables. Methods from Deits and Mellinger have relatively large control effort because optimization on time or interval allocation is not supported. Note that some methods need preassigned total flight duration.



(a) Trajectories from different methods within a long SFC of the office.



(b) The velocity and acceleration magnitude for different methods.

Fig. 13. Trajectory profiles with large weight on time regularization. Only the Proposed⋆ one and the ground truth from Patterson⋆ generate persistently tight trajectories, considering the continuous-time constraints on norms of derivatives.

on control points of Bézier curves. Its temporal profile is then optimized by an SOCP for TOPP under (92).

- Deits⋆ [30]: The jerk energy and interval allocation of a trajectory is optimized by an MISOCP. Safety constraints and dynamic limits on L_1-norm of trajectory derivatives are exactly enforced through SOS conditions. Each trajectory piece is a 3-degree polynomial.
- Deits: Details are the same as Deits⋆ except that intervals are allocated heuristically. No integer variable exists.
- Tordesillas⋆ [33]: Details are the same as Deits⋆ except that safety is ensured by linear constraints on control points of Bézier curves. An MIQP is solved instead. The total time is determined by a well-designed algorithm.
- Mellinger [11]: A trajectory is optimized in a QP formed by quadratic cost on jerk and linear safety constraints on sampled points. Its time allocation is generated with trapezoidal velocity profiles. Dynamic limits in (92) are enforced by time scaling [38].
- Sun⋆ [35]: A trajectory is optimized in a bilevel framework. The low-level QP is exact the same as Tordesillas⋆ except that 6-degree polynomials are used. Its time allocation is optimized in the upper level optimization using analytical sensitivity of the low-level one.

A method is asterisked if it supports optimizing time allocation or interval allocation. Dynamic limits are treated as the same for either L_1-norm or L_2-norm. Thus, constraints are indeed much tighter on methods from the Proposed⋆, Gao⋆, Mellinger, and Patterson⋆, that restrict L_2-norm of derivatives. As for total time, Deits⋆ and Sun⋆ need preassigned values, thus, we set their total time using trapezoidal velocity profiles. Patterson⋆ handles the original problem directly, taking advantage of the exponential convergence of global collocation [24]. Therefore, we take its trajectory as the ground truth.

The benchmark is conducted in randomly generated environments, one of which is shown in Fig. 11. The corridor size $M_{\mathcal{P}}$ ranges from 2 to 64 where 10 SFCs are generated for each size. The facet number of $\mathcal{P}_i^{\mathcal{H}}$ ranges from 8 to 30. We set $K = 1$, $k_\rho = 1024.0$, $v_{max} = 5.0m/s$, $a_{max} = 7.0m/s^2$, $\kappa_i = 16$, the timeout as 3 minutes, and the relative tolerance as 10^{-4}. Static boundary conditions are assumed. As for programs, methods from the Proposed⋆ and Mellinger are both implemented in C++11 with a single thread for sequential computing. The general-purpose solver [24] is directly adopted for Patterson⋆. A C++11 re-implementation of the original MATLAB one [30] is adopted for both Deits⋆ and Deits. Methods from Gao⋆, Tordesillas⋆, and Sun⋆ are taken from their open-source implementations. Besides, the commercial solver Gurobi [66] is used by Deits⋆, Deits, and Tordesillas⋆ with 6 threads enabled for parallel computing. The commercial solver MOSEK [67] is used by both Gao⋆ and Sun⋆.

The computation efficiency is provided in Fig. 10. Clearly, Deits⋆ and Tordesillas⋆ have to optimize integer variables, thus possessing approximately exponential complexity as $M_{\mathcal{P}}$ grows. Nonetheless, Tordesillas⋆ achieves acceptable performance for small $M_{\mathcal{P}}$ by using a more conservative but easier constraints than Deits⋆. Methods from Deits and Mellinger achieve satisfactory performance by tackling time allocation or interval allocation heuristically. Methods from Gao⋆ and Sun⋆ performs well in their scalability while the overhead for small $M_{\mathcal{P}}$ does not suit real-time applications. The method from Patterson⋆ suits offline scenarios where computation time is far less important than solution quality. The Proposed⋆ method improves the speed by more than an order of magnitude, while retaining optimization on time allocation.

The geometrical profile of trajectories is provided in Fig. 11. Methods that do not optimize time or interval allocation are more likely to deviate from the ground truth. Trajectories by Deits⋆ and Tordesillas⋆ also deviate a lot from the ground

(a) Hardware settings of the vehicle. (b) An onboard camera image.

Fig. 14. Left: Our autonomous multicopter equipped with an onboard computer and a LiDAR. Right: A snapshot of the first person view during our high-speed flight experiment in a garage.

truth because of the limited resolution of intervals. The success rates, relative control effort, and flight durations are all given in Fig. 12. Interval allocation based methods have relatively low success rates. All control effort are normalized by that of the Proposed* one, whose total time is fixed accordingly for fairness. Clearly, heuristic time or interval allocation causes relatively high control effort. Besides, the flight duration from the Proposed* method is the closest to the ground truth.

To explore the temporal profile, we also test four complete methods in a long-distance flight as shown in Fig. 13. The trajectory from Gao* is less aggressive than the others. The trajectory from Tordesillas* has discontinuous jerk since 3-degree polynomials are used. The results from the Proposed* one have nearly the same quality as the ground truth. Profiting from the effectiveness of the penalty functional, our method can also achieve the maximum speed persistently.

In simulations, our method achieves comparable trajectory quality to the collocation based method [24] in both the geometrical and temporal profile, while having superior computational speed against all benchmarked ones.

We conduct experimental validation of our framework by enabling high-speed autonomous flights of a multicopter in an underground garage. All computations are performed by an onboard computer with an Intel Core i7-8550U CPU, which is shown in Fig. 14. We utilize FAST-LIO2 [68] for highly robust LiDAR-based localization. Polyhedron-shaped safe flight corridors are generated by following [31]. Our method generates a $343.57m$ global trajectory in only $0.29s$ in the first track. The planning results are provided in Fig. 1(c). We believe this computation time validates our framework's efficiency even for long-distance trajectory planning. In this experiment, the vehicle speed reaches $12.0m/s$ while ensuring its safety among obstacles and keeping a low thrust-to-weight ratio. We further compare planning results for different parameters on v_{max}. It turns out that our method can always squeeze the capability of v_{max} and a_{max} if k_ρ is large. More details about this experiment are given in the attached multimedia.

C. SE(3) *Motion Planning in Quotient Space*

In dense obstacle environments, safe motions often do not exist for narrow spaces unless a multicopter agilely adjusts its attitude to avoid collisions. Therefore, we consider SE(3) motion planning in our framework. An important property for planning in SE(3) as a manifold with structure $\mathbb{R}^3 \times \mathrm{SO}(3)$ is the necessary condition that a feasible pose for a rigid body at least contains a feasible translation for a dimensionless point. The subspace $\mathbb{R}^3$ is referred to as a *Quotient Space* [69]. Exploiting such a quotient-space decomposition [70], we consider the rotational safety based on a translational trajectory, instead of handling them jointly. Therefore, we can relax assumptions for (14) such that $\tilde{\mathcal{F}}$ is just a free region in the quotient space without considering multicopter's actual size.

We consider simplified quadcopter dynamics whose configuration is defined by its translation p and rotation $\mathbf{R}$:

$$\begin{cases} \dot{p} = v, \\ \bar{m}\dot{v} = -\bar{m}\bar{g}e_3 + \mathbf{R}\tilde{f}e_3, \\ \dot{\mathbf{R}} = \mathbf{R}\hat{\omega}. \end{cases} \tag{93}$$

where e_i is the i-th column of $\mathbf{I}_3$, $\bar{g}$ the gravitational acceleration, $\tilde{f}$ the thrust, ω the body rate input, and $\bar{m}$ the vehicle mass. The hat map $\hat{\cdot}: \mathbb{R}^3 \mapsto \mathbb{R}^{3\times 3}$ is defined by $\hat{a}b = a \times b$ for all $a, b \in \mathbb{R}^3$. Moreover, we model the geometrical shape of a symmetric multicopter as its outer Löwner-John ellipsoid [37],

$$\mathcal{E}(t) = \left\{ \mathbf{R}(t)\mathbf{Q}x + p(t) \;\middle|\; \|x\|_2 \le 1 \right\} \tag{94}$$

where $\mathbf{Q} = \mathrm{Diag}\{r_e, r_e, h_e\}$. r_e and h_e are the radius and the height of multicopter, respectively.

A feasible motion satisfies the safety and dynamic limits. By safety we mean $\mathcal{E}(t) \subset \tilde{\mathcal{F}},\ \forall t \in [0, T]$, where T is the total time of the motion. However, this safety constraint is indeed hard to enforce. We further make an assumption on $\mathcal{F}$ that all $\mathcal{P}_i^{\mathcal{H}}$ or their intersections are able to contain at least one ellipsoid of the multicopter. This assumption can be reasonably satisfied when $\tilde{\mathcal{F}}$ is generated incrementally. As a result, we can ensure safety through

$$\forall t \in [0, T],\ \exists 1 \le i \le M_p,\ s.t.\ \mathcal{E}(t) \subset \mathcal{P}_i^{\mathcal{H}}. \tag{95}$$

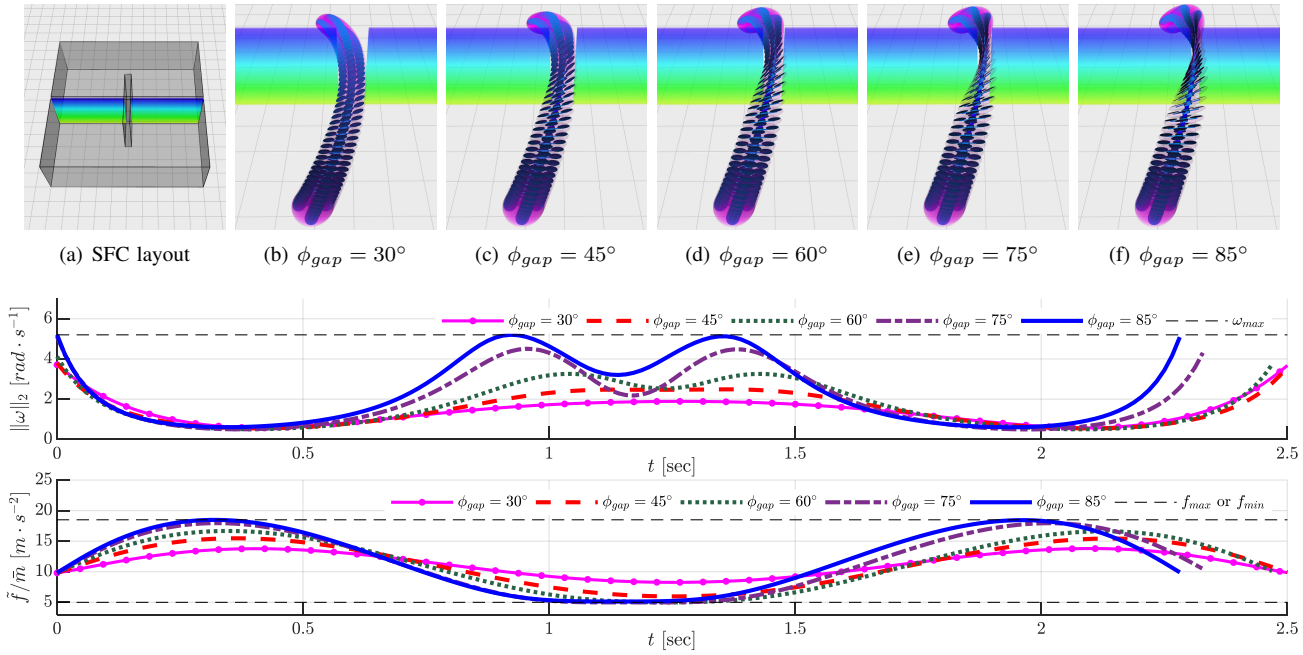
By dynamic limits we mean the velocity, thrust and body rate should have reasonable magnitude,

$$\begin{cases} \|p^{(1)}(t)\|^2 \le v_{max}^2, & \forall t \in [0, T], \\ f_{min} \le \tilde{f}(t) \le f_{max}, & \forall t \in [0, T], \\ \|\omega(t)\|_2^2 \le \omega_{max}^2, & \forall t \in [0, T]. \end{cases} \tag{96}$$

Given a quotient-space trajectory $p(t) : [0, T] \mapsto \mathbb{R}^3$, state-control trajectories of p, v, $\mathbf{R}$, and ω are all algebraically computed by flatness maps Ψ_x and Ψ_u of the dynamics (93). The concrete forms of the algebraic maps are detailed in [11] with fixes on the body rate [14] for simple quadcopters thus are omitted here. Consequently, the entire SE(3) trajectory is also obtained. Denote by $\mathbf{R}(t)$ its rotational part. To generate $p(t)$ in $\tilde{\mathcal{F}}$, we follow the methodology of our previous experiment but with different constraints here.

The i-th trajectory piece $p_i(t) : [0, T_i] \mapsto \mathbb{R}^3$ is assigned to the polytope $\mathcal{P}_j^{\mathcal{H}}$ defined in (17) with $j = \lceil i/K \rceil$. We denote by $\mathcal{E}_i(t)$ the ellipsoid induced by $p_i(t)$ and the corresponding $\mathbf{R}_i(t)$ as is defined in (94). As proposed by Wu et. al. [71], ensuring safety by confining the vehicle ellipsoid in a polyhedron also has an analytical form. Specifically,

$$\mathcal{E}_i(t) \in \mathcal{P}_j^{\mathcal{H}},\ j = \lceil i/K \rceil,\ \forall t \in [0, T_i], \tag{97}$$



(a) SFC layout (b) $\phi_{gap} = 30°$ (c) $\phi_{gap} = 45°$ (d) $\phi_{gap} = 60°$ (e) $\phi_{gap} = 75°$ (f) $\phi_{gap} = 85°$

(g) The magnitude of angular velocity and the normalized thrust for different SE(3) trajectories.

Fig. 15. SFC layout for a narrow gap, SE(3) trajectories under different widths of gaps, and control inputs for different motions. As the gap becomes narrower, larger angular rates and higher thrust are needed for a safe flight. The proposed method persistently enforces limits on these control inputs under different settings while retains millisecond-level computation time.

is equivalent to

$$\left[[\mathbf{A}_j \mathbf{R}_i(t)\mathbf{Q}]^2 \mathbf{1}\right]^{\frac{1}{2}} + \mathbf{A}_j p_i(t) - b_j \preceq \mathbf{0}, \tag{98a}$$

$$j = \lceil i/K \rceil,\ \forall t \in [0, T_i], \tag{98b}$$

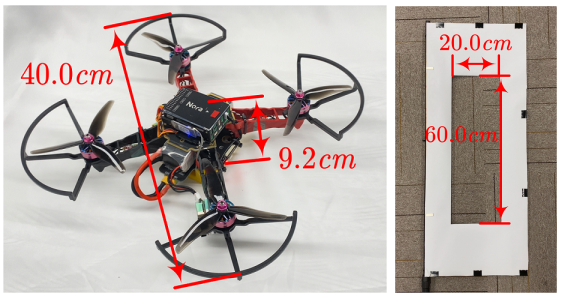
where $\mathbf{1}$ is an all-ones vector with an appropriate length, $[\cdot]^2$ and $[\cdot]^{\frac{1}{2}}$ are entry-wise square and square root, respectively. Finally, we obtained the state-control constraint $\mathcal{G}_{\mathcal{D}}$ in (8) for the considered dynamics in (93). We choose to minimize $s = 3$ because it is the highest derivative order for flatness of (93) and also helpful in smoothing the angular rate.

We validate our framework in simulations where a relatively large quadcopter is required to fly through a narrow gap with much smaller width as shown in Fig. 15. The settings are $r_e = 0.5m$, $h_e = 0.1m$, $f_{min}/\bar{m} = 5.0m/s^2$, $f_{max}/\bar{m} = 18.5m/s^2$, $v_{max} = 6.5m/s$, and $\omega_{max} = 5.2rad/s$. Intuitively, the quadcopter can only achieve no more than 1 revolution per second (rps), making it less agile than small quadcopters [72] that can achieve $5rps$. The computation times, required roll angles, and SE(3) motions for different d_{gap} are shown in the Table I and Fig. 15(b)-15(f).

TABLE I
COMPUTATION TIMES AND ROLL ANGLES FOR DIFFERENT GAPS

d_{gap}	$0.88m$	$0.76m$	$0.60m$	$0.40m$	$0.25m$
ϕ_{gap}	$30°$	$45°$	$60°$	$75°$	$85°$
$t_{comp.}$	$4.7ms$	$4.4ms$	$6.0ms$	$6.6ms$	$7.4ms$

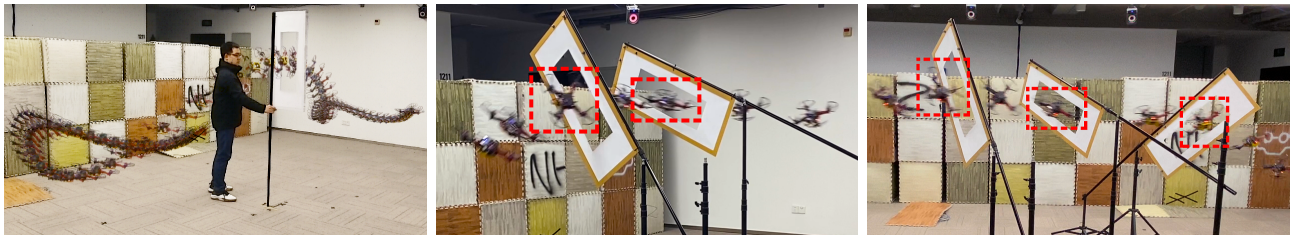
As the gap becomes narrower, the required roll angle becomes larger and the feasible space becomes smaller in view of dynamic limits. Our method is still able to find all the feasible motions. The superior computation speed makes it possible to solving SE(3) planning at a high frequency (at least $100Hz$). Constraint functions are visualized in Fig. 15(g). The body rate and thrust satisfy dynamic limits all the time. The continuous-time tightness of f_{min} for $\phi_{gap} \in \{60°, 75°, 85°\}$ shows the effectiveness of our penalty functional.



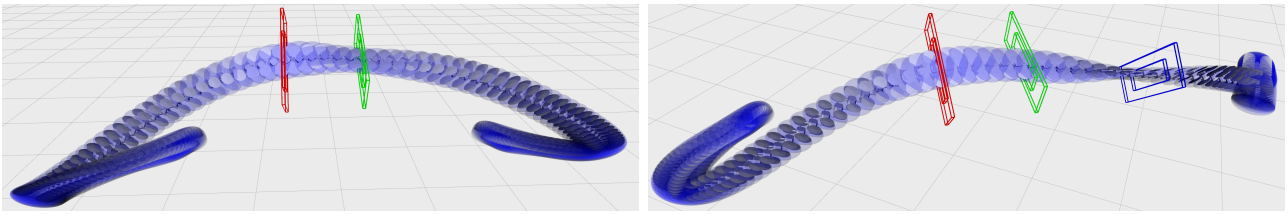
(a) The custom-made quadcopter. (b) The window.

Fig. 16. Sizes of the quadcopter and the narrow window.

We evaluate the performance of our planner in a real-world experiment where a quadcopter flies through several narrow windows. Sizes of the quadcopter and windows are given in Fig. 16. The quadcopter weights $794.2g$. The safety margin of the short side is only $5.4cm$, implying that the feasible motion space is extremely small. The settings are $r_e = 20.0cm$, $h_e = 4.6cm$, $v_{max} = 4.0m/s$, $f_{min}/\bar{m} = 3.0m/s^2$, $f_{max}/\bar{m} = 18.0m/s^2$, $\omega_{max} = 6.0rad/s$, and $K = 2$. The flying space is a restricted volume of $6.5 \times 6.0 \times 2.0m^3$. All poses of narrow windows and the quadcopter are provided by a motion



(a) The interactive scenario. (b) Flying through two consecutive windows. (c) Flying through three consecutive windows.



(d) SE(3) planning for two windows. (e) SE(3) planning for three windows.

Fig. 17. Experiment results for three SE(3) planning scenarios. The Fig. 17(a) gives a snapshot for the interactive scenario. Fig. 17(b)-17(c) show two snapshots for real flights through consecutive windows. Fig. 17(d)-17(e) show corresponding SE(3) trajectories generated by the proposed method.

capture system running at $100Hz$. The obstacle-free region $\tilde{\mathcal{F}}$ is geometrically computed for multiple narrow windows in the free volume. The planner is run on an offboard computer where a human operator arbitrarily chooses the goal position. We adopt the control algorithm by Faessler et al. [14] for onboard SE(3) trajectory tracking.

The first scenario contains consecutive windows with roll angles ranging from 30° to 90°. The quadcopter has to fly through them and reach a randomly selected goal as shown in Fig. 17(b)-17(e). The second scenario is an interactive one where a human operator randomly holds a narrow window for real-time planning as given in Fig. 17(a). The third scenario requires the quadcopter persistently fly back and forth through multiple windows for a long duration as shown in Fig. 1(b) and Fig. 1(d). Our planner guides the quadcopter to fly back and forth through windows for about $20.0s$ while ensuring the safety and physical limits all the time. More details about this experiment are given in the attached multimedia.

In this experiment, the short distance between consecutive windows, the small acceleration/deceleration space, and the limited vehicle maneuverability are challenges that our planner must confront. We believe that these results constitute a strong evidence for its constraint fidelity, motion quality, computation efficiency, and robustness. However, we do observe the limitation of optimization-based methods. For example, if two 90° windows are asymmetrically placed, a multicopter has to pass them in sequence. Each window only allows two roll angles $\pm 90^\circ$. The combinations are 4 locally optimal maneuvers but only one can be the global optimum. Thus, the other three are shallow local minima inevitable for local methods.

VII. Discussion and Conclusion

A. Extensions

Profiting from the flexibility and efficiency, our framework has many applicative and algorithmic extensions. First, no assumption is ever made on concrete forms of vehicle dynamics and $\mathcal{G}_\mathcal{D}$. More accurate dynamics such as the rotor drag [14] can be adopted to fully exploit physical limits via real-time high-fidelity planning and control. Time-dependent constraints for moving obstacles can also be supported by $\mathcal{G}_\mathcal{D}$. Second, our framework is inherently parallelizable to further squeeze its performance. Computation-demanding operations on $I_\mathcal{G}[p]$ are independent at each timestamp, thus parallelization can effectively speedup our optimization. Moreover, it is possible to extend our methodology to other vehicle types whose flat-output space overlaps the configuration space. An example is the fixed-wing aircraft in [28] whose flights are mainly restricted by the trajectory curvature. Bry et al. propose Dubins–Polynomial trajectories [28] for this restriction while the curvature constraint is a special case of $\mathcal{G}$ for MINCO.

To demonstrate the extendibility, we apply our framework to a swarm of multicopters to enable their autonomous navigation in unknown environments. All details of the formulation (11) and real-world flights are given in a technical report [73].

B. Limitations

Our framework, like most optimization-based ones, focuses on local solutions of trajectory planning, thus suffering from shallow local minima. This can be alleviated by interleaving sampling-based or graph-search-based strategies into our framework, as proposed in [74]–[76]. A major limitation of the framework originates from MINCO itself. If $\mathcal{G}$ exist, optimal solutions cannot in general be represented by polynomial splines, let alone MINCO. Thus optimizing MINCO is just a relaxation to the original problem. However, our results show that MINCO can still represent high-quality solutions comparable to the ground truth, but with several orders of magnitudes faster computing. There are also limitations caused by the penalty functional. To achieve zero constraint violations, an unbounded smoothing factor or penalty weight and an unbounded quadrature resolution are both required. However, small constraint violations are empirically acceptable for

multicopter navigation. As a reward, this method does not need initial feasible guesses.

C. Conclusion

In this article, we proposed a flexible multicopter trajectory planning framework powered by several core features, such as the MINCO trajectory based on our optimality conditions, constraint elimination schemes based on smooth maps, the penalty functional method based on constraint transcription, and the backward differentiation of the flatness maps from flat outputs. All these components enjoy the efficiency and generality originating from low complexity and less preliminary assumptions. We performed extensive benchmarks against many kinds of multicopter trajectory planning methods to show the speedup over orders of magnitude and the top-level solution quality. A variety of applications demonstrated the versatility of our framework. We also presented further discussions about several unlisted applications or extensions as future work.

VIII. Acknowledgment

The authors would like to thank Shaohui Yang for his profound insight into the experiment design and applications of this framework, Hongkai Ye and Yuwei Wu for their help in the benchmark, and Yuman Gao, Tiankai Yang, and Neng Pan for the hardware platform for high-speed flights.

Appendix

A. Proof of Sufficiency in Theorem 2

Proof. We consider the space of M-piece polynomial $2s$-order splines defined over $[t_0, t_M]$ where consecutive pieces on any $x : [t_0, t_M] \mapsto \mathbb{R}$ satisfy $x_{i-1}^{(j)}(t_i) = x_i^{(j)}(t_i)$ for $0 \le j < \bar{d}_i$ and $1 \le i < M$. In (18), $d_i \le s$ holds for each i. For brevity, we define $D_{i,j}$ as $D_{i,j} = i \cdot s + \sum_{k=1}^{j} d_k$. According to Theorem 4.4 in [77], this spline space is actually a linear space of dimension $\bar{D} = D_{2,M-1}$.

Moreover, an explicit basis of the space exists. Based on the original partition $t_0 < t_1 < \cdots < t_M$, we define an *extended partition* $\bar{t}_1 \le \bar{t}_2 \le \cdots \le \bar{t}_{\bar{M}}$ of length $\bar{M} = D_{4,M-1}$ as

$$\bar{t}_i = \begin{cases} t_0 & \text{if } 1 \le i \le D_{2,0}, \\ t_j & \text{if } D_{2,j-1} < i \le D_{2,j}, \\ t_M & \text{if } D_{2,M-1} < i \le \bar{M}. \end{cases} \tag{99}$$

Based on this extended partition, Theorem 4.9 in [77] explicitly constructs $\bar{D}$ functions $\{B_i(t) : [t_0, t_M] \mapsto \mathbb{R}\}_{i=1}^{\bar{D}}$ which form a basis for the considered spline space.

Now we consider (18c) and (18d) in the spanned linear space. These conditions specify derivative values on timestamps of the original partition to be interpolated by the basis $\{B_i(t)\}_{i=1}^{\bar{D}}$. We only needs the specified orders along with their timestamps instead of the specified derivative values. Denote by τ_i the i-th specified timestamps, where

$$\tau_i = \begin{cases} t_0 & \text{if } 1 \le i \le D_{1,0}, \\ t_j & \text{if } D_{1,j-1} < i \le D_{1,j}, \\ t_M & \text{if } D_{1,M-1} < i \le \bar{D}. \end{cases} \tag{100}$$

Denote by ν_i the specified order at τ_i, written as

$$\nu_i = \begin{cases} i - 1 & \text{if } 1 \le i \le D_{1,0}, \\ i - 1 - D_{1,j-1} & \text{if } D_{1,j-1} < i \le D_{1,j}, \\ i - 1 - D_{1,M-1} & \text{if } D_{1,M-1} < i \le \bar{D}. \end{cases} \tag{101}$$

Then, the conditions (18c) and (18d) generate a linear equation system on the basis, whose coefficient matrix is

$$\mathbf{B} = \begin{pmatrix} B_1^{(\nu_1)}(\tau_1) & B_2^{(\nu_1)}(\tau_1) & \cdots & B_{\bar{D}}^{(\nu_1)}(\tau_1) \\ B_1^{(\nu_2)}(\tau_2) & B_2^{(\nu_2)}(\tau_2) & \cdots & B_{\bar{D}}^{(\nu_2)}(\tau_2) \\ \vdots & \vdots & \ddots & \vdots \\ B_1^{(\nu_{\bar{D}})}(\tau_{\bar{D}}) & B_2^{(\nu_{\bar{D}})}(\tau_{\bar{D}}) & \cdots & B_{\bar{D}}^{(\nu_{\bar{D}})}(\tau_{\bar{D}}) \end{pmatrix}. \tag{102}$$

It is obvious that $\mathbf{B}$ is a square matrix for any possible solution to Theorem 2 in each dimension.

According to Theorem 4.67 in [77], $\mathbf{B}$ is nonsingular if and only if

$$\tau_i \in \delta_i = \begin{cases} [\bar{t}_i, \bar{t}_{i+2s}) & \text{if } \nu_i + \alpha_i - 2s \ge 0, \\ (\bar{t}_i, \bar{t}_{i+2s}) & \text{if } \nu_i + \alpha_i - 2s < 0, \end{cases} \tag{103}$$

holds for any $i = 1, \ldots, \bar{D}$, where α_i is defined as

$$\alpha_i = \{max\ j\ :\ \bar{t}_i = \cdots = \bar{t}_{i+j-1}\}. \tag{104}$$

We show that (103) is always true in our case. It is obvious that α_i can be computed as

$$\alpha_i = \begin{cases} D_{2,0} - i + 1 & \text{if } 1 \le i \le D_{2,0}, \\ D_{2,j} - i + 1 & \text{if } D_{2,j-1} < i \le D_{2,j}. \end{cases} \tag{105}$$

Combining (101) and (105), we know that $\nu_i < s$ and $\alpha_i \le s$ always hold for $i > s$, which means

$$\begin{cases} \nu_i + \alpha_i - 2s = 0 & \text{if } 1 \le i \le s, \\ \nu_i + \alpha_i - 2s < 0 & \text{if } s < i \le \bar{D}. \end{cases} \tag{106}$$

Thus, the interval δ_i is computed as

$$\delta_i = \begin{cases} [\bar{t}_i, \bar{t}_{i+2s}) & \text{if } 1 \le i \le s, \\ (\bar{t}_i, \bar{t}_{i+2s}) & \text{if } s < i \le \bar{D}. \end{cases} \tag{107}$$

Consequently, we have

$$\tau_i = t_0 \in [t_0, t_1) \subseteq [\bar{t}_i, \bar{t}_{i+2s}) = \delta_i,\ 1 \le i \le s. \tag{108}$$

When $i > s$, we denote $\bar{t}_i = t_k$, $\bar{t}_{i+2s} = t_l$ and $\tau_i = t_j$. As is shown in (99) and (100), we have

$$D_{2,k-1} < i,\ (i + 2s) \le D_{2,l},\ D_{1,j-1} < i \le D_{1,j}. \tag{109}$$

Due to the fact that $d_i \le s$ holds for any $1 \le i < M$, the following two inequalities always hold.

$$D_{2,k-1} < i \le D_{1,j} = (D_{2,j} - s) \le D_{2,j-1}, \tag{110}$$

$$D_{2,j} = (D_{1,j} + s) \le (D_{1,j-1} + 2s) < (i + 2s) \le D_{2,l}. \tag{111}$$

Inequalities (110) and (111) imply $k < j$ and $j < l$, thus

$$\tau_i = t_j \in (t_k, t_l) = (\bar{t}_i, \bar{t}_{i+2s}) = \delta_i,\ s < i \le \bar{D}, \tag{112}$$

always holds. Combining (108) and (112) gives (103). Therefore, the coefficient matrix $\mathbf{B}$ on basis is always nonsingular for settings on the original problem, implying the existence and uniqueness of solution.

The optimality conditions guarantee one unique solution in each decoupled dimension, which gives its sufficiency. □

B. Proof of Proposition 2

Proof. Denote by $\mathbf{J}$ the Jacobian of $\mathbf{G}$. For any $x \in \mathbb{D}_{\mathrm{F}}$ and $y \in \mathbb{R}^N$, satisfying $x = \mathbf{G}(y)$ or $y = \mathbf{G}^{-1}(x)$, we have

$$\nabla H(y) = \mathbf{J}(y)^{\mathrm{T}} \nabla F(x). \tag{113}$$

Then, the nonsingularity of $\mathbf{J}$ implies that the first statement always holds. Denote by $\mathbf{K}_i$ the Hessian of the i-th entry in $\mathbf{G}$. If x and y are stationary points, the Hessian of H is

$$\begin{aligned} \nabla^2 H(y) &= \mathbf{J}(y)^{\mathrm{T}} \nabla^2 F(x) \mathbf{J}(y) + \sum_{i=1}^{N} \frac{\partial F(x)}{\partial x_i} \mathbf{K}_i(y) \\ &= \mathbf{J}(y)^{\mathrm{T}} \nabla^2 F(x) \mathbf{J}(y). \end{aligned} \tag{114}$$

Then, the nonsingular $\mathbf{J}$ implies that $\nabla^2 F(x)$ and $\nabla^2 H(y)$ are congruent [45]. Thus the second statement holds. □

REFERENCES

[1] M. Ryll, J. Ware, J. Carter, and N. Roy, "Efficient trajectory planning for high speed flight in unknown environments," in *IEEE International Conference on Robotics and Automation*, Montreal, Canada, 2019, pp. 732–738.

[2] H. Oleynikova, C. Lanegger, Z. Taylor, M. Pantic, A. Millane, R. Siegwart, and J. Nieto, "An open-source system for vision-based micro-aerial vehicle mapping, planning, and flight in cluttered environments," *Journal of Field Robotics*, vol. 37, no. 4, pp. 642–666, 2020.

[3] J. Zhang, C. Hu, R. G. Chadha, and S. Singh, "Falco: Fast likelihood-based collision avoidance with extension to human-guided navigation," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1300–1313, 2020.

[4] L. Campos-Macías, R. Aldana-López, R. de la Guardia, J. I. Parra-Vilchis, and D. Gómez-Gutiérrez, "Autonomous navigation of mavs in unknown cluttered environments," *Journal of Field Robotics*, vol. 38, no. 2, pp. 307–326, 2021.

[5] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "EGO-Planner: An ESDF-free gradient-based local planner for quadrotors," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 478–485, 2021.

[6] P. Foehn, D. Brescianini, E. Kaufmann, T. Cieslewski, M. Gehrig, M. Muglikar, and D. Scaramuzza, "AlphaPilot: Autonomous drone racing," *Autonomous Robots*, pp. 1–14, 2021.

[7] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: Introductory theory and examples," *International Journal of Control*, vol. 61, no. 6, pp. 1327–1361, 1995.

[8] M. J. Van Nieuwstadt and R. M. Murray, "Real-time trajectory generation for differentially flat systems," *International Journal of Robust and Nonlinear Control*, vol. 8, no. 11, pp. 995–1020, 1998.

[9] P. Martin, R. M. Murray, and P. Rouchon, "Flat systems, equivalence and trajectory generation," California Institute of Technology, Pasadena, Calif, USA, Tech. Rep. CDS 2003-008, 2003.

[10] J.-C. Ryu and S. K. Agrawal, "Differential flatness-based robust control of mobile robots in the presence of slip," *The International Journal of Robotics Research*, vol. 30, no. 4, pp. 463–475, 2011.

[11] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation*, Shanghai, China, May 2011, pp. 2520–2525.

[12] M. Watterson and V. Kumar, "Control of quadrotors using the Hopf fibration on SO(3)," in *International Symposium on Robotics Research*, Hanoi, Vietnam, 2019.

[13] J. Ferrin, R. Leishman, R. Beard, and T. McLain, "Differential flatness based control of a rotorcraft for aggressive maneuvers," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, USA, 2011, pp. 2688–2693.

[14] M. Faessler, A. Franchi, and D. Scaramuzza, "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 620–626, 2018.

[15] B. Mu and P. Chirarattananon, "Trajectory generation for underactuated multirotor vehicles with tilted propellers via a flatness-based method," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2019, pp. 1365–1370.

[16] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[17] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University, Tech. Rep. TR 98-11, 1998.

[18] L. E. Kavraki, M. N. Kolountzakis, and J.-C. Latombe, "Analysis of probabilistic roadmaps for path planning," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 166–171, 1998.

[19] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, pp. 846–894, 2011.

[20] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast Marching Tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 883–921, 2015.

[21] Y. Li, Z. Littlefield, and K. E. Bekris, "Asymptotically optimal sampling-based kinodynamic planning," *The International Journal of Robotics Research*, vol. 35, no. 5, pp. 528–564, 2016.

[22] J. D. Gammell, T. D. Barfoot, and S. S. Srinivasa, "Informed sampling for asymptotically optimal path planning," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 966–984, 2018.

[23] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. SIAM, 2010.

[24] M. A. Patterson and A. V. Rao, "GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Transactions on Mathematical Software*, vol. 41, no. 1, pp. 1–37, 2014.

[25] B. Houska, H. J. Ferreau, and M. Diehl, "ACADO toolkit—an open-source framework for automatic control and dynamic optimization," *Optimal Control Applications and Methods*, vol. 32, no. 3, pp. 298–312, 2011.

[26] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.

[27] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[28] A. Bry, C. Richter, A. Bachrach, and N. Roy, "Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 969–1002, 2015.

[29] A. J. Barry, T. Jenks, A. Majumdar, H.-T. Lin, I. G. Ros, A. A. Biewener, and R. Tedrake, "Flying between obstacles with an autonomous knife-edge maneuver," in *IEEE International Conference on Robotics and Automation*, Hong Kong, China, 2014, pp. 2559–2559.

[30] R. Deits and R. Tedrake, "Efficient mixed-integer planning for UAVs in cluttered environments," in *IEEE International Conference on Robotics and Automation*, Seattle, USA, 2015, pp. 42–49.

[31] F. Gao, L. Wang, B. Zhou, X. Zhou, J. Pan, and S. Shen, "Teach-Repeat-Replan: A complete and robust system for aggressive flight in complex environments," *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1526–1545, 2020.

[32] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2318–2327, 2009.

[33] J. Tordesillas, B. T. Lopez, and J. P. How, "FASTER: Fast and safe trajectory planner for flights in unknown environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Macau, China, 2019, pp. 1934–1940.

[34] J. Tordesillas and J. P. How, "MINVO Basis: Finding simplexes with minimum volume enclosing polynomial curves," *arXiv preprint arXiv:2010.10726*, 2020.

[35] W. Sun, G. Tang, and K. Hauser, "Fast UAV trajectory optimization using bilevel optimization with analytical gradients," in *American Control Conference*, 2020, pp. 82–87.

[36] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Athena-Scientific, 1995.

[37] C. D. Toth, J. O'Rourke, and J. E. Goodman, *Handbook of Discrete and Computational Geometry*. CRC Press, 2017.

[38] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robotics and Automation Letters*, pp. 1688–1695, 2017.

[39] E. Verriest and F. Lewis, "On the linear quadratic minimum-time problem," *IEEE Transactions on Automatic Control*, vol. 36, no. 7, pp. 859–863, 1991.

[40] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.

[41] S. Liu, N. Atanasov, K. Mohta, and V. Kumar, "Search-based motion planning for quadrotors using linear quadratic minimum time control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, Canada, 2017, pp. 2872–2879.

[42] Z. Zhang, J. Tomlinson, and C. Martin, "Splines and linear control theory," *Acta Applicandae Mathematica*, vol. 49, no. 1, pp. 1–34, 1997.

[43] M. Egerstedt and C. Martin, *Control Theoretic Splines: Optimal Control, Statistics, and Path planning*. Princeton University Press, 2009.

[44] A. V. Dmitruk and A. M. Kaganovich, "The hybrid maximum principle is a consequence of pontryagin maximum principle," *Systems & Control Letters*, vol. 57, no. 11, pp. 964–970, 2008.

[45] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 2012.

[46] G. H. Golub and F. V. Loan, *Matrix Computations*. The Johns Hopkins University Press, 2013.

[47] Z. Zhang and D. Scaramuzza, "Perception-aware receding horizon navigation for MAVs," in *IEEE International Conference on Robotics and Automation*, Brisbane, Australia, 2018, pp. 2534–2541.

[48] T. Nägeli, J. Alonso-Mora, A. Domahidi, D. Rus, and O. Hilliges, "Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1696–1703, 2017.

[49] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2006.

[50] J. Lee, *Introduction to Smooth Manifolds*. Springer, 2012.

[51] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996.

[52] R. Seidel, "Small-dimensional linear programming and convex hulls made easy," *Discrete & Computational Geometry*, vol. 6, no. 3, pp. 423–434, 1991.

[53] F. S. Sisser, "Elimination of bounds in optimization problems by transforming variables," *Mathematical Programming*, vol. 20, no. 1, pp. 110–121, 1981.

[54] J. Warren, S. Schaefer, A. N. Hirani, and M. Desbrun, "Barycentric coordinates for convex sets," *Advances in Computational Mathematics*, vol. 27, no. 3, pp. 319–338, 2007.

[55] D. P. Bertsekas, *Nonlinear Programming*. Athena-Scientific, 2016.

[56] L. S. Jennings and K. L. Teo, "A computational algorithm for functional inequality constrained optimization problems," *Automatica*, vol. 26, no. 2, pp. 371–375, 1990.

[57] A. Griewank and A. Walther, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM, 2008.

[58] W. Baur and V. Strassen, "The complexity of partial derivatives," *Theoretical Computer Science*, vol. 22, no. 3, pp. 317–330, 1983.

[59] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, 1989.

[60] D. Burke, A. Chapman, and I. Shames, "Generating minimum-snap quadrotor trajectories really fast," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, USA, 2020, pp. 1487–1492.

[61] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, pp. 1–36, 2020.

[62] J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, and J. W. Liu, "A supernodal approach to sparse partial pivoting," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 3, pp. 720–755, 1999.

[63] W. Hönig, J. A. Preiss, T. S. Kumar, G. S. Sukhatme, and N. Ayanian, "Trajectory planning for quadrotor swarms," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 856–869, 2018.

[64] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on Point Clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.

[65] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic Foundations of Robotics XI*. Springer, 2015, pp. 109–124.

[66] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2020. [Online]. Available: https://www.gurobi.com

[67] MOSEK Aps, "MOSEK Optimizer API for C," 2020. [Online]. Available: https://www.mosek.com

[68] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, pp. 1–21, 2022.

[69] A. Orthey and M. Toussaint, "Rapidly-exploring quotient-space trees: Motion planning using sequential simplifications," in *International Symposium on Robotics Research*, Hanoi, Vietnam, 2019.

[70] A. Orthey, A. Escande, and E. Yoshida, "Quotient-space motion planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Madrid, Spain, 2018, pp. 8089–8096.

[71] Y. Wu, Z. Ding, C. Xu, and F. Gao, "External forces resilient safe motion planning for quadrotor," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8506–8513, 2021.

[72] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a swarm of agile micro quadrotors," *Autonomous Robots*, vol. 35, no. 4, pp. 287–300, 2013.

[73] X. Zhou, Z. Wang, X. Wen, J. Zhu, C. Xu, and F. Gao, "Decentralized spatial-temporal trajectory planning for multicopter swarms," *arXiv preprint arXiv:2106.12481*, 2021.

[74] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9, pp. 1164–1193, 2013.

[75] L. Campos-Macías, D. Gómez-Gutiérrez, R. Aldana-López, R. de la Guardia, and J. I. Parra-Vilchis, "A hybrid method for online trajectory planning of mobile robots in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 935–942, 2017.

[76] R. Natarajan, H. Choset, and M. Likhachev, "Interleaving graph search and trajectory optimization for aggressive quadrotor flight," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5357–5364, 2021.

[77] L. Schumaker, *Spline Functions: Basic Theory*. Cambridge University Press, 2007.

Zhepei Wang received the B.Eng. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2017.

He is currently working toward the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China. His research interests include motion planning, discrete and computational geometry, numerical optimization, and autonomous navigation of unmanned vehicles.

Xin Zhou received the B.Eng. degree in electrical engineering and automation from China University of Mining and Technology, Xuzhou, China, in 2019.

He is currently working toward the Ph.D. degree in control engineering from Zhejiang University, Hangzhou, China. His research interests include motion planning and mapping for aerial swarm robotics.

Chao Xu received the Ph.D. degree in mechanical engineering from Lehigh University, Bethlehem, PA, USA, in 2010.

He is the Professor of Cyber-Systems & Robotics, the Associate Dean of the College of Control Science and Engineering, and the Founding Dean of Huzhou Institute of Zhejiang University. He founded the FAST (Field Autonomous System and Computing) Lab. His research interests are robot mechanics and control. He is currently the Managing-Editor for the Journal of Industrial and Management Optimization, and the Founding Managing-Editor for IET Cyber-Systems & Robotics.

Fei Gao received the Ph.D. degree in electronic and computer engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2019.

He is currently an Assistant Professor with the College of Control Science and Engineering, Zhejiang University, where he co-directs the FAST (Field Autonomous System and Computing) Lab and leads the FAR (Flying Autonomous Robotics) Group. His research interests include aerial robots, swarms, autonomous navigation, motion planning, and localization and mapping.