Geometrically Constrained Trajectory Optimization for Multicopters

Journal Title
XX(X):1–30

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Zhepei Wang, Xin Zhou, Chao Xu and Fei Gao

Abstract

We present an optimization-based framework for multicopter trajectory planning subject to geometrical spatial 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 wide range 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, the proposed 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 frequency and constraint fidelity for a multicopter with limited resources and maneuvering capability. Its generality and robustness are both demonstrated by applications and experiments for different 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 magnitudes. Details and source code of our framework will be freely available at: http://zju-fast.com/gcopter.

Keywords

Nonholonomic Motion Planning, Aerial Robotics, Autonomous Agents

Introduction

Multicopters rely on robust and efficient trajectory planning for safe yet agile autonomous navigation in complex environments (Bry et al. 2015; Oleynikova et al. 2020; Zhang et al. 2020; Zhou et al. 2021). For robotics, precisely incorporating dynamics, smoothness, and safety is essential to generate high-quality motions. Moreover, lightweight robots, such as multicopters under the 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 guarantees high-frequency online planning while also considers 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.

For high-performance planning mentioned above, four major algorithmic challenges are possessed. First, ensuring motion safety usually involves frequent interactions with a large volume of discrete environment data which has extensive nonsmoothness. 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, making the requirements of 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 paper, we overcome all these challenges by designing a lightweight and flexible optimization framework to meet various motion requirements based on a novel trajectory class.

As the theoretical foundation of the proposed framework, we present optimality conditions to the concerned multistage control effort minimization problem in a general form for the unconstrained linear dynamics. More importantly, the proof of both the necessity and sufficiency are provided for the first time. The optimality conditions are easy-to-use in that the unique optimal solution can be directly constructed with linear complexity in both time and space aspects. As a result, the computational efficiency is improved by orders of magnitudes against conventional schemes (Mellinger and Kumar 2011; Bry et al. 2015; Burke et al. 2020). Besides, the existence and uniqueness for these conditions further provide crucial information on the smoothness of sensitivity of problem parameters.

To ease the computation burden in trajectory planning without sacrificing the solution quality, it is essential to use sparse parameterization while retaining the flexibility to

Zhejiang University, Hangzhou, China

Corresponding author:
Fei Gao, State Key Laboratory of Industrial Control Technology, Institute of Cyber-Systems and Control, College of Control Science and Engineering, Zhejiang University, Hangzhou, Zhejiang, 310027, China.
Email: fgaoaa@zju.edu.cn

suits the system dynamics. Therefore, we design a novel trajectory class based on the optimality conditions. Any elements in this trajectory class is already an unconstrained control effort minimizer, thus we name it MINCO (Minimum Control). This newly designed trajectory class is different from conventional splines that majorly focus on the smoothness of the geometrical shape, such as B-Splines and Bézier curves. MINCO is designed to directly control the spatial and temporal profile of a trajectory, which are of equal importance while satisfying dynamic feasibility of multicopters. Specifically, MINCO takes the intermediate points and time vector as a compact representation. Besides, a spatial-temporal deformation scheme is also designed such that the trajectory in MINCO can be optimized under any user-defined objective function. A gradient propagation scheme with linear complexity is also developed to make the deformation computationally efficient.

The proposed framework utilizes geometrical approximation of low-dimensional free space based on sampling-based or search-based global methods. The safety is ensured by direct constraints between geometrical primitives of the free space and that of a multicopter. For direct constraints on parameters, an elimination method is proposed such that the deformation of MINCO can be carried out conveniently and efficiently through unconstrained optimization. This method exactly eliminates geometrical constraints without introducing extra local minima.

Reliable motion planning often requires that all dynamic constraints satisfied at a high resolution. To achieve this, we propose a systematic way to ensure continuous-time constraints for sparse trajectory representations without resorting to a fine discretization. Conventionally, most existing methods use differential flatness to decouple multicopter dynamics from their planning. The system states and inputs are only algebraically evaluated in the feedback control stage instead of the planning. Differently, we further exploit the differentiation of this algebraic procedure in a backward direction such that the constraint violation can be reflected in their gradient w.r.t. decision variables of the sparse representation. The efficiency is also ensured by *Baur-Strassen Theorem* (Baur and Strassen 1983), which shows the same complexity for an algebraic function on its evaluation and gradient computation. Moreover, a time integral penalty functional with its differentiation is also proposed to convert continuous-time constraints into finite-dimensional ones.

Our framework focuses on computationally efficient yet high-quality trajectory planning where there are complex constraints for safety, critical limits on dynamics, and task-specified requirements for a multicopter. 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 real-world experiments to show the practical performance of the proposed approach. For instance, real-time $\mathrm{SE}(3)$ planning such that a multicopter safely and agilely maneuvers through several narrow tilted gaps sequentially. The stability and robustness of our framework are also validated by its persistent real-time interactions with a gap that is arbitrarily placed.

Summarizing our contributions in this paper:

- 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 via corresponding well-formed 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.

Related Work

Despite various multicopter trajectory planning approaches in existing literature, there has yet to emerge a complete framework to accomplish high-frequency large-scale planning while incorporating user-defined continuous-time constraints on state and control. The strong need for such a framework comes from the fact that a high-quality motion subject to complex dynamic constraints requires adequate computation time and resources. The requirement is difficult to fulfill, especially in the time-critical flights of a lightweight multicopter. To bridge this gap, we propose a computationally efficient framework specialized for multicopters, exploring and exploiting different capabilities from both optimal control and motion planning.

Differentially Flat Multicopters

The concept of differential flatness of nonlinear systems has been introduced by Fliess et al. (1995) and drawn great attentions in robotics trajectory planning (Van Nieuwstadt and Murray 1998; Martin et al. 2003; Ryu and Agrawal 2011). This property makes it possible to recover the full state and input of a flat system from finite derivatives of its flat outputs. As for multicopters, Mellinger and Kumar (2011) prove the differential flatness of regular quadcopters with aligned propellers, which takes three-dimensional torque together with thrust as inputs. They utilize the flatness to generate trajectories that meet differential constraints and compute the feedforward term of a tracking controller. Mueller et al. (2015) show the flatness of simplified dynamics suffices for quadcopter trajectory generation, where body angular rates and thrust are taken as inputs. Ferrin et al. (2011) show the flatness of a hexacopter whose inputs are desired orientation and total thrust. They utilize the flatness to compute the nominal state and input where the system is linearized for Linear Quadratic Regulator (LQR). Faessler et al. (2018) consider more complicated system dynamics with aerodynamic effect to design a high-speed flight controller. The linear rotor drag effect is incorporated into both the translational and rotational parts via linear and angular accelerations. They further present the flatness

of the parallel-rotor multicopter system subject to the rotor drag. Moreover, Mu and Chirarattananon (2019) 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 system, if holds, is beneficial to trajectory generation and tracking in that the reference state and input can be algebraically computed. Based on this property, literature mentioned above directly eliminates differential constraints from system dynamics. However, commonly-used differential constraints are only considered to be valid in specific regions of the state space with admissible control inputs. Therefore, constraints on both state and control are important to ensure the feasibility considering a real physical system. The framework proposed in this paper is also based on the flatness of multicopters but differs from previous works in that a general form of state-input constraints is formally considered and flexibly handled.

Sampling-Based Motion Planning

Sampling-based motion planning possesses the nature of exploration and exploitation in the solution space. It well suits the seeking for global solutions of non-convex problems where the complexity mainly originates from the configuration space. The Probabilistic Roadmap (PRM) (Kavraki et al. 1996) constructs an obstacle-free graph from randomly sampled feasible vertices in a static configuration space for multiple queries of shortest paths. The Rapidly-exploring Random Tree (RRT) (LaValle 1998) instead incrementally builds a tree by online sampling based on a single query's information. Both of them are probabilistically complete (Kavraki et al. 1998; Frazzoli et al. 2002) since their probability of failure decays to zero exponentially as the sample number goes to infinity. A milestone is that Karaman and Frazzoli (2011) analyze the asymptotic optimality conditions for sampling-based algorithms. They also propose asymptotically optimal variants of PRM and RRT, known as PRM* and RRT*. These variants ensure the convergence to the globally optimal solution as the sample number goes to infinity. Successful applications of them also exist in kinodynamic planning (or trajectory planning) problems (LaValle and Kuffner Jr 2001; Frazzoli et al. 2002; Karaman and Frazzoli 2010) if a steering function based on system dynamics is provided. There are also algorithms (Janson et al. 2015; Li et al. 2016; Gammell et al. 2018) that further explore either the efficiency or the applicability of randomized motion planning.

An optimal sampling-based planner is promising, albeit time-consuming, when the robot has many Degrees of Freedom (DoFs) and the configuration space is highly nonconvex. The proposed framework does not confront the difficulties from both the configuration space and the multicopter dynamics concurrently. It exploits the capability of a sampling-based one to overcome the complexity from environments. Only a low-dimensional collision-free path is required for a geometrical approximation of the local free space. Our framework then accomplishes the optimization of motions that are homotopic to the 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 environment and dynamics are divided and conquered in the *front end* and the *back end*, respectively.

Optimization-Based Motion Planning

Optimization-based planners often focus on local solutions of motion planning problems where objectives or optimality criteria are specified. Consequently, they are often built on specific environment pre-processing methods such that obstacle information is encoded into the optimization as objectives or constraints.

Trajectory optimization, viewed as an optimal control problem, has been extensively studied for general dynamic systems in the control community (Betts 1998). There are various general-purpose solvers that are able to compute high-quality solutions, such as the shooting-based solver ACADO (Houska et al. 2011), and the collocation-based one GPOPS-II (Patterson and Rao 2014). With these methods, the original optimization is often discretized and transcribed into a Nonlinear Programming (NLP), which is then solved by well-established NLP methods such as SNOPT (Gill et al. 2005) or IPOPT (Wächter and Biegler 2006). However, robotics motion planning may impose hard-to-formulate constraints, nonsmoothness, and even integer variables. Besides, general-purpose methods often take a relatively long computation time, making it inappropriate for time-critical applications. For example, Bry et al. (2015) 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 (Barry et al. 2014). Therefore, specialized methods are on calling to overcome these difficulties.

To obtain smooth trajectories for robots, Elastic Band Quinlan and Khatib (1993) iteratively minimizes the strain energy of a spline within bubbles, i.e., the union of ball-shaped regions. These bubbles are constructed to cover the free space approximately near an initial feasible path. Thus the resultant trajectory is both smooth and safe within bubbles. Nonetheless, they only ensure smoothness in a narrow sense, which does not necessarily meet general robots' dynamic feasibility. Covariant Hamiltonian Optimization for Motion Planning (CHOMP) is proposed by Zucker et al. (2013) for high-DoF robotic systems, such as manipulators or legged robots. It formulates a parameterization-invariant cost functional, which is then minimized numerically by preconditioned gradient descent. CHOMP uses Signed Distance Field (SDF), i.e., the difference between an Euclidean distance field (Felzenszwalb and Huttenlocher 2012) and its complement, to compute collision penalties. However, the inherent nonsmoothness of SDF and its further discretization make the optimization cumbersome, thus slowing down the optimization theoretically (Nesterov 2018) and practically. There are also variants of CHOMP, such as Stochastic Trajectory Optimization for Motion Planning (STOMP) (Kalakrishnan et al. 2011) and Incremental Trajectory Optimization (ITOMP) (Park et al. 2012), which focus on stochastic optimization or dynamic obstacles, respectively. Schulman et al. (2014) formulate the trajectory planning into sequential convex programming with L_1 penalty terms. Obstacles are represented as convex geometrical primitives.

They penalize the distance between primitives as a hinge loss based on GJK-EPA collision detection (Gilbert et al. 1988; Van Den Bergen 2001). Their method comparably increases the success rate and computation efficiency in high-DoF trajectory planning problems.

For differentially flat multicopters, motion planning can be practically transformed into low-dimensional optimization on the premise of enough smoothness. Mellinger and Kumar (2011) generate fix-duration trajectories of quadcopter with splines. A Quadratic Programming (QP) is formulated with a quadratic cost of snap and linear constraints of safety. However, perturbation problems need to be solved such that the gradient of time allocation can be estimated by finite differencing, and the actuator constraints are also oversimplified. Bry et al. (2015) derive a closed-form solution for the unconstrained QP formed by the integral of squared derivatives. They heuristically add waypoints from a feasible path of RRT* to computed the solution until the safety is satisfied. This method is admittedly efficient but cannot guarantee high-quality solutions in obstacle-rich environments. Moreover, this method requires the inverse of a matrix whose non-singularity is not discussed. Deits and Tedrake (2015b) approximate the free space using polytopes. Safety of each piece of trajectory is equivalent to a Sum-of-Square (SOS) condition if it entirely lies inside a polytope. Due to the interval assignment, they solve the trajectory by Mixed Integer Second Order Cone Programming (MISOCP). This method generates globally optimal trajectories while the computation time is unacceptable. Gao et al. (2020) 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). There are also variants that propose improvements over the above methods. For example, Tordesillas et al. (2019) improve the efficiency of the method (Deits and Tedrake 2015b) by substituting SOS conditions on polynomials with linear constraints on Bézier curves at the cost of conservativeness (Tordesillas and How 2020). Sun et al. (2020) further eliminate its integer variables by optimizing time allocation instead, where the gradient is estimated from the sensitivity of a bilevel optimization.

The aforementioned specialized methods for multicopters utilize continuous-time parameterization, such as splines or Bézier curves, to overcome the computation burden resulting from fine discretization. However, this parameterization presents additional problems that are unresolved in existing methods. Firstly, either the time allocation or interval assignment of the representation lacks an effective and flexible optimization scheme for different scenarios. Secondly, enforcing continuous-time feasibility is either expensive or conservative for existing trajectory representations. Thirdly, dynamic limits on the nonlinear multicopter system are only heuristically fulfilled by restricting norms of trajectory derivatives. In comparison, methods that incorporate high-fidelity dynamics are free of these problems (Ryou et al. 2020; Foehn and Scaramuzza 2020), while their high-quality solutions demand a large amount of computation resources and are only applicable offline. In this paper, although a sparse trajectory representation is also proposed, it efficiently solves these previously existed disadvantages by introducing unified techniques. Moreover, the trajectory quality is comparable with that of the general-purpose optimal control solvers.

Preliminaries

Differential Flatness

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 $u \in \mathbb{R}^m$. For convenience, the map g is assumed to have rank m. The system is said to be *differentially flat* (Fliess et al. 1995), 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 $\Psi_x : (\mathbb{R}^m)^{s-1} \mapsto \mathbb{R}^n$ and $\Psi_u : (\mathbb{R}^m)^{s} \mapsto \mathbb{R}^m$ are induced by both f and g for parameterization. Intuitively, the state and control variables can be determined from z without explicit integration of the system dynamics (1).

Leveraging the differential flatness of a system, the trajectory generation is convenient when there only exist the 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

$$u = \Psi_u(z^{[s-1]}, v) \tag{5}$$

exactly linearizes the original flat system in to m decoupled chains of s-integrators (Isidori 2013). Let z_i denote the i-th entry in z, v_i the i-th entry in v and

$$z_i^{[s-1]} = (z_i, \dot{z}_i, \ldots, z_i^{(s-1)})^{\mathrm{T}}. \tag{6}$$

Then, the i-th integrator chain can be expressed as

$$\dot{z}_i^{[s-1]} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_{s-1} \\ 0 & \mathbf{0}^{\mathrm{T}} \end{pmatrix} z_i^{[s-1]} + \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix} v_i, \tag{7}$$

where $\mathbf{0}$ and $\mathbf{I}$ are an all-zeros matrix and an identity matrix with appropriate sizes, respectively. Provided with an initial state and a goal state, boundary values of each integrator chain (7) can be algebraically computed. Satisfying the boundary conditions, any trajectory integrated from these m integrator chains can be transformed into a feasible trajectory for the original flat system (Van Nieuwstadt and Murray 1998) by following transformations (2) and (3).

For multicopter dynamics where $m \ll n$, the transformations Ψ_x and Ψ_u much reduce the trajectory dimension and eliminate the differential constraints (1). An illustration is also provided in the Figure 1. As a side effect, nonlinearity coming from both Ψ_x and Ψ_u brings additional difficulties in trajectory generation for z when constraints exist on x and u of the original system. However, such an effect is much relieved if the flat-output space coincides with the configuration space of the relevant planning problem.

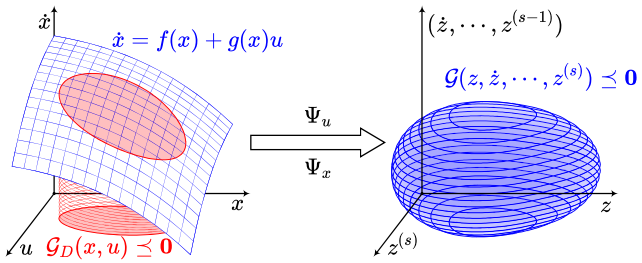


Figure 1. The transform Ψ_u and Ψ_x of a flat system eliminate differential constraints (blue surface) from dynamics in the state-input space (left coordinate). The original state-input constraint $\mathcal{G}_\mathcal{D}$ (red area) is also transformed into a new constraint $\mathcal{G}$ (blue volume) in the flat-output space (right coordinate).

Direct Optimization in Flat-Output Space

Fortunately, the differential flatness for multicopters has been well studied and is shown to have physically meaningful flat-output space which overlaps with the concerned configuration space. As is studied by Mellinger and Kumar (2011); Faessler et al. (2018); Mu and Chirarattananon (2019), many underactuated multicopters have the following flat output:

$$z = (p_x, p_y, p_z, \psi)^{\mathrm{T}} \tag{8}$$

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

To generate a feasible motion for the multicopter, we optimize a trajectory $z(t) : [0, T] \mapsto \mathbb{R}^m$ in the flat-output space such that most of the spatial constraints are directly enforced. For trajectory smoothness, we adopt the quadratic control effort (Verriest and Lewis 1991) with time regularization as the cost functional over $z(t)$. For feasibility, we split general constraints on multicopter system into two parts. The first part consists of spatial constraints coming from the configuration space. Normally, a collision-free motion implies

$$z(t) \in \mathcal{F},\ \forall t \in [0, T], \tag{9}$$

where $\mathcal{F}$ is the concerned *obstacle-free* region in configuration space. The second part consists of any other constraints on state and control of system dynamics, such as actuator limits or task-specific constraints. All these user-defined constraints in the second part are denoted by

$$\mathcal{G}_\mathcal{D}(x(t), u(t)) \preceq \mathbf{0},\ \forall t \in [0, T]. \tag{10}$$

Due to existence of Ψ_x and Ψ_u, corresponding constraints on $z(t)$ are computed as

$$\mathcal{G}_\mathcal{D}(\Psi_x(z^{[s-1]}(t)), \Psi_u(z^{[s]}(t))) \preceq \mathbf{0},\ \forall t \in [0, T]. \tag{11}$$

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 (11) hereafter by

$$\mathcal{G}(z(t), \dot{z}(t), \ldots, z^{(s)}(t)) \preceq \mathbf{0},\ \forall t \in [0, T], \tag{12}$$

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

Problem Formulation

Concluding above descriptions gives the following problem:

$$\min_{z(t),T} \int_0^T v(t)^{\mathrm{T}} \mathbf{W} v(t) \mathrm{d}t + \rho(T), \tag{13a}$$

$$s.t.\ \ v(t) = z^{(s)}(t),\ \forall t \in [0, T], \tag{13b}$$

$$\mathcal{G}(z(t), \ldots, z^{(s)}(t)) \preceq \mathbf{0},\ \forall t \in [0, T], \tag{13c}$$

$$z(t) \in \mathcal{F},\ \forall t \in [0, T], \tag{13d}$$

$$z^{[s-1]}(0) = \bar{z}_o,\ z^{[s-1]}(T) = \bar{z}_f, \tag{13e}$$

where $\mathbf{W} \in \mathbb{R}^{m \times m}$ is a diagonal matrix with positive entries, $\rho : [0, \infty) \mapsto [0, \infty]$ the time regularization term, $\bar{z}_o \in \mathbb{R}^{ms}$ the initial condition and $\bar{z}_f \in \mathbb{R}^{ms}$ the terminal condition. The control $z^{(s)}$ of the integrator chain is allowed to be discontinuous in a finite number of time instants, as is commonly assumed in existing literature (Bertsekas 1995) for convention.

The trajectory optimization in (13) is nontrivial because of the continuous-time nonlinear constraints $\mathcal{G}$ and nonconvex set constraint $\mathcal{F}$. We specify some reasonable conditions to make it a well defined problem. As for the time regularization ρ, it trades off between the control effort and the expectation of total time,

$$\rho_s(T) = \sum_{i=0}^{M_T} b_i T^i, \tag{14}$$

where b_{M_T} is strictly 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:

$$\rho_f(T) = \begin{cases} 0 & if\ T = T_\Sigma, \\ \infty & if\ T \neq T_\Sigma. \end{cases} \tag{15}$$

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

$$\mathcal{F} \simeq \tilde{\mathcal{F}} = \bigcup_{i=1}^{M_\mathcal{P}} \mathcal{P}_i. \tag{16}$$

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

$$\begin{cases} \mathcal{P}_i \cap \mathcal{P}_j = \varnothing & if\ |i - j| = 2, \\ \mathrm{Int}\,(\mathcal{P}_i \cap \mathcal{P}_j) \neq \varnothing & if\ |i - j| \leq 1, \end{cases} \tag{17}$$

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:

$$\mathcal{P}_i^\mathcal{B} = \left\{ x \in \mathbb{R}^m \,\middle|\, \|x - o_i\|_2 \leq r_i \right\}, \tag{18}$$

or, more generally, a bounded convex polytope described by its $\mathcal{H}$-representation (Toth et al. 2017) with potentially redundant constraints:

$$\mathcal{P}_i^\mathcal{H} = \left\{ x \in \mathbb{R}^m \,\middle|\, \mathbf{A}_i x \preceq b_i \right\}. \tag{19}$$

For the optimization in (13), we aim to construct an efficient solver capable of online computing the solution trajectory at a high frequency.

Multi-Stage Control Effort Minimization

In this section, we analyze the multi-stage control effort minimization problem without functional constraints. For this problem, we propose easy-to-use optimality conditions in the general case, which are proved to be both 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 the conditions, a novel trajectory class along with linear-complexity spatial-temporal deformation is designed to meet user-defined objectives in various trajectory planning scenarios.

Unconstrained Control Effort Minimization

Control effort minimization has extensive studies and applications (Bertsekas 1995). A *Linear Quadratic Minimum-Time* (LQMT) problem (Verriest and Lewis 1991) considers quadratic control effort with linear time regularization, where only initial and terminal conditions are specified. The relevant result has many applications in aerial robotics such as the work by Mueller et al. (2015) and Liu et al. (2017a), where the LQMT trajectory for a chain of s-integrators is indeed a $2s-1$ degree polynomial. This kind of problem only considers a single stage with state specified at either start or end of the stage.

When there are spatial constraints $\mathcal{F}$, a trajectory with multiple pieces is much more applicable because additional constraints can be applied on intermediate waypoints (Bry et al. 2015) or control points (Tordesillas et al. 2019) to ensure safety. When there are actuator constraints in $\mathcal{G}$, adjusting the time allocation also helps to satisfy the dynamic limits (Liu et al. 2017b). These spatial and temporal parameters are vital to a flexible trajectory representation. A natural problem is to ensure the local smoothness of a trajectory if the global settings are fixed on these two kinds of parameters. Another problem is to appropriately adjust these global settings according to the requirements of the original problem (13).

We firstly study the former subproblem where spatial constraints $\mathcal{F}$ and functional constraints $\mathcal{G}$ are not considered temporarily. Specifically, the intermediate points and the time vector are given in advance. Consider the following M-stage control effort minimization for a chain of s-integrators,

$$\min_{z(t)} \int_{t_0}^{t_M} v(t)^{\mathrm{T}}\mathbf{W}v(t)\mathrm{d}t, \tag{20a}$$

$$s.t.\ \ v(t) = z^{(s)}(t),\ \forall t \in [t_0, t_M], \tag{20b}$$

$$z^{[s-1]}(t_0) = \bar{z}_o,\ z^{[s-1]}(t_M) = \bar{z}_f, \tag{20c}$$

$$z^{[d_i-1]}(t_i) = \bar{z}_i,\ 1 \le i < M, \tag{20d}$$

$$t_{i-1} < t_i,\ 1 \le i \le M. \tag{20e}$$

In this problem, the interval $[t_0, t_M]$ is split into M stages by $M+1$ fixed time stamps, with specified initial and terminal conditions. Besides, some derivatives of the flat output are fixed as $\bar{z}_i \in \mathbb{R}^{md_i}$ at t_i the end of each stage, where $(d_i - 1) < s$ is the highest order of specified derivatives.

Although this multi-stage problem has been widely studied and used, existing work mainly focuses on some necessary conditions for special cases of the problem (20), which lack sufficiency and simplicity. In aerial robotics applications, methods like the QP-based one (Mellinger and Kumar 2011) and the closed-form one (Bry et al. 2015) implicitly or explicitly optimize the unknown boundary conditions for each piece, using the optimal parameterization as *a priori*. This extra computation actually makes them less efficient. In control area, the special case where all $d_i = 1$ is also considered by Zhang et al. (1997) and Egerstedt and Martin (2009). They assume $d_i = 1$ but for general standard linear systems. Consequently, their solution constructed from controllability Gramian is less intuitive and efficient but suitable for systems where nonpolynomial solutions exist.

Different from previous works, we propose optimality conditions that enjoy necessity, sufficiency, and simplicity for the problem (20) with arbitrary settings of d_i. The optimal solution can be directly constructed in linear complexity without cost functional evaluation. What's more, the existence and uniqueness of the conditions provide well-formed approaches to further optimize problem parameters under user-defined objectives while retaining the efficiency and nondegeneracy of the solution.

Optimality Conditions

To analyze the optimality conditions for (20), we transform it from the Lagrangian form into Mayer form (Clarke 2013) by state augmentation. The new state $y \in \mathbb{R}^{ms+1}$ augmented by a scalar $\tilde{y} \in \mathbb{R}$ is defined as

$$y = \begin{pmatrix} z^{[s-1]} \\ \tilde{y} \end{pmatrix}. \tag{21}$$

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

$$\dot{y} = \begin{pmatrix} \bar{\mathbf{A}} & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & 0 \end{pmatrix} y + \begin{pmatrix} \mathbf{0} \\ v \\ v^{\mathrm{T}}\mathbf{W}v \end{pmatrix}, \tag{22}$$

where

$$\bar{\mathbf{A}} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_m & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I}_m \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{pmatrix} \in \mathbb{R}^{ms \times ms}. \tag{23}$$

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 $\tilde{y}$ part is reset as

$$\tilde{y}_{[i+1]}(t_i) = 0,\ 0 \le i < M, \tag{24}$$

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

$$z_{[i]}^{[s-1]}(t_i) = z_{[i+1]}^{[s-1]}(t_i),\ 1 \le i < M. \tag{25}$$

The conditions in (20c) and (20d) are still satisfied, i.e.,

$$z_{[1]}^{[s-1]}(t_0) = \bar{z}_o,\ z_{[M]}^{[s-1]}(t_M) = \bar{z}_f, \tag{26}$$

$$z_{[i]}^{[d_i-1]}(t_i) = \bar{z}_i,\ 1 \le i < M. \tag{27}$$

In this running process, the original cost functional in (20) is converted into the sum of terminal cost for the augmented system in each stage, i.e., $\sum_{i=1}^{M} \tilde{y}_{[i]}(t_i)$. Therefore, the optimal trajectories for both the augmented system and the original one are identical in their $z^{[s-1]}$ part.

Now we utilize the following *Hybrid Maximum Principle* to analyze the necessary conditions for an optimal trajectory minimizing the terminal cost.

Theorem 1. Hybrid Maximum Principle (Dmitruk and Kaganovich 2008). *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*

$$x_\Sigma = \left(x_1^{\mathrm{T}}(t_0), x_1^{\mathrm{T}}(t_1), \ldots, x_M^{\mathrm{T}}(t_{M-1}), x_M^{\mathrm{T}}(t_M)\right)^{\mathrm{T}}. \tag{28}$$

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

$$\min_{u_k, x_k}\ J(x_\Sigma), \tag{29a}$$

$$s.t.\ \ \dot{x}_k(t) = f_k(x_k(t), u_k(t)), \tag{29b}$$

$$u_k(t) \in U_k \subseteq \mathbb{R}^{r_k}, \tag{29c}$$

$$\forall t \in \Delta_k,\ k = 1, \ldots, M, \tag{29d}$$

$$\eta(x_\Sigma) = \mathbf{0}, \tag{29e}$$

where the functions $f_k : \mathbb{R}^{n_k} \times \mathbb{R}^{r_k} \mapsto \mathbb{R}^{n_k}$, $J : \mathbb{R}^{\bar{n}} \mapsto \mathbb{R}$ 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 the optimal process for (29) by $(x^*(t), u^*(t))$. Then, there exists a collection $(\alpha, \gamma, \psi_1, \ldots, \psi_M)$, where $\alpha \ge 0$, $\gamma \in \mathbb{R}^q$ and $\psi_k : \Delta_k \mapsto \mathbb{R}^{n_k}$ are Lipschitz continuous. It generates M Pontryagin functions

$$H_k(\psi_k, x_k, u_k) = \psi_k^{\mathrm{T}} f_k(x_k, u_k),\ t \in \Delta_k, \tag{30}$$

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: $(\alpha, \gamma^{\mathrm{T}}) \neq \mathbf{0}$;*
- *Adjoint equations: for almost all $t \in \Delta_k$,*

$$\dot{\psi}_k(t) = -\frac{\partial H_k}{\partial x_k}(\psi_k(t), x_k^*(t), u_k^*(t)); \tag{31}$$

- *Transversality conditions:*

$$\begin{cases} \psi_k(t_{k-1}) = L_{x_k(t_{k-1})}(x_\Sigma^*), \\ \psi_k(t_k) = -L_{x_k(t_k)}(x_\Sigma^*); \end{cases} \tag{32}$$

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

$$\begin{aligned} &H_k(\psi_k(t), x_k^*(t), u_k^*(t)) \\ &= \sup_{u_k \in U_k} H_k(\psi_k(t), x_k^*(t), u_k) \\ &= 0. \end{aligned} \tag{33}$$

Proof. The proof can be directly adapted from Theorem 4 in Dmitruk and Kaganovich (2008). 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

$$\psi_{[i]} = \begin{pmatrix} \lambda_{[i]} \\ \mu_{[i]} \end{pmatrix} = \left(\lambda_{[i]_1}, \lambda_{[i]_2}, \ldots, \lambda_{[i]_s}, \mu_{[i]}\right)^{\mathrm{T}}, \tag{34}$$

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 can be generated from (22) as

$$\begin{aligned} H_i(\psi_{[i]}, y_{[i]}, v_{[i]}) &= \psi_{[i]}^{\mathrm{T}} \hat{f}(y_{[i]}, v_{[i]}) \\ &= \lambda_{[i]}^{\mathrm{T}} \bar{\mathbf{A}} z_{[i]}^{[s-1]} + \lambda_{[i]_s}^{\mathrm{T}} v_{[i]} + \mu_{[i]} v_{[i]}^{\mathrm{T}} \mathbf{W} v_{[i]}. \end{aligned} \tag{35}$$

By applying the adjoint equation in (31) 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 $v_{[i]}$,

$$H_i(\psi_{[i]}, y_{[i]}, v_{[i]}) = \lambda_{[i]}^{\mathrm{T}} \bar{\mathbf{A}} z_{[i]}^{[s-1]} + \lambda_{[i]_s}^{\mathrm{T}} v_{[i]} + \bar{\mu}_i v_{[i]}^{\mathrm{T}} \mathbf{W} v_{[i]}. \tag{36}$$

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

$$\dot{\lambda}_{[i]} = -\bar{\mathbf{A}}^{\mathrm{T}} \lambda_{[i]}, \tag{37}$$

which is expanded as

$$\begin{pmatrix} \dot{\lambda}_{[i]_1} \\ \vdots \\ \dot{\lambda}_{[i]_s} \end{pmatrix} = - \begin{pmatrix} \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{I}_m & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_m & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I}_m & \mathbf{0} \end{pmatrix} \begin{pmatrix} \lambda_{[i]_1} \\ \vdots \\ \lambda_{[i]_s} \end{pmatrix}. \tag{38}$$

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

According to maximality conditions (33), the supremum of H_i is always 0 in Δ_i. Thus the positive definiteness of $\mathbf{W}$ implies $\bar{\mu}_i \le 0$. If $\bar{\mu}_i = 0$, then (36) 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 (38), $\psi_{[i]}(t) = \mathbf{0}$ holds for all t in Δ_i. In such a case, a contradiction occurs that the nontrivality condition and the transversality conditions (32) 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

$$\frac{\partial H_i}{\partial v_{[i]}}(\psi_{[i]}, y_{[i]}^*, v_{[i]}^*) = \lambda_{[i]_s} + 2\bar{\mu}_i \mathbf{W}^{\frac{1}{2}} v_{[i]}^* = \mathbf{0}, \tag{39}$$

i.e.,

$$v_{[i]}^*(t) = -\frac{1}{2\bar{\mu}_i} \mathbf{W}^{-\frac{1}{2}} \lambda_{[i]_s}(t),\ \forall t \in \Delta_i. \tag{40}$$

Then, $z_{[i]}^*$ produced by a chain of s-integrators from $\lambda_{[i]_s}(t)$, is a $2s-1$ degree polynomial, which matches the classical result from a single stage LQMT problem.

To further explore structures of the solution, we generate the Lagrange function using the cost of augmented system

along with all constraints in (25), (26) and (27). We have

$$\begin{aligned} L(y_\Sigma) = &\alpha \sum_{i=1}^{M} \tilde{y}_{[i]}(t_i) + \sum_{i=0}^{M-1} \gamma_i \tilde{y}_{[i+1]}(t_i) \\ &+ \sum_{i=1}^{M-1} (\zeta_i^{\mathrm{T}}, \sigma_i^{\mathrm{T}})(z_{[i]}^{[s-1]}(t_i) - z_{[i+1]}^{[s-1]}(t_i)) \\ &+ \theta_o^{\mathrm{T}}(z_{[1]}^{[s-1]}(t_0) - \bar{z}_o) + \theta_f^{\mathrm{T}}(z_{[M]}^{[s-1]}(t_M) - \bar{z}_f) \\ &+ \sum_{i=1}^{M-1} \theta_i^{\mathrm{T}}(z_{[i]}^{[d_i-1]}(t_i) - \bar{z}_i), \end{aligned} \tag{41}$$

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 (28). Following transversality conditions (32), taking the derivative of L w.r.t. y_Σ gives the boundary values of costates $\psi_{[i]}$ and $\psi_{[i+1]}$, i.e.,

$$\lambda_{[i]}(t_i) = -\begin{pmatrix} \zeta_i + \theta_i \\ \sigma_i \end{pmatrix}, \ \lambda_{[i+1]}(t_i) = -\begin{pmatrix} \zeta_i \\ \sigma_i \end{pmatrix}, \tag{42}$$

$$\mu_{[i]}(t_i) = \mu_{[i+1]}(t_{i+1}) = -\alpha. \tag{43}$$

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

$$\bar{\mu}_i = -\alpha, \ 1 \le i \le M. \tag{44}$$

Finally, by substituting (38), (42) and (44) into (40), we obtain that the optimal control of two consecutive stages satisfying

$$v_{[i]}^{*(j)}(t_i) = v_{[i+1]}^{*(j)}(t_i), \ 0 \le j < (s - d_i). \tag{45}$$

We finally know that the optimal control of the problem (20) 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 necessary optimality conditions derived from both (40) and (45) in the following theorem, which is also proved to be sufficient.

Theorem 2. Optimality Conditions. *A trajectory, denoted by $z^*(t) : [t_0, t_M] \mapsto \mathbb{R}^m$, is optimal for the problem (20), 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 $1 \le i \le M$;*
- *The boundary conditions in (20c);*
- *The intermediate conditions in (20d);*
- *$z^*(t)$ is $\bar{d}_i - 1$ times continuously differentiable at t_i for any $1 \le i < M$ where $\bar{d}_i = 2s - d_i$.*

Moreover, a unique trajectory exists for these conditions.

Proof. The necessity of the optimality conditions is evident in the previous analysis. The sufficiency of the optimality conditions, along with the existence and the uniqueness of solution 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 waypoints ($d_i = 1$) 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.

Minimization Without Cost Functional

Necessary and sufficient optimality conditions in Theorem 2 provide a straightforward way to compute the unique optimal trajectory, which enjoys linear complexity in time and space. The computation even does not 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:

$$p_i(t) = \mathbf{c}_i^{\mathrm{T}} \beta(t), \ t \in [0, T_i], \tag{46}$$

where $\mathbf{c}_i \in \mathbb{R}^{2s \times m}$ is the coefficient matrix of the piece, T_i the piece time, and $\beta(t) = (1, t, \ldots, t^N)^{\mathrm{T}}$ the natural basis. Given a coefficient matrix $\mathbf{c} \in \mathbb{R}^{2Ms \times m}$ defined by

$$\mathbf{c} = (\mathbf{c}_1^{\mathrm{T}}, \ldots, \mathbf{c}_M^{\mathrm{T}})^{\mathrm{T}}, \tag{47}$$

and a time vector $\mathbf{T} \in \mathbb{R}_{>0}^M$ defined by

$$\mathbf{T} = (T_1, \ldots, T_M)^{\mathrm{T}}, \tag{48}$$

the M-piece trajectory $p : [0, T] \mapsto \mathbb{R}^m$ is parameterized as

$$p(t) = p_i(t - t_{i-1}), \ \forall t \in [t_{i-1}, t_i), \tag{49}$$

where $t_i = \sum_{j=1}^{i} T_j$ and $T = t_M$. To compute the unique solution for the problem (20), we directly enforce optimality conditions on the coefficient matrix $\mathbf{c}$. For simplicity, we 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 $\mathbf{E}_i, \mathbf{F}_i \in \mathbb{R}^{2s \times 2s}$:

$$\begin{pmatrix} \mathbf{E}_i & \mathbf{F}_i \end{pmatrix} \begin{pmatrix} \mathbf{c}_i \\ \mathbf{c}_{i+1} \end{pmatrix} = \begin{pmatrix} \mathbf{D}_i \\ \mathbf{0}_{\bar{d}_i \times m} \end{pmatrix}, \tag{50}$$

where

$$\begin{aligned} \mathbf{E}_i = (&\beta(T_i), \ldots, \beta^{(d_i-1)}(T_i), \\ &\beta(T_i), \ldots, \beta^{(\bar{d}_i-1)}(T_i))^{\mathrm{T}}, \end{aligned} \tag{51}$$

$$\mathbf{F}_i = (\mathbf{0}, -\beta(0), \ldots, -\beta^{(\bar{d}_i-1)}(0))^{\mathrm{T}}. \tag{52}$$

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

$$\mathbf{F}_0 = (\beta(0), \ldots, \beta^{(s-1)}(0))^{\mathrm{T}}, \tag{53}$$

$$\mathbf{E}_M = (\beta(T_M), \ldots, \beta^{(s-1)}(T_M))^{\mathrm{T}}. \tag{54}$$

The linear system for the optimal coefficient matrix can then be written as

$$\mathbf{M}\mathbf{c} = \mathbf{b} \tag{55}$$

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

$$\mathbf{M} = \begin{pmatrix} \mathbf{F}_0 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{E}_1 & \mathbf{F}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{E}_2 & \mathbf{F}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{F}_{M-1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{E}_M \end{pmatrix}, \tag{56}$$

$$\mathbf{b}=(\mathbf{D}_0^{\mathrm{T}},\mathbf{D}_1^{\mathrm{T}},\mathbf{0}_{m\times\bar{d}_1},\ldots,\mathbf{D}_{M-1}^{\mathrm{T}},\mathbf{0}_{m\times\bar{d}_{M-1}},\mathbf{D}_M^{\mathrm{T}})^{\mathrm{T}}. \tag{57}$$

It is essential that the uniqueness of conditions in Theorem 2 ensures the nonsingularity of $\mathbf{M}$ for any time vector $\mathbf{T}\succ\mathbf{0}$. Consequently, the unique solution of $\mathbf{c}$ can be obtained by solving the linear equation system (55) with a banded matrix $\mathbf{M}$, i.e., a *banded system*. As for a nonsingular banded system, its *Banded PLU Factorization* always exists (Horn and Johnson 2012), which can be employed to compute the solution in $O(M)$ linear time and space complexity (Golub and Loan 2013). Besides, for any time vector $\mathbf{T}$ and condition $\mathbf{D}_i$ of problem size M, constructing both $\mathbf{b}$ and the band in $\mathbf{M}$ also costs linear complexity. Therefore, without the need of cost functional, the unique optimal trajectory for the multi-stage control effort minimization (20) is obtained in the minimal complexity, by directly applying our optimality conditions.

MINCO Trajectories with Spatial-Temporal Deformation

For a multicopter, the safety of a flight trajectory is usually dominated by its spatial profile, while the dynamic limits are dominated by its temporal profile. Therefore, we divide parameters of a trajectory into two groups. The first group consists intermediate points that the trajectory should pass through. The second contains the time vector for different pieces of the trajectory. Based on these parameters, all coefficients of a polynomial spline can be generated from conditions in Theorem 2. As a result, the trajectory naturally belongs to a minimum control effort class, which is named by MINCO for convenience hereafter in this paper.

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. The time vector are denoted by $\mathbf{T}=(T_1,\ldots,T_M)^{\mathrm{T}}$ where each $T_i=t_i-t_{i-1}$ is a strictly positive time for the i-th piece. The MINCO trajectory class $\mathfrak{T}_{\mathrm{MINCO}}$ is defined as

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

For brevity, the dimension m, system order s, initial and terminal conditions are omitted here. Intuitively, all trajectories in $\mathfrak{T}_{\mathrm{MINCO}}$ are compactly parameterized by only $\mathbf{q}$ and $\mathbf{T}$. Evaluating an entire trajectory from $\mathbf{q}$ and $\mathbf{T}$ can be done via the aforementioned linear-complexity formulation. If constraints exist, adjusting $\mathbf{q}$ yields the spatial deformation within $\mathfrak{T}_{\mathrm{MINCO}}$ such that the trajectory is collision-free. Adjusting $\mathbf{T}$ yields the temporal deformation within $\mathfrak{T}_{\mathrm{MINCO}}$ such that dynamic limits are satisfied.

In trajectory planning, task-specific objectives other than feasibility constraints are often considered, such as the demand on high perception quality in active SLAM (Zhang and Scaramuzza 2018), the requirement on low occlusion in aerial videography (Nägeli et al. 2017), or so on. For trajectories in $\mathfrak{T}_{\mathrm{MINCO}}$, it is important to adjust their parameters flexibly and adaptively to meet these user-defined requirements. Fortunately, the existence and uniqueness of conditions in Theorem 2 guarantee the smoothness of the parameter sensitivity for $\mathfrak{T}_{\mathrm{MINCO}}$. We design an iterative procedure to conduct spatial-temporal deformation within $\mathfrak{T}_{\mathrm{MINCO}}$ with linear time complexity in each iteration.

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

$$H(\mathbf{q},\mathbf{T})=F(\mathbf{c}(\mathbf{q},\mathbf{T}),\mathbf{T}). \tag{58}$$

To accomplish deformation within $\mathfrak{T}_{\mathrm{MINCO}}$, the function H together with its gradient $\partial H/\partial\mathbf{q}$ and $\partial H/\partial\mathbf{T}$ are needed for a high-level optimizer to optimize the objective. Obviously, evaluating H shares the same complexity as evaluating any trajectory in $\mathfrak{T}_{\mathrm{MINCO}}$. The key procedure is to compute the gradient. Now we give a linear-complexity way to compute $\partial H/\partial\mathbf{q}$ and $\partial H/\partial\mathbf{T}$ from the corresponding $\partial F/\partial\mathbf{c}$ and $\partial F/\partial\mathbf{T}$. We first rewrite the linear equation system (55) as

$$\mathbf{M}(\mathbf{T})\mathbf{c}(\mathbf{q},\mathbf{T})=\mathbf{b}(\mathbf{q}). \tag{59}$$

Without causing ambiguity, we omit the parameters in $\mathbf{M},\mathbf{b},\mathbf{c},F$ and H temporarily for simplicity. Any notation involves $\mathbf{c}$ 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 (59) w.r.t. $q_{i,j}$, which gives

$$\frac{\partial\mathbf{c}}{\partial q_{i,j}}=\mathbf{M}^{-1}\frac{\partial\mathbf{b}}{\partial q_{i,j}}. \tag{60}$$

Then,

$$\begin{aligned}\frac{\partial H}{\partial q_{i,j}}&=\mathrm{Tr}\left\{\left(\frac{\partial\mathbf{c}}{\partial q_{i,j}}\right)^{\mathrm{T}}\frac{\partial F}{\partial\mathbf{c}}\right\}\\&=\mathrm{Tr}\left\{\left(\mathbf{M}^{-1}\frac{\partial\mathbf{b}}{\partial q_{i,j}}\right)^{\mathrm{T}}\frac{\partial F}{\partial\mathbf{c}}\right\}\\&=\mathrm{Tr}\left\{\left(\frac{\partial\mathbf{b}}{\partial q_{i,j}}\right)^{\mathrm{T}}\left(\mathbf{M}^{-\mathrm{T}}\frac{\partial F}{\partial\mathbf{c}}\right)\right\},\end{aligned} \tag{61}$$

where $\mathrm{Tr}\,(\cdot)$ is the trace operation. The definition of $\mathbf{b}(\mathbf{q})$ in (57) 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, by stacking all the resultant scalars, we have

$$\frac{\partial H}{\partial q_i}=\left(\mathbf{M}^{-\mathrm{T}}\frac{\partial F}{\partial\mathbf{c}}\right)^{\mathrm{T}}e_{(2i-1)s+1}, \tag{62}$$

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

$$\mathbf{M}^{\mathrm{T}}\mathbf{G}=\frac{\partial F}{\partial\mathbf{c}}. \tag{63}$$

We only need to compute $\mathbf{G}$ once to obtain $\partial H/\partial q_i$ 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 (Horn and Johnson 2012). Specifically, $\mathbf{M}^{\mathrm{T}} = \bar{\mathbf{L}}\bar{\mathbf{U}}\mathbf{P}^{\mathrm{T}}$, where

$$\bar{\mathbf{L}} = \mathbf{U}^{\mathrm{T}} \left(\mathbf{U} \circ \mathbf{I}\right)^{-1}, \ \bar{\mathbf{U}} = \left(\mathbf{I} \circ \mathbf{U}\right) \mathbf{L}^{\mathrm{T}}, \tag{64}$$

where the inverse is only 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 $\mathbf{G}$ as

$$\mathbf{G} = \left(\mathbf{G}_0^{\mathrm{T}}, \mathbf{G}_1^{\mathrm{T}}, \ldots, \mathbf{G}_{M-1}^{\mathrm{T}}, \mathbf{G}_M^{\mathrm{T}}\right)^{\mathrm{T}} \tag{65}$$

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 $1 \leq i < M$. After that, the gradient of H w.r.t. $\mathbf{q}$ is determined as

$$\frac{\partial H}{\partial \mathbf{q}} = \left(\mathbf{G}_1^{\mathrm{T}} e_1, \ldots, \mathbf{G}_{M-1}^{\mathrm{T}} e_1\right), \tag{66}$$

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 (59) w.r.t. T_i gives

$$\frac{\partial \mathbf{M}}{\partial T_i}\mathbf{c} + \mathbf{M}\frac{\partial \mathbf{c}}{\partial T_i} = \mathbf{0}. \tag{67}$$

Thus,

$$\begin{aligned}\frac{\partial H}{\partial T_i} &= \frac{\partial F}{\partial T_i} + \mathrm{Tr}\left\{\left(\frac{\partial \mathbf{c}}{\partial T_i}\right)^{\mathrm{T}} \frac{\partial F}{\partial \mathbf{c}}\right\} \\ &= \frac{\partial F}{\partial T_i} - \mathrm{Tr}\left\{\left(\frac{\partial \mathbf{M}}{\partial T_i}\mathbf{c}\right)^{\mathrm{T}} \mathbf{M}^{-\mathrm{T}} \frac{\partial F}{\partial \mathbf{c}}\right\} \\ &= \frac{\partial F}{\partial T_i} - \mathrm{Tr}\left\{\mathbf{G}^{\mathrm{T}} \frac{\partial \mathbf{M}}{\partial T_i}\mathbf{c}\right\}\end{aligned} \tag{68}$$

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

$$\mathbf{G}^{\mathrm{T}} \frac{\partial \mathbf{M}}{\partial T_i}\mathbf{c} = \mathbf{G}_i^{\mathrm{T}} \frac{\partial \mathbf{E}_i}{\partial T_i}\mathbf{c}_i. \tag{69}$$

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

$$\frac{\partial H}{\partial T_i} = \frac{\partial F}{\partial T_i} - \mathrm{Tr}\left\{\mathbf{G}_i^{\mathrm{T}} \frac{\partial \mathbf{E}_i}{\partial T_i}\mathbf{c}_i\right\}, \tag{70}$$

where $\partial \mathbf{E}_i / \partial T_i$ has an analytic form derived from (51). Computing (70) for $1 \leq i \leq M$ gives $\partial H / \partial \mathbf{T}$.

Finally, we finishes the gradient computation for both $\partial H/\partial \mathbf{q}$ and $\partial H/\partial \mathbf{T}$. It can be verified from (66) and (70) that the gradient propagation is done in $O(M)$ linear time and space complexity. As for the objective F, we make no assumption on its concrete form. Actually, the smoothness of F is not even required if only the resultant H is C^2. In other word, the linear-complexity gradient propagation enjoys both efficiency and flexibility. By incorporating it into a common optimizer, we can accomplish spatial-temporal deformation within $\mathfrak{T}_{\mathrm{MINCO}}$ for a wide range of planning purposes while maintaining the local smoothness of the trajectory.

Geometrically Constrained Flight Trajectory Optimization

In this section, we provide an unified framework for flight trajectory optimization with different time regularization

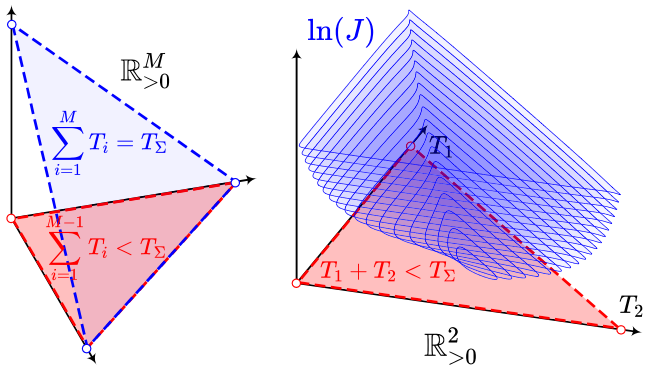


Figure 2. The left illustration shows the domain of J on an M-piece trajectory with total time fixed as T_Σ. The domain is indeed the relative interior of an $(M-1)$-simplex in $\mathbb{R}^M_{>0}$. The right one gives the contour of $\ln J$ with $M = 3$. The function goes to infinity as the time vector approaches the boundary of the open domain in $\mathbb{R}^2_{>0}$.

$\rho(T)$, spatial constraints $\tilde{\mathcal{F}}$ and continuous-time constraints $\mathcal{G}$. This framework indeed relaxes the original problem into the trajectory class $\mathfrak{T}_{\mathrm{MINCO}}$. The spatial-temporal deformation is utilized to meet various feasibility constraints while maintaining the local smoothness. For geometrical constraints, an elimination method is proposed such that the spatial-temporal deformation is free from the *Combinatorial Difficulty* of inequalities (Nocedal and Wright 2006). For continuous-time constraints on the trajectory, a time integral penalty functional is proposed to ensure the feasibility under a prescribed resolution without sacrificing the scalability. Finally, an efficient and flexible gradient propagation turns the constrained optimization into an unconstrained one which can be reliably solved.

Temporal Constraint Elimination

For a trajectory in $\mathfrak{T}_{\mathrm{MINCO}}$ defined by $\mathbf{q}$ and $\mathbf{T}$, the time regularized control effort in (13) can be written as

$$J(\mathbf{q}, \mathbf{T}) = J_q(\mathbf{q}, \mathbf{T}) + \rho(\|\mathbf{T}\|_1), \tag{71}$$

where $J_q(\mathbf{q}, \mathbf{T}) := J_c(\mathbf{c}(\mathbf{q}, \mathbf{T}), \mathbf{T})$ and $J_c(\mathbf{c}, \mathbf{T})$ is the control effort of a polynomial spline defined by $\mathbf{c}$ and $\mathbf{T}$. An analytic expression of J_c is provided by Bry et al. (2015). For the sake of completeness, we derive the analytic gradient $\partial J_c/\partial \mathbf{c}$ and $\partial J_c/\partial \mathbf{T}$ in Appendix B. Obviously, the evaluation of J_q, $\partial J_q/\partial \mathbf{q}$ and $\partial J_q/\partial \mathbf{T}$ can be done in $O(M)$ time and space, as belonging to the deformation of $\mathfrak{T}_{\mathrm{MINCO}}$.

Now that $\partial J/\partial \mathbf{T}$ can be computed from $\partial J_q/\partial \mathbf{T}$, it is natural to use quasi-Newton methods to optimize $\mathbf{T}$. However, two implicit temporal constraints do harm to the optimization process. The first one is that $J_q(\mathbf{q}, \mathbf{T})$ is defined on $\mathbb{R}^M_{>0}$ which requires the strict positiveness of each entry in $\mathbf{T}$. The second one is that $\rho = \rho_f$ in (15) adds an extra linear constraint on $\mathbf{T}$, which further restricts the domain of J.

The first temporal constraint comes from the fact that J_q behaves like a barrier function, i.e., J_q goes to infinity as any T_i approaches to 0^+ for nontrivial $\mathbf{q}$ where there are no consecutive repeating points. It results from the unbounded magnitude of high order derivatives on the trajectory according to the mean value theorem. The second temporal constraint originates from ρ_f which makes the

feasible region much more limited. As illustrated in Figure 2, the domain of J is bounded by $\sum_{i=1}^{M-1} T_i < T_\Sigma$. In these cases, the ill conditioning of Hessian and the clipping of infeasible step at the domain boundary both slow down the convergence of the optimization (Nocedal and Wright 2006).

We propose a diffeomorphism based method to eliminate these temporal constraints for both kinds of time regularization. Consider the following domain for ρ_f in (15),

$$\mathcal{T}_f = \left\{ \mathbf{T} \in \mathbb{R}^M \;\middle|\; \|\mathbf{T}\|_1 = T_\Sigma,\ \mathbf{T} \succ \mathbf{0} \right\}. \tag{72}$$

It is easy to know that $J(\mathbf{q}, \cdot)$ takes a finite value for a nontrivial $\mathbf{q}$ if and only if $\mathbf{T} \in \mathrm{RelInt}(\mathcal{T}_f)$, i.e., the relative interior of $\mathcal{T}_f$. We eliminate this domain constraint by using the diffeomorphism given in the below proposition.

Proposition 1. *The set $\mathcal{T}_f$ given in (72) is diffeomorphic to $\mathbb{R}^{M-1}$. Denote by $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_{M-1})$ an element in $\mathbb{R}^{M-1}$. The C^∞ diffeomorphism is given by the following map,*

$$T_i = \frac{e^{\tau_i}}{1 + \sum_{j=1}^{M-1} e^{\tau_j}} T_\Sigma, \tag{73}$$

for any $1 \leq i < M$, and

$$T_M = T_\Sigma - \sum_{i=1}^{M-1} T_i. \tag{74}$$

Proof. See Appendix C. □

Using the diffeomorphism in Proposition 1, we can directly optimize the cost function J in the whole $\mathbb{R}^{M-1}$ of $\boldsymbol{\tau}$, because the aforementioned two temporal constraints from J_q and ρ_f are always satisfied by default.

While optimizing $\boldsymbol{\tau}$, the forward cost evaluation and the backward gradient propagation are both needed. In the forward direction, the evaluation of $\mathbf{T}$ for any $\boldsymbol{\tau}$ directly follows the analytic expression of the diffeomorphism. Thus the cost function J can also be evaluated. In the backward direction, we show that the gradient propagation also has analytic form. We partition the gradient as $\partial J_q / \partial \mathbf{T} = (g_a^{\mathrm{T}}, g_b)^{\mathrm{T}}$, where the sub-block $g_a \in \mathbb{R}^{M-1}$ and $g_b \in \mathbb{R}$. Differentiating (73) and (74) gives the Jacobian of $\mathbf{T}$ over $\boldsymbol{\tau}$. Directly product it with $\partial J_q / \partial \mathbf{T}$ yields the gradient of J w.r.t. $\boldsymbol{\tau}$,

$$\frac{\partial J}{\partial \boldsymbol{\tau}} = \frac{(g_a - g_b \cdot \mathbf{1}) \circ e^{[\boldsymbol{\tau}]}}{1 + \|e^{[\boldsymbol{\tau}]}\|_1} - \frac{\left(g_a^{\mathrm{T}} e^{[\boldsymbol{\tau}]} - g_b \|e^{[\boldsymbol{\tau}]}\|_1\right) e^{[\boldsymbol{\tau}]}}{\left(1 + \|e^{[\boldsymbol{\tau}]}\|_1\right)^2}, \tag{75}$$

where $e^{[\cdot]}$ is the entry-wise natural exponential map. The notation $\mathbf{1}$ denotes an all-ones vector with an appropriate length. The forward cost evaluation and backward gradient propagation are frequently needed in time allocation optimization. We deduce analytic expressions for both of them. Moreover, both parts only need $O(M)$ linear time and space complexity as can be seen from their expressions.

It is often the case that the optimization needs to start from an initial guess $\mathbf{T}$. In such a case, the backward evaluation of $\boldsymbol{\tau}$ for a given $\mathbf{T}$ needs to be called only once during initialization. This can be done by directly using the inverse map of the diffeomorphism, given by $\tau_i = \ln(T_i / T_M)$ for $1 \leq i < M$.

When the time regularization is used as $\rho = \rho_s$ in (14), there only remains a temporal constraint of the first kind, i.e., $\mathbf{T} \succ \mathbf{0}$. In such a case, it suffices to use $\mathbf{T} = e^{[\boldsymbol{\tau}]}$ as the diffeomorphism between $\mathbb{R}^M$ and $\mathbb{R}^M_{>0}$. The forward cost evaluation and backward gradient propagation in this case are much simpler than those with fixed total time. Note that the gradient of $\rho_s(\|\mathbf{T}\|)_1)$ should also be included.

All temporal constraints are eliminated for either ρ_f or ρ_s, using explicit diffeomorphisms. For any case, we denote by $\mathbf{T}(\boldsymbol{\tau})$ the diffeomorphism from $\boldsymbol{\tau}$ to $\mathbf{T}$. We can conduct unconstrained optimization on $\boldsymbol{\tau}$ to minimize $J(\mathbf{q}, \mathbf{T}(\boldsymbol{\tau}))$ hereafter. It is worth noting that our constraint elimination method may turn a convex problem into a nonconvex one. Nevertheless, the original function $J(\mathbf{q}, \mathbf{T})$ is already nonconvex considering the structure of (59). Therefore, the only concern is whether the diffeomorphism $\mathbf{T}(\boldsymbol{\tau})$ generates new local minima in the space of $\boldsymbol{\tau}$ or eliminates local minima previously existed in the space of $\mathbf{T}$. Technically, the underlying property of diffeomorphism ensures that these two problems do not occur, as shown in below proposition.

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 D. □

Naturally, $J(\mathbf{q}, \mathbf{T})$ can be viewed as a function of top $M-1$ pieces of time when $\rho = \rho_f$ or the whole time vector $\mathbf{T}$ when $\rho = \rho_s$. The corresponding open domain is evident. According to Proposition 2, it is clear that the diffeomorphism preserves the first/second-order necessary optimality conditions and the second-order sufficient optimality conditions (Nocedal and Wright 2006). Now that only the information up to second order is considered here, 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.

Spatial Constraint Elimination

The geometrical constraints $\tilde{\mathcal{F}}$ should also be enforced to ensure a collision-free trajectory. It requires the whole continuous-time trajectory to be contained by an union of obstacle-free geometrical regions. This requirement is fulfilled in two stages. In the first stage, intermediate points are sequentially assigned to these convex regions so that the trajectory approximately lies in free space. In the second stage, the whole trajectory is contained by the geometrical constraints under any given relative resolution. The first stage is detailed here while the second stage follows the same methodology for continuous-time nonlinear constraints $\mathcal{G}$.

Consider the constraint $q \in \mathcal{P}$ where $q \in \mathbb{R}^n$ is a point and the set $\mathcal{P} \subset \mathbb{R}^n$ is a closed ball or a closed convex polytope.

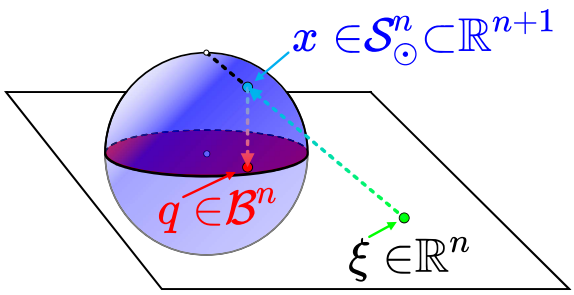


Figure 3. The inverse stereographic projection f_s maps the Euclidean space $\mathbb{R}^n$ onto a sphere without north pole $\mathcal{S}^n_\odot$ in an $(n+1)$-dimensional space. The orthographic projection f_o maps $\mathcal{S}^n_\odot$ onto an n-dimensional ball $\mathcal{B}^n$. The variable ξ moves 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.

The dimension n of the constraint is not necessarily identical to m because a low dimension constraint $n \leq m$ can also be used in $\mathbb{R}^m$. For nonconvex problems with constraints of this type, general methods like barrier method need to solve subproblems with additional parameters. In our paper, we aim to handle the constraints in a lightweight fashion, utilizing the geometrical property of constraints in common planning scenarios. Therefore, we propose a spatial constraint elimination method here to preserve the exact description of the hard constraints.

If $\mathcal{P}$ is a closed ball $\mathcal{P}^{\mathcal{B}}$ centered at point o with radius r,

$$\mathcal{P}^{\mathcal{B}} = \left\{ x \in \mathbb{R}^n \;\middle|\; \|x - o\|_2 \leq r \right\}, \tag{76}$$

We utilize a smooth surjection to map $\mathbb{R}^n$ to $\mathcal{P}^{\mathcal{B}}$. Then, the optimization over the unconstrained Euclidean space implicitly satisfying the constraint described by $\mathcal{P}^{\mathcal{B}}$. As illustrated in Figure 3, the smooth surjection 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 an unit n-sphere without the north pole, i.e.,

$$\mathcal{S}^n_\odot = \left\{ x \in \mathbb{R}^{n+1} \;\middle|\; \|x\|_2 = 1,\ x_{n+1} < 1 \right\}. \tag{77}$$

The inverse stereographic projection f_s is define as

$$f_s(x) = \frac{(2 \cdot x^{\mathrm{T}}, x^{\mathrm{T}}x - 1)^{\mathrm{T}}}{x^{\mathrm{T}}x + 1} \in \mathcal{S}^n_\odot,\ \forall x \in \mathbb{R}^n. \tag{78}$$

Note that f_s is a diffeomorphism between $\mathbb{R}^n$ and $\mathcal{S}^n_\odot$ (Lee 2012). It is obvious that we can project $\mathcal{S}^n_\odot$ from $\mathbb{R}^{n+1}$ back in $\mathbb{R}^n$ to obtain a closed unit ball

$$\mathcal{B}^n = \left\{ x \in \mathbb{R}^n \;\middle|\; \|x\|_2 \leq 1 \right\}. \tag{79}$$

The map is described by

$$f_o(x) = (x_1, \ldots, x_n)^{\mathrm{T}} \in \mathcal{B}^n,\ \forall x \in \mathcal{S}^n_\odot. \tag{80}$$

Note that f_o is 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

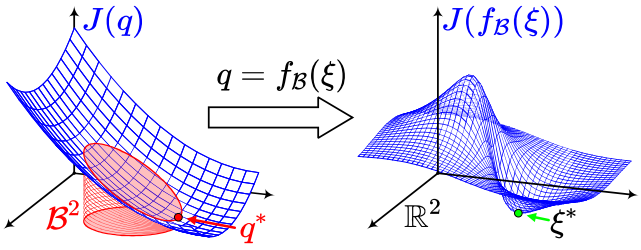


Figure 4. This illustration shows the constrained minimum q^* of a convex function $J(q)$ within a 2-dimensional ball. Transformed by $f_{\mathcal{B}}$, the resultant function $J(f_{\mathcal{B}}(\xi))$ becomes nonconvex but it preserves the local minimum ξ^* satisfying $q^* = f_{\mathcal{B}}(\xi^*)$ with no additional local minimum introduced.

composition of f_s, f_o, and a linear transformation, is also a surjection:

$$f_{\mathcal{B}}(x) = o + \frac{2r \cdot x}{x^{\mathrm{T}}x + 1} \in \mathcal{P}^{\mathcal{B}},\ \forall x \in \mathbb{R}^n. \tag{81}$$

From the geometrical interpretation of f_s and f_o, we can know that $f_{\mathcal{B}}$ maps both regions outside and inside $\mathcal{B}^n$ onto $\mathrm{Int}(\mathcal{P}^{\mathcal{B}})$. The boundary of $\mathcal{B}^n$ is directly mapped onto the boundary of $\mathcal{P}^{\mathcal{B}}$.

The smooth surjection $f_{\mathcal{B}}$ introduces a new coordinate, denoted by ξ, such that optimizing ξ over $\mathbb{R}^n$ always satisfies the constraint on q described by $\mathcal{P}^{\mathcal{B}}$. More importantly, the boundary of $\mathcal{P}^{\mathcal{B}}$ is also attainable. 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 $\mathbf{q}$. Due to the independence between all q_i, all forward cost evaluation and backward gradient propagation can be done separately. In the forward direction, each q_i is evaluated as $f_{\mathcal{B}}(\xi_i)$ using the corresponding center o_i and radius r_i, thus the cost evaluation can be obtained through original decision variables. In the backward direction, the gradient $\partial J/\partial \mathbf{q} = \partial J_q/\partial \mathbf{q}$ is already computed through our general gradient propagation for $\mathfrak{T}_{\mathrm{MINCO}}$. Denote by g_i the i-th component $\partial J/\partial q_i$ in $\partial J/\partial \mathbf{q}$. Using the Jacobian of $f_{\mathcal{B}}$, the gradient is propagated as

$$\frac{\partial J}{\partial \xi_i} = \frac{2r_i \cdot g_i}{\xi_i^{\mathrm{T}}\xi_i + 1} - \frac{4r_i(\xi_i^{\mathrm{T}} g_i) \cdot \xi_i}{(\xi_i^{\mathrm{T}}\xi_i + 1)^2}. \tag{82}$$

The propagation is computed for each q_i, thus it also has an $O(M)$ linear time and space complexity.

When the optimization needs to start from an initial guess $\mathbf{q}$, the backward evaluation of $\boldsymbol{\xi}$ for the given $\mathbf{q}$ needs to be called only once in initialization. This can be done by using a local inverse of $f_{\mathcal{B}}$, given by

$$\xi_i = \frac{r_i - \sqrt{r_i^2 - \|q_i - o_i\|_2^2}}{\|q_i - o_i\|_2^2} \cdot (q_i - o_i) \tag{83}$$

for $1 \leq i < M$.

Similarly, we analyze influences that the smooth surjection $f_{\mathcal{B}}$ imposes onto the constrained local minima in $\mathcal{P}^{\mathcal{B}}$. Unlike diffeomorphisms, $f_{\mathcal{B}}$ lacks the one-to-one correspondence because any point in the boundary of $\mathcal{P}^{\mathcal{B}}$, denoted by $\partial\mathcal{P}^{\mathcal{B}}$, is a critical point where the Jacobian of $f_{\mathcal{B}}$ is singular. The below proposition shows that $f_{\mathcal{B}}$ actually preserves the first-order necessary optimality condition even in $\partial\mathcal{P}^{\mathcal{B}}$.

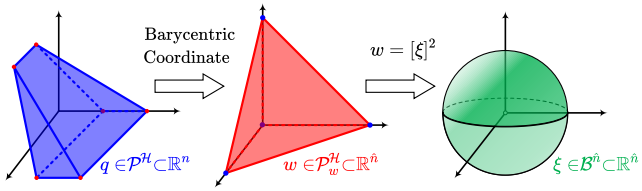


Figure 5. The illustration gives transformations on a convex polytope. A convex polytope $\mathcal{P}^{\mathcal{H}}$ with $\hat{n}+1$ vertices is indeed a standard $\hat{n}$-simplex in the barycentric coordinate. The simplex $\mathcal{P}_w^{\mathcal{H}}$ is then the image of an entry-wise square map $[\cdot]^2$ with ball-shaped domain, which can be eliminated as in Figure 3.

Proposition 3. *Denote by $F : \mathbb{R}^n \mapsto \mathbb{R}$ any C^2 function with $n \geq 2$. For any $o \in \mathbb{R}^n$ and $r \in \mathbb{R}_{>0}$, define a smooth surjection $f_{\mathcal{B}}$ as (81) and a function $G : \mathbb{R}^n \mapsto \mathbb{R}$ as*

$$G(x) = (x-o)^{\mathrm{T}}(x-o) - r^2. \tag{84}$$

Besides, define a composite function $H(x) = F(f_{\mathcal{B}}(x))$ for any $x \in \mathbb{R}^n$. The following statements always hold:

- *If q is a local minimum of $F(x)$ subject to $G(x) \leq 0$, then $\nabla H(\xi) = \mathbf{0}$ holds for any ξ satisfying $f_{\mathcal{B}}(\xi) = q$.*
- *If ξ is a local minimum of $H(x)$, then $q = f_{\mathcal{B}}(\xi)$ satisfies Karush–Kuhn–Tucker (KKT) conditions considering minimization on $F(x)$ subject to $G(x) \leq 0$.*

Proof. See Appendix E. □

Proposition 3 ensures that any local minimum, either in $\mathcal{P}^{\mathcal{B}}$ or in $\mathbb{R}^n$, guarantees the first-order necessary optimality condition to hold in the other one. As is illustrated in Figure 4, the constrained minimum within a 2-dimensional ball is transformed into an unconstrained minimum in the Euclidean space.

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

$$\mathcal{P}^{\mathcal{H}} = \left\{ x \in \mathbb{R}^n \;\middle|\; \mathbf{A}x \preceq b \right\}. \tag{85}$$

where $\mathrm{Int}(\mathcal{P}^{\mathcal{H}}) \neq \varnothing$ according to (17). Traditional methods uses the $\mathcal{H}$-representation for $\mathcal{P}^{\mathcal{H}}$, i.e., bounded intersection of halfspaces in a Cartesian coordinate, which is treated as linear inequality constraints in common optimization algorithms. However, due to the difficulties in a nonconvex cost functional J and nonlinear constraints $\mathcal{G}$, we intend to eliminate direct constraints on parameters of $\mathfrak{T}_{\mathrm{MINCO}}$ such that the trajectory can be freely deformed. To achieve this, we use the $\mathcal{V}$-representation for $\mathcal{P}^{\mathcal{H}}$ instead, where any point $q \in \mathcal{P}^{\mathcal{H}}$ has a (general) barycentric coordinate, i.e., a convex combination of vertices. To obtain the vertices, we apply the efficient convex hull algorithm (Barber et al. 1996) to the dual of $\mathcal{P}^{\mathcal{H}}$ based on an interior point calculated by Seidel's algorithm (Seidel 1991). Note that this procedure produces negligible overhead in our case ($n \leq 4$).

Now we present the procedure to eliminate a polytope constraint as is illustrated in Figure 5. We denote all $\hat{n}+1$ vertices of $\mathcal{P}^{\mathcal{H}}$ by $(v_0, \ldots, v_{\hat{n}})$, 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 $\hat{\mathbf{V}} = (\hat{v}_1, \ldots, \hat{v}_{\hat{n}})$, then the position for any intermediate point is calculated as

$$q = v_0 + \hat{\mathbf{V}}w, \tag{86}$$

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

$$\mathcal{P}_w^{\mathcal{H}} = \left\{ w \in \mathbb{R}^{\hat{n}} \;\middle|\; w \succeq \mathbf{0},\ \|w\|_1 \leq 1 \right\}. \tag{87}$$

According to the *Main Theorem for Polytopes* (Ziegler 2012), the set $\mathcal{P}_w^{\mathcal{H}}$ and the polytope $\mathcal{P}^{\mathcal{H}}$ are totally equivalent through the coordinate transformation in (86), which means

$$\mathcal{P}^{\mathcal{H}} = \left\{ v_0 + \hat{\mathbf{V}}w \;\middle|\; w \in \mathcal{P}_w^{\mathcal{H}} \right\}. \tag{88}$$

In other words, the polytope constraint is exactly converted into a standard $(\hat{n}+1)$-simplex constraints. It is done by simply adding auxiliary variables and a linear map into the original decision variables 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 standard $(\hat{n}+1)$-simplex constraint (87) can also be eliminated using nonlinear transformations. We first use an entry-wise square map $[\cdot]^2 : \mathbb{R}^{\hat{n}} \mapsto \mathbb{R}^{\hat{n}}$ proposed by Sisser (1981) to eliminate nonnegativity constraints by substituting $w = [x]^2$. Consequently, the constraint $\mathcal{P}_w^{\mathcal{H}}$ on w is transformed into a closed unit ball $\mathcal{B}^{\hat{n}}$ on x,

$$\mathcal{B}^{\hat{n}} = \left\{ x \in \mathbb{R}^{\hat{n}} \;\middle|\; \|x\|_2 \leq 1 \right\}. \tag{89}$$

Consequently, we can utilize the previously-analyzed smooth surjection $f_{\mathcal{B}}$ given in (81) again. The composition of the barycentric coordinate, the entry-wise square map and $f_{\mathcal{B}}$ yields a smooth surjection $f_{\mathcal{H}}$ from $\mathbb{R}^{\hat{n}}$ onto $\mathcal{P}^{\mathcal{H}}$:

$$f_{\mathcal{H}}(x) = v_0 + \frac{4\hat{\mathbf{V}}[x]^2}{(x^{\mathrm{T}}x+1)^2} \in \mathcal{P}^{\mathcal{H}},\ \forall x \in \mathbb{R}^{\hat{n}}. \tag{90}$$

The smooth surjection $f_{\mathcal{H}}$ introduces a new coordinate, denoted by ξ. Similarly, optimizing ξ over $\mathbb{R}^{\hat{n}}$ ensures that $q \in \mathcal{P}^{\mathcal{H}}$ is always satisfied. Besides, the boundary of $\mathcal{P}^{\mathcal{H}}$ is also attainable. In the forward direction, the new coordinate $\boldsymbol{\xi}$ for the whole $\mathbf{q}$ can be obtained by evaluating $q_i = f_{\mathcal{H}}(\xi_i)$ for each given ξ_i. Then the cost is evaluated using the computed $\mathbf{q}$. In the backward direction, the gradient propagation uses the Jacobian of $f_{\mathcal{H}}$. Denote by g_i the i-th gradient $\partial J/\partial q_i$ in $\partial J/\partial \mathbf{q}$, then

$$\frac{\partial J}{\partial \xi_i} = \frac{8\xi_i \circ \hat{\mathbf{V}}^{\mathrm{T}} g_i}{(\xi_i^{\mathrm{T}}\xi_i+1)^2} - \frac{16 g_i^{\mathrm{T}}\hat{\mathbf{V}}[\xi_i]^2}{(\xi_i^{\mathrm{T}}\xi_i+1)^3} \cdot \xi_i, \tag{91}$$

The propagation requires computations for each q_i, forming an $O(M)$ linear time and space complexity operation.

When an initial guess $\mathbf{q}$ is specified for the optimization, the backward evaluation of $\boldsymbol{\xi}$ for the given $\mathbf{q}$ needs to be called only once in initialization. This requires finding an local inverse of $f_{\mathcal{H}}$. To accomplish this, we first compute the barycentric coordinate of each q_i by following the analytic approach from Warren et al. (2007). 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 smooth surjection $f_{\mathcal{H}}$ is a composite map, of which the entry-wise square map $[\cdot]^2$ brings additional nonlinearity into optimization. Fortunately, the following proposition shows that it totally preserves constrained local minima during the transformation from the simplex $\mathcal{P}_w^{\mathcal{H}}$ to a ball $\mathcal{B}^{\hat{n}}$.

Proposition 4. *Consider the map $f_c : \mathbb{R}^{\hat{n}} \mapsto \mathbb{R}^{\hat{n}}_{\geq 0}$ defined as $f_c(x) = [x]^2$, the set $\mathcal{P}_w^{\mathcal{H}}$ defined in (87) and the set $\mathcal{B}^{\hat{n}}$ defined in (89). For any C^2 function $F : \mathbb{R}^{\hat{n}} \mapsto \mathbb{R}$, define a function $H(x) = F(f_c(x))$ for any $x \in \mathbb{R}^{\hat{n}}$. Then, the following two statements are equivalent.*

- *$w \in \mathbb{R}^n$ is a local minimum of F over $\mathcal{P}_w^{\mathcal{H}}$;*
- *There exists a local minimum $v \in \mathbb{R}^{\hat{n}}$ of H over $\mathcal{B}^{\hat{n}}$ such that $f_c(v) = w$.*

Proof. See Appendix F. □

It is confirmed that the additional nonlinearity in $f_{\mathcal{H}}$ does not exclude the desired minimum or produce undesired minimum practically. One drawback is that saddle points not present in original problem can also be produced by $f_{\mathcal{H}}$. We deal with this issue by invoking a local search on the computed solution (Gill et al. 1981). Intuitively, perturbations are applied to find if there is any improved solution near the obtained one.

All direct spatial constraints on decision variables are eliminated for either $\mathcal{P}^{\mathcal{B}}$ or $\mathcal{P}^{\mathcal{H}}$ cases, using smooth surjections. For any case, we denote by $\mathbf{q}(\boldsymbol{\xi})$ the smooth map from $\boldsymbol{\xi}$ to $\mathbf{q}$. We can conduct unconstrained optimization on $\boldsymbol{\xi}$ to minimize $J(\mathbf{q}(\boldsymbol{\xi}), \mathbf{T}(\boldsymbol{\tau}))$ hereafter.

Time Integral Penalty Functional

By eliminating constraints on parameters of $\mathfrak{T}_{\text{MINCO}}$, the trajectory can be deformed freely to further meet general constraints $\mathcal{G}$. As defined in (12), $\mathcal{G}$ contains infinite many inequality constraints, which cannot be directly solved by constrained optimization. Inspired by the constraint transcription (Jennings and Teo 1990), we transform $\mathcal{G}$ into finite constraints via integral of constraint violations. For any trajectory $p : [0, T] \mapsto \mathbb{R}^m$, we define

$$I_{\mathcal{G}}^k[p] := \int_0^T \max\left[\mathcal{G}(p(t), \dots, p^{(s)}(t)), \mathbf{0}\right]^k \mathrm{d}t, \tag{92}$$

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. Note that a line integral can also be implemented by adding a multiplier $\|\dot{p}(t)\|$ to the integrand without affect the structure of our framework. For simplicity, we consider this factor to be included in $\mathcal{G}$. The functional-type constraint is then equivalent to following inequality constraints

$$I_{\mathcal{G}}^k[p] \preceq \mathbf{0}, \tag{93}$$

where $I_{\mathcal{G}}^k[p]$ is indeed a function of trajectory parameters. Note that this function is in general non-smooth if $k \leq 1$. When $k > 1$, the constraints in (93) become degenerate because common constraint qualifications are not satisfied. To address this issue, a smoothing technique is also proposed by Jennings and Teo (1990) to handle (93) while retaining optimality. In consideration of two reasons, we directly adopt an C^2 exterior penalty instead to approximately handle the constraint. Firstly, the integral in (92) can only be evaluated numerically, making the approximation inevitable. Secondly, penalty methods have no requirement on a feasible initial guess which is nontrivial to construct. Thus, we utilize a penalty with $k = 3$ hereafter. By the way, a smoothed exact penalty can still be adopted if a higher resolution is required for constraints.

Consequently, for a trajectory p, we define its time integral penalty functional as

$$I_{\mathcal{G}}[p] := \chi^{\mathrm{T}} I_{\mathcal{G}}^3[p]. \tag{94}$$

where $\chi \in \mathbb{R}^{n_g}_{\geq 0}$ is a vector of penalty weights. It should be noted that the cubic penalty is used here for twice continuous differentiability. Normally, the constant vector χ should contain large entries. 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 satisfied within acceptable tolerance.

Practically, $I_{\mathcal{G}}[p]$ can only be evaluated via quadrature. For a trajectory piece $p_i(t)$ parameterized as (46), the constraint violation $\mathcal{G}(p_i(t), \dots, p_i^{(s)}(t))$ is also a function of $\mathbf{c}_i$ and T_i. To conduct the quadrature, we first define a sampled function $\mathcal{G}_v : \mathbb{R}^{2s\times m} \times \mathbb{R}_{>0} \times [0, 1] \mapsto \mathbb{R}^{n_g}$ as

$$\mathcal{G}_v(\mathbf{c}_i, T_i, \hat{t}) = \mathcal{G}\left(\mathbf{c}_i^{\mathrm{T}}\beta(T_i \cdot \hat{t}), \dots, \mathbf{c}_i^{\mathrm{T}}\beta^{(s)}(T_i \cdot \hat{t})\right), \tag{95}$$

where $\hat{t} \in [0, 1]$ is the normalized timestamp. The quadrature of $I_{\mathcal{G}}[p_i]$, denoted by $I : \mathbb{R}^{2s\times m} \times \mathbb{R}_{>0} \times \mathbb{Z}_{\geq 0} \mapsto \mathbb{R}_{>0}$, is computed as a weighted sum of the penalized sampled function:

$$I(\mathbf{c}_i, T_i, \kappa_i) = \frac{T_i}{\kappa_i} \sum_{j=0}^{\kappa_i} \bar{\omega}_j \chi^{\mathrm{T}} \max\left[\mathcal{G}_v(\mathbf{c}_i, T_i, \frac{j}{\kappa_i}), \mathbf{0}\right]^3, \tag{96}$$

where the constant integer κ_i directly controls the relative resolution of the quadrature, and $(\bar{\omega}_0, \bar{\omega}_1, \dots, \bar{\omega}_{\kappa_i-1}, \bar{\omega}_{\kappa_i}) = (1/2, 1, \dots, 1, 1/2)$ are the quadrature coefficients from the trapezoidal rule (Press et al. 2007). Intuitively, $I(\mathbf{c}_i, T_i, \kappa_i)$ gives a reasonable approximation to $I_{\mathcal{G}}[p_i]$, whose precision is adjustable through κ_i.

The quadrature of $I_{\mathcal{G}}[p]$ is then defined as

$$I_{\Sigma}(\mathbf{c}, \mathbf{T}) = \sum_{i=1}^{M} I(\mathbf{c}_i, T_i, \kappa_i). \tag{97}$$

Apparently, the C^2 continuity of $I_{\Sigma}(\mathbf{c}, \mathbf{T})$ is preserved, making its second-order information usable. The forward penalty evaluation follows the analytical form in (97). The backward gradient w.r.t. $\mathbf{c}$ and $\mathbf{T}$ can also be analytically derived from $I_{\Sigma}(\mathbf{c}, \mathbf{T})$. Once the computation is done, the penalty values or gradients are directly combined as one for each piece. Thus, the output size is fully determined by M, which is irrelevant to any κ_i. Finally, the quadrature of $I_{\mathcal{G}}[p]$ for a trajectory in $\mathfrak{T}_{\text{MINCO}}$ is computed as $I_{\Sigma}(\mathbf{c}(\mathbf{q}, \mathbf{T}), \mathbf{T})$. Its gradient computation only requires $O(M)$ time and space, which is the same as the deformation of $\mathfrak{T}_{\text{MINCO}}$.

Trajectory Optimization via Unconstrained NLP

Due to the general nonlinearity in (13), the optimal trajectory parameterization is generally hard to known. Unlike traditional methods approximating optimal solutions via large numbers of variables and constraints (Betts 2010), we propose a lightweight relaxed trajectory optimization of $\mathfrak{T}_{\text{MINCO}}$ via an unconstrained NLP, where the spatial-temporal deformation is applied to meet general constraints.

The relaxation to (13) is defined as

$$\min_{\boldsymbol{\xi},\boldsymbol{\tau}} \; J(\mathbf{q}(\boldsymbol{\xi}),\mathbf{T}(\boldsymbol{\tau}))+I_{\Sigma}(\mathbf{c}(\mathbf{q}(\boldsymbol{\xi}),\mathbf{T}(\boldsymbol{\tau})),\mathbf{T}(\boldsymbol{\tau})), \quad (98)$$

where J is the time-regularized control effort (71) for $\mathfrak{T}_{\text{MINCO}}$ and I_{Σ} is the quadrature of penalty functional (97) for splines. In comparison with the original optimization (13), the relaxation uses compact parameterization of $\mathfrak{T}_{\text{MINCO}}$, constraint elimination by variable transformations, and softened penalty for continuous-time constraints. It is important to note that any task-specific requirement on a trajectory, other than user-defined constraints in $\mathcal{G}$, can also be added into (98) without affecting the structure of our framework.

Firstly, the optimization is conducted within $\mathfrak{T}_{\text{MINCO}}$. If the time vector is fixed, $\mathfrak{T}_{\text{MINCO}}$ becomes a linear space with $m(M-1)$ dimensions as is shown in the proof of Theorem 2. Its dimension is comparably low considering the entire parameter space of splines. Besides, any element in this space is already an unconstrained minimizer of the control effort. These two properties greatly reduce the difficulty coming from both the dimension and nonlinearity. Secondly, the variable transformations exactly eliminate the need for geometrical constraints on decision variables. The elimination method only serves as a layer in the evaluation for function and gradient, which is both lightweight and flexible. Thirdly, the quadrature of time integral penalty provides an adjustable way to enforce infinite dimensional constraints. For a fixed setting of resolution, the quadrature term itself is also differentiable w.r.t. parameters of $\mathfrak{T}_{\text{MINCO}}$, making it convenient to apply the spatial-temporal deformation under restrictions.

To generate trajectories for any flat multicopter, the maps Ψ_x in (2) and Ψ_u in (3) should be known in advance. Utilizing the optimality conditions, we parameterize a trajectory into $\mathbf{q}$ and $\mathbf{T}$. After assigning a fixed number of pieces into each $\mathcal{P}_i$, the variable transformation is then applied to eliminate all direct constraints. User-defined state-input constraints on systems are also transformed into $\mathcal{G}$ via Ψ_x and Ψ_u. Finally, we obtain the relaxed cost function in (98). Apparently, the gradient computation is also derived for all layers except $\mathcal{G}$. Now that $\mathcal{G}$ is composed by system constraints, Ψ_x, and Ψ_u, deriving its analytic gradient enjoys the same complexity as evaluating $\mathcal{G}$ analytically (Baur and Strassen 1983). One can either utilize the mature *Automatic Differentiation* (AD) techniques as detailed by Griewank and Walther (2008) or derive it analytically by following the reverse mode of AD. After obtaining all the gradients, the relaxation (98) is then efficiently solved by the Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS) (Liu and Nocedal 1989) with a modified updating rule (Li and Fukushima 2001) to ensure stability of the nonconvex minimization.

Applications

Large-Scale Unconstrained Control Effort Minimization

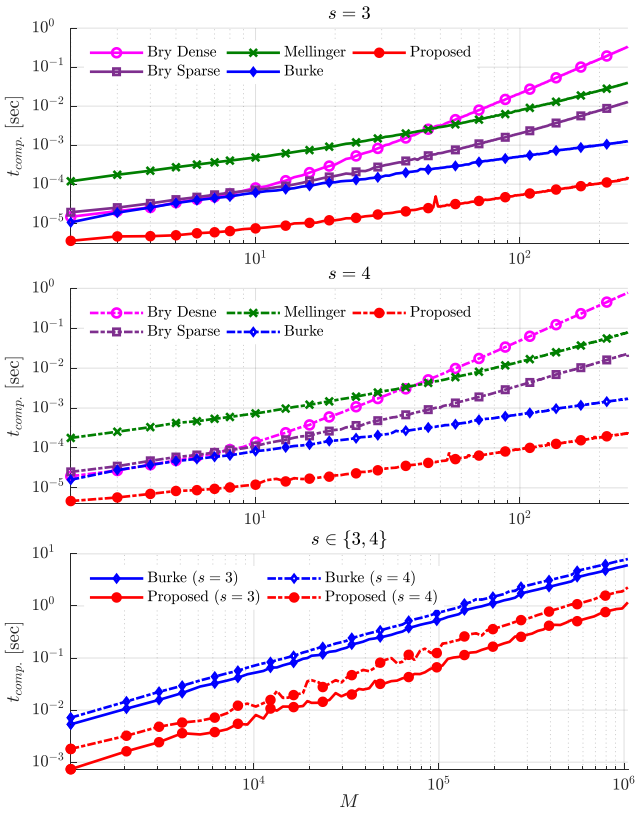


Figure 6. Computation time $t_{comp.}$ under different piece numbers M. The top and the middle figure give the performance for jerk minimization ($s = 3$) and snap minimization ($s = 4$), respectively. The bottom figure shows the efficiency of two linear-complexity schemes for very large-scale problems.

The concerned unconstrained problem (20) is a classic topic since the work by Bry et al. (2015), which has long been analyzed and solved from the perspective of optimization. In this paper, the solution can be directly constructed following the evaluation of $\mathfrak{T}_{\text{MINCO}}$, thus it is possible to solve very large-scale trajectory when there is no extra constraint.

To demonstrate the efficiency of our scheme, we benchmark it with several traditional ones, including the QP formulation by Mellinger and Kumar (2011), the closed-form solution by Bry et al. (2015), and the linear-complexity solution by Burke et al. (2020) where the problem structure is exploited. As for Mellinger's scheme, we utilize the operator splitting solver (Stellato et al. 2020) to solve the QP with equality and inequality constraints. As for Bry's scheme, the closed-form solution is evaluated through a dense solver as well as a sparse solver (Demmel et al. 1999). As for Burke's scheme, we re-implement their algorithm here such that the fairness is ensured. Technically, the re-implementation cost less than 10 seconds for snap minimization with $500,000$ waypoints, while the original one is reported to cost more than 2 minutes (Burke et al. 2020). All the involved schemes are implemented in C++11 without any intentional hardware

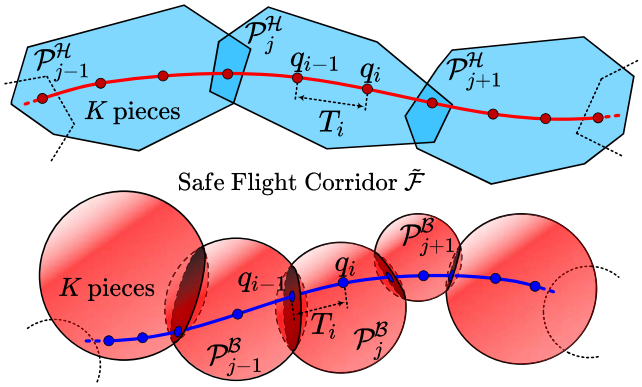


Figure 7. Piece assignment for a trajectory within different kinds of safe flight corridors in $\mathbb{R}^n$. Each geometrical primitive is assigned with K pieces. An intermediate point q_i is assigned to $\mathcal{P}_{\lceil i/K \rceil} \cap \mathcal{P}_{\lceil (i+1)/K \rceil}$. For ball-shaped corridors, a point is further anchored to an $(n-1)$-dimensional disk if it is assigned to the intersection of two n-dimensional balls.

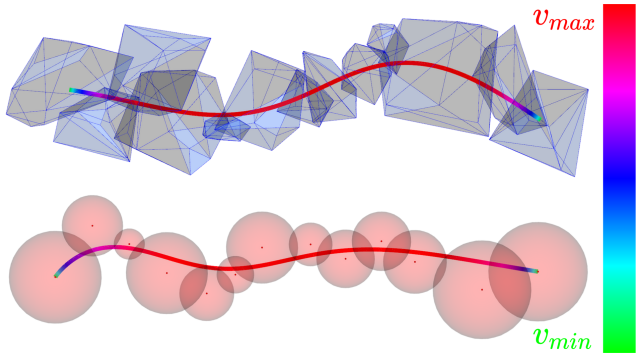


Figure 8. The optimized trajectories within different kinds of 3-dimensional SFCs. The speed profile is colored according to its magnitude. The proposed method generates smooth trajectories within randomly generated SFCs. The speed persistently attains the maximum even if the SFCs are both narrow and twisted.

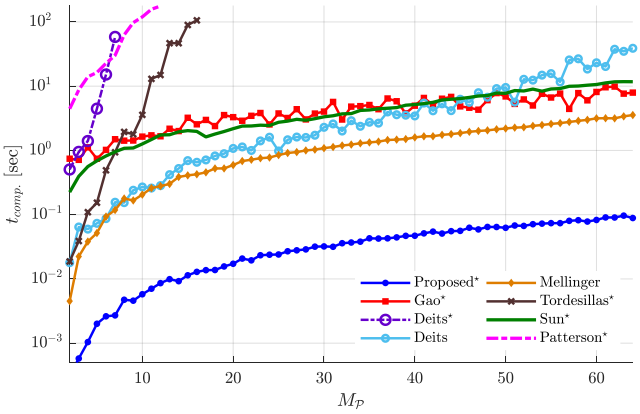


Figure 9. The benchmark on computation efficiency. The Proposed⋆ one outperforms other methods by orders of magnitudes. Methods from Tordesillas⋆ and Deits⋆ suffer from combinatorial explosion, but they are faster than Patterson⋆ on relatively small scale problems. Methods that do not support optimizing time or interval allocation consume less computation time, while they are still slower than the Proposed⋆ one.

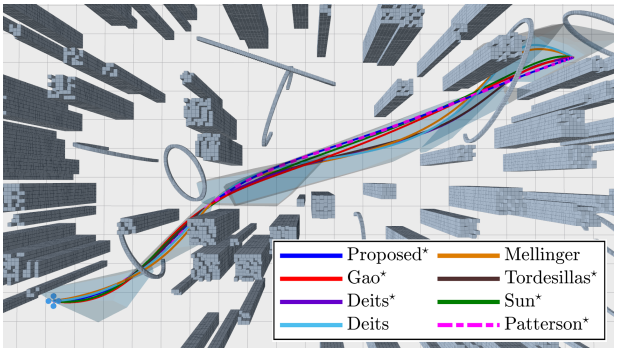


Figure 10. The comparison of trajectories from different methods. The Proposed⋆ one generate a smoother trajectory than the other specialized methods, taking the one from Patterson⋆ as the ground truth.

acceleration. The benchmark is conducted on an Intel Core i7-8700 CPU under Linux.

The performance is reported in Figure 6 for 5 schemes. Both jerk $s = 3$ and snap $s = 4$ are minimized as special cases of the problem (20). Mellinger's scheme (Mellinger and Kumar 2011) only performs better than the dense evaluation of Bry's closed-form solution (Bry et al. 2015) on middle-scale problems $10^1 < M < 10^3$. Burke's scheme (Burke et al. 2020) benefits from its linear complexity, thus it is able to solve large-scale problems ($10^4 < M < 10^6$). The proposed scheme improves the computation speed by at least an order of magnitude against all other ones at any problem scale $10^1 < M < 10^6$, which also possesses $O(M)$ complexity.

In conclusion, our conditions in Theorem 2 provides a practical way to directly construct the solution of problem (20), which possesses simplicity, efficiency, stability and scalability. The resultant trajectory class, $\mathfrak{T}_{\text{MINCO}}$, along with its evaluation and deformation, can not only serve as a reliable submodule of our trajectory optimization framework, but can also promote the performance of other applications where the unconstrained control effort minimization comes into effect.

Trajectory Generation within Safe Flight Corridors

As a special case of our problem (13), optimizing trajectories within 3-dimensional Safe Flight Corridors (SFCs) is widely adopted in various applications (Hönig et al. 2018; Mohta et al. 2018; Gao et al. 2019, 2020). An SFC is usually generated by the *front end* of a motion planning framework as an abstraction of the concerned configuration space. Optimizing dynamically feasible trajectories within SFCs is usually taken as a *back end* of such kind of frameworks.

We assume that an SFC, either polyhedron-shaped or ball-shaped, is already obtained through a frond end such as the Regional Inflation by Line Search (RILS) algorithm by Liu et al. (2017b), the Iterative Regional Inflation by Semidefinite programming (IRIS) by Deits and Tedrake (2015a), the Sphere Flipping based method by Zhong et al. (2020) or the Safe-region RRT* Expansion Algorithm by Gao et al. (2019). For simplicity, we omit the process for generating SFCs in these front ends. Consequently, the configuration of an SFC is naturally described by Eq.(16) and (17).

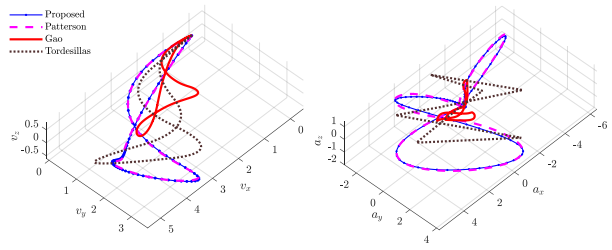


Figure 11. The velocity and acceleration profile for some trajectories in Figure 10. The Proposed* is most consistent with the ground truth from Patterson*. The other two trajectories is comparably conservative and less smooth in higher derivatives.

As is illustrated in Figure 7, we consider different kinds of SFCs. Each geometrical primitive is assigned with K trajectory pieces. That is to say, the numbers of pieces and geometrical primitives satisfy $M = K \cdot M_{\mathcal{P}}$. The i-th trajectory piece $p_i(t) : [0, T_i] \mapsto \mathbb{R}^3$ is assigned to $\mathcal{P}_{\lceil i/K \rceil}$. Following the methodology in previous sections, we treat $\mathbf{q} = (q_1, \ldots, q_{M-1})$ and $\mathbf{T} = (T_1, \ldots, T_M)^{\mathrm{T}}$ as trajectory parameters. Consequently, the i-th intermediate point q_i is assigned to $\mathcal{P}_{\lceil i/K \rceil} \cap \mathcal{P}_{\lceil (i+1)/K \rceil}$. The entire trajectory is automatically determined in $\mathfrak{T}_{\text{MINCO}}$. Applying the constraint elimination, direct constraints on $\mathbf{T}$ and $\mathbf{q}$ are automatically satisfied, such as $\mathbf{T} \in \mathbb{R}_{>0}$ for ρ_s, $\|\mathbf{T}\|_1 < T_\Sigma$ for ρ_f, as well as $q_i \in \mathcal{P}_{\lceil i/K \rceil} \cap \mathcal{P}_{\lceil (i+1)/K \rceil}$ for all i. The continuous-time constraints $\mathcal{G}$ are specified as follows to ensure both safety and dynamic limits:

$$\begin{cases} p_i(t) \in \mathcal{P}_{\lceil i/K \rceil}, & \forall t \in [0, T_i],\ \forall 1 \le i \le M, \\ \|p_i^{(1)}(t)\|^2 \le v_{max}^2, & \forall t \in [0, T_i],\ \forall 1 \le i \le M, \\ \|p_i^{(2)}(t)\|^2 \le a_{max}^2, & \forall t \in [0, T_i],\ \forall 1 \le i \le M, \end{cases} \tag{99}$$

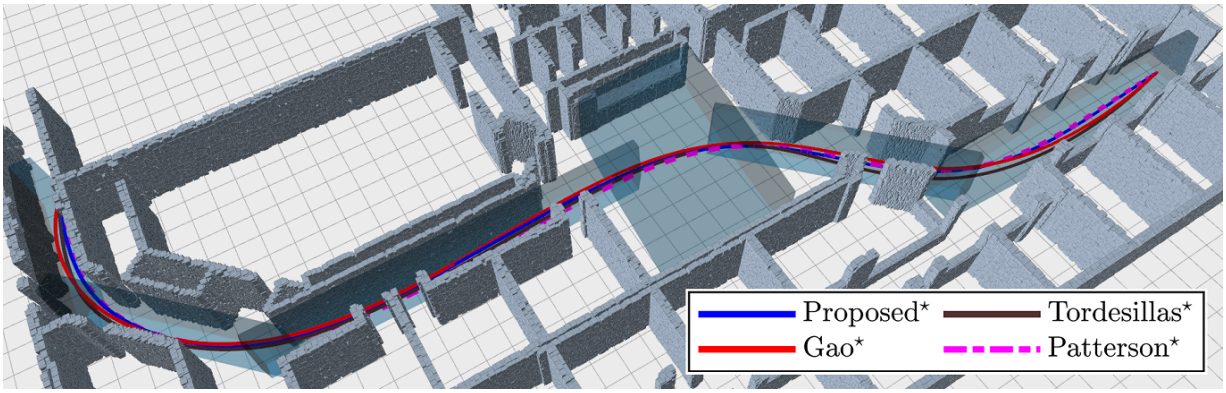
where v_{max} and a_{max} are constants for dynamic limits. The constraints in (99) are either linear or quadratic constraints for a specific t and i, thus the time integral penalty with gradient can be easily derived then applied. Consequently, the trajectory generation in $\tilde{\mathcal{F}}$ can also be conducted by solving the unconstrained NLP in (98). We show some optimization results in Figure 8 for randomly generated SFCs. Both the polyhedron-shaped and the ball-shaped SFCs are handled.

To further show effectiveness of our method, we benchmark several existing methods over polyhedron-shaped SFCs. Technical details for all benchmarked methods, including the proposed one, are listed as follows:

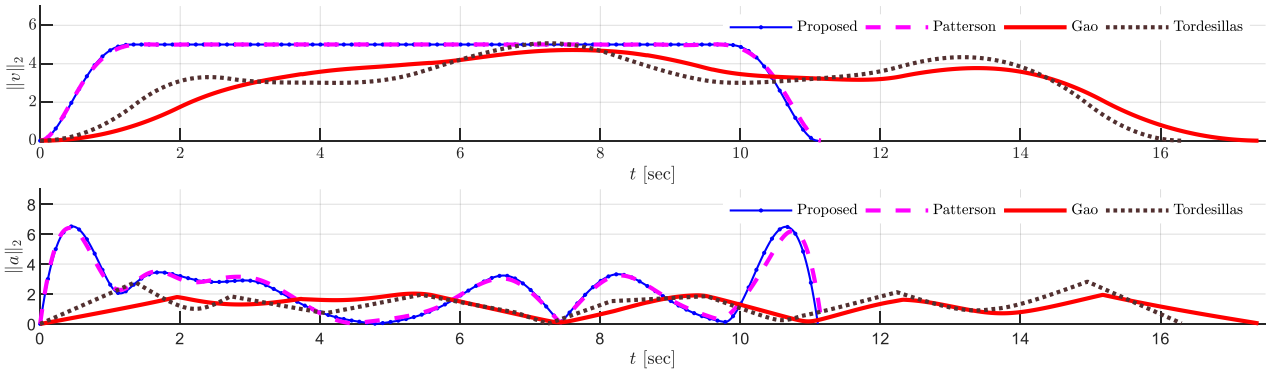
- Proposed*: Jerk minimzation is conducted with either linear time regularization or fixed total time. Constraints in (99) are enforced.
- Gao* (Gao et al. 2020): A geometrical curve is optimized in a QP formed by quadratic cost on jerk and linear safety constraints on control points of Bézier curves. Its temporal profile is then optimized in an SOCP to accomplish TOPP (Verscheure et al. 2009) under dynamic limits in (99).
- Deits* (Deits and Tedrake 2015b): A trajectory is generated via an MISOCP. Its interval allocation is optimized via integer variables. Jerk minimization is conducted. 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: The details are the same as Deits* except that intervals are allocated heuristically, thus no integer variable exists.
- Mellinger (Mellinger and Kumar 2011): 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 (Lynch and Park 2017). Dynamic limits in (99) are enforced by time scaling (Liu et al. 2017b).
- Tordesillas* (Tordesillas et al. 2019): The details are the same as Deits* except that all constraints are enforced through linear constraints on control points of Bézier curves. An MIQP is solved instead. The interval time is also determined by a well-designed algorithm.
- Sun* (Sun et al. 2020): 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.
- Patterson* (Patterson and Rao 2014): A trajectory is optimized in a constrained NLP via Gauss pseudospectral method. The system dynamics is taken as an s-integrator. Each trajectory phase is confined within one polytope. Dynamic limits are enforced through path constraints. The cost functional is set as the same as LQMT problems.

A method is marked with a star if it can optimize either the time allocation or the interval allocation. As for dynamic limits, the values are the same for either L_1-norm or L_2-norm of trajectory derivatives. Thus, constraints are indeed much tighter on methods from the Proposed*, Gao*, Mellinger and Patterson*, that strictly follow constraints in (99). As for total time, Deits* and Sun* need preassigned values. However, no procedure is ever mentioned to determine such a value while ensuring the existence of a feasible solution. Thus, we assign them with same total time from trapezoidal velocity profiles. The method from Patterson* handles the original problem directly, taking advantage of the exponential convergence of global collocation (Patterson and Rao 2014). Therefore, we take its trajectory as the ground truth.

The benchmark is conducted in randomly generated environments as is shown in Figure 10. The SFC 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 the assignment number as $K = 1$, the weight for linear time regularization as $k_\rho = 1024.0$, the dynamic limits as $v_{max} = 5.0m/s, a_{max} = 7.0m/s$, the resolution for time integral as $\kappa_i = 16$, the timeout as 3 minutes, the relative tolerance as 10^{-4}, the initial and terminal condition to be static. Besides, the proposed method is only implemented in a sequential way. Methods from Deits*, Deits, and Sun* utilize the commercial solver Gurobi (Gurobi Optimization, LLC 2020) with 6 threads enabled for parallel computing. Two methods from Gao* and Sun* utilize the commercial solver MOSEK (MOSEK Aps 2020).



(a) Trajectories generated by different methods within a long SFC of the office environment



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

Figure 12. This figure shows 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.

The computation performance is provided in Figure 9. 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⋆ only suits offline scenarios where computation time is far less important than solution quality. The Proposed⋆ method improves the performance by more than an order of magnitude, while retaining optimization on time allocation.

The trajectory quality is provided in Figure 11. Based on the geometrical profile in Figure 10, methods that do not optimize time or interval allocation are more likely to deviate from the ground truth of Patterson⋆. Trajectories from Deits⋆ and Tordesillas⋆ also deviate a lot from the ground truth because of the limited resolution of intervals. Besides, the temporal profile are provided in Figure 11 for four complete methods where all trajectory parameters are considered. The trajectory generated by Gao⋆ is less aggressive than the other three. The trajectory from Tordesillas⋆ has discontinuous jerk since only 3-degree polynomials are used. The results from the Proposed⋆ method have nearly the same quality as the ground truth. To explore the temporal profile under constraints, we also test four complete methods in a long-distance flight as is shown in Figure 12. Profiting from the effectiveness of the time integral penalty functional, our method can also achieve the maximum speed persistently.

The success rates, relative control effort, and flight durations are also provided in Figure 13. Considering the success rate, interval allocation based ones have the lowest robustness. As for control effort, we normalize the effort from each method using that from the Proposed⋆. Its trajectory time is fixed accordingly for the sake of fairness. Clearly, heuristic time or interval allocation causes relatively high control effort. As for flight duration, the solution from the Proposed⋆ one is the closest to the ground truth.

In conclusion, our method achieves comparable solution quality to the collocation based method from Patterson and Rao (2014) in both geometrical and temporal profile of trajectories, while having superior computational speed against all benchmarked ones.

$\mathrm{SE}(3)$ *Motion Planning in Quotient Space*

In complex environments, it is often conservative to take the position in $\mathbb{R}^3$ as the whole configuration space. Safe motion will not exist for narrow spaces unless a multicopter agilely adjusts its rotation to avoid unnecessary collisions. Therefore, we consider $\mathrm{SE}(3)$ motion planning in our framework where a multicopter is considered as a rigid body.

An important property for motion planning in $\mathrm{SE}(3) = \mathbb{R}^3 \times \mathrm{SO}(3)$ is the necessary condition that a feasible pose

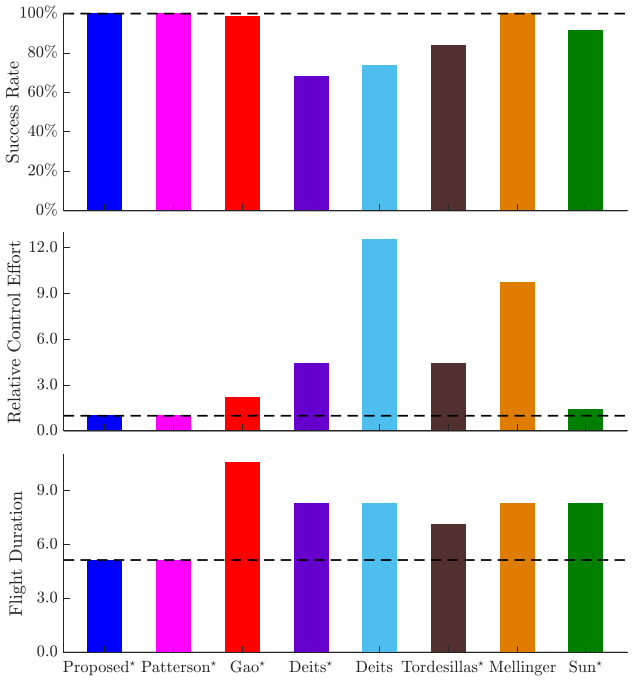


Figure 13. Benchmark on success rates, relative control effort, and flight durations. Methods from Deits* and Tordesillas* have relatively low success rates because they optimize interval allocation which involves integer variables. Methods from Deits and Mellinger have relatively large control effort because optimization on time or interval allocation is not supported. Note that some methods need preassigned total flight duration.

for a rigid body at least contains a feasible translation for a dimensionless point. The subspace $\mathbb{R}^3$ is often referred to as a *Quotient Space* (Orthey and Toussaint 2019). Exploiting such a quotient-space decomposition (Orthey et al. 2018), we only consider the rotational feasibility based on an translational trajectory, instead of handling them jointly. Consequently, we can relax the assumption in Eq. (16), where $\tilde{\mathcal{F}}$ is only assumed to be the obstacle-free region in the quotient space $\mathbb{R}^3$ without considering the actual size of a multicopter.

In this subsection, we consider the simplified dynamics proposed by Mueller et al. (2015) for a multicopter with aligned propellers. Configuration of the vehicle is defined by translation its p and rotation $\mathbf{R}$. Translational motion depends on the gravitational acceleration $\bar{g}$ as well as the thrust $\tilde{f}$ produced by the vehicle. Rotational motion takes the body rate ω as input. The simplified dynamics can be written as

$$\begin{cases} \dot{p} = v, \\ \bar{m}\dot{v} = -\bar{m}\bar{g}e_3 + \mathbf{R}\tilde{f}e_3, \\ \dot{\mathbf{R}} = \mathbf{R}\hat{\omega}. \end{cases} \tag{100}$$

where e_i is the i-th column of $\mathbf{I}_3$ and $\bar{m}$ is 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 $a, b \in \mathbb{R}^3$, where $\hat{a}$ is a skew-symmetric matrix. Moreover, we model the geometrical shape of a symmetric multicopter as its outer Löwner-John ellipsoid (Toth et al. 2017),

$$\mathcal{E}(t) = \left\{ \mathbf{R}(t)\mathbf{Q}x + p(t) \;\middle|\; \|x\|_2 \le 1 \right\} \tag{101}$$

where

$$\mathbf{Q} = \begin{pmatrix} r_e & 0 & 0 \\ 0 & r_e & 0 \\ 0 & 0 & h_e \end{pmatrix}, \tag{102}$$

r_e is the radius of the multicopter and h_e is the height.

The basic requirement for a feasible motion is to ensure safety and dynamic limits. By safety we mean the ellipsoid of multicopter is always confined in the convex decomposition of the obstacle-free region, i.e.,

$$\mathcal{E}(t) \subset \tilde{\mathcal{F}},\ \forall t \in [0, T] \tag{103}$$

where T is the total time of the motion. The safety constraint in (103) is indeed nonconvex and 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 the safety through the following condition,

$$\forall t \in [0, T],\ \exists 1 \le i \le M_p,\ s.t.\ \mathcal{E}(t) \subset \mathcal{P}_i^{\mathcal{H}}. \tag{104}$$

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

$$\begin{cases} \|p^{(1)}(t)\|^2 \le v_{max}^2, & \forall t \in [0, T], \\ f_{min} \le \tilde{f}(t) \le f_{max}, & \forall t \in [0, T], \\ \|\omega(t)\|_2^2 \le \omega_{max}^2, & \forall t \in [0, T]. \end{cases} \tag{105}$$

Given a quotient-space trajectory $p(t) : [0, T] \mapsto \mathbb{R}^3$, the trajectory in $\mathrm{SE}(3)$ is algebraically determined by flatness of the dynamics (100). Denote by $\mathbf{R}(t)$ its rotational part. The algebraic procedure along with gradient propagation is provided in Appendix G for determining $\mathbf{R}(t)$ and $\dot{\mathbf{R}}(t)$ from $p(t)$ and its derivatives. To generate $p(t)$ within $\tilde{\mathcal{F}}$, we follow the aforementioned methodology, except that more complicated constraints are used 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 (19) with $j = \lceil i/K \rceil$. Thus, we need to compute the value and the gradient for constraint functions in both (104) and (105), such that our framework comes into effect. 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 (101). As proposed by Wu et al. (2021), ensuring the safety by confining the vehicle ellipsoid in a polyhedron also has an analytical form. Specifically,

$$\mathcal{E}_i(t) \in \mathcal{P}_j^{\mathcal{H}},\ j = \lceil i/K \rceil,\ \forall t \in [0, T_i], \tag{106}$$

is equivalent to

$$\left[[\mathbf{A}_j \mathbf{R}_i(t)\mathbf{Q}]^2 \mathbf{1} \right]^{\frac{1}{2}} + \mathbf{A}_j p_i(t) - b_j \preceq \mathbf{0}, \tag{107a}$$

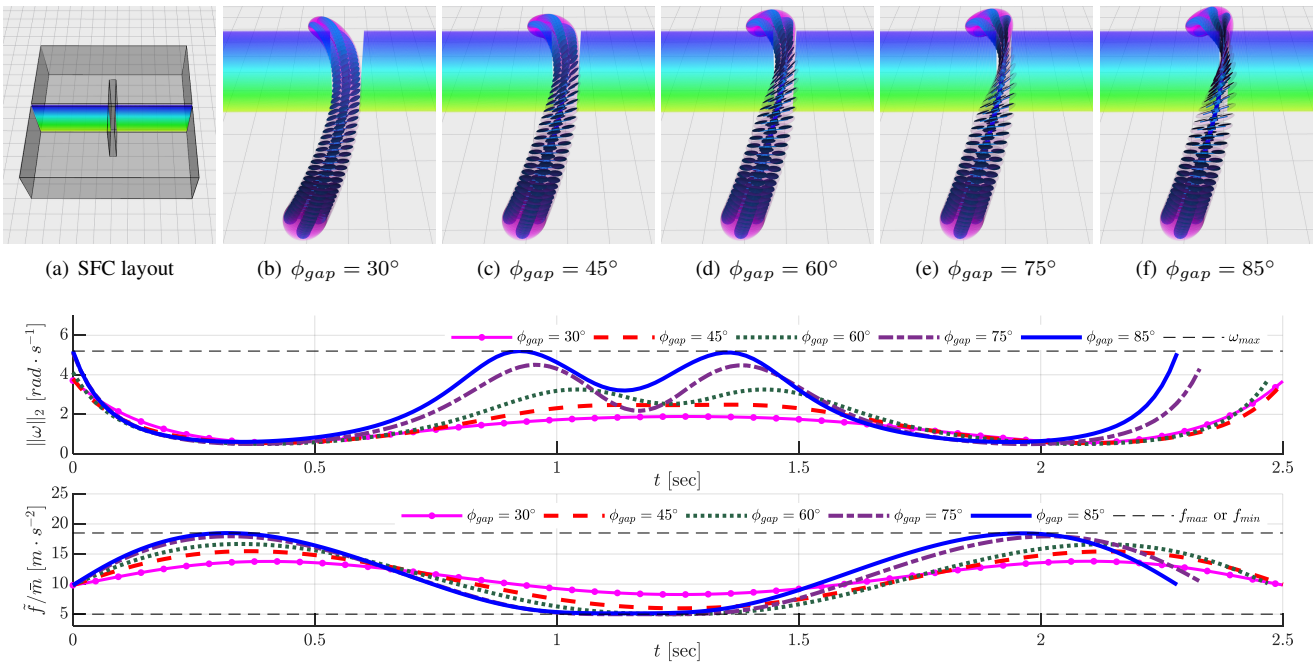
$$j = \lceil i/K \rceil, \forall t \in [0, T_i], \tag{107b}$$

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. As for dynamic limits, differentiating (100) gives the thrust

$$\tilde{f}_i(t) = \bar{m}\|p_i^{(2)}(t) + \bar{g}e_3\|, \tag{108}$$

and the body rate

$$\omega_i(t) = \left(\mathbf{R}_i^{\mathrm{T}}(t)\dot{\mathbf{R}}_i(t) \right)^{\vee}, \tag{109}$$



(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 $\mathrm{SE}(3)$ trajectories.

Figure 14. These figures show the SFC layout for a narrow gap, the $\mathrm{SE}(3)$ trajectories under different widths of gaps, and the 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.

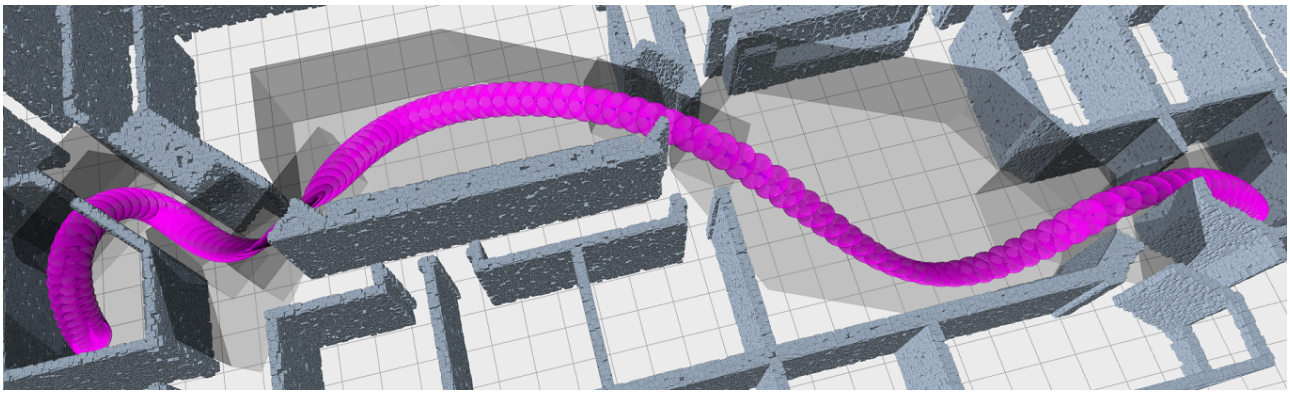


Figure 15. Long range $\mathrm{SE}(3)$ trajectory for a large-size quadcopter in an office environment. The vehicle has to pass through several narrow corridors and doors to reach the goal position. Gray polytopes represent the geometrical approximation of the free quotient space in $\mathbb{R}^3$.

where the map $\cdot^{\vee} : \mathbb{R}^{3\times3} \mapsto \mathbb{R}^3$ is defined as the inverse of the aforementioned hat map $\hat{\cdot} : \mathbb{R}^3 \mapsto \mathbb{R}^{3\times3}$.

Thus, we transform (105) into constraints on $p_i(t)$, $\mathbf{R}_i(t)$ and their derivatives:

$$\begin{cases} \|p_i^{(1)}(t)\|^2 - v_{max}^2 \leq 0, & \forall t \in [0, T_i], \\ \bar{m}^2\|p_i^{(2)}(t) + \bar{g}e_3\|_2^2 - f_{max}^2 \leq 0, & \forall t \in [0, T_i], \\ f_{min}^2 - \bar{m}^2\|p_i^{(2)}(t) + \bar{g}e_3\|_2^2 \leq 0, & \forall t \in [0, T_i], \\ \|(\mathbf{R}_i^{\mathrm{T}}(t)\dot{\mathbf{R}}_i(t))^{\vee}\|_2^2 - \omega_{max}^2 \leq 0, & \forall t \in [0, T_i]. \end{cases} \tag{110}$$

Finally, we obtain the continuous-time constraints $\mathcal{G}$ on the i-th piece along with its gradient by combining (107), (110) and the Appendix G. Besides, we still choose to minimize $s = 3$ because it is the highest derivative order for flatness of (100) and also helpful in smoothing the angular rate (109).

To demonstrate the efficiency of the proposed method, we consider the planning problem to enable a relatively large quadcopter to fly through a narrow gap whose width is smaller than that of the vehicle. Apparently, the yaw rotation ψ of $Z(\psi) - X(\phi) - Y(\theta)$ Euler angles does not affect the ellipsoid in the considered case. Moreover, as analyzed by Mueller et al. (2015), the angle ψ does not affect the translational motion either. Consequently, we assume $\psi = 0$ here and focus on the safety constraints and dynamic limits mainly caused by tilting rotation.

As is shown in Figure 14(a), only three polytopes are used to approximate the free quotient space, each of which is assigned with only a single trajectory piece. The ellipsoid that contains the quadcopter has the size $r_e = 0.5m$ and $h_e = 0.1m$. Due to the large vehicle size,

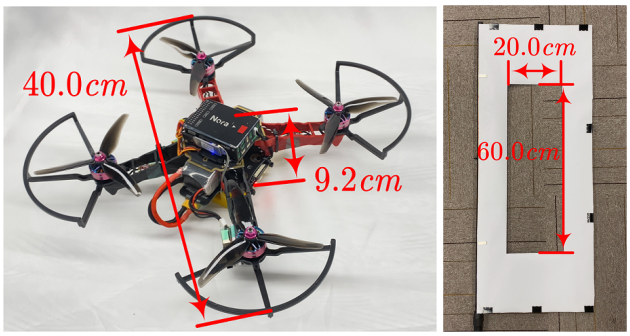
we set $f_{min}/\bar{m} = 5.0m/s^2$, $f_{max}/\bar{m} = 18.5m/s^2$, $v_{max} = 6.5m/s$ and $\omega_{max} = 5.2rad/s$. Intuitively, the quadcopter can only rotate no more than 1 revolution per second (rps), making it less agile than small quadcopters (Kushleyev et al. 2013) that can achieve $5rps$. We apply our method in several cases with different widths of gaps d_{gap}. The resultant computation time, maximum roll angle, and the trajectories on SE(3) are given in the following table and Figure 14(b)-14(f).

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 maximum roll angle during flight becomes larger such that no collision occurs. Intuitively, the solution space becomes smaller when dynamic limits exist. In all these cases, our method is still able to find safety solutions. The superior computation speed makes it possible to solving SE(3) planning at a considerably high frequency (at least $100Hz$). The curve for constraint functions are also provided in Figure 14(g) accordingly. The body rate and thrust satisfy the dynamic limits all the time. The continuous-time tightness of f_{min} for $\phi_{gap} \in \{60°, 75°, 85°\}$ shows the effectiveness of our time integral penalty functional.

We further solve a large-scale SE(3) planning problem to enable the same quadcopter to cover a long distance in an office environment. The resultant motion on SE(3) is shown in Figure 15, where the vehicle consecutively flies through several narrow gaps. The computation for this long trajectory only costs $62.88ms$ on the same aforementioned platform, making it possible to conduct large-scale full-body planning in real time. On SE(3) planning, our framework shows its great potential in constraint fidelity, motion quality and computation efficiency by enabling real-time agile flight planning for the considered less agile vehicle. It can also be applied to muticopters with various shapes by formulating collision constraints on vertices of their convex hull instead of its outter ellipsoid.

Agile Flight through Narrow Windows



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

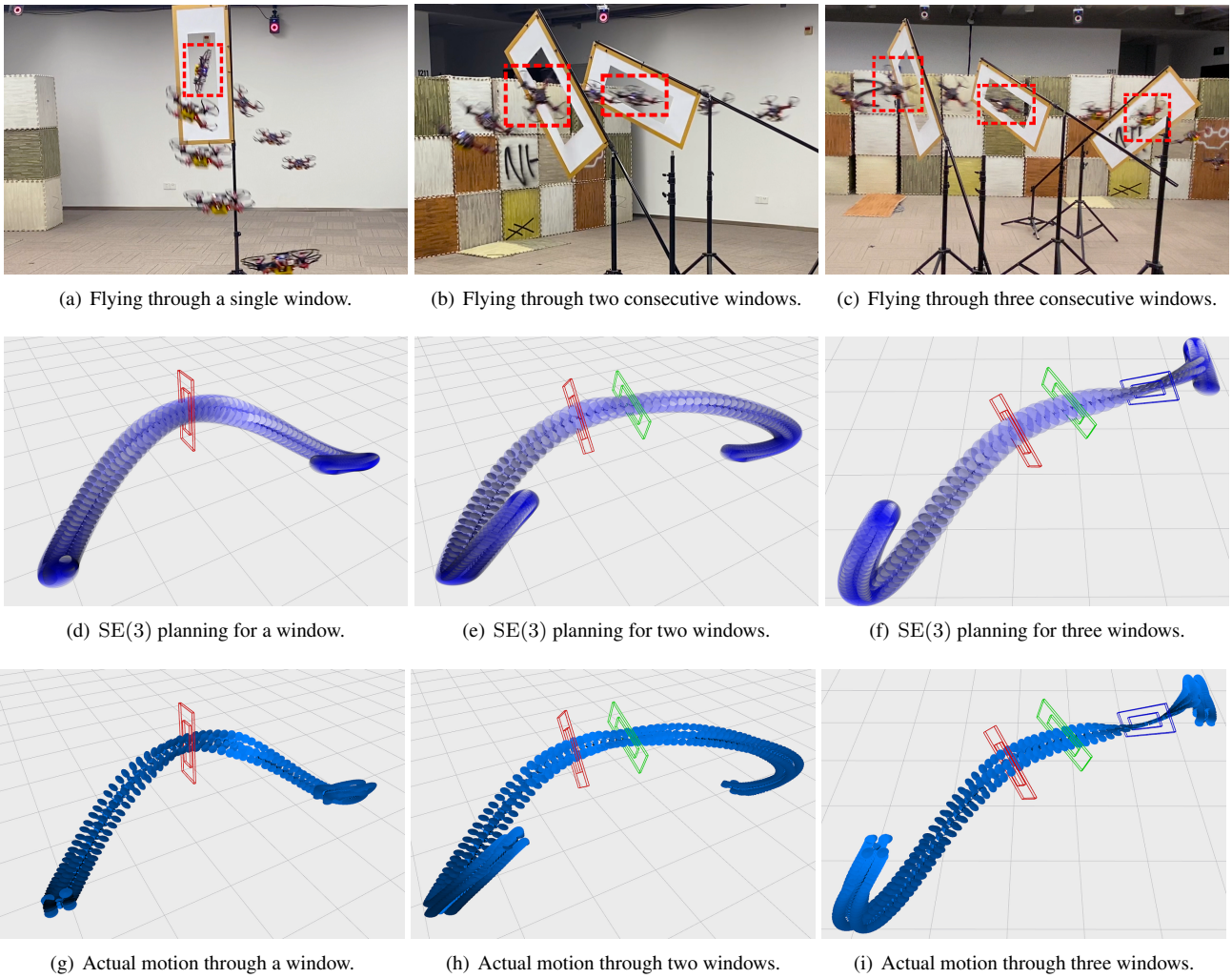
Figure 16. Sizes of the quadcopter and the narrow window.

We validate the practical performance of our framework on real-world experiments where a quadcopter flies through consecutive narrow windows. As is shown in Figure 16, the dimensions of the custom-made quadcopter are $40.0cm \times 9.2cm$, while its overall weight is $794.2g$. The width of the window is $20.0cm$. The quadcopter can never safely traverse even a single window if it does not fly at a large roll angle. When the vehicle is passing the window, the safety margin of the short side is only $5.4cm$. This implies the limited solution space of SE(3) motion when there are multiple windows to pass through.

The motion planner we adopt is the same as in the previous application except that we add the aforementioned margin into the safety constraints. The ellipsoid for the quadcopter takes $r_e = 20.0cm$ and $h_e = 4.6cm$ as its shape parameters. The feasibility constraints for velocity, thrust, and body rate are set as $v_{max} = 4.0m/s$, $f_{min}/\bar{m} = 3.0m/s^2$, $f_{max}/\bar{m} = 18.0m/s^2$, and $\omega_{max} = 6.0rad/s$, respectively. The flying area is a limited volume of $6.5 \times 6.0 \times 2.0m^3$. The poses of narrow windows and the quadcopter are all provided by external motion capture system running at $100Hz$. To ensure the vehicle safety, the obstacle-free region $\tilde{\mathcal{F}}$ is geometrically computed for multiple narrow windows in the free volume. Each local convex region is assigned with two trajectory pieces, i.e., $K = 2$. We run the planner on an offboard computer such that a human operator can arbitrarily choose the goal position for robustness testing. The control algorithm proposed by Faessler et al. (2018) is adopted for onboard SE(3) trajectory tracking. We disable its rotor drag effect then change its attitude computation following our Appendix G to eliminate the singularity at a right roll angle $\phi = 90°$.

The first scenario contains windows whose roll angle ranges from $30°$ to $90°$. The quadcopter has to pass through these windows to reach a randomly selected goal. The scenario is designed to test the validness of our planner. The experiment results are partially provided in Figure 17. There are several factors that bring challenges to our planner in this scenario, which are the large roll angles of windows, the short distance between consecutive windows, the small acceleration/deceleration space and the limited maneuvering capability of the quadcopter. Our planner still generates safe and smooth motion in real time for multiple window passing without unrealistic requirements on body rate and thrust. The resultant SE(3) trajectories are shown in Figure 17(d)-17(f). The actual flight trajectories of our quadcopter are provided in Figure 17(g)-17(i). The estimated actual maximum speed and body rate reach $4.1m/s$ and $8.4rad/s$, respectively. As can be seen from the results, all these motions from our planner are practically feasible even considering the high-fidelity quadcopter dynamics.

The second scenario requires the quadcopter persistently fly through multiple windows. This scenario is designed to test the robustness of our planner. The experiment results are provided in Figure 18. Our planner guides the quadcopter to fly back and forth through narrow windows for $20.0s$ while ensuring the safety and dynamic feasibility all the time. We believe the results constitute a strong evidence for the robustness. In addition, we also conduct an interactive scenario where a human operator randomly holds the narrow window for real-time planning. A snapshot is given in Figure 18(a). More details about the experiment are given in the attached multimedia.



(a) Flying through a single window. (b) Flying through two consecutive windows. (c) Flying through three consecutive windows.

(d) $\mathrm{SE}(3)$ planning for a window. (e) $\mathrm{SE}(3)$ planning for two windows. (f) $\mathrm{SE}(3)$ planning for three windows.

(g) Actual motion through a window. (h) Actual motion through two windows. (i) Actual motion through three windows.

Figure 17. The experiment results for flying through consecutive narrow windows. The first row shows some snapshots for real flights. The second row and the third row give some $\mathrm{SE}(3)$ motions of the proposed planner and the actual poses during flight. The position of narrow windows are visualized by red, green, and blue markers. The proposed planner automatically handles the acceleration/deceleration in a small flying space and the maneuvers under limited actuator constraints while ensuring the motion safety and smoothness.

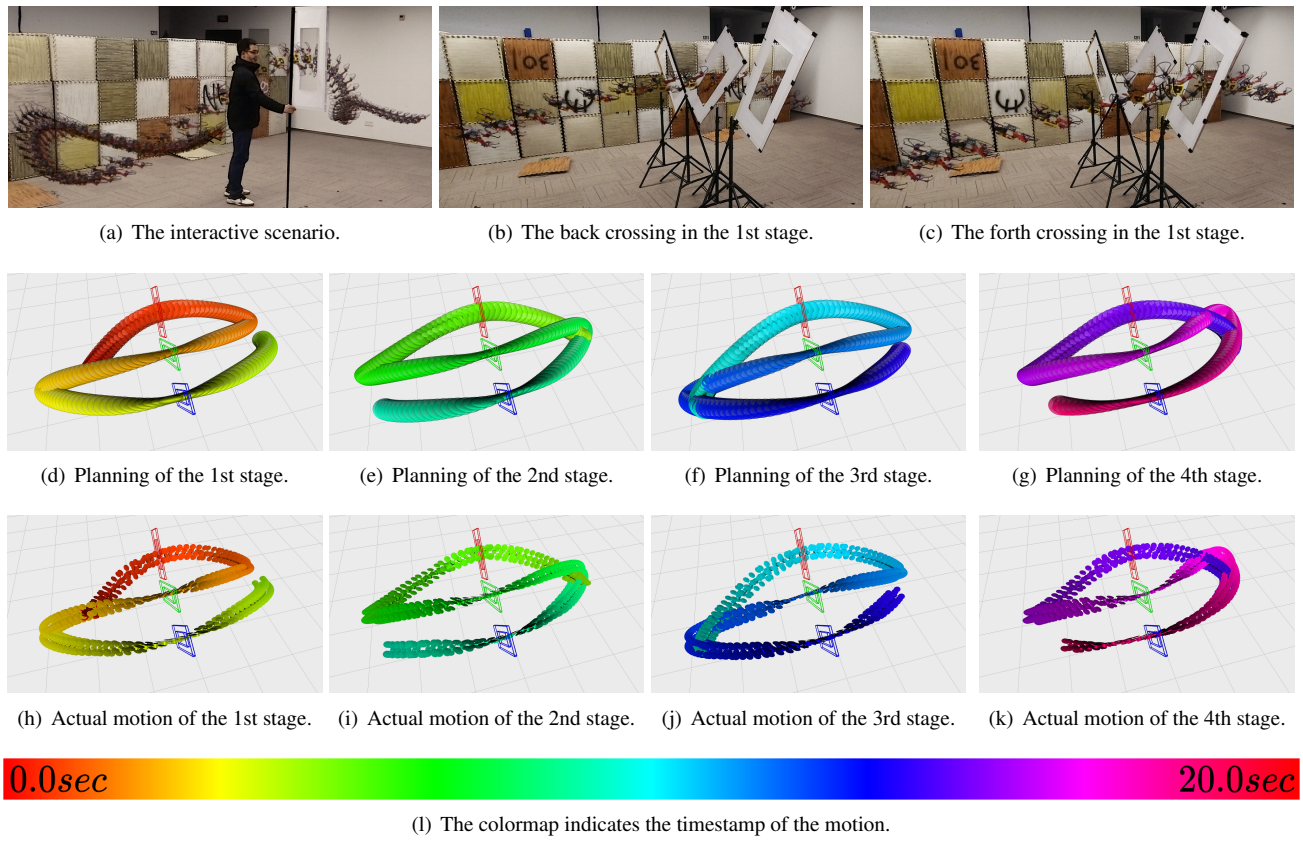
In this experiments, we only provides the planner with the geometrical approximation $\tilde{\mathcal{F}}$, the vehicle ellipsoid $\mathcal{E}$, and the information on vehicle dynamics $\mathcal{G}$. All the $\mathrm{SE}(3)$ motions are direct solutions of the optimization. It should be highlighted that our method differs from traditional planners for narrow gaps (Falanga et al. 2017; Loianno et al. 2016) in its flexibility, efficiency and high fidelity. The flexibility lies in that $\tilde{\mathcal{F}}$ can be directly generated from raw sensor data without prior information on the narrow passage. The efficiency comes from the fact that it only costs several milliseconds for a complete motion through a narrow window. The high fidelity comes from the capability to directly enforce $\mathrm{SE}(3)$ safety and state-input feasibility constraints.

Discussion and Conclusion

Extensions

Profiting from the flexibility, efficiency, and quality, our planner have many applicative and algorithmic extensions in autonomous vehicle motion planning including but not limited to multicopters.

The MINCO trajectory class is designed for general-purpose applications. In other words, many methods that utilize spline trajectories can take MINCO as a candidate for sparse spatial-temporal optimization. For example, it is often required to optimize a trajectory based on a discrete field, such as the Fisher Information Field (FIF) for perception-aware planning (Zhang and Scaramuzza 2020) or the Euclidean Signed Distance Field (ESDF) for clearance-aware planning (Zhou et al. 2020). MINCO brings convenience to these applications since its spatial-temporal deformation suits almost any user-defined objective along with gradient. Besides, the induced objective is often required to be evaluated uniformly on either the time or curve length. In such a case, a time-uniform variant of MINCO can be obtained by setting the same duration for each piece of the trajectory. An approximately space-uniform variant of MINCO can also be obtained by restricting the variance of piece lengths obtained from quadrature.



(a) The interactive scenario. (b) The back crossing in the 1st stage. (c) The forth crossing in the 1st stage.

(d) Planning of the 1st stage. (e) Planning of the 2nd stage. (f) Planning of the 3rd stage. (g) Planning of the 4th stage.

(h) Actual motion of the 1st stage. (i) Actual motion of the 2nd stage. (j) Actual motion of the 3rd stage. (k) Actual motion of the 4th stage.

(l) The colormap indicates the timestamp of the motion.

Figure 18. These figures show the experiment results where a quadcopter is required to consecutively cross the narrow gaps. The first row shows some snapshots of the flights. The second row and the third row give the planned $\mathrm{SE}(3)$ trajectories and the actual motions in different experiment stages. The timestamp is indicated by the colormap in the last figure. The robustness of our planner is demonstrated in this experiment.

Our framework makes no assumption on a concrete form of continuous-time constraints $\mathcal{G}$ and vehicle dynamics. Consequently, it is applicable to incorporate more accurate dynamics and actuator limits for any differentially flat multicopters by considering more model details such as the rotor drag (Faessler et al. 2018). In this way, it is possible to fully exploit the vehicle limit via real-time high-fidelity planning and control. Moreover, the proposed framework is also extendable to a swarm of multicopters or moving obstacles. For example, a shared time vector can be used for a swarm of multicopters in centralized motion planning such that the collision cost can be expressed at any timestamp. When moving obstacles exist, the time-dependent collision cost is also supported in $\mathcal{G}$ for collision avoidance in both time and space.

Our method is also inherently parallelizable to further squeeze its performance. Specifically, the most computation-demanding part lies in the calculation and differentiation of the time integral penalty functional $I_{\mathcal{G}}[p]$. Fortunately, these operations are independent at each sampled timestamps, thus a more efficient way is to conduct the operations concurrently. In this way, parallelization can effectively speedup the optimization especially when dense quadratures are required for complicated constraints $\mathcal{G}$.

Currently, we only study the trajectory planning for differentially flat multicopters. However, as described in our paper, the framework can be at least partially extended for any vehicle whose flat output space overlaps the configuration space. An example is the fixed-wing aircraft as is studied by Bry et al. (2015). Different from multicopter, the motion of fixed-wing aircraft is mainly limited by the trajectory curvature. To handle this, they propose the Dubins–Polynomial trajectory (Bry et al. 2015) for bounded curvature planning. In our framework, the curvature constraint is indeed a special case of $\mathcal{G}$. Consequently, provided with a sufficient number of pieces, MINCO can also be deformed to meet the restriction on curvature.

Limitations

The proposed framework, like most optimization-based ones, focuses on a relatively local trajectory planning problem for multicopters. Although a sampling-based methods can be utilized to find a roughly proper local solution space, there are still nonconvexity coming from the system dynamics and objective functional. Our framework suffers from shallow local minima while it is intractable to find the global minimum by using local optimization. This issue can be alleviated by providing a relative rational initial guess or adding exploration strategy to the framework as is done in CHOMP (Zucker et al. 2013).

A major limitation of the framework originates from the MINCO trajectory class itself. When constraints $\mathcal{G}$ exist, the optimal solution cannot in general be represented by polynomial splines, let alone MINCO. Thus, optimizing

within $\mathfrak{T}_{\mathrm{MINCO}}$ is just a relaxation to the original problem. However, our benchmark shows that the spatial-temporal deformation of MINCO is able to generate high-quality solutions comparable to the ground truth from DC, but with several orders of magnitudes faster computing.

There are also limitations caused by the penalty method in this framework. For the sake of differentiability, we adopt an inexact penalty such that unbounded penalty weights are required for zero constraint violations. Moreover, the framework is unable to determine whether a feasible solution exists for a specific problem. However, small constraint violations in the penalty are empirically acceptable for multicopter dynamics. A feasible margin also helps to eliminate the constraint violation. As a reward, the penalty does not need an initial feasible guess, making it suitable for online applications.

Conclusion

In this paper, we propose a flexible trajectory optimization framework for multicopter motion planning. Our framework is powered by several core features, such as the MINCO trajectory class derived from optimality conditions, the constraint elimination based on smooth maps, the time-integral penalty functional based on constraint transcription, and the backward differentiation of the transformation for flat outputs. All these components enjoy the efficiency and generality originating from low-complexity and less preliminary assumptions.

We perform extensive benchmarks against many kinds of multicopter trajectory planning methods to show the speedup over orders of magnitude and the top-level solution quality. Various kinds of applications demonstrate the versatility of our framework. We also present further discussions about several unlisted applications or extensions as future work.

References

Barber CB, Dobkin DP and Huhdanpaa H (1996) The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software* 22(4): 469–483.

Barry AJ, Jenks T, Majumdar A, Lin HT, Ros IG, Biewener AA and Tedrake R (2014) Flying between obstacles with an autonomous knife-edge maneuver. In: *IEEE International Conference on Robotics and Automation*. Hong Kong, China, pp. 2559–2559.

Baur W and Strassen V (1983) The complexity of partial derivatives. *Theoretical Computer Science* 22(3): 317–330.

Bertsekas DP (1995) *Dynamic Programming and Optimal Control*. Athena-Scientific.

Betts JT (1998) Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics* 21(2): 193–207.

Betts JT (2010) *Practical methods for optimal control and estimation using nonlinear programming*. SIAM.

Bry A, Richter C, Bachrach A and Roy N (2015) Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments. *The International Journal of Robotics Research* 34(7): 969–1002.

Burke D, Chapman A and Shames I (2020) Generating minimum-snap quadrotor trajectories really fast. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Las Vegas, USA, pp. 1487–1492.

Clarke F (2013) *Functional Analysis, Calculus of Variations and Optimal Control*. Springer.

Deits R and Tedrake R (2015a) Computing large convex regions of obstacle-free space through semidefinite programming. In: *Algorithmic foundations of robotics XI*. Springer, pp. 109–124.

Deits R and Tedrake R (2015b) Efficient mixed-integer planning for UAVs in cluttered environments. In: *IEEE International Conference on Robotics and Automation*. Seattle, USA, pp. 42–49.

Demmel JW, Eisenstat SC, Gilbert JR, Li XS and Liu JW (1999) A supernodal approach to sparse partial pivoting. *SIAM Journal on Matrix Analysis and Applications* 20(3): 720–755.

Dmitruk AV and Kaganovich AM (2008) The hybrid maximum principle is a consequence of pontryagin maximum principle. *Systems & Control Letters* 57(11): 964–970.

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

Faessler M, Franchi A and Scaramuzza D (2018) Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robotics and Automation Letters* 3(2): 620–626.

Falanga D, Mueggler E, Faessler M and Scaramuzza D (2017) Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision. In: *IEEE International Conference on Robotics and Automation*. Singapore, Singapore, pp. 5774–5781.

Felzenszwalb PF and Huttenlocher DP (2012) Distance transforms of sampled functions. *Theory of Computing* 8(1): 415–428.

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

Fliess M, Lévine J, Martin P and Rouchon P (1995) Flatness and defect of non-linear systems: Introductory theory and examples. *International Journal of Control* 61(6): 1327–1361.

Foehn P and Scaramuzza D (2020) CPC: Complementary progress constraints for time-optimal quadrotor trajectories. *arXiv preprint arXiv:2007.06255* .

Frazzoli E, Dahleh MA and Feron E (2002) Real-time motion planning for agile autonomous vehicles. *Journal of Guidance, Control, and Dynamics* 25(1): 116–129.

Gammell JD, Barfoot TD and Srinivasa SS (2018) Informed sampling for asymptotically optimal path planning. *IEEE Transactions on Robotics* 34(4): 966–984.

Gao F, Wang L, Zhou B, Zhou X, Pan J and Shen S (2020) Teach-Repeat-Replan: A complete and robust system for aggressive flight in complex environments. *IEEE Transactions on Robotics* 36(5): 1526–1545.

Gao F, Wu W, Gao W and Shen S (2019) Flying on Point Clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments. *Journal of Field Robotics* 36(4): 710–733.

Gilbert EG, Johnson DW and Keerthi SS (1988) A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal on Robotics and Automation* 4(2): 193–203.

Gill PE, Murray W and Saunders MA (2005) SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review* 47(1): 99–131.

Gill PE, Murray W and Wright MH (1981) *Practical Optimization*. SIAM.

Golub GH and Loan FV (2013) *Matrix Computations*. The Johns Hopkins University Press.

Griewank A and Walther A (2008) *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM.

Gurobi Optimization, LLC (2020) Gurobi Optimizer Reference Manual. URL https://www.gurobi.com.

Hönig W, Preiss JA, Kumar TS, Sukhatme GS and Ayanian N (2018) Trajectory planning for quadrotor swarms. *IEEE Transactions on Robotics* 34(4): 856–869.

Horn RA and Johnson CR (2012) *Matrix Analysis*. Cambridge University Press.

Houska B, Ferreau HJ and Diehl M (2011) ACADO toolkit—an open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods* 32(3): 298–312.

Isidori A (2013) *Nonlinear Control Systems*. Springer.

Janson L, Schmerling E, Clark A and Pavone M (2015) Fast Marching Tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *The International Journal of Robotics Research* 34(7): 883–921.

Jennings LS and Teo KL (1990) A computational algorithm for functional inequality constrained optimization problems. *Automatica* 26(2): 371–375.

Kalakrishnan M, Chitta S, Theodorou E, Pastor P and Schaal S (2011) STOMP: Stochastic trajectory optimization for motion planning. In: *IEEE International Conference on Robotics and Automation*. Shanghai, China, pp. 4569–4574.

Karaman S and Frazzoli E (2010) Optimal kinodynamic motion planning using incremental sampling-based methods. In: *IEEE Conference on Decision and Control*. Atlanta, Georgia, USA, pp. 7681–7687.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research* 30: 846 – 894.

Kavraki LE, Kolountzakis MN and Latombe JC (1998) Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation* 14(1): 166–171.

Kavraki LE, Svestka P, Latombe JC and Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation* 12(4): 566–580.

Kushleyev A, Mellinger D, Powers C and Kumar V (2013) Towards a swarm of agile micro quadrotors. *Autonomous Robots* 35(4): 287–300.

LaValle SM (1998) Rapidly-exploring random trees: A new tool for path planning. Technical Report TR 98-11, Iowa State University.

LaValle SM and Kuffner Jr JJ (2001) Randomized kinodynamic planning. *The International Journal of Robotics Research* 20(5): 378–400.

Lee J (2012) *Introduction to Smooth Manifolds*. Springer.

Li DH and Fukushima M (2001) A modified BFGS method and its global convergence in nonconvex minimization. *Journal of Computational and Applied Mathematics* 129(1-2): 15–35.

Li Y, Littlefield Z and Bekris KE (2016) Asymptotically optimal sampling-based kinodynamic planning. *The International Journal of Robotics Research* 35(5): 528–564.

Liu DC and Nocedal J (1989) On the limited memory BFGS method for large scale optimization. *Mathematical Programming* 45(1-3): 503–528.

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

Liu S, Watterson M, Mohta K, Sun K, Bhattacharya S, Taylor CJ and Kumar V (2017b) Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments. *IEEE Robotics and Automation Letters* : 1688–1695.

Loianno G, Brunner C, McGrath G and Kumar V (2016) Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and imu. *IEEE Robotics and Automation Letters* 2(2): 404–411.

Lynch KM and Park FC (2017) *Modern Robotics*. Cambridge University Press.

Martin P, Murray RM and Rouchon P (2003) Flat systems, equivalence and trajectory generation. Technical Report CDS 2003-008, California Institute of Technology, Pasadena, Calif, USA.

Mellinger D and Kumar V (2011) Minimum snap trajectory generation and control for quadrotors. In: *IEEE International Conference on Robotics and Automation*. Shanghai, China, pp. 2520–2525.

Mohta K, Watterson M, Mulgaonkar Y, Liu S, Qu C, Makineni A, Saulnier K, Sun K, Zhu A, Delmerico J, Karydis K, Atanasov N, Loianno G, Scaramuzza D, Daniilidis K, Taylor CJ and Kumar V (2018) Fast, autonomous flight in gps-denied and cluttered environments. *Journal of Field Robotics* 35(1): 101–120.

MOSEK Aps (2020) MOSEK Optimizer API for C. URL https://www.mosek.com.

Mu B and Chirarattananon P (2019) Trajectory generation for underactuated multirotor vehicles with tilted propellers via a flatness-based method. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. pp. 1365–1370.

Mueller MW, Hehn M and D'Andrea R (2015) A computationally efficient motion primitive for quadrocopter trajectory generation. *IEEE Transactions on Robotics* 31(6): 1294–1310.

Nägeli T, Alonso-Mora J, Domahidi A, Rus D and Hilliges O (2017) Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. *IEEE Robotics and Automation Letters* 2(3): 1696–1703.

Nesterov Y (2018) *Lectures on Convex Optimization*. Springer.

Nocedal J and Wright S (2006) *Numerical Optimization*. Springer.

Oleynikova H, Lanegger C, Taylor Z, Pantic M, Millane A, Siegwart R and Nieto J (2020) An open-source system for vision-based micro-aerial vehicle mapping, planning, and flight in cluttered environments. *Journal of Field Robotics* 37(4): 642–666.

Orthey A, Escande A and Yoshida E (2018) Quotient-space motion planning. In: *IEEE/RSJ International Conference on Intelligent*

Robots and Systems. Madrid, Spain, pp. 8089–8096.

Orthey A and Toussaint M (2019) Rapidly-exploring quotient-space trees: Motion planning using sequential simplifications. In: *International Symposium on Robotics Research*. Hanoi, Vietnam.

Park C, Pan J and Manocha D (2012) ITOMP: Incremental trajectory optimization for real-time replanning in dynamic environments. In: *International Conference on Automated Planning and Scheduling*. pp. 207–215.

Patterson MA and Rao AV (2014) 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* 41(1): 1–37.

Press WH, Teukolsky SA, Vetterling WT and Flannery BP (2007) *Numerical Recipes*. Cambridge University Press.

Quinlan S and Khatib O (1993) Elastic bands: Connecting path planning and control. In: *IEEE International Conference on Robotics and Automation*. Atlanta, USA, pp. 802–807.

Ryou G, Tal E and Karaman S (2020) Multi-fidelity black-box optimization for time-optimal quadrotor maneuvers. In: *Robotics: Science and Systems*. Oregon, USA, pp. 5774–5781.

Ryu JC and Agrawal SK (2011) Differential flatness-based robust control of mobile robots in the presence of slip. *The International Journal of Robotics Research* 30(4): 463–475.

Schulman J, Duan Y, Ho J, Lee A, Awwal I, Bradlow H, Pan J, Patil S, Goldberg K and Abbeel P (2014) Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research* 33(9): 1251–1270.

Schumaker L (2007) *Spline Functions: Basic Theory*. Cambridge University Press.

Seidel R (1991) Small-dimensional linear programming and convex hulls made easy. *Discrete & Computational Geometry* 6(3): 423–434.

Sisser FS (1981) Elimination of bounds in optimization problems by transforming variables. *Mathematical Programming* 20(1): 110–121.

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

Sun W, Tang G and Hauser K (2020) Fast UAV trajectory optimization using bilevel optimization with analytical gradients. In: *American Control Conference*. pp. 82–87.

Tordesillas J and How JP (2020) MINVO basis: Finding simplexes with minimum volume enclosing polynomial curves. *arXiv preprint arXiv:2010.10726* .

Tordesillas J, Lopez BT and How JP (2019) FASTER: Fast and safe trajectory planner for flights in unknown environments. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Macau, China, pp. 1934–1940.

Toth CD, O'Rourke J and Goodman JE (2017) *Handbook of Discrete and Computational Geometry*. CRC Press.

Van Den Bergen G (2001) Proximity queries and penetration depth computation on 3d game objects. In: *Game Developers Conference*, volume 170.

Van Nieuwstadt MJ and Murray RM (1998) Real-time trajectory generation for differentially flat systems. *International Journal of Robust and Nonlinear Control* 8(11): 995–1020.

Verriest E and Lewis F (1991) On the linear quadratic minimum-time problem. *IEEE Transactions on Automatic Control* 36(7): 859–863.

Verscheure D, Demeulenaere B, Swevers J, De Schutter J and Diehl M (2009) Time-optimal path tracking for robots: A convex optimization approach. *IEEE Transactions on Automatic Control* 54(10): 2318–2327.

Wächter A and Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106(1): 25–57.

Warren J, Schaefer S, Hirani AN and Desbrun M (2007) Barycentric coordinates for convex sets. *Advances in Computational Mathematics* 27(3): 319–338.

Wu Y, Ding Z, Xu C and Gao F (2021) External forces resilient safe motion planning for quadrotor. *arXiv preprint arXiv:2103.11178* .

Zhang J, Hu C, Chadha RG and Singh S (2020) Falco: Fast likelihood-based collision avoidance with extension to human-guided navigation. *Journal of Field Robotics* 37(8): 1300–1313.

Zhang Z and Scaramuzza D (2018) Perception-aware receding horizon navigation for MAVs. In: *IEEE International Conference on Robotics and Automation*. Brisbane, Australia, pp. 2534–2541.

Zhang Z and Scaramuzza D (2020) Fisher Information Field: An efficient and differentiable map for perception-aware planning. *arXiv preprint arXiv:2008.03324* .

Zhang Z, Tomlinson J and Martin C (1997) Splines and linear control theory. *Acta Applicandae Mathematica* 49(1): 1–34.

Zhong X, Wu Y, Wang D, Wang Q, Xu C and Gao F (2020) Generating large convex polytopes directly on point clouds. *arXiv preprint arXiv:2010.08744* .

Zhou B, Pan J, Gao F and Shen S (2020) Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight. *arXiv preprint arXiv:2007.03465* .

Zhou X, Wang Z, Ye H, Xu C and Gao F (2021) EGO-Planner: An ESDF-free gradient-based local planner for quadrotors. *IEEE Robotics and Automation Letters* 6(2): 478–485.

Ziegler GM (2012) *Lectures on Polytopes*. Springer.

Zucker M, Ratliff N, Dragan AD, Pivtoraiko M, Klingensmith M, Dellin CM, Bagnell JA and Srinivasa SS (2013) CHOMP: Covariant hamiltonian optimization for motion planning. *The International Journal of Robotics Research* 32(9): 1164–1193.

Appendix A: Proof of Theorem 2

Proof. The necessity of optimality conditions are already given by the analysis in the aforementioned content. As for the sufficiency, we need to prove the existence and the uniqueness of solution for these conditions.

The first part of these conditions indicates that the optimal trajectory is a polynomial spline. The rest three parts are all linear constraints on the spline, which can be decoupled in m dimensions. In this proof, we only consider a single dimension hereafter. boundary and intermediate conditions provide $2s$ and $\sum_{i=1}^{M-1} d_i$ constraints, respectively. The continuous differentiability to a certain orders gives $\sum_{i=1}^{M-1} \bar{d}_i$ constraints. Consequently, there are $2Ms$ constraints in total, which matches the number

of parameters exactly. The uniqueness is then equivalent to the nonsingularity of this linear equation system. However, its coefficient matrix is indeed a function of the time vector of the spline. Showing the nonsingularity of this matrix-valued function over its entire domain is generally nontrivial. Thus, we prove the uniqueness of the solution based on spline theory instead.

Consider the space of M-piece polynomial $2s$-order splines defined on $[t_0, t_M]$ where two consecutive pieces of any spline $x : [t_0, t_M] \mapsto \mathbb{R}$ satisfy

$$x_{i-1}^{(j)}(t_i) = x_i^{(j)}(t_i),\ 0 \le j < \bar{d}_i, \tag{111}$$

for any $1 \le i < M$. In the original problem, $d_i \le s$ holds for each i. Note that the order of spline is its polynomial degree plus one. For brevity, we define $D_{i,j}$ as

$$D_{i,j} = i \cdot s + \sum_{k=1}^{j} d_k. \tag{112}$$

According to Theorem 4.4 by Schumaker (2007), this space is a linear space of dimension $\bar{D} := D_{2,M-1}$. Moreover, explicit bases for this linear space can be constructed. 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

$$\bar{t}_i = \begin{cases} t_0 & \text{if } 1 \le i \le D_{2,0}, \\ t_j & \text{if } D_{2,j-1} < i \le D_{2,j}, \\ t_M & \text{if } D_{2,M-1} < i \le \bar{M}. \end{cases} \tag{113}$$

Using *divided differences* of *Green's functions* over this extended partition, Theorem 4.9 by Schumaker (2007) 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 the boundary conditions (20c) and intermediate conditions (20d) 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

$$\tau_i = \begin{cases} t_0 & \text{if } 1 \le i \le D_{1,0}, \\ t_j & \text{if } D_{1,j-1} < i \le D_{1,j}, \\ t_M & \text{if } D_{1,M-1} < i \le \bar{D}. \end{cases} \tag{114}$$

Denote by ν_i the specified order at τ_i, written as

$$\nu_i = \begin{cases} i - 1 & \text{if } 1 \le i \le D_{1,0}, \\ i - 1 - D_{1,j-1} & \text{if } D_{1,j-1} < i \le D_{1,j}, \\ i - 1 - D_{1,M-1} & \text{if } D_{1,M-1} < i \le \bar{D}. \end{cases} \tag{115}$$

Then, the conditions (20c) and (20d) generate a linear equation system on the basis, whose coefficient matrix is

$$\mathbf{B} = \begin{pmatrix} B_1^{(\nu_1)}(\tau_1) & B_2^{(\nu_1)}(\tau_1) & \cdots & B_{\bar{D}}^{(\nu_1)}(\tau_1) \\ B_1^{(\nu_2)}(\tau_2) & B_2^{(\nu_2)}(\tau_2) & \cdots & B_{\bar{D}}^{(\nu_2)}(\tau_2) \\ \vdots & \vdots & \ddots & \vdots \\ B_1^{(\nu_{\bar{D}})}(\tau_{\bar{D}}) & B_2^{(\nu_{\bar{D}})}(\tau_{\bar{D}}) & \cdots & B_{\bar{D}}^{(\nu_{\bar{D}})}(\tau_{\bar{D}}) \end{pmatrix}. \tag{116}$$

According to Theorem 4.67 by Schumaker (2007), $\mathbf{B}$ is nonsingular if and only if

$$\tau_i \in \delta_i := \begin{cases} [\bar{t}_i, \bar{t}_{i+2s}) & \text{if } \nu_i + \alpha_i - 2s \ge 0, \\ (\bar{t}_i, \bar{t}_{i+2s}) & \text{if } \nu_i + \alpha_i - 2s < 0, \end{cases} \tag{117}$$

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

$$\alpha_i = \{max\ j\ :\ \bar{t}_i = \cdots = \bar{t}_{i+j-1}\}. \tag{118}$$

We show that (117) is always true in our case. It is obvious that α_i can be computed as

$$\alpha_i = \begin{cases} D_{2,0} - i + 1 & \text{if } 1 \le i \le D_{2,0}, \\ D_{2,j} - i + 1 & \text{if } D_{2,j-1} < i \le D_{2,j}. \end{cases} \tag{119}$$

Combining (115) and (119), we know that $\nu_i < s$ and $\alpha_i \le s$ always hold for $i > s$, which means

$$\begin{cases} \nu_i + \alpha_i - 2s = 0 & \text{if } 1 \le i \le s, \\ \nu_i + \alpha_i - 2s < 0 & \text{if } s < i \le \bar{D}. \end{cases} \tag{120}$$

Thus, the interval δ_i is computed as

$$\delta_i = \begin{cases} [\bar{t}_i, \bar{t}_{i+2s}) & \text{if } 1 \le i \le s, \\ (\bar{t}_i, \bar{t}_{i+2s}) & \text{if } s < i \le \bar{D}. \end{cases} \tag{121}$$

Consequently, we have

$$\tau_i = t_0 \in [t_0, t_1) \subseteq [\bar{t}_i, \bar{t}_{i+2s}) = \delta_i,\ 1 \le i \le s. \tag{122}$$

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 (113) and (114), we have

$$D_{2,k-1} < i,\ (i + 2s) \le D_{2,l}, \tag{123}$$

$$D_{1,j-1} < i \le D_{1,j}. \tag{124}$$

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

$$D_{2,k-1} < i \le D_{1,j} = (D_{2,j} - s) \le D_{2,j-1}, \tag{125}$$

$$D_{2,j} = (D_{1,j} + s) \le (D_{1,j-1} + 2s) < (i + 2s) \le D_{2,l}. \tag{126}$$

The inequality (125) and (126) implies $k < j$ and $j < l$, respectively. That is to say,

$$\tau_i = t_j \in (t_k, t_l) = (\bar{t}_i, \bar{t}_{i+2s}) = \delta_i,\ s < i \le \bar{D}, \tag{127}$$

always holds. Combining (122) and (127) gives (117). 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. □

Appendix B: Analytic Gradient of $J_c(\mathbf{c}, \mathbf{T})$

Without loss of generality, consider a single dimension of the i-th piece $p_i(t) : [0, T_i] \mapsto \mathbb{R}$ of a polynomial spline. We denote by $J_c(c_i, T_i)$ the control effort on $p_i(t)$ where

$c_i \in \mathbb{R}^{2s}$ is and $T_i \in \mathbb{R}_{>0}$. Substituting $p_i(t) = c_i^{\mathrm{T}}\beta(t)$ gives

$$J_c(c_i, T_i) = \int_0^{T_i} p_i^{(s)}(t)^2 \mathrm{d}t \tag{128}$$

$$= \int_0^{T_i} c_i^{\mathrm{T}} \beta^{(s)}(t)\beta^{(s)}(t)^{\mathrm{T}} c_i \mathrm{d}t \tag{129}$$

$$= c_i^{\mathrm{T}} \left(\int_0^{T_i} \beta^{(s)}(t)\beta^{(s)}(t)^{\mathrm{T}} \mathrm{d}t \right) c_i. \tag{130}$$

where $\beta(t) = (1, t, \ldots, t^{2s-1})^{\mathrm{T}}$. Defining

$$\mathbf{R}(\tau) = \beta^{(s)}(\tau)\beta^{(s)}(\tau)^{\mathrm{T}}, \tag{131}$$

$$\mathbf{Q}(\tau) = \int_0^{\tau} \mathbf{R}(t)\mathrm{d}t, \tag{132}$$

we have

$$J_c = c_i^{\mathrm{T}} \mathbf{Q}(T_i) c_i, \tag{133}$$

and the gradient

$$\frac{\partial J_c}{\partial c_i} = 2\mathbf{Q}(T_i)c_i, \ \frac{\partial J_c}{\partial T_i} = c_i^{\mathrm{T}}\mathbf{R}(T_i)c_i. \tag{134}$$

Specifically, computing $\mathbf{R}(\tau)$ and $\mathbf{Q}(\tau)$ from $\beta(\tau)$ gives their entries at the i-th row and the j-th column,

$$\mathbf{R}_{ij}(\tau) = \begin{cases} \frac{(i-1)!(j-1)!}{(i-s-1)!(j-s-1)!}\tau^{i+j-2s-2} & \text{if } i,j > s, \\ 0 & \text{otherwise}, \end{cases} \tag{135}$$

$$\mathbf{Q}_{ij}(\tau) = \begin{cases} \frac{(i-1)!(j-1)!}{(i-s-1)!(j-s-1)!(i+j-2s-1)}\tau^{i+j-2s-1} & \text{if } i,j > s, \\ 0 & \text{otherwise}. \end{cases} \tag{136}$$

Appendix C: Proof of Proposition 1

Proof. An open unit ball with L_p-norm is defined as

$$\mathcal{B}_p = \left\{ x \in \mathbb{R}^{M-1} \;\middle|\; \|x\|_p < 1 \right\}. \tag{137}$$

It is widely known that $\mathcal{B}_p$ and $\mathbb{R}^{M-1}$ are diffeomorphic (Lee 2012), where the smooth bijection is given by

$$f_p(x) = \frac{x}{\left(\|x\|_p^p + 1\right)^{\frac{1}{p}}} \in \mathcal{B}_p, \ \forall x \in \mathbb{R}^{M-1}. \tag{138}$$

All the normalized time vectors $(T_1/T_\Sigma, \ldots, T_{M-1}/T_\Sigma)^{\mathrm{T}}$ constitute the following open set

$$\mathcal{B}_1^{>0} := \left\{ x \in \mathbb{R}_{>0}^{M-1} \;\middle|\; \|x\|_1 < 1 \right\} \tag{139}$$

which is diffeomorphic to $\mathcal{T}_f$. Here $\mathcal{B}_1^{>0}$ is the intersection of $\mathcal{B}_1$ and $\mathbb{R}_{>0}^{M-1}$. Notice that $\mathcal{B}_1^{>0}$ and $\mathbb{R}_{>0}^{M-1}$ are also diffeomorphic by using the map f_p with L_1-norm, i.e., $\tilde{T} = f_1(x) \in \mathcal{B}_1^{>0}$ for any $x \in \mathbb{R}_{>0}^{M-1}$. Besides, the diffeomorphism from $\mathbb{R}^{M-1}$ to $\mathbb{R}_{>0}^{M-1}$ is the entry-wise exponential map. Following $\mathbb{R}^{M-1} \to \mathbb{R}_{>0}^{M-1} \to \mathcal{B}_1^{>0} \to \mathcal{T}_f$, the composition of an entry-wise natural exponential map, f_1 and a scaling map gives the diffeomorphism in (73) and (74). The smoothness and bijectivity of the resultant map is evident. □

Appendix D: 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

$$\nabla H(y) = \mathbf{J}(y)^{\mathrm{T}} \nabla F(x) \tag{140}$$

always holds. 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 computed as

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

$$= \mathbf{J}(y)^{\mathrm{T}} \nabla^2 F(x) \mathbf{J}(y). \tag{141}$$

Then, due to the nonsingularity of $\mathbf{J}$, the positive-definiteness (or positive-semidefiniteness) of either $\nabla^2 F(x)$ or $\nabla^2 H(y)$ implies that of the other one. □

Appendix E: Proof of Proposition 3

Proof. The linear independence constraint qualification always holds because there is only one constraint. In the first statement, KKT conditions always hold at the constrained local minimum q. Then there is a Lagrange multiplier $\lambda \geq 0$, such that the following conditions hold:

$$\nabla F(q) + \lambda \nabla G(q) = \mathbf{0}, \tag{142}$$

$$\lambda \cdot G(q) = 0. \tag{143}$$

Denote by $\mathbf{J}$ the Jacobian of $f_{\mathcal{B}}$. For any ξ satisfying $f_{\mathcal{B}}(\xi) = q$, we have

$$\nabla H(\xi) = \mathbf{J}(\xi)^{\mathrm{T}} \nabla F(q) = -\lambda \mathbf{J}(\xi)^{\mathrm{T}} \nabla G(q). \tag{144}$$

If $\lambda = 0$, we obtain $\nabla H(\xi) = \mathbf{0}$. If $\lambda > 0$, the complementarity condition (143) implies $G(q) = 0$. Substituting $f_{\mathcal{B}}(\xi)$ into $G(q) = 0$ gives $\xi^{\mathrm{T}}\xi = 1$, $q = r \cdot \xi + o$ and $\nabla G(q) = 2r \cdot \xi$. By explicitly computing $\mathbf{J}(\xi)$, we have

$$\mathbf{J}(\xi) = \frac{2r \cdot \mathbf{I}_n}{\xi^{\mathrm{T}}\xi + 1} - \frac{4r \cdot \xi\xi^{\mathrm{T}}}{(\xi^{\mathrm{T}}\xi + 1)^2} = r \cdot (\mathbf{I}_n - \xi\xi^{\mathrm{T}}). \tag{145}$$

Now that ξ has unit length, we define an orthonormal matrix $\mathbf{B}_\xi = (\xi, \boldsymbol{\xi}^{\perp})$ where $\boldsymbol{\xi}^{\perp} \in \mathbb{R}^{n \times (n-1)}$ is an orthonomal basis of the null space of ξ^{T}. A singular value decomposition (SVD) of $\xi\xi^{\mathrm{T}}$ is then given by

$$\xi\xi^{\mathrm{T}} = \mathbf{B}_\xi \left(\mathbf{I}_1 \oplus \mathbf{0}_{n-1}\right) \mathbf{B}_\xi^{\mathrm{T}} \tag{146}$$

with $n \geq 2$, thus

$$\mathbf{J}(\xi) = \mathbf{B}_\xi \left(\mathbf{0}_1 \oplus r\mathbf{I}_{n-1}\right) \mathbf{B}_\xi^{\mathrm{T}}. \tag{147}$$

Then, we obtain

$$\nabla H(\xi) = -2\lambda r \cdot \mathbf{B}_\xi \left(\mathbf{0}_1 \oplus r\mathbf{I}_{n-1}\right) e_1 = \mathbf{0}, \tag{148}$$

where e_1 is the first column vector of $\mathbf{I}_n$. Therefore, the first statement always holds.

In the second statement, we have $\nabla H(\xi) = \mathbf{0}$ at the local minimum ξ. If $\xi^{\mathrm{T}}\xi \neq 1$, $\mathbf{J}(\xi)$ is always nonsingular and thus $\nabla F(q) = \mathbf{J}(\xi)^{-\mathrm{T}} \nabla H(\xi) = \mathbf{0}$. If $\xi^{\mathrm{T}}\xi = 1$, $q = f_{\mathcal{B}}(\xi)$ implies $G(q) = 0$, $q = r \cdot \xi + o$ and $\nabla G(q) = 2r \cdot \xi$.

Therefore, (147) also holds in this case. Because $\nabla H(\xi) = \mathbf{J}(\xi)^{\mathrm{T}}\nabla F(q)$, we have

$$\mathbf{B}_\xi \left(\mathbf{0}_1 \oplus r\mathbf{I}_{n-1}\right) \mathbf{B}_\xi^{\mathrm{T}} \nabla F(q) = \mathbf{0}. \tag{149}$$

For convenience, we decompose $\nabla F(q)$ on to the basis $\mathbf{B}_\xi$ as

$$\nabla F(q) = \mathbf{B}_\xi \begin{pmatrix} c \\ c^\perp \end{pmatrix}, \tag{150}$$

where $c \in \mathbb{R}$ and $c^\perp \in \mathbb{R}^{n-1}$. Thus,

$$\mathbf{B}_\xi \left(\mathbf{0}_1 \oplus r\mathbf{I}_{n-1}\right) \mathbf{B}_\xi^{\mathrm{T}} \mathbf{B}_\xi \begin{pmatrix} c \\ c^\perp \end{pmatrix} = \mathbf{0} \tag{151}$$

implies $c^\perp = \mathbf{0}$. Equivalently, we have $\nabla F(q) = c \cdot \xi$. Consider a scalar function defined as

$$H_\xi(s) = H(s \cdot \xi) = F(\frac{2rs}{s^2+1}\xi + o). \tag{152}$$

Computing the second-order derivative of H_ξ at $s = 1$ yields

$$\left.\frac{\mathrm{d}^2 H_\xi}{\mathrm{d}s^2}\right|_{s=1} = -r \cdot \xi^{\mathrm{T}} \nabla F(r \cdot \xi + o) = -rc. \tag{153}$$

Because ξ is a local minimum, we have $-rc \geq 0$, which means $c \leq 0$. By choosing $\lambda = -c/2r \geq 0$, we obtain the KKT conditions $\lambda \cdot G(q) = 0$ and

$$\nabla F(q) + \lambda \nabla G(q) = c \cdot \xi - \frac{c}{2r} \cdot 2r \cdot \xi = \mathbf{0}, \tag{154}$$

Therefore, the second statement always holds. □

Appendix F: Proof of Proposition 4

Proof. Define a neighborhood $\mathcal{N}_p^x(r)$ of a point $x \in \mathbb{R}^{\hat{n}}$ as

$$\mathcal{N}_p^x(r) = \left\{ y \in \mathbb{R}^{\hat{n}} \;\middle|\; \|y - x\|_p < r \right\}. \tag{155}$$

For any $\mathcal{X}, \mathcal{Y} \subseteq \mathbb{R}^{\hat{n}}$, we denote

$$f_c(\mathcal{X}) = \left\{ y \in \mathbb{R}^{\hat{n}} \;\middle|\; y = f_c(x),\ x \in \mathcal{X} \right\}, \tag{156}$$

$$f_c^{-1}(\mathcal{Y}) = \left\{ x \in \mathbb{R}^{\hat{n}} \;\middle|\; f_c(x) \in \mathcal{Y} \right\}. \tag{157}$$

Apparently, we have $\mathcal{P}_w^{\mathcal{H}} = f_c(\mathcal{B}^{\hat{n}})$ and $\mathcal{B}^{\hat{n}} = f_c^{-1}(\mathcal{P}_w^{\mathcal{H}})$. Moreover, $\forall w \in \mathcal{P}_w^{\mathcal{H}},\ f_c^{-1}(\{w\}) \neq \varnothing$. Now we show the proof by contradiction.

Assume that only the second statement is false. In other words, any v in $f_c^{-1}(\{w\})$ is never a local minimum. Then, $\forall \delta > 0,\ \exists \tilde{v} \in \mathcal{N}_2^v(\delta) \cap \mathcal{B}^{\hat{n}},\ s.t.\ H(\tilde{v}) < H(v)$. Besides, we have $f_c(\mathcal{N}_2^v(\delta) \cap \mathcal{B}^{\hat{n}}) \subseteq f_c(\mathcal{B}^{\hat{n}}) = \mathcal{P}_w^{\mathcal{H}}$. For any element $x \in \mathcal{N}_2^v(\delta) \cap \mathcal{B}^{\hat{n}}$, the Cauchy-Schwarz inequality implies

$$\left\|[x]^2 - [v]^2\right\|_1 \leq \|x + v\|_2 \|x - v\|_2 < 2 \cdot \delta. \tag{158}$$

Thus, $f_c(\mathcal{N}_2^v(\delta) \cap \mathcal{B}^{\hat{n}}) \subseteq \mathcal{N}_1^w(2\delta)$. Consequently, we have $\forall \delta > 0,\ \exists \tilde{w} = f_c(\tilde{v}) \in \mathcal{N}_1^w(2\delta) \cap \mathcal{P}_w^{\mathcal{H}},\ s.t.\ F(\tilde{w}) < F(w)$, which contradicts the first statement. Therefore, the first statement implies the second one.

Assume that only the first statement is false. In other words, we have $\forall \delta > 0,\ \exists \tilde{w} \in \mathcal{N}_2^w(\delta) \cap \mathcal{P}_w^{\mathcal{H}},\ s.t.\ F(\tilde{w}) < F(w)$. Equivalently, $\forall \delta > 0,\ \exists \tilde{w} \in \mathcal{N}_2^w(\delta) \cap \mathcal{P}_w^{\mathcal{H}},\ \forall \tilde{v} \in f_c^{-1}(\{\tilde{w}\}),\ s.t.\ H(\tilde{v}) < H(v)$. Obviously, there exists an element $\tilde{v} \in f_c^{-1}(\{\tilde{w}\})$ such that $\tilde{v} \circ v \succeq \mathbf{0}$. For such a $\tilde{v}$, the below inequality holds,

$$\left\|[\tilde{v}]^2 - [v]^2\right\|_2 \geq \|(\tilde{v} - v) \circ (\tilde{v} - v)\|_2 = \|\tilde{v} - v\|_4^2. \tag{159}$$

Due to $\|[\tilde{v}]^2 - [v]^2\|_2 < \delta$, we have $\tilde{v} \in \mathcal{N}_4^v(\sqrt{\delta})$. Then, $\forall \delta > 0,\ \exists \tilde{v} \in \mathcal{N}_4^v(\sqrt{\delta}) \cap \mathcal{B}^{\hat{n}},\ s.t.\ H(\tilde{v}) < H(v)$, which contradicts the second statement. Therefore, the second statement implies the first one. □

Appendix G: Calculation and Differentiation for $\mathbf{R}(t)$ and $\dot{\mathbf{R}}(t)$

For simplicity, we denote by p the translational trajectory of $p(t)$ and by ψ the yaw trajectory $\psi(t)$ of $Z(\psi) - X(\phi) - Y(\theta)$ Euler angles. The three columns of $\mathbf{R}(t)$ is denoted as x_b, y_b and z_b. Note that a dotted symbol stands for its derivative over t. For example, $\dot{\psi}$ and $\dddot{p}$ stands for $\mathrm{d}\psi/\mathrm{d}t$ and $\mathrm{d}^3 p/\mathrm{d}t^3$, respectively.

By calculation for $\mathbf{R}(t)$ and $\dot{\mathbf{R}}(t)$ we mean computing the value of $\mathbf{R}$ and $\dot{\mathbf{R}}$ using $\ddot{p}$, $\dddot{p}$, ψ, and $\dot{\psi}$. According to the simplified dynamics (100), z_b is aligned with the direction of thrust, thus

$$\begin{aligned} z_b &= f_{\mathcal{N}}(\ddot{p} + \bar{g}e_3), \\ \dot{z}_b &= f_{\mathcal{DN}}(\ddot{p} + \bar{g}e_3)\dddot{p}, \end{aligned}$$

where $f_{\mathcal{N}} : \mathbb{R}^3 \mapsto \mathbb{R}^3$ and $f_{\mathcal{DN}} : \mathbb{R}^3 \mapsto \mathbb{R}^{3\times 3}$ are defined by

$$\begin{aligned} f_{\mathcal{N}}(x) &= x / \|x\|_2, \\ f_{\mathcal{DN}}(x) &= \left(\mathbf{I}_3 - \frac{xx^{\mathrm{T}}}{x^{\mathrm{T}}x}\right) / \|x\|_2. \end{aligned}$$

We introduce some temporary vectors

$$\begin{aligned} x_c &= (\cos\psi, \sin\psi, 0)^{\mathrm{T}}, \\ \dot{x}_c &= (-\sin\psi, \cos\psi, 0)^{\mathrm{T}}\dot{\psi}, \\ y_c &= z_b \times x_c, \\ \dot{y}_c &= \dot{z}_b \times x_c + z_b \times \dot{x}_c, \end{aligned}$$

where x_c is the heading vector with only yaw rotation and y_c is collinear with y_b for our Euler intrinsic rotations. Therefore, we have

$$\begin{aligned} y_b &= f_{\mathcal{N}}(y_c), \\ \dot{y}_b &= f_{\mathcal{DN}}(y_c)\dot{y}_c, \\ x_b &= y_b \times z_b, \\ \dot{x}_b &= \dot{y}_b \times z_b + y_b \times \dot{z}_b. \end{aligned}$$

Finally, we obtain

$$\mathbf{R} = (x_b, y_b, z_b), \ \dot{\mathbf{R}} = (\dot{x}_b, \dot{y}_b, \dot{z}_b).$$

By differentiation for $\mathbf{R}(t)$ and $\dot{\mathbf{R}}(t)$ we mean computing the value of $\partial\mathbf{R}/\partial\varepsilon$ and $\partial\dot{\mathbf{R}}/\partial\varepsilon$ using $\partial\ddot{p}/\partial\varepsilon$, $\partial\dddot{p}/\partial\varepsilon$, $\partial\psi/\partial\varepsilon$, $\partial\dot{\psi}/\partial\varepsilon$, $\ddot{p}$, $\dddot{p}$, ψ, and $\dot{\psi}$ for any intermediate variable $\varepsilon \in \mathbb{R}$. Differentiating $\mathbf{R}$ and $\dot{\mathbf{R}}$ gives

$$\begin{aligned} \frac{\partial\mathbf{R}}{\partial\varepsilon} &= \left(\frac{\partial x_b}{\partial\varepsilon}, \frac{\partial y_b}{\partial\varepsilon}, \frac{\partial z_b}{\partial\varepsilon}\right), \\ \frac{\partial\dot{\mathbf{R}}}{\partial\varepsilon} &= \left(\frac{\partial\dot{x}_b}{\partial\varepsilon}, \frac{\partial\dot{y}_b}{\partial\varepsilon}, \frac{\partial\dot{z}_b}{\partial\varepsilon}\right). \end{aligned}$$

The column vectors are computed as follows,

$$
\begin{aligned}
&\frac{\partial z_b}{\partial \varepsilon} = f_{\mathcal{DN}}(\ddot{p} + \bar{g}e_3)\frac{\partial \ddot{p}}{\partial \varepsilon},\\
&\frac{\partial \dot{z}_b}{\partial \varepsilon} = f_{\mathcal{D}^2\mathcal{N}}(\ddot{p} + \bar{g}e_3, \frac{\partial \ddot{p}}{\partial \varepsilon})\dddot{p} + f_{\mathcal{DN}}(\ddot{p} + \bar{g}e_3)\frac{\partial \dddot{p}}{\partial \varepsilon},\\
&\frac{\partial x_c}{\partial \varepsilon} = (-\sin\psi, \cos\psi, 0)^{\mathrm{T}}\frac{\partial \psi}{\partial \varepsilon},\\
&\frac{\partial \dot{x}_c}{\partial \varepsilon} = -(\cos\psi, \sin\psi, 0)^{\mathrm{T}}\frac{\partial \psi}{\partial \varepsilon}\dot{\psi} + (-\sin\psi, \cos\psi, 0)^{\mathrm{T}}\frac{\partial \dot{\psi}}{\partial \varepsilon},\\
&\frac{\partial y_c}{\partial \varepsilon} = \frac{\partial z_b}{\partial \varepsilon} \times x_c + z_b \times \frac{\partial x_c}{\partial \varepsilon},\\
&\frac{\partial \dot{y}_c}{\partial \varepsilon} = \frac{\partial \dot{z}_b}{\partial \varepsilon} \times x_c + \dot{z}_b \times \frac{\partial x_c}{\partial \varepsilon} + \frac{\partial z_b}{\partial \varepsilon} \times \dot{x}_c + z_b \times \frac{\partial \dot{x}_c}{\partial \varepsilon},\\
&\frac{\partial y_b}{\partial \varepsilon} = f_{\mathcal{DN}}(y_c)\frac{\partial y_c}{\partial \varepsilon},\\
&\frac{\partial \dot{y}_b}{\partial \varepsilon} = f_{\mathcal{D}^2\mathcal{N}}(y_c, \frac{\partial y_c}{\partial \varepsilon})\dot{y}_c + f_{\mathcal{DN}}(y_c)\frac{\partial \dot{y}_c}{\partial \varepsilon},\\
&\frac{\partial x_b}{\partial \varepsilon} = \frac{\partial y_b}{\partial \varepsilon} \times z_b + y_b \times \frac{\partial z_b}{\partial \varepsilon},\\
&\frac{\partial \dot{x}_b}{\partial \varepsilon} = \frac{\partial \dot{y}_b}{\partial \varepsilon} \times z_b + \dot{y}_b \times \frac{\partial z_b}{\partial \varepsilon} + \frac{\partial y_b}{\partial \varepsilon} \times \dot{z}_b + y_b \times \frac{\partial \dot{z}_b}{\partial \varepsilon},
\end{aligned}
$$

where $f_{\mathcal{D}^2\mathcal{N}} : \mathbb{R}^3 \times \mathbb{R}^3 \mapsto \mathbb{R}^{3\times 3}$ is defined by

$$
f_{\mathcal{D}^2\mathcal{N}}(x, y) = \frac{x^{\mathrm{T}}y}{\|x\|_2^3}\left(\frac{3xx^{\mathrm{T}}}{x^{\mathrm{T}}x} - \mathbf{I}_3\right) - \frac{xy^{\mathrm{T}} + yx^{\mathrm{T}}}{\|x\|_2^3}.
$$