2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

TOPPQuad: Dynamically-Feasible Time-Optimal Path Parametrization for Quadrotors

Katherine Mao, Igor Spasojevic, M. Ani Hsieh, and Vijay Kumar

***Abstract*— Planning time-optimal trajectories for quadrotors in cluttered environments is a challenging, non-convex problem. This paper addresses minimizing the traversal time of a given collision-free geometric path without violating actuation bounds of the vehicle. Previous approaches have either relied on convex relaxations that do not guarantee dynamic feasibility or have generated overly conservative time parametrizations. We propose TOPPQuad, a time-optimal path parameterization algorithm for quadrotors which explicitly incorporates quadrotor rigid body dynamics and constraints, such as bounds on inputs (including motor thrusts) and state of the vehicle (including the pose, linear and angular velocity and acceleration). We demonstrate the ability of the planner to generate faster trajectories that respect hardware constraints of the robot compared to planners with relaxed notions of dynamic feasibility in both simulation and hardware. We also demonstrate how TOPPQuad can be used to plan trajectories for quadrotors that utilize bidirectional motors. Overall, the proposed approach paves a way towards maximizing the efficacy of autonomous micro aerial vehicles while ensuring their safety.**

I. Introduction

Autonomous micro aerial vehicles (MAVs) have demonstrated an immense potential to transform logistics in numerous domains. Package delivery in large metropolitan areas with dense traffic, aid distribution in disaster-stricken environments, and inventory management in large warehouses are only a few such examples. The unique combination of size and aerial agility makes MAVs ideal for navigating cluttered 3-D environments beyond the reach of other robots. However, key to realizing their full potential in boosting task productivity while ensuring safe environmental interaction lies in planning missions that respect their physical limitations. This work addresses a class of problems in planning dynamically-feasible time-optimal trajectories for MAVs.

Planning time-optimal trajectories for robots subject to actuation constraints in cluttered environments is a challenging task. First, the presence of obstacles renders the underlying optimization problem non-convex, even for systems with linear dynamics. Second, the high-dimensional non-Euclidean state space of most aerial vehicles makes trajectory synthesis subject to actuation constraints formidable, especially in obstacle-rich environments. One method of addressing these hurdles is a decoupled approach to motion planning [1]. First, a sufficiently smooth collision-free geometric path is established. A corresponding optimal time parametrization respecting actuation constraints is then determined. Our work focuses on the second stage, the minimization of the traversal time of a given geometric path while respecting individual motor thrusts bounds of the MAV.

Fig. 1. A time-optimal Lissajous curve trajectory computed with the TOPPQuad algorithm and tracked by a CrazyFlie quadrotor with the SE(3) Geometric Controller [8] in a Vicon motion capture system. TOPPQuad is able to plan trajectories at speeds otherwise infeasible by conventional trajectory planners.

A landmark result [2] previously established the *differential flatness* [3] of quadrotor dynamics. This initiated a line of computationally-efficient approaches [2], [4], [5] that optimize trajectories in the space of (piecewise) polynomials of quadrotor positions and yaw angles. It also demonstrated the highly non-convex dependence of motor thrusts on the underlying flat outputs, posing challenges in determining trajectories with explicit constraints on motor thrusts. Most previous works either (a) use (convex) relaxations of actuation constraints [6], (b) use bounds on the total thrust and angular velocity instead of actual motor thrusts [7], or (c) plan dynamically feasible trajectories without clear bounds on suboptimality [5]. Our approach can be viewed as a generalization of [5] in that we use a *spatially-varying* dilation of time to transform any sufficiently smooth trajectory into a feasible one.

Optimizing spatially-varying time dilations of a trajectory is also known as the Time Optimal Path Parametrization (TOPP) problem. Early works addressed this in the context of planning trajectories for manipulators subject to bounds on its joint velocities and torques [9]. Verscheure *et al* introduced the square speed profile, uncovering the hidden convexity behind such problems [10]. This approach has been adopted to many other robotic systems [11], [12] and various

This works was supported by NSF Grant CCR-2112665, the ARL DCIST CRA W911NF-17-2-0181, and ONR Grant N00014-20-1-2822. Katherine Mao, Igor Spasojevic, M. Ani Hsieh, and Vijay Kumar are with the GRASP Laboratory, University of Pennsylvania, PA, 19104, USA {maokat, igorspas, mya, kumar}@seas.upenn.edu

computational improvements to the original method have been proposed [13]. However, none of these approaches deal with the bounds involved in planning the critical rotational components of vehicle trajectories. This is due to the non-convex constraints introduced by the inclusion of rotation, even with the aid of the square speed profile. Addressing the rotational components of trajectory generation for MAVs is one of the core themes of this paper.

Various existing works have explored planning time-optimal quadrotor trajectory with individual motor thrust constraints. Ryou *et al* use Gaussian Processes to learn a time-optimal obstacle-free trajectory from repeated simulation results, but any changes to the set of waypoints or obstacle space requires re-training the network [14]. Foehn *et al* formulate the planning problem by trajectory 'progress' to find time-optimal trajectories and allows minor waypoint deviation, but in doing so assumes an sparse environment [15]. A defining characteristic of these methods is the modification of the original geometric path, a potentially problematic attribute for planning time-optimal trajectories in obstacle-dense environments.

Towards this end, we propose TOPPQuad, a TOPP algorithm that considers the full rigid body dynamics of the quadrotor. **The contributions of this paper are as follows:**

- A trajectory planner capable of refining time-optimal trajectories obtained from any arbitrary planner.
- A comparison of the resulting trajectories with existing planners in simulation and real-world experiments. We show that inclusion of rotational variables leads to faster dynamically feasible trajectories.
- A method of planning time-optimal trajectories for a bidirectional quadrotor (i.e. one whose motor thrusts can be both parallel and antiparallel to its body z-axis), that seamlessly bypasses switching between flatness diffeomorphisms.
- A demonstration of the trackability of our trajectories in hardware experiments

II. Problem Formulation

We consider the problem of minimizing the traversal time of an arbitrary sufficiently smooth geometric path for an actuation-constrained quadrotor. A *geometric* path

$$\gamma : [0, S_{end}] \to \mathbb{R}^3 \tag{1}$$

is one not necessarily parametrized by time, but rather by any abstract parameter denoted by $s \in [0, S_{end}]$. We model the quadrotor using the standard rigid body dynamics model given by

$$\underbrace{\begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{v}} \\ \dot{\mathbf{R}} \\ \dot{\omega} \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} \mathbf{v} \\ \mathbf{R}\mathbf{e}_3 \, \frac{c}{m} + \mathbf{g} \\ \mathbf{R}[\omega\times] \\ \mathbf{J}^{-1}(\tau - \omega \times \mathbf{J}\omega) \end{bmatrix}}_{\mathbf{f}(\mathbf{x},\mathbf{u})}, \tag{2}$$

where the total thrust $c \in \mathbb{R}$ and torque $\tau \in \mathbb{R}^3$ are given by

$$\begin{bmatrix} c \\ \tau \end{bmatrix} = \underbrace{\mathbf{F}}_{\in \mathbf{R}^{4\times 4}} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}. \tag{3}$$

Let $[\omega\times]$ be defined as

$$[\omega\times] = \begin{bmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{bmatrix}. \tag{4}$$

In Eq. (3), $\mathbf{F}$ is a constant *scaled* control matrix, and $\mathbf{u} = [u_1, u_2, u_3, u_4]^T \in \mathbb{R}^4$ represents control inputs — the individual motor thrusts. The position of the vehicle's center of mass with respect to (w.r.t.) the world frame is denoted by $\mathbf{p} \in \mathbb{R}^3$, its velocity by $\mathbf{v}$, the columns of $\mathbf{R} \in SO(3)$ encode its body axes as linear combinations of world axes $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$, $\omega \in \mathbb{R}^3$ its angular velocity expressed in the *body* frame of the vehicle, and $\mathbf{g}$ the gravity vector.

The minimum-time path traversal problem can then be specified as an optimization problem over the total execution time T and a sufficiently smooth time parametrization function $\chi(\cdot)$:

$$\begin{aligned} \min_{\substack{T>0,\\ \chi:[0,T]\to[0,S_{end}]}} \quad & T \\ s.t.\ & \chi(0) = 0,\ \chi(T) = S_{end}, \\ & \chi(\cdot) \text{ increasing on } [0, S_{end}], \\ & \mathbf{p}(t) = \gamma(\chi(t)), \quad \text{(path following constraint)} \\ & \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \quad \text{(dynamics constraint)} \\ & \mathcal{H}(\mathbf{x}(t), \chi(t)) \le 0, \qquad \text{(state constraints)} \\ & u_{min} \le \mathbf{u}(t) \le u_{max} \quad \text{(actuation constraint).} \end{aligned} \tag{5}$$

The relation $\mathcal{H}(\mathbf{x}(t), \chi(t)) \le 0$ collects state constraints such as bounds on the velocity, acceleration, and angular velocity of the vehicle. Additional requirements could include pose constraints at specified points along γ that ensure the robot can pass through narrow gaps.

III. Methodology

A. Problem Reparametrization

Solving Problem (5) for many types of robots is easier upon introduction of the *square speed profile* [10], [11], defined as $h(s) := \left(\frac{ds}{dt}(s)\right)^2 \ \forall s \in [0, S_{end}]$. h represents the square speed of the robot as a function of distance traversed along the path. One motivation for the square speed profile comes from the elementary relation $\Delta v^2 = 2a\Delta s$; a bound on the maximum acceleration is equivalent to a bound on the Lipschitz constant of v^2 (i.e. h) w.r.t. s — a linear (and thus convex) constraint [10], [16], [17].

In addition, we rewrite the orientation kinematics ($\dot{\mathbf{R}} = \mathbf{R}[\omega\times]$) as

$$\dot{\mathbf{q}}(t) = \frac{1}{2}\Omega(\omega)\mathbf{q}(t), \tag{6}$$

where $\mathbf{q}(t) \in S^3$ is a quaternion representing the rotation matrix $\mathbf{R}$, and

$$\Omega(\omega) := \begin{bmatrix} 0 & -\omega^T \\ \omega & -[\omega\times] \end{bmatrix}. \tag{7}$$

Let $(\cdot)' = d(\cdot)/ds$ and $(\dot{\cdot}) = d(\cdot)/dt$. Using the relation

$$\frac{d}{dt} = \frac{ds}{dt}\frac{d}{ds} = \sqrt{h(s)}\frac{d}{ds}, \tag{8}$$

we rewrite the objective and dynamics as follows. The objective function can be recovered as

$$T = \int_0^{S_{end}} dt = \int_0^{S_{end}} \frac{dt}{ds} ds = \int_0^{S_{end}} \frac{1}{\sqrt{h(s)}} ds. \tag{9}$$

We next unpack and rewrite the dynamic constraints in Problem (5). For the translational dynamics, the crucial relation is $\mathbf{p}(s) = \gamma(s)$. Differentiating, we get

$$\dot{\mathbf{p}} = \sqrt{h(s)}\mathbf{p}'(s) = \sqrt{h(s)}\gamma'(s), \tag{10}$$

$$\dot{\mathbf{v}} = \frac{1}{2}\gamma'(s)h'(s) + \gamma''(s)h(s). \tag{11}$$

Substituting these into the second line of Eq. (2),

$$\frac{1}{2}\gamma'(s)h'(s) + \gamma''(s)h(s) = \mathbf{R}(s)\mathbf{e}_3 \ \frac{c(s)}{m} + \mathbf{g}. \tag{12}$$

For the rotational dynamics, we write

$$\mathbf{q}'(s) = \frac{1}{2}\Omega([\omega(s)\times])\mathbf{q}(s). \tag{13}$$

where the relationship between ω from Eq. (6) and (13) is

$$\omega(t) = \sqrt{h(s)}\omega(s). \tag{14}$$

Noting the parallel between Eq. (10) and (14), and defining

$$\alpha(s) := \omega'(s), \tag{15}$$

we get the relation

$$\dot{\omega} = \frac{1}{2}\omega(s)h'(s) + \alpha(s)h(s). \tag{16}$$

This ultimately allows us to rewrite line 4 of Eq. (2) as

$$\tau = \mathbf{J}\underbrace{\left(\frac{1}{2}\omega(s)h'(s) + \alpha(s)h(s)\right)}_{=\ \dot{\omega}} + \omega(s)\times\mathbf{J}\omega(s)\ h(s). \tag{17}$$

Combining Eq. (12) and (17), we get that the individual motor thrusts at point s along the path are related to the translational and angular accelerations via

$$\begin{bmatrix} m(\frac{1}{2}\gamma'(s)h'(s) + \gamma''(s)h(s) - \mathbf{g}) \\ \mathbf{J}\left(\frac{1}{2}\omega(s)h'(s) + \alpha(s)h(s)\right) + \omega(s)\times\mathbf{J}\omega(s)\ h(s) \end{bmatrix} = \begin{bmatrix} \mathbf{R}(s)\mathbf{e}_3 & 0 \\ 0 & \mathbf{I}_3 \end{bmatrix}\mathbf{F}\mathbf{u} \tag{18}$$

B. Numerical Implementation

To solve Problem (5), we use the following decision variables:

$$h(\cdot),\ h'(\cdot),\ \mathbf{q}(\cdot),\ \omega(\cdot),\ \alpha(\cdot),\ \mathbf{u}(\cdot), \tag{19}$$

which are all functions on $[0, S_{end}]$ as related by equations in III-A. We approximately represent these *functional* decision variables by their values at a finite set of N grid points $D = (0 = s_0 < s_1 < \cdots < s_N = S_{end})$ spaced Δs apart. Thus, $h_i := h(s_i)$ and likewise for the remaining variables in Eq. (19). The differential constraints are approximated using forward Euler integration. As such, the effective dynamic constraints for $h(\cdot)$ and $\omega(\cdot)$ become

$$\begin{bmatrix} h_{i+1} \\ \omega_{i+1} \end{bmatrix} = \begin{bmatrix} h_i + h_i' \ \Delta s \\ \omega_i + \alpha_i \ \Delta s \end{bmatrix} \tag{20}$$

$$\begin{bmatrix} m(\frac{1}{2}\gamma_i' h_i' + \gamma_i'' h_i - \mathbf{g}) \\ J\left(\frac{1}{2}\omega_i h_i' + \alpha_i h_i\right) + \omega_i \times J\omega_i \ h_i \end{bmatrix} = \begin{bmatrix} \mathbf{R}_i\mathbf{e}_3 & 0 \\ 0 & \mathbf{I}_3 \end{bmatrix}\mathbf{F}\mathbf{u}_i. \tag{21}$$

Actuation constraints amount to $\mathbf{u}_i \in [u_{min}, u_{max}]$, and the objective function, the total execution time, equals

$$T = \sum_{i=0}^{N-1} \frac{2\Delta s}{\sqrt{h_i} + \sqrt{h_{i+1}}}. \tag{22}$$

The critical numerical approximation is in the rotational kinematics. We approximate the dynamics of q via

$$\mathbf{q}_{i+1} = \frac{(\mathbf{I}_4 + \frac{\Delta s}{2}\Omega(\omega_i))\ \mathbf{q}_i}{\sqrt{1 + \frac{\Delta s^2}{4}||\omega_i||_2^2}}, \tag{23}$$

where $\mathbf{I}_4$ denotes the 4×4 identity matrix. The numerator of Eq. (23) performs a first-order Euler integration of quaternion kinematics, then the denominator projects this value back onto S^3, the manifold of quaternions.

IV. Simulation Setup

We benchmark TOPPQuad against a family of flatness-based planners and alternative TOPP procedures based on convex relaxations of the dynamic constraints. All planners are written in Python and computed on a laptop with an i7-6700HQ CPU. We use quadrotor parameters from the CrazyFlie 2.0 from Bitcraze [18], with actuation constraints $\mathbf{u}_i = [0, 0.14375]N$. The flatness-based planners are implemented with a linear least-squares program [19], the TOPP methods with `cvxpy` [20], and our TOPPQuad algorithm with the CasADi interface to IPOPT [21]. The planners are compared on a set of 200 randomized trajectories, each formed by interpolating four waypoints sampled from within a $10m \times 10m \times 10m$ box.

A. Baselines

A defining characteristic of TOPPQuad is the generation of time-optimal trajectories that do not modify the geometric path of the original trajectory. As such, we compare only against trajectory planners that maintain this characteristic.

We select three sets of commonly-used differential flatness-based planners as baselines: minimum snap, minimum jerk, and minimum acceleration [5]. Within each set, we apply the following three types of constraints, while ensuring the geometric path remains fixed:

1) An unconstrained flatness-based trajectory planner

Waypoint visit times are pre-assigned, with the time interval between consecutive waypoints determined from their Euclidean distance and a nominal velocity v. All geometric paths are based on this output.

2) An upper bound on the maximum speed along the path

A convex relaxation of the TOPP problem on the path from 1) is solved by placing an equivalent bound on h.

$$\begin{aligned} &\min \sum_{i=0}^{N-1} \frac{2\Delta s}{\sqrt{h_i}+\sqrt{h_{i+1}}} + \lambda \sum_{i=0}^{N} (h_i^{(3)})^2 \\ &s.t. \quad 0 \leq h_i \leq \frac{v_{max}^2}{||\gamma_i'||_2^2} \quad \forall i \leq N. \end{aligned} \tag{24}$$

3) An upper bound on the maximum speed and total thrust

Additional bounds on maximum total thrust are added to the TOPP problem in 2).

$$\begin{aligned} &\min \sum_{i=0}^{N-1} \frac{2\Delta s}{\sqrt{h_i}+\sqrt{h_{i+1}}} + \lambda \sum_{i=0}^{N} (h_i^{(3)})^2 \\ &s.t. \quad 0 \leq h_i \leq \frac{v_{max}^2}{||\gamma_i'||_2^2} \\ &\qquad \left|\left| \frac{1}{2}\gamma_i^{'} h_i^{'} + \gamma_i^{''} h_i - \mathbf{g} \right|\right|_2 \leq 4\frac{u_{max}}{m} \quad \forall i \leq N. \end{aligned} \tag{25}$$

For 2) and 3), we represent the square speed profile as a third-order integrator to ensure a $\mathcal{C}^3$ trajectory and thus continuity up to jerk. We add a regularization term $\lambda \sum_{i=0}^{N} (h_i^{(3)})^2$ to the objectives of Eq. (24) and (25) for a small positive λ to ensure our optimization problems are numerically well conditioned. The sub-optimality of the solutions of convex relaxations decreases as $\lambda \downarrow 0$, but non-vanishing values of λ are necessary for the solver to converge. In our experiments, we chose $\lambda = 10^{-1}$.

We refer to "α-scaling" (α) as the method introduced in [5], which slows down the time uniformly across the trajectory until all motor thrusts lie within allowed ranges. This is equivalent to multiplying the square speed profile and its spatial derivatives by a suitable quantity (less than one). We perform α-scaling on each of the baselines mentioned above to enforce dynamic feasibility.

B. Initial Guess

TOPPQuad requires an initial guess of variables in Eq. (19), the quality of which can greatly affect optimizer performance. For the following comparisons, the initial guess is supplied from the baseline trajectories. To determine a 'good' initial guess, we examine the effect of three parameters: the planner's nominal velocity v, the constraints on the planner, and Δs determined by the number of discretization points. Failure is considered to be one of two cases: when the CasADi solver fails to find a solution, or when the solver settles on a local minimum with a significantly slower traversal time than the initial guess.

We see that of three nominal velocities, $v = 1m/s$ offers the best performance in both iteration count and success rate, as shown in Table I. A further examination shows that of the three cases considered, only at $v = 1m/s$ do the motor thrusts of the initial guess consistently stay within the input bounds of the quadrotor.

Nominal Velocity (v)	**1m/s**	**3m/s**	**5m/s**
Avg Iterations	473	659	1138
Success Rate	0.981	0.918	0.609

TABLE I

TOPPQUAD INITIAL GUESS SUCCESS - VELOCITY COMPARISON

We then compare in Table II the algorithm's optimization success rate given initial guesses from four planners: 1) a minimum snap (MS) trajectory computed with $v = 1m/s$, 2) an α-scaled minimum snap trajectory computed with some higher v, 3) an α-scaled TOPP trajectory with only speed bounds, and 4) an α-scaled TOPP trajectory with speed and acceleration bounds. We see that the initial guesses from 1) have the highest success rates.

	MS (1 m/s)	**α MS**	**α TOPP vel**	**α TOPP acc**
Success Rate	0.991	0.981	0.936	0.891

TABLE II

TOPPQUAD INITIAL GUESS SUCCESS - PLANNER COMPARISON

Based on these two observations, we postulate that an initial guess with motor speeds far from the bounds of the input constraints is more likely to succeed, and in fewer iterations. We proceed using the $v = 1m/s$ flatness-based planner initial guess for the remaining experiments.

For the trajectories in this experiment, we find $N = 300$ ($\Delta s \approx 0.025$) leads to adequate optimizer convergence. Selecting N too coarsely will see IPOPT frequently fail to converge, as the discrete dynamics in Eq. (21), (23) no longer accurately model the continuous dynamics in Eq. (18). On the other hand, selecting N too finely will drastically increase run time for diminishing return in trajectory feasibility with a controller in the loop.

V. COMPARISON OF PLANNING ALGORITHMS

In this section, we first demonstrate the effectiveness of our algorithm along two metrics: dynamic feasibility and time optimality. For dynamic feasibility, we compare our optimized trajectories against the aforementioned baselines. For time optimality, we compare our algorithm against only the dynamically feasible α-scaled baselines. Next, we extend our formulation to trajectory generation for bidirectional quadrotors. In addition, we demonstrate the flexibility of the initial guess with real-world trajectories from odometry data.

For a fair comparison, we use $v = 5m/s$ as the nominal velocity for the flatness-based planners and likewise. Thus, we show that given the same maximum velocity, our TOPPQuad trajectories are able to take fuller advantage of

the quadrotor's flight envelope and achieve higher average speeds. The computed trajectories average $27.5m$ in length for the minimum snap planner, $26.8m$ for the minimum jerk planner, and $21.4m$ for the minimum acceleration planner.

A. Dynamic Feasibility

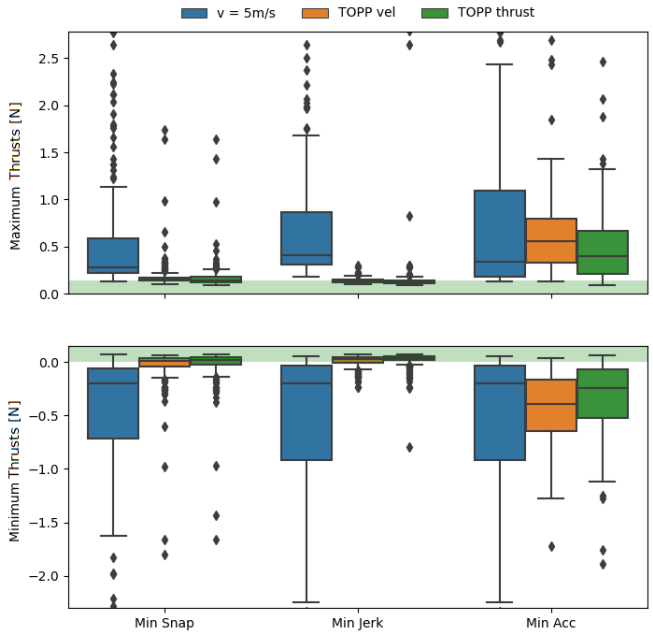

Fig. 2. The distribution of maximum (**top**) and minimum (**bottom**) thrusts over 200 randomized trajectories for three sets of planners: minimum snap (**left**), minimum jerk (**middle**), and minimum acceleration (**right**). In each set, we compare a flatness-based planner at $v = 5m/s$ (**blue**), a TOPP planner subject to velocity constraints (**orange**), and a TOPP planner subject to velocity and thrust constraints (**dark green**). No planner consistently plans trajectories that stay within motor thrust bounds (**light green**).

We measure dynamic feasibility by the percentage of trajectories that stay within motor thrust bounds. Fig. 2 shows that although trajectories have constraints on velocity and acceleration, commonly used convex approximations, they often require inputs that strongly exceed allowed motor thrust bounds. In Fig. 3. we focus only on the motor thrusts of dynamically feasible planners, including TOPPQuad. While all planners generate trajectories that respect motor thrust bounds, the three baseline methods are conservative compared to the TOPPQuad trajectories, and do not always take full advantage of the quadrotor's flight capabilities.

B. Time Optimality

Next, we focus on the time optimality of the dynamically feasible trajectory planners: the α-scaled methods and TOPPQuad. Fig. 4 shows that our TOPPQuad method consistently generates faster trajectories than the α-scaled methods. We see anywhere between a $2-5$ sec decrease from the unconstrained trajectories and about a 2 sec decrease from the TOPP trajectories. This corresponds to an approximate $1.8m/s$ and $0.8m/s$ increase in respective average trajectory velocity. Note that the shorter average speed of the minimum

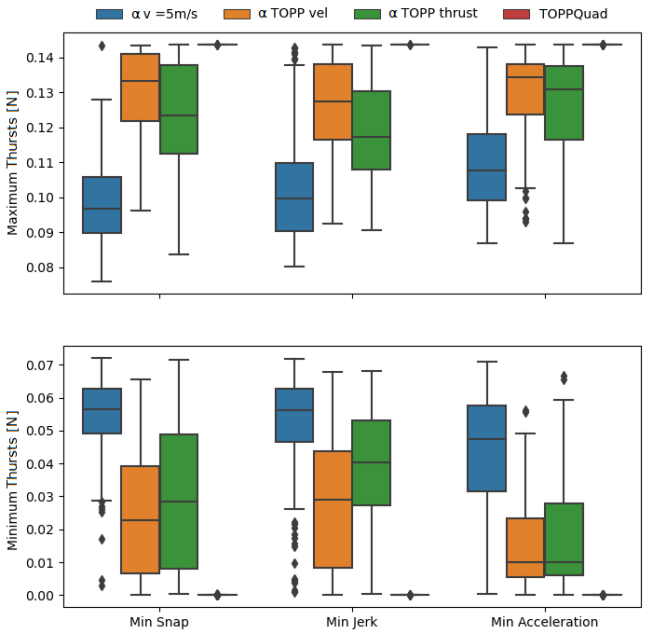

Fig. 3. The distribution of maximum (**top**) and minimum (**bottom**) thrusts over 200 randomized trajectories for three sets of α-scaled planners: minimum snap (**left**), minimum jerk (**middle**), and minimum acceleration (**right**). In each set, we compare against an α-scaled flatness-based planner (**blue**), an α-scaled TOPP planner subject to velocity constraints (**orange**), and an α-scaled TOPP planner subject to velocity and thrust constraints (**green**). Only TOPPQuad (**red**) is able to take full range of the quadrotor's inputs.

acceleration trajectories is due to the faster average trajectory, not algorithm performance.

In Fig. 5, we show the percentage time decrease between our method and the three dynamically feasible baselines for the minimum snap case, as well as the $v = 1m/s$ initial guess. For the α-scaled minimum snap planner, the majority of trajectories see a time decrease of 40% or more, and even the TOPP methods see at least a 10% decrease.

C. Bidirectionality

The TOPPQuad planner can be easily adapted for optimizing trajectories for bidirectional quadrotors while sidestepping the difficulties of planning for a flatness-based model [22]. Such vehicles are governed by the same set of rigid body dynamics as 'unidirectional' quadrotors, but can also exert thrust along the negative body z-axis of the robot.

We explore how much efficiency can be gained with bidirectional motors. At $v_{max} = 5m/s$, we see a traversal time performance on par with or only marginally faster than the unidirectional quad. This can likely be attributed to the saturation of motor thrusts and the maximum velocity constraint, preventing further time improvements.

In Fig. 6, we show the same trajectory computed for a time-optimal unidirectional and bidirectional quadrotor, with the maximum speed capped at $v = 10m/s$. While the total traversal time improved by only around 0.2s, the orientations along the path of the bidirectional trajectory appear smoother than their unidirectional counterparts. We suspect this is due to the quadrotor's ability to temporarily

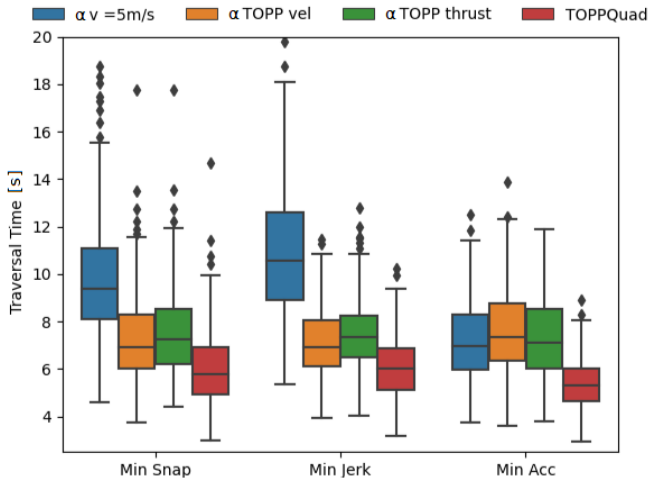

Fig. 4. The distribution of traversal times over 200 randomized trajectories for three sets of α-scaled planners: minimum snap (**left**), minimum jerk (**middle**), and minimum acceleration (**right**). In each set, we compare the traversal time for a α-scaled flatness-based planner (**blue**), an α-scaled TOPP planner subject to velocity constraints (**orange**), an α-scaled TOPP planner subject to velocity and thrust constraints (**green**), and our TOPPQuad algorithm (**red**).

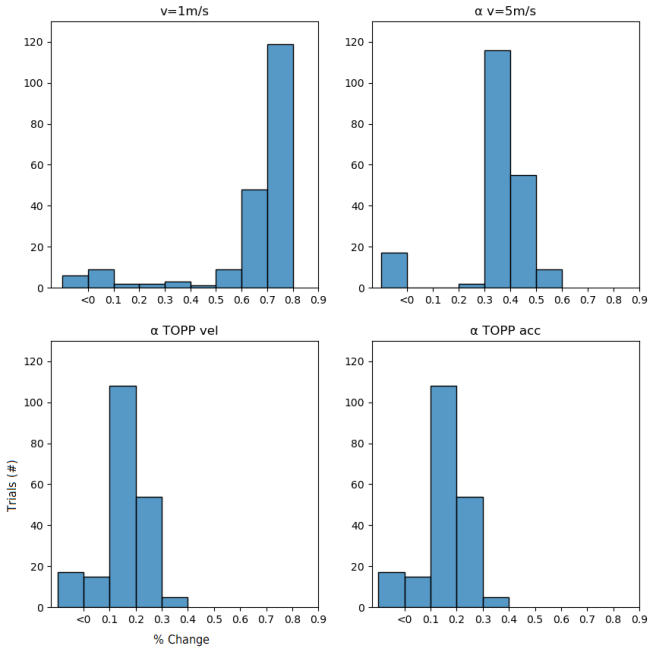

Fig. 5. The distribution of traversal time improvement over the four dynamically feasible planners when refined with TOPPQuad: min snap $v = 1m/s$ (**top left**), α-scaled $v = 5m/s$ min snap (**top right**), α-scaled TOPP w/velocity constraints (**bottom left**), and α-scaled TOPP w/acceleration constraints (**bottom right**). Values are computed against the initial $v = 1m/s$ trajectory guess. Our algorithm consistently sees improvements in total traversal time, even against other time-optimal methods.

dip into the negative thrust regime, thus allowing for more rapid orientation change and longer saturation of maximum motor thrusts. The addition of motor thrust rate of change constraints may dampen this effect, but this initial exploration demonstrates the potential for quadrotors with bidirectional thrust in traversing time optimal trajectories.

D. TOPPQuad in the Wild

Here, we demonstrate the performance of our planner on real-world flight data, when not all initial guess values are known and the flight path may not be smooth. We apply TOPPQuad to a trajectory flown in the Wharton State Forest, NJ, USA, where a Falcon IV quadrotor autonomously navigated under the canopy of a large, tree-dense environment ([23], [24]). We recover the geometric path and the initial guess for TOPPQuad from the flight's odometry data. As we do not have estimates of motor speeds or angular acceleration, we use the values of the hover configuration ($4108\ rad/s, 0\ rad/s^2$) along the full trajectory. Fig. 7 shows the increase in speed along the path. The total trajectory time decreases from 236 sec to 136 sec across a total trajectory length of 336.2 m.

VI. Deployment Results

We validate the performance of our TOPPQuad trajectories with deployment on a real-world quadrotor platform, compared against minimum snap trajectories. We use a Crazyflie 2.0 from Bitcraze, with a nominal weight of $32g$. All experiments were performed in a VICON motion capture system to collect the ground truth state information. Trajectories were computed offboard and commands sent to the quadrotor from a computer base-station at a frequency of 100 Hz.

We use the $SE(3)$ Geometric Controller [8] for trajectory tracking. PD controller gains are listed in Table III. Given that the controller samples at a higher frequency than the TOPPQuad trajectory discretization, we rely on an interpolation of points to determine the requisite flat outputs at each time stamp. For the positional flat outputs (position, velocity, and yaw), we fit a $5th$-order polynomial to the corresponding pair of positions, velocitues, and accelerations. For yaw, we fit a quaternion spline [25] to the corresponding pair of quaternions and angular velocities. This approach naturally results in sub-time-optimal trajectory, as the interpolation will smooth away any high-frequency behavior that may occur between sampled points of the TOPPQuad trajectory. Nevertheless, the resulting trajectories demonstrate good trackability and a run-time improvement over their minimum snap counterparts.

	K_p	K_d	K_R	K_ω
x	8	5.5	2812	128
y	8	5.5	2812	128
z	19	8.7	163	73

TABLE III
SE(3) Controller Gains

The following four trajectories are shown: a line (Fig. 8), an L-shaped curve (Fig. 9), an X-shaped curve in 3D space (Fig. 10), and a Lissajous curve (Fig. 1, 11). For the first three trajectories, we compare the trackability of two minimum snap (MS) trajectories, with heuristic velocities of $1m/s$ and $2m/s$, against a TOPPQuad trajectory with a maximum velocity constraint of $2m/s$. We show a consistent decrease

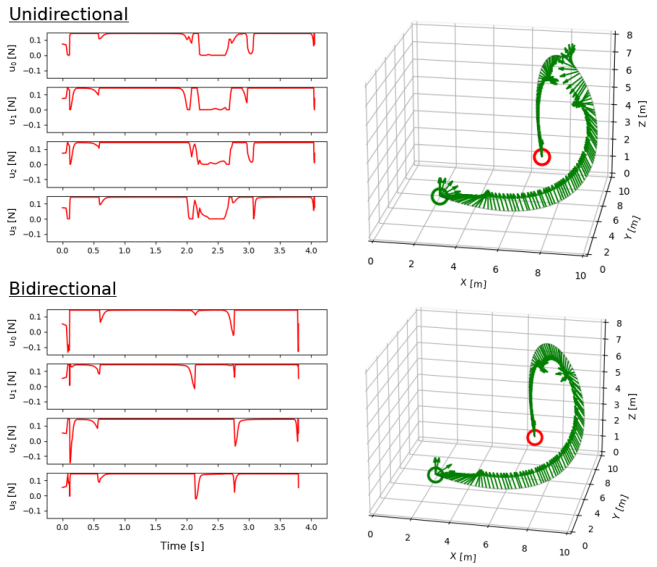

Fig. 6. A sample trajectory computed with TOPPQuad for both a unidirectional (**top**) and bidirectional (**bottom**) quadrotor, and the respective motor speeds. The green vectors represent the quadrotor's $\mathbf{R}\mathbf{e}_3$ vector at each point along the trajectory.

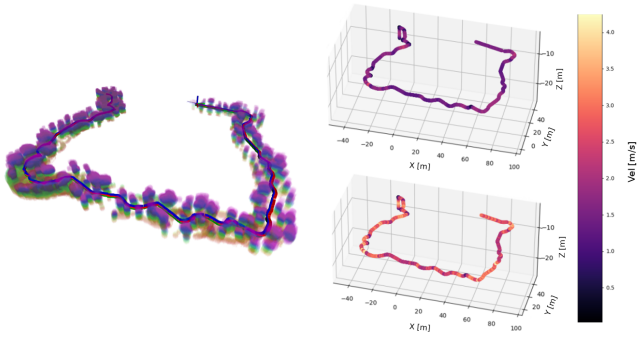

Fig. 7. Obstacle-free path through a forest (**left**) . We show the quadrotor speed along the original path (**top right**) and refined TOPPQuad path (**Top left**). Maximum velocity is bounded at $v = 5m/s$.

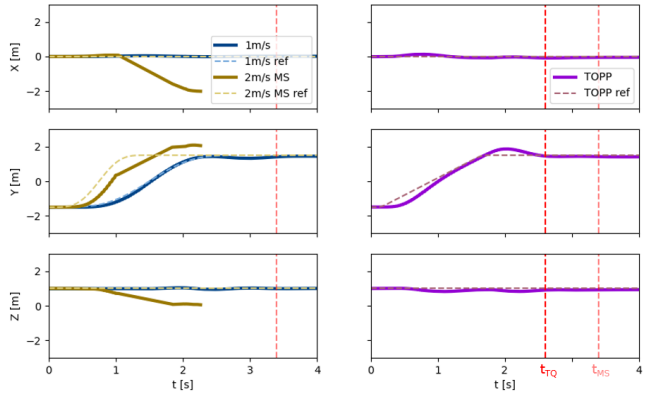

Fig. 8. **Line**. *Left:* Plots of the tracked (blue, purple) and reference (orange, yellow) $1m/s$ MS and $2m/s$ MS trajectories in X, Y, Z. *Right:* Plots of tracked (purple) and reference (blue) TOPPQuad trajectory. With the controller in the loop, the TOPPQuad trajectory was executed in $t_{TQ} = 2.6s$, the $1m/s$ MS trajectory executed in $t_{MS} = 3.4s$, and the $2m/s$ MS trajectory crashed.

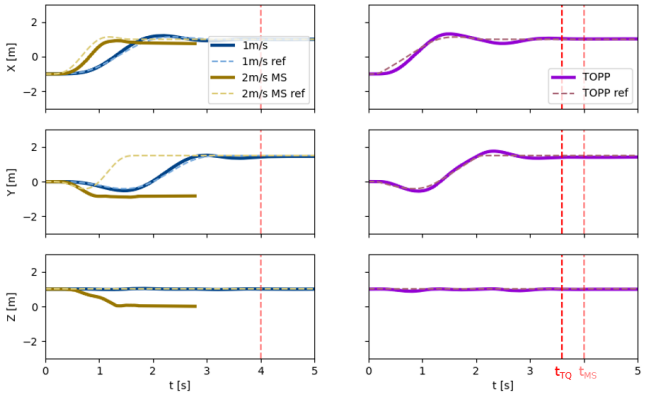

Fig. 9. **L-Curve**. *Left:* Plots of the tracked (blue, purple) and reference (orange, yellow) $1m/s$ MS and $2m/s$ MS trajectories in X, Y, Z. *Right:* Plots of tracked (purple) and reference (blue) TOPPQuad trajectory. With the controller in the loop, the TOPPQuad trajectory was executed in $t_{TQ} = 3.6s$, the $1m/s$ MS trajectory executed in $t_{MS} = 4s$, and the $2m/s$ MS trajectory crashed.

in execution time between the $1m/s$ MS and TOPPQuad trajectories and improved trackability between the $2m/s$ MS (which all crashed) and TOPPQuad trajectories. In addition, we compare our planner with a more demanding Lissajous trajectory. We show that we are able to track the TOPPQuad-refined Lissajous curve while the original trajectory fails. In all experiments, we consider a trajectory executed once the state of the robot settles to the terminal state of the trajectory.

VII. Conclusion

We present TOPPQuad, an optimization algorithm for finding time-optimal quadrotor trajectories. We explicitly enforce state and input constraints such as motor thrusts instead of relying on convex relaxations, which we demonstrate are not always sufficient. The key to our algorithm is an optimization of the total trajectory time given the full dynamics of the quadrotor. We show the ability to refine trajectories that come from a variety of commonly used differential flatness-based planners, expand motor thrust bounds to automatically generate trajectories for bidirectional quadrotors, and demonstrate our planner's performance on real world data. This offers a complimentary approach to existing time-optimal quadrotor planning methods.

However, our algorithm still has room for improvement time. Firstly, the computation time takes on the order of half a minute. Secondly, by virtue of solving a non-convex problem, our planner is not complete. However, fall-backs such as α-scaling offer completeness, although without guarantees of time-optimality. Finally, our model does not account for drag or motor dynamics. Future work will focus on examining implementations for improving algorithm computation time and will focus on a higher fidelity model by incorporating representations of air resistance and motor dynamics into the optimization framework.

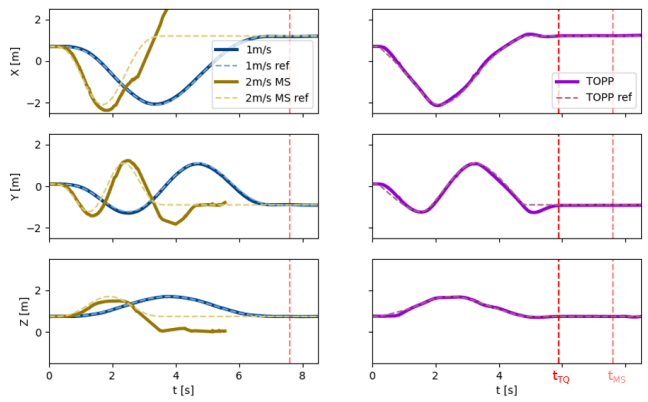

Fig. 10. **X-Curve**. *Left:* Plots of the tracked (blue, purple) and reference (orange, yellow) $1m/s$ MS and $2m/s$ MS trajectories in X, Y, Z. *Right:* Plots of tracked (purple) and reference (blue) TOPPQuad trajectory. With the controller in the loop, the TOPPQuad trajectory was executed in $t_{TQ} = 5.9s$, the $1m/s$ MS trajectory executed in $t_{MS} = 7.6s$, and the $2m/s$ MS trajectory crashed.

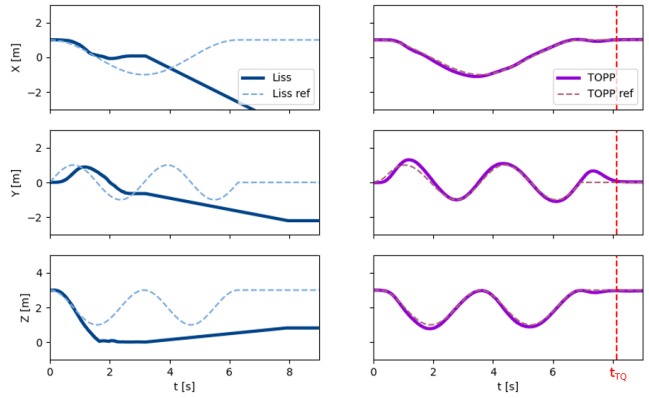

Fig. 11. **Lissajous Curve**. *Left:* Plots of the tracked (blue) and reference (orange) Lissajous trajectory in X, Y, Z. *Right:* Plots of tracked (purple) and reference (blue) TOPPQuad trajectory. With the controller in the loop, the TOPPQuad trajectory was executed in $t_{TQ} = 9.1s$, while the Lissajous trajectory crashed.

References

[1] S. M. LaValle, "Planning algorithms." Cambridge university press, 2006, ch. 14, pp. "841–855".

[2] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 2520–2525.

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

[4] M. Hehn and R. D'Andrea, "Real-time trajectory generation for quadrocopters," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 877–892, 2015.

[5] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Robotics Research: The 16th International Symposium ISRR*. Springer, 2016, pp. 649–666.

[6] S. Liu, K. Mohta, N. Atanasov, and V. Kumar, "Search-based motion planning for aggressive flight in se (3)," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2439–2446, 2018.

[7] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, 2022.

[8] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se(3)," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 5420–5425.

[9] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "Time-optimal control of robotic manipulators along specified paths," *The international journal of robotics research*, vol. 4, no. 3, pp. 3–17, 1985.

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

[11] T. Lipp and S. Boyd, "Minimum-time speed optimisation over a fixed path," *International Journal of Control*, vol. 87, no. 6, pp. 1297–1311, 2014.

[12] H. Nguyen and Q.-C. Pham, "Time-optimal path parameterization of rigid-body motions: Applications to spacecraft reorientation," *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 7, pp. 1667–1671, 2016.

[13] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, 2018.

[14] G. Ryou, E. Tal, and S. Karaman, "Multi-fidelity black-box optimization for time-optimal quadrotor maneuvers," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1352–1369, 2021.

[15] P. Foehn, A. Romero, and D. Scaramuzza, "Time-optimal planning for quadrotor waypoint flight," *Science Robotics*, vol. 6, no. 56, p. eabh1221, 2021.

[16] L. Consolini, M. Locatelli, A. Minari, and A. Piazzi, "An optimal complexity algorithm for minimum-time velocity planning," *Systems & Control Letters*, vol. 103, pp. 50–57, 2017.

[17] I. Spasojevic, V. Murali, and S. Karaman, "Asymptotic optimality of a time optimal path parametrization algorithm," *IEEE Control Systems Letters*, vol. 3, no. 4, pp. 835–840, 2019.

[18] W. Giernacki, M. Skwierczyński, W. Witwicki, P. Wroński, and P. Kozierski, "Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering," in *2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR)*, 2017, pp. 37–42.

[19] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[20] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[21] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[22] K. Mao, J. Welde, M. A. Hsieh, and V. Kumar, "Trajectory planning for the bidirectional quadrotor as a differentially flat hybrid system," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1242–1248.

[23] L. Jarin-Lipschitz, X. Liu, Y. Tao, and V. Kumar, "Experiments in adaptive replanning for fast autonomous flight in forests," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 8185–8191.

[24] X. Liu, G. V. Nardari, F. C. Ojeda, Y. Tao, A. Zhou, T. Donnelly, C. Qu, S. W. Chen, R. A. F. Romero, C. J. Taylor, and V. Kumar, "Large-scale autonomous flight with real-time semantic slam under dense forest canopy," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5512–5519, 2022.

[25] M.-J. Kim, M.-S. Kim, and S. Y. Shin, "A general construction scheme for unit quaternion curves with simple high order derivatives," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 1995, pp. 369–376.