Abstract—In this paper, we present a framework to do optimal time allocation for quadrotor trajectory generation. Using this method, we can generate minimum-time piecewise polynomial trajectories for quadrotor flights. We decouple the quadrotor trajectory generation problem into two folds. Firstly we generate a smooth and safe curve which is parameterized by a virtual variable. This curve named spatial trajectory is independent of time and has fixed spatial properties. Then a mapping function which decides how the quadrotor moves along the spatial trajectory respecting kinodynamic limits is found by minimizing total trajectory time. The mapping function maps the virtual variable to time is named temporal trajectory. We formulate the minimum-time temporal trajectory generation problem as a convex program which can be efficiently solved. We show that the proposed method can incorporate with various types of previous trajectory generation method to obtain the optimal time allocation. The proposed method is integrated into a customized light-weight quadrotor platform and is validated by presenting autonomous flights in indoor and outdoor environments. We release our code for time optimization as an open-source ros-package.

I. INTRODUCTION

In recent years, micro aerial vehicles (MAVs), especially quadrotors, have drawn increasing attention on various applications. Thanks to their mobility, agility, and flexibility, quadrotors can fly rapidly and safely through complex environments while avoiding collisions. One of the most crucial issues for quadrotor flights is the motion/path planning, which coordinates the quadrotor to navigate from a start point to a target location in cluttered environments. Beyond the safety requirement, which has been satisfied in many approaches [1], [6], time optimality and kinodynamic feasibility are also significant issues we consider. On the premise of satisfying kinodynamic limits, for most applications of quadrotors, optimality in time means efficiency in operation.

For quadrotor motion planning and trajectory generation, the piecewise polynomial-based trajectory has been widely adopted since [3] and [1], because of its superior representative capability and concise formulation. Although variants of piecewise polynomial trajectory have been proposed, time allocation is still a bottleneck for it. We can use the scenario of a car/quadrotor racing in a sharp turn for illustration. No matter how high is the speed before entering the sharp turn, the acceleration of the vehicle has to be bounded while turning. In this case, a poor time allocation would take a significantly longer time for passing the sharp turn. In this paper, we present a framework to generate minimum time trajectories under the constraints of physical limits. The proposed framework is decoupled into two stages. Firstly we generate trajectories in the spatial aspect. Instead of parameterizing the trajectory directly by time, we generate time independent trajectories in a virtual domain, with fixed geometric distributions. Secondly, we bridge the spatial trajectory to the temporal information. The relations between the virtual variable to time variable is found by minimizing the total flight time globally considering the dynamical limits. Therefore vehicles with our proposed time allocation method can pass the sharp turn with minimal time; related tests are given in Sect. V-C.2.

This work is motivated by the observation that for a robotics trajectory generation application, most of the cases the geometrical and temporal information are not necessarily coupled. For safety consideration, the trajectory should be wrapped to avoid obstacles [4] [5] or be bounded within free space [6] [7]. To this end, geometrical properties of the trajectory are the only concern. On the other side, for dynamical feasibility consideration, temporal information of the trajectory, such as velocity and acceleration of the quadrotor should be bounded within the kinodynamic limits. In many works, although the kinodynamic is constrained at the same time with the generation of the geometrical trajectory [7] [8], it is highly dependent on the time allocation and is often too conservative to fully utilize the actuators for high-speed navigations. In this paper, we propose a method to get the optimal time allocation for quadrotor trajectory

All authors are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong. China. fgaoaa@ust.hk, wwuar@connect.ust.hk, jpanai@connc.ust.hk, uv.boyuzhou@gmail.com, eeshaojie@ust.hk
generation, and we summarize our contributions as:

1) Decouple the trajectory generation problem into spatial and temporal sub-problems. Moreover, utilize our previous methods [9], [7] to generate trajectories with spatial safety guarantee in a virtual domain.

2) Formulate the temporal trajectory optimization problem as a convex SOCP (Second Order Conic Program) based on [10] and extend the approach to piecewise trajectory formulation for a jerk-controlled quadrotor.

3) Integrate the proposed method into an autonomous quadrotor platform and present real-time onboard demonstrations. Fast autonomous flights in indoor and outdoor environments are presented. The source code will be released as an add-on to any other piecewise polynomial trajectory generation methods.

We discuss related literature in Sect. II. Our safe spatial trajectory generation and minimal-time temporal trajectory methods are detailed in Sect. III and Sect. IV, respectively. In Sect. V, benchmark results as well as indoor and outdoor experimental results which show fast autonomous flights are presented. The paper is concluded in Sect. VI.

II. RELATED WORK

For safe trajectory generation, many methods were proposed, ranging from soft constrained formulation to hard constrained formulation. Gradient-based methods like CHOMP [4] formulates the trajectory optimization problem as a nonlinear optimization over the penalty of the safety and the smoothness. In [5] and [9] the gradient-based methods are applied on piecewise polynomial trajectories. Mellinger et al. [3] pioneered a minimum-snap trajectory generation algorithm, where piecewise polynomial functions represent the trajectory, and the trajectory generation problem is formulated as a quadratic program (QP) problem. Closed form solution for minimum snap trajectory generation is obtained in [1], where the safety of the trajectory is achieved by iteratively adding intermediate waypoints in a discrete path found by RRT*. Our previous work [7] carves a flight corridor consisting of convex regions against the environments and confines the trajectory within the corridor by utilizing the convex hull property of Bernstein polynomial basis.

The time parameterization or so-called time allocation dominates the progress of the trajectory in the time domain, and is regarded as one of the main reasons leads to sub-optimality [1]. One way to achieve reasonable time allocation is by iteratively refining allocated times using gradient descent [1]. However, iteratively regenerating the trajectory prevents this method to be used in real-time, especially when the trajectory generation is costly. In our previous work [11], we allocate times for each piece of the trajectory according to the Euclidean distance between the segmentation points and rely on the overlapping regions of the flight corridor to adjust the time allocation. Another standard way is using some heuristic [12] [13] to allocate the time. All of the above methods have no guarantee on the optimality of the time.

Minimum time trajectory or so-called time-optimal trajectory generation aims at fully utilizing the ability of actuators of mobile robots to travel as fast as possible without violating kinodynamic limits. Methods can be divided into direct methods [14] which directly generate spatial-temporal optimal trajectory and indirect methods [15] which generate the trajectory independent of time firstly and then find the relationship between the trajectory and time. The proposed approach in this paper is an indirect method. For direct methods, a recent result is presented in [14], where the trajectory generation problem is formulated as a minimum-time optimal control based graph search. The linear model of the quadrotor is established, and a graph consists of minimum time controlled motion primitives is formed. This method considers the time optimality when searching a safe and dynamical feasible trajectory towards the goal. However, the complexity of the search grows exponentially as the graph size increases. Also, linearizing the quadrotor model at hover point can lead to high error when flying at high acceleration. In [15], the geometric shape of the trajectory is fixed at a virtual domain firstly, which is the same as our method. Then the mapping function from the virtual parameter to time is optimized by nonlinear optimization. Compared to our approach, this method needs a kinodynamic feasible initial solution, and the convergence rate is slow. Also, it cannot guarantee the global optimality. The approach proposed in [16] also fixes the geometric trajectory and finds the mapping function by iteratively adds mapping points along the spatial trajectory to squeeze the kinodynamic feasibility. However, this method also needs numerous numerical iterations, making it unsuitable for onboard usage. Our proposed method in this paper is based on an indirect method used for robotic-arm path tracking [10]. For finding the optimal mapping function from the virtual domain to the time domain, the virtual variable is discretized. This discretization is essential since it makes the problem easily tractable in a convex formulation. In this paper, we develop a framework based on [10] to get the optimal time allocation of piecewise polynomial trajectory for quadrotors and apply the proposed method on applications of fast quadrotor flights.

III. SAFE SPATIAL TRAJECTORY GENERATION

A. Corridor-Based Method

As presented in our previous work [7], we firstly extract the free space in environments to form a flight corridor consisting of convex shapes and then constrain piecewise Bézier curve entirely within these convex shapes. The Bézier curve consists of Bernstein polynomial basis with respect to a virtual parameter $s$ is written as:

$$ B_j(s) = c_0^j b_0^n(s) + \ldots + c_j^j b_j^n(s) = \sum_{i=0}^{n} c_j^i b_i^n(s), \quad (1) $$

where $[c_0^j, c_1^j, \ldots, c_j^j]$ is the set of control points of the $j^{th}$ piece of the Bézier curve. $b(n)$ is the Bernstein polynomial basis, $n$ is the degree. Suppose we denote the piecewise Bézier curve in one dimension $\mu$ out of $x, y, z$ as $f_\mu(s)$. In our applications, since we only want to fix the geometrical shape of the trajectory in $s$ domain, we simply set the duration of $s$
in each piece of the trajectory as 1. For generating a minimal jerk curve, the objective is written as:

\[ J = \sum_{\mu} \int_{0}^{s(T)} \left( \frac{d^3 f_{\mu}(s)}{ds^3} \right)^2 ds, \]  

(2)

where \(T\) is the total time of the trajectory and \(s(T)\) is the corresponding \(s\) value. Note here although the minimum jerk objective can efficiently smooth the trajectory, it has a totally different physical meaning compared to the one in time domain. In practice, the minimal 3rd order objective in \(s\) domain often results an un-necessary longer trajectory since we set \(s \in [0, 1]\) for all pieces of the trajectory ignoring the length of the segment. To overcome this disadvantage, we add a minimal arc length regularization term into this objective. The arc length of a given curve \(f_{\mu}(s)\) is:

\[ L = \int_{0}^{s(T)} \sqrt{f_x'(s)^2 + f_y'(s)^2 + f_z'(s)^2} ds, \]  

(3)

which is a line integral and can not be written in a closed form. We instead use \(L^2\) as the regularization term and the objective function is \(w_s \cdot J + w_l \cdot L^2\), where \(w_s\) and \(w_l\) are relative weights for minimizing jerk in \(s\) domain and length of the curve. The detailed objective is in a quadratic formation and is omitted here for brevity. Utilizing the convex hull property, the entire curve can be bounded within the flight corridor by adding safety constraints on all control points. The trajectory generation problem is a quadratic program (QP) which can be solved in polynomial time. For details about the formulation and the application we refer readers to our previous publication [7].

B. Gradient-Based Method

For another category of the trajectory generation, the trajectory is optimized as a nonlinear optimization which takes into account the smoothness, safety and dynamical feasibility of the trajectory. By using the spatial-temporal decoupling strategy, the dynamical feasibility terms can be dropped. Thus the objective function can be written as \(J = w_s \cdot J + w_l \cdot C\), where \(J\) is exactly the same as Equ.3. The collision cost \(C\) is the line integral of the arc length along the trajectory and can be discretized for numerical calculation. The collision cost \(C\) in \(s\) domain is:

\[ C = \sum_{k=0}^{N} c(p(s_k)) \int_{s_k}^{s_{k+1}} \sqrt{f_x'(s)^2 + f_y'(s)^2 + f_z'(s)^2} ds \]  

(4)

\[ = \sum_{k=0}^{N} c(p(s_k)) \int_{s_k}^{s_{k+1}} \sqrt{f_x(s_k)^2 + f_y(s_k)^2 + f_z(s_k)^2} ds, \]

where \(c(p)\) is the penalty function and is designed as an exponential function of distance in our previous work [9].

The trajectory generation problem is in a nonlinear optimization formulation. By using numerical optimization techniques such as Levenberg-Marquardt method, the nonlinear optimization results in a smooth and collision-free trajectory. Note here that the dropping of dynamical feasibility terms significantly reduces computation time and improves the convergence rate, which is detailed in V-A.

IV. Minimum-Time Temporal Trajectory Generation

A. Representing the Temporal Trajectory

For a piecewise spatial trajectory defined in virtual domain \(s\), the geometric properties such as the \(x, y, z\) positions and their derivatives are fixed. However, the kinodynamic properties of the quadrotor such as the velocity and acceleration are decided by a piecewise mapping function which maps each \(s\) value to a time value \(t\). In this paper, we name the trajectory of velocity and acceleration in the time domain as temporal trajectory. Our motivation in this section is to design a mapping function which can generate motion as fast as possible without violating the kinodynamic limits of the quadrotor. Suppose we have already solved the spatial trajectory \(f_{\mu}(s)\) and denote the relation between the variable \(s\) of the trajectory to time \(t\) as a function \(s(t)\). Obviously, \(s(t)\) is also a piecewise function which can be written as follows:

\[ s(t) = \begin{cases} s_0(t), & s_1(0) = 0, s_0(T_0) = 1, s_0 \in [0, 1], \\ s_1(t), & s_1(0) = 0, s_1(T_1) = 1, s_1 \in [0, 1], \\ \vdots & \vdots \\ s_m(t), & s_m(0) = 0, s_m(T_m) = 1, s_m \in [0, 1], \end{cases} \]  

(5)

where \(T_0, T_1, ..., T_m\) are unknown time durations in each piece of the temporal trajectory.

Another obvious observation is that \(s(t)\) must be a monotonically increasing function since physically variable time can only increase. Therefore the following condition holds:

\[ \dot{s}(t) \geq 0. \]  

(6)

By chain rule, we can re-write the temporal trajectory, i.e. the velocity and acceleration of the quadrotor as:

\[ v_{\mu}(t) = f_{\mu}'(s) = f_{\mu}'(s) \cdot \dot{s}, \]  

\[ a_{\mu}(t) = f_{\mu}''(s) = f_{\mu}''(s) + f_{\mu}'(s) \cdot \ddot{s}. \]  

(7)

Here we use the notation \(c = dc/dt, \dot{c} = d^2c/dt^2\) to indicate taking the 1st and 2nd order derivative of a variable \(c\) with respect to time, and use \(c' = dc/ds, c'' = d^2c/ds^2\) for derivatives with respect to \(s\).

B. Piecewise Minimum-Time Optimization

1) Objective: The objective \(J\) of the temporal trajectory generation is to minimize the total time, which is written as:

\[ J = T = \int_{0}^{T} 1 dt. \]  

(8)

With the fact \(s = ds/dt\) and in each piece of the trajectory \(s_i(t) \in [0, 1]\), the time \(T\) can be re-written as:

\[ T = \sum_{i=0}^{m} \int_{0}^{1} \frac{1}{s_i} ds. \]  

(9)

In minimum-time optimal control problem [17], the energy of the control input is penalized in the objective function to trade-off the time optimality and control extremeness. In
our problem, we can also incorporate the regularization on changing rate of \( s \) in our objective:

\[
J = \sum_{i=0}^{m} \int_{0}^{1} \left( \frac{1}{s_i} + \rho \cdot \dot{s}_i^2 \right) ds,
\]

where \( \rho \) is the weighting parameter. Following the direct transcription method in [10], we introduce two additional functions \( a(s) \) and \( b(s) \) which are also piecewise and satisfy:

\[
a(s) = \dot{s}, \tag{11}
\]

\[
b(s) = s^2. \tag{12}
\]

Then the objective is derived as to minimize:

\[
J = \sum_{i=0}^{m} \int_{0}^{1} \left( \frac{1}{\sqrt{b_i(s)}} + \rho \cdot a_i(s)^2 \right) ds, \tag{13}
\]

where the subscript \( i \) of \( a_i(s) \) and \( b_i(s) \) shares the same meaning as in \( s_i \). Several observations obviously hold following the formulation above, including:

\[
b_i(s) \geq 0, \tag{14}
\]

\[
b_i'(s) = 2 \cdot a_i(s). \tag{15}
\]

2) Continuity Constraints: The generated spatial-temporal trajectory must be continuous at several order derivatives to meet the requirements in quadrotor control. According to our experience and the actuator model of the quadrotor, we require the continuity up to acceleration holds between each pieces of the trajectory.

The continuity of position is achieved by the spatial trajectory in Sec. III. For velocity and acceleration, the continuities are enforced by setting equality constraints on the joint point between two consecutive pieces of the temporal trajectory, for the \( i^{th} \) and \( (i+1)^{th} \) piece trajectories, in dimension \( \mu \in \{x, y, z\} \) we have:

\[
f_{i,\mu}'(1) \cdot \sqrt{b_i(1)} = f_{i+1,\mu}'(0) \cdot \sqrt{b_{i+1}(0)}, \tag{16}
\]

\[
\begin{align*}
f_{i,\mu}'(1) \cdot a_i(1) + f_{i,\mu}''(1) \cdot b_i(1) \\
= f_{i+1,\mu}'(0) \cdot a_{i+1}(0) + f_{i+1,\mu}''(0) \cdot b_{i+1}(0)
\end{align*} \tag{17}
\]

3) Kinodynamic Constraints: To ensure the kinodynamic feasibility, we enforce constraints on velocity and acceleration in each piece of the trajectory. Constraints hold in \( s \in [0, 1] \) at \( x, y, z \) dimensions are written as:

\[
- v_{\text{max}} \leq f_{i,\mu}'(s) \cdot \dot{b}(s) \leq v_{\text{max}}, \tag{18}
\]

\[
- a_{\text{max}} \leq f_{i,\mu}'(s) \cdot a(s) + f_{i,\mu}''(s) \cdot b(s) \leq a_{\text{max}}, \tag{19}
\]

where \( v_{\text{max}} \) and \( a_{\text{max}} \) are the kinodynamic limits of the quadrotor decided by the actuator models.

4) Boundary Constraints: In practice, the trajectory is generated from the initial state of the quadrotor to a final state (normally but not necessarily zero state). For the temporal trajectory, the boundary condition is to meet the initial velocity and acceleration \( a^0, v^0 \), and the terminal velocity and acceleration \( a^f, v^f \). The constraints are:

\[
\begin{align*}
f_{0,\mu}'(0) \cdot \sqrt{b_0(0)} &= v_{\mu}^0, \tag{20} \\
f_{m,\mu}'(1) \cdot \sqrt{b_m(1)} &= v_{\mu}^f, \tag{21} \\
f_{0,\mu}'(0) \cdot a_0(0) + f_{0,\mu}''(0) \cdot b_0(0) &= a_{\mu}^0, \tag{22} \\
f_{m,\mu}'(1) \cdot a_m(1) + f_{m,\mu}''(1) \cdot b_m(1) &= a_{\mu}^f. \tag{23}
\end{align*}
\]

Note that if the initial and final states are both static, there must exist a solution satisfying the boundary constraints. Otherwise, the optimization program may be infeasible.

C. Convex SOCP Reformulation

To make the above optimization problem easily solvable, we would like to re-formulate the objective and constraints under the disciplined convex formalism. To this end, the key step is to discretize the virtual parameter \( s \) into grids [10] for each piece of the trajectory. Then \( a(s) \) is regarded as piecewise constant in each grid of \( s \) and therefore \( b(s) \) is piecewise linear according to Equ. 15. For each piece of the trajectory, \( s_i \in [0, 1] \) is discretized to \( s_k^{i} \), where \( k = 0, 1, ..., K \), according to a given number \( K \). We have \( s_k^{i} - s_{k-1}^{i} = \Delta s = 1/K \). For each piece, \( a_i(s) \) and \( b_i(s) \) are modeled by introducing discrete \( a_k^i \) and \( b_k^i \), for which \( b_k^i \) is assigned at each \( s_k^i \) and \( a_k^i \) is assigned at the middle of \( s_k^i \) and \( s_{k+1}^i \). Then \( b(s) \) is written as piecewise linear function:

\[
b(s) = b_k^i + \frac{b_{k+1}^i - b_k^i}{\Delta s} \cdot (s - s_k^i). \tag{24}
\]

According to Equs. 14 and 15, we have

\[
b_k^i \geq 0, \tag{25}
\]

\[
b_{k+1}^i - b_k^i = 2 \cdot \Delta s \cdot a_k^i. \tag{26}
\]

The objective function in Equ. 13 is derived as:

\[
J = \sum_{i=0}^{m} \sum_{k=0}^{K-1} \left( \frac{2}{\sqrt{b_k^i} + \sqrt{b_k^i}} + \rho \cdot a_k^i \right) \cdot \Delta s, \tag{27}
\]

which is equivalent to the quadratic function:

\[
\sum_{i=0}^{m} \sum_{k=0}^{K-1} \left( 2 \cdot d_k^i + \rho \cdot a_k^i \right) \cdot \Delta s, \tag{28}
\]

by introducing slack variables \( d_k^i \), plus additional constraints:

\[
\frac{1}{\sqrt{b_{k+1}^i} + \sqrt{b_k^i}} \leq d_k^i, \quad k = 0, ...K - 1, i = 0, ...m. \tag{29}
\]

Equ. 29 are transformed to quadratic forms as

\[
\frac{1}{e_k^i + e_k^i} \leq d_k^i, \quad k = 0, ...K - 1, i = 0, ...m. \tag{30}
\]

\[
e_k^i \leq \sqrt{b_k^i}, \quad k = 0, ...K, i = 0, ...m. \tag{31}
\]

by introducing slack variables \( e_k^i \) in each segment.

Equ. 30 is equivalent to the typical formulation of rotated quadratic cones:

\[
2 \cdot d_k^i \cdot \left( e_k^i + c_k^i \right) \geq \sqrt{2}, \tag{32}
\]
which is shorthanded in
\[ (d^k_t, t^k_{t+1} + c^k, \sqrt{2}) \in Q^3, \]
(33)
And Eq. 31 can be written in the typical formulation of quadratic cones:
\[ (b^k_t + 1)^2 \geq (b^k_t - 1)^2 + (2 \cdot c^k)^2, \]
(34)
and be shorthanded as
\[ (b^k_t + 1, b^k_t - 1, 2c^k) \in Q^3. \]
(35)
Finally, an additional slack variable \( t \) is added to convert the quadratic objective in Eq. 28 to an affine form as
\[
\sum_{i=0}^{m} \sum_{k=0}^{K-1} 2\Delta s \cdot d^k_t + \rho \cdot \Delta s \cdot t, 
\]
with a rotated quadratic cone
\[ 2 \cdot t \cdot 1 \geq \sum_{i=0}^{m} \sum_{k=0}^{K-1} (a^k_t)^2, \]
(37)
\[ i.e. \]
\[ (t, 1, a) \in Q^{2+K(m+1)}, \]
(38)
where \( a = [a^0_1, a^0_2, \ldots, a^{K-1}_m] \) is the vectorized variable consisting of all \( a^k_t \) assigned to each segment.

Up to now, the original objective function in Eq. 13 is converted to an affine function (Eq. 36) with several induced rotated second-order cones (Eqs. 33 and 38) and second-order cones (Eqs. 35). Back to constraints in the original problem, the kinodynamic constraints (Sec. IV-B.3) are written in discrete form as:
\[ -v_{max} \leq f'_{i,\mu}(s_i^k) \cdot \sqrt{b^k_i} \leq v_{max}, k = 0, \ldots, K, \]
(39)
\[ -a_{max} \leq f''_{i,\mu}(s_i^{k+1/2}) \cdot a^{0}_t + f''_{i,\mu}(s_i^{k+1/2}) \cdot b^{k+1/2}_i \leq a_{max}, k = 0, \ldots, K - 1 \]
(40)
where \( s_i^{k+1/2} = (s_i^k + s_i^{k+1})/2 \) and \( b^{k+1/2}_i = (b^k_i + b^{k+1}_i)/2 \) corresponding to \( a^k_t \), for \( i = 0, \ldots, m \). Equation 40 is in affine and Equation 39 can also be re-written as affine:
\[ f''_{i,\mu}(s_i^k) \cdot b^k_i \leq v_{max}^2, \]
(41)
Continuity constraints (Sec. IV-B.2) on velocity are written in discrete form as:
\[ f'_{i,\mu}(s_i^k) \cdot \sqrt{b^k_i} = f'_{i+1,\mu}(s^0_{i+1}) \cdot \sqrt{b^0_{i+1}}, \]
(42)
Since the spatial trajectory is continuous at the joint position, which means \( f'_{i,\mu}(s^k_i) = f'_{i+1,\mu}(s^0_{i+1}) \). Equation 42 is equivalent to
\[ b^k_i = b^0_{i+1}. \]
(43)

Unlike \( b^k_i \) is evaluated exactly at \( s^k_i \), \( a^k_t \) is evaluated at \( (s^k_i + s^k_{i+1})/2 \), so no \( a^k_t \) is assigned exactly at the joint position of each two pieces of the temporal trajectory. Instead, the last element \( a^{K-1}_i \) in the \( i^{th} \) segment is separated by \( \Delta s \) with the first element \( a^0_{i+1} \) in the \((i + 1)^{th} \) segment, as shown in Fig. 2. Thus we set in-equality constraints to bound the transitions of accelerations between every two segments. Besides, although we assume \( a^k_t \) is piecewise constant, we would like to bound the changing rate of acceleration considering the aggressiveness in control. We write the affine in-equality constraints for bounding the transitions of acceleration whether within one segment or across two segments:
\[ -\lambda_a \leq f'_{i,\mu}(s_i^{k+1/2}) \cdot a^k_t + f''_{i,\mu}(s_i^{k+1/2}) \cdot b^{k+1/2}_i \]
\[ f''_{i,\mu}(s_i^{k+3/2}) \cdot a^{k+1}_t - f''_{i,\mu}(s_i^{k+3/2}) \cdot b^{k+3/2}_i \leq \lambda_a, \]
\[ -\lambda_a \leq f''_{i-1,\mu}(s_i^{0+1/2}) \cdot a^0_t + f''_{i-1,\mu}(s_i^{0+1/2}) \cdot b^{0+1/2}_i \]
\[ f''_{i-1,\mu}(s_i^{K-1/2}) \cdot a^{K-1}_t - f''_{i-1,\mu}(s_i^{K-1/2}) \cdot b^{K-1/2}_i \leq \lambda_a, \]
where \( \lambda_a \) is a given parameter to limit the changing rate of acceleration along the temporal trajectory. Note here \( \lambda_a \) is not jerk, since time difference between \( s_i^k \) and \( s_{i+1}^k \) cannot be determined in the optimization.

For boundary constraints, velocity constraints are applied directly as:
\[ f'_{0,\mu}(s^0_0) \cdot \sqrt{b^0_0} = v^0_0, \]
(46)
\[ f'_{m,\mu}(s^0_m) \cdot \sqrt{b^m_0} = v^m_0. \]
(47)
For acceleration’s boundary conditions, we set in-equality constraints for the same reason as in Eqs. 44 and 45.
\[ -\lambda_a \leq f''_{0,\mu}(s_0^{1/2}) \cdot a^0_t + f''_{0,\mu}(s_0^{1/2}) \cdot b^{1/2}_0 - a^0_0 \leq \lambda_a, \]
(48)
\[ -\lambda_a \leq f''_{m,\mu}(s_m^{K-1/2}) \cdot a^{K-1}_t + f''_{m,\mu}(s_m^{K-1/2}) \cdot b^{K-1/2}_m - a^m_m \leq \lambda_a. \]
(49)
Up to now, the objective is written in affine \( (h^T d + \rho \cdot t) \) alongside with second order cones. In-equality and equality constraints are abstracted and written in affine \( (A_{eq} \cdot x = b_{eq}) \), where \( d \) consists of all variables \( d^k_i \) and \( x \) vectorizes all optimized variables \( a^k_t, b^k_i, c^k_i, d^k_i \). We can write the temporal trajectory generation problem as:
\[
\min \ h^T d + \rho \cdot t, \\
\text{s.t.} \ A_{eq} \cdot x = b_{eq}, \ A_{ie} \cdot x \leq b_{ie}, \\
\]
which is a typical convex Second Order Cone Program (SOCP) and can be solved in polynomial time by off-the-shelf convex solver.

V. RESULTS

A. Comparisons with Time-parameterized trajectory

We present the comparisons between the proposed time optimization method in this paper to our previous work [7], in which the kinodynamic limits are enforced based on given time allocations. In the comparison $w_1$ and $w_2$ are set as 1.0 and 2.0. The velocity and acceleration limits are $\pm 2m/s$ and $\pm 1m/s^2$. Results are given in Fig. 3. We can see from the results that our proposed method gives a much shorter time for finishing the trajectory respecting the kinodynamic limits. In the test, the optimal time calculated by the proposed method in this paper is 31.47s and the time based on the heuristic in our previous work is 56.52s, respectively. The generated velocities and accelerations approach the limits as much as possible by using our proposed approach. In our previous method [7], although trajectories generated based on a given time allocation can be entirely constrained within the physical bound, it is almost always no possibility to fully utilize the quadrotor’s actuators to do a full-speed cruise. In our previous work, the constraints on kinodynamic feasibility are met easily when one of the segments of the trajectory has reached its limit. Moreover, if an over-aggressiveness time allocation is given, the trajectory generation problem is infeasible at all. While in this paper we optimize the time allocation globally and the fastest motion concerning the kinodynamical bounds are finally obtained.

Fig. 3. Comparisons of the proposed method against our previous method [7]. Trajectories are generated in a random forest with targets randomly selected as in Figs. 3(a), 3(b). Resulted velocities and accelerations against time are given in Fig. 3(b).

B. Study on Parameter Settings

Here we show the results of optimizing the temporal trajectory with different parameter settings. Two key parameters affecting the performance most are the control weighting coefficients $\rho$ which regularizes $\dot{s}$, and $K$ which decides the resolution of $b(s)$ and $a(s)$.

1) Varying $\rho$: The weighting coefficient $\rho$ of the control energy regularization term balances the expectation of minimal time and aggressiveness of control effort. Results of applying different $\rho$ values on a fixed spatial trajectory with $K = 50$ are given in Figs. 5(a) to 5(d). As is shown in

Decoupling the trajectory generation problem into spatial and temporal layer not only achieves time optimality but also benefits the optimization procedure in the solely spatial aspect. Here we present numerical tests by using our previous gradient-based method [9] to show the differences with/without the spatial-temporal decoupling. Under the proposed framework, the dynamical feasibility cost term is removed from the nonlinear optimization, makes convergence significantly improved. We do 500 random tests in each different obstacle density. As shown in Fig. 4, by separating the temporal information, the objective value converges much faster, and the resulted trajectories are not only smoother but also have higher clearance. We also compare the cost reduction, which is defined as the dropping ratio of the objective from the beginning of the nonlinear optimization, in a given time 50ms. As in Fig. 4(d), by separating the temporal trajectory, the objective converges to much lower values in all obstacle densities quickly, while the time-dependent trajectory converges much slower, especially when the obstacle density is high. The reason is that the removal of the dynamic feasibility cost terms saves much time of evaluation in each iteration, and also reduces the non-convexity of the objective function.

B. Study on Parameter Settings

Here we show the results of optimizing the temporal trajectory with different parameter settings. Two key parameters affecting the performance most are the control weighting coefficients $\rho$ which regularizes $\dot{s}$, and $K$ which decides the resolution of $b(s)$ and $a(s)$.

1) Varying $\rho$: The weighting coefficient $\rho$ of the control energy regularization term balances the expectation of minimal time and aggressiveness of control effort. Results of applying different $\rho$ values on a fixed spatial trajectory with $K = 50$ are given in Figs. 5(a) to 5(d). As is shown in
the result, when \( \rho = 0 \), the resulted temporal trajectory has the most aggressiveness. There are peaks in acceleration and sharp transitions in velocity. As \( \rho \) increases, the acceleration and velocity become smoother, and all sharp transitions are eliminated, as a cost, the total time of the trajectory is longer.

2) Varying \( K \): In our approach, the parameter \( K \) directly decides the number of grids in \( s \in [0, 1] \). Here we present results of optimizing the temporal trajectory with different \( K \) values with \( \rho = 10 \). As in Figs. 5(e) to 5(h), increasing \( K \) improves the optimality of the final solution significantly when \( K \) is small. However, when \( K \) is large, the improvement is not very notable. The number of \( K \) can be viewed as the resolution of the optimal solution. As more grids in the \( s \) domain are divided, more freedom (decision variables) is provided to solve the problem with the cost that the computational complexity increases. According to our experience, the computational time of the SOCP increases linearly with \( K \). Moreover, we suggest to set \( K \) in \( 30 \sim 50 \) for normal applications.

C. On-board Flight Tests

The method proposed in this paper\(^1\) is implemented in C++11 using a general convex solver Mosek\(^2\). The flight experiments are done on a self-developed quadrotor platform which is specially designed for high-speed flights. All processings are done on an onboard dual-core 2.40 GHz Intel i7-5500U processor, which has 8 GB RAM and 64 GB SSD. In the test, several waypoints are sent to the quadrotor, and a spatial trajectory in the virtual domain \( s \) with fixed geometric shape is generated onboard. The spatial trajectory is then be parameterized to time \( t \) by using our proposed time optimization method, and the quadrotor is commanded to track the trajectory. The procedure repeats in several loops. We assume our quadrotor has different kinodynamic limits and optimize the minimal-time temporal trajectory to approach such limits in each loop. In experiments, the kinodynamic limits are raised in each loop.

1) Indoor Test: We conduct indoor flights with the pose feedback by the indoor localization system OptiTrack\(^3\). A figure shows the flight is in Fig. 1(a). In this test, a series of waypoints are sent to the quadrotor to generate a spatial trajectory. Initially, the kinodynamic limits \( v_{\text{max}} \) and \( a_{\text{max}} \) are \( 1 \text{m/s} \) and \( 0.5 \text{m/s}\^2 \) in velocity and acceleration. And \( \lambda_\alpha \) and \( \rho \) are set as 0.05 and 0. \( K \) is set as 30. The quadrotor repeats the tracking of the trajectory. In each loop, \( v_{\text{max}}, a_{\text{max}} \) and \( \lambda_\alpha \) are increased by \( 1 \text{m/s} \), \( 0.5 \text{m/s}\^2 \) and 0.1 until velocity limit reaches \( 6 \text{m/s} \) and acceleration limit reaches \( 6 \text{m/s}\^2 \). The result presents the profiles of velocity and acceleration in the 12th loop of the flight is given in Fig. 6. Average computing time in generating the optimal temporal trajectory is \( 62.41 \text{ms} \).

2) Outdoor Test: We use visual-inertial fusion \(^[18]\) with the minimum sensing suite: one camera and one IMU, for pose feedback in outdoor experiments. The kinodynamic limits are increased by \( 1.5 \text{m/s} \) and \( 2.0 \text{m/s}\^2 \) from the initial value of \( 2.0 \text{m/s} \) and \( 3.0 \text{m/s}\^2 \). Other parameters are the same as indoor tests. We set virtual obstacles in the outdoor experiment to mimic the complex trajectory in fully autonomous flights, as shown in Fig. 7. A snapshot shows the flight is in Fig. 1(b). The result presents the profile of velocity and acceleration in the 3rd loop of the flight is given in Fig. 8. Average computing time in generating the optimal temporal trajectory is \( 108.94 \text{ms} \) in this experiment. We also make an experiment as a response to the sharp turning case in Sect. I. The result is shown in Fig. 9. More details and online visualizations about the indoor and outdoor experiment are given in the attached video.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a method to generate the piecewise minimal-time polynomial trajectory for quadrotors. We decouple the traditional time-dependent trajectory generation problem into spatial and temporal layers and propose a method to generate time-optimal trajectory efficiently. The effectiveness and efficiency of the proposed method are validated in both numerical simulations and onboard experiments. The decomposition and optimization achieve the full usage of actuators for quadrotors to cruise in full-speed. We plan to integrate the proposed method into a

\(^1\)Source code of the proposed method will be released in https://github.com/HKUST-Aerial-Robotics/TimeOptimizer after the publishing of this paper.

\(^2\)https://www.mosek.com

\(^3\)http://optitrack.com/
We are going to investigate the feasibility in the future.

When the initial and final states of the temporary optimization block are not both zero, there may not exist a feasible solution.

The biggest challenge is, whether we can develop a complete autonomous quadrotor with online perception like the vehicle shown in green and yellow arrows.

**Fig. 8.** The result of a sample flight in the outdoor experiment. In the flight, the maximum velocity and acceleration allowed for the quadrotor is set as 5.0 m/s and 7.0 m/s². The generated highest velocity and acceleration in one axis are 5.0 m/s and 7.0 m/s².

**Fig. 9.** The experiment of the sharp turning for the case in Sect. I. The color of the trajectory indicates the magnitude of speed. Red color indicates the highest speed (5m/s) while the green indicates the lowest speed in the flight. The quadrotor is shown in a black model. The velocity and acceleration of the vehicle are shown in green and yellow arrows.

We are going to investigate the feasibility in the future.

**REFERENCES**


