Real-Time Trajectory Generation for Quadrocopters

Markus Hehn, *Member, IEEE*, and Raffaello D'Andrea, *Fellow, IEEE*

***Abstract*—This paper presents a trajectory generation algorithm that efficiently computes high-performance flight trajectories that are capable of moving a quadrocopter from a large class of initial states to a given target point that will be reached at rest. The approach consists of planning separate trajectories in each of the three translational degrees of freedom, and ensuring feasibility by deriving decoupled constraints for each degree of freedom through approximations that preserve feasibility. The presented algorithm can compute a feasible trajectory within tens of microseconds on a laptop computer; remaining computation time can be used to iteratively improve the trajectory. By replanning the trajectory at a high rate, the trajectory generator can be used as an implicit feedback law similar to model predictive control. The solutions generated by the algorithm are analyzed by comparing them with time-optimal motions, and experimental results validate the approach.**

***Index Terms*—Motion control, optimal control, quadrocopter, trajectory generation, unmanned aerial vehicles.**

I. Introduction

QUADROCOPTERS are popular aerial robotic platforms for applications in which the ability to hover and move freely in a three-dimensional (3-D) space is important [1]. Trajectory generation remains a problem, however, as flight paths that are feasible under the complex dynamic and input constraints of the vehicles must be computed.

A. Goal and Motivation

In well-known controlled static environments, quadrocopters can be flown using preplanned flight paths and feedback control to track these flight paths (see, for example, [2] and the references therein). Many of the potential applications for aerial robots (such as inspection tasks, disaster coordination, and journalism), however, do not offer such environments: the task may change dynamically (e.g., a target point might be continuously updated based on incoming information); there may be significant disturbances in the actual flight path (e.g., a large wind gust might push the vehicle too far off course for the feedback control to recover efficiently); and knowledge of the environment may be inaccurate (e.g., the existence, position, size, and shape of obstacles may only become available midflight). When dynamic changes such as these are encountered, the original preplanned flight path may become suboptimal or even infeasible and, therefore, impossible to execute.

If a system is to operate in dynamic environments or execute dynamically changing tasks, it must be able to quickly update the planned flight trajectory according to new information as it becomes available, and it must be able to do this while the vehicle is in motion. Such trajectory generation methods are often referred to as "real time" [3], "online" [4], or "reactive" [5] for their ability to accommodate changing constraints by replanning almost instantaneously. Executing the initial control inputs of the continuously updated trajectory also forms an implicit feedback law that can be used to control the vehicle in a fashion similar to model predictive control (MPC)[6]. The use of a trajectory generation algorithm in real-time settings results in several additional requirements when compared with the generation of preplanned flight trajectories, as follows.

1) *Large range of initial conditions:* Preplanning allows boundary conditions to be carefully designed, e.g., by limiting the planning to trajectories that start and end with the vehicle at rest. If trajectories are replanned dynamically, it is necessary to account for the non-rest initial state of the vehicle even if a disturbance has caused it to significantly deviate from its planned flight path.
2) *Computational complexity:* Updating the planned trajectory at a high frequency and with little delay helps to improve reaction time, and it follows that the computation time of the trajectory generation algorithm should be as short as possible. With typical quadrotor position control loops running at rates on the order of 50–100 Hz [7], [8], computation times of a few milliseconds are desirable. In many scenarios, the use of higher level path planning algorithms involves evaluating large numbers of potential trajectories (for example, in rapidly exploring random tree or probabilistic road map algorithms, see e.g. [9]), also making short computation times desirable.

B. Related Work

The problem of quadrocopter trajectory generation has received significant attention in recent years, and a number of algorithms have been presented. Broadly speaking, a possible categorization of the algorithms is as follows:

A first group of algorithms can be considered as primarily geometric. The trajectory generation process consists of first generating a path in space from a class of path primitives, and thereafter parameterizing the generated path in time such that the dynamic constraints of the quadrocopter are enforced. Examples of such algorithms have been presented using path primitives such as lines [10], polynomials [11], or splines [12].

Manuscript received February 11, 2014; revised November 1, 2014; accepted May 9, 2015. This paper was recommended for publication by Associate Editor T. Hamel and Editor T. Murphey upon evaluation of the reviewers' comments. This work was supported by and builds upon prior contributions by past and present collaborators of the ETH Zurich Flying Machine Arena Project and the Swiss National Science Foundation. A list of collaborators is available online at http://flyingmachinearena.org. This work was supported in part by Swiss National Science Foundation.

The authors are with the Institute for Dynamic Systems and Control, ETH Zurich, 8092 Zürich, Switzerland (e-mail: hehnm@alumni.ethz.ch; rdandrea@ethz.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2015.2432611

A second group of algorithms is based on the design of trajectories that minimize a derivative of the position trajectory (or combinations thereof). Because these derivatives can be related to the control input constraints of the quadrocopter through its differential flatness property, the feasibility of the trajectory depends on these derivatives. Examples of such methods include minimum snap trajectory generation [13] and the minimization of a weighted sum of derivatives [14]. More real-time-focused implementations are based on MPC methods with learnt linear dynamics [15] or linear dynamics based on a decoupling of the system [16]. An algorithm that provides particularly low computational complexity has been presented for the case where it is sufficient to consider constraints and verify the feasibility of the trajectory *a posteriori* [17].

A third group consists of algorithms that numerically solve an optimal control problem that directly considers the nonlinear dynamics of the quadrocopter. Optimal solutions are then found through the application of well-established optimal control methods such as Pontryagin's minimum principle [18] or numerical optimal control [19].

C. Contributions

This paper presents and analyzes a trajectory generation algorithm that allows the fast computation of high-performance flight trajectories to move the quadrocopter from a large class of initial states to a given target location at rest. The specific performance criterion considered herein is the duration of the flight maneuver until the vehicle reaches the target.

The algorithm is designed to be computationally efficient in order to provide a provably feasible trajectory quickly enough for its use in real-time settings. The design is further targeted at flight in relatively small spaces, where peak velocities remain sufficiently small that aerodynamic effects (such as drag) do not become dominant. It is also assumed that an underlying body rate control loop is available to accurately track commanded rotational rates and, therefore, allow the rotational rates to be modeled as control inputs.

In order to efficiently generate trajectories under these assumptions, a decoupling approach is used to reformulate the quadrocopter trajectory problem in the computation of time-optimal trajectories for acceleration- and jerk-limited triple integrators. The fact that time-optimal trajectories for input-affine systems are bang singular is well known from the optimal control theory [20], [21]. In the context of trajectory generation for robotic applications, the bang-singular structure is attractive because these trajectories often provide a relatively simple parameterization and the possibility of computing the corresponding parameters efficiently. Similar approaches of bang-singular trajectory generation have been demonstrated for robotic arms with many joints [4] and for generic trajectories, where the derivatives of the position are constrained [22]. A similar decoupling approach with time-optimal bang-singular trajectories has been used for omnidirectional ground robots in the RoboCup competition [23].

Furthermore, the algorithm provides means to tradeoff computational complexity and the performance of generated trajectories: After a first trajectory has been computed, remaining computation time can be used to iteratively increase the performance (i.e., reduce the duration) of the trajectory. The computational effort of this iterative optimization can be determined *a priori*, but it is also possible to adapt to changing availability of computational resources: The iteration process can be aborted at any time, and the best trajectory generated up until that time can be used as a solution while still maintaining feasibility.

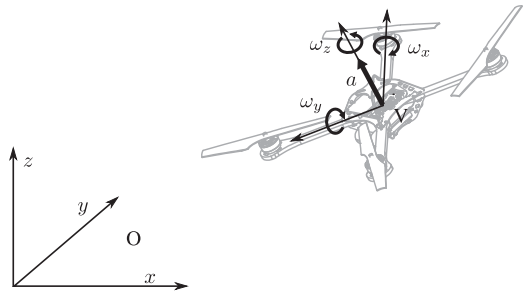

Fig. 1. Inertial coordinate system O, the body-fixed coordinate system V, and the control inputs of the quadrocopter. The rotational rates ω_x, ω_y, and ω_z are tracked by an on-board controller using gyroscope feedback. The collective mass-normalized thrust a acts along the third body axis of the vehicle.

Preliminary results of the decoupling approach were presented in a previous conference paper [24]. This paper extends these results by: 1) introducing an iterative method for optimally choosing the decoupling parameters, 2) a rigorous way to account for a large class of initial conditions without violating the dynamic constraints of the vehicle, and 3) an evaluation of the computational requirements and performance of the algorithm.

D. Outline

The remainder of this paper is organized as follows: Section II introduces the model of the quadrocopter dynamics that is used throughout this paper. The trajectory generation problem is then formally presented in Section III. Section IV derives the feasibility conditions from the dynamic model and the decoupling is used to simplify the trajectory generation problem while guaranteeing feasibility. Section V introduces the basic proposed trajectory generation algorithm that satisfies the trajectory generation problem. Section VI presents the iterative optimization of the decoupling parameters to improve performance when more computation time is available. The use of the trajectory generation algorithm as an implicit feedback law is discussed in Section VII. Section VIII presents results on the computational performance of the method and characterizes the properties of planned trajectories in comparison with time-optimal trajectories. An experimental validation of the method is presented in Section IX, and the conclusion as well as an outlook are given in Section X.

II. Dynamic Model

The quadrocopter is described by six degrees of freedom: The translational position $(x,\ y,\ z)$ is measured in the inertial coordinate system O, as shown in Fig. 1. The vehicle attitude V is defined by the rotation matrix ${}^{\mathrm{O}}_{\mathrm{V}}R$. The rotation matrix is

defined such that when multiplying it with a vector v in the vehicle coordinate system V, the same vector, described in the inertial coordinate system O, is obtained

$$^{\mathrm{O}}v = {}^{\mathrm{O}}_{\mathrm{V}}R\,{}^{\mathrm{V}}v. \tag{1}$$

A. Control Inputs

The control inputs of the quadrocopter are the rotational rates about the vehicle body axes, ω_x, ω_y, and ω_z, and the mass-normalized collective thrust a, as shown in Fig. 1.

It is assumed that high-bandwidth controllers on the vehicle track the rotational rates (this is usually achieved using feedback from gyroscopes). Typical quadrocopters have very low rotational inertia, and can produce high torques due to the outward mounting of the propellers [25]. This results in very high achievable rotational accelerations $\dot{\omega}_x$ and $\dot{\omega}_y$; the rotational rate control loops can, therefore, achieve very fast response times to changes in the commanded rotational rates.[1] In the following, it is, therefore, assumed that the vehicle body rates are directly controllable. Rotational accelerations $\dot{\omega}_z$ are created by causing a drag difference between propellers rotating in opposite directions, and achievable values are typically significantly lower. However, it will be shown that ω_z does not greatly influence the translational dynamics of the vehicle in this trajectory generation problem.

The four control inputs $(\omega_x, \omega_y, \omega_z, a)$ are subject to saturation. The magnitude of the vehicle body rates are limited (such limitations can be caused, for example, by the range of the gyroscopes used for feedback, or performance limitations of the body rate tracking controllers) as

$$|\omega_x| \leq \omega_{xy,\max} \tag{2}$$

$$|\omega_y| \leq \omega_{xy,\max} \tag{3}$$

$$|\omega_z| \leq \omega_{z,\max}\,. \tag{4}$$

The collective thrust is limited by a minimum and a maximum thrust

$$a_{\min} \leq a \leq a_{\max} \tag{5}$$

where $a_{\min} > 0$. This limitation is motivated by the fact that typical quadrocopters have fixed-pitch propellers, the direction of rotation of which cannot be reversed during flight.

B. Equations of Motion

The translational acceleration of the vehicle is dictated by the attitude of the vehicle and the total thrust produced by the four propellers a. The translational acceleration in the inertial frame is

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = {}^{\mathrm{O}}_{\mathrm{V}}R \begin{bmatrix} 0 \\ 0 \\ a \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -\mathrm{g} \end{bmatrix}. \tag{6}$$

[1]Experiments by the authors with vehicles of approximately 0.5 kg have shown rotational accelerations on the order of 200 rad s^{-2} and body rate tracking time constants on the order of 20 ms.

The change of vehicle attitude is related to the rotational control inputs through the kinematic relationship [26]

$${}^{\mathrm{O}}_{\mathrm{V}}\dot{R} = {}^{\mathrm{O}}_{\mathrm{V}}R \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{7}$$

Note that the above equations of motion neglect well-known aerodynamic effects that act on quadrocopters, such as rotor damping [27] and drag-like effects [28]. While such effects can be significant at high flight speeds or when significant wind affects flight, the algorithm presented herein is mainly intended for navigating relatively small spaces where flight speeds are limited and external wind effects are relatively small. Omitting these effects eliminates the need to quantify them over a large range of flight regimes and also greatly simplifies the problem. This motivates their omission in this paper. These simplifying assumptions can be considered similar to model reduction approaches for MPC (see e.g., [29]) in that the model accuracy is traded off for computational performance.

III. Problem Statement

The trajectory generation problem considered herein can be described as follows: Let the quadrocopter have a given initial state (consisting of initial vehicle position x_0, y_0, z_0, velocity $\dot{x}_0, \dot{y}_0, \dot{z}_0$, and attitude ${}^{\mathrm{O}}_{\mathrm{V}}R_0$). Generate a trajectory that satisfies the initial state constraints and drives the vehicle to a target point (taken to be, without loss of generality, the origin), with the vehicle reaching the target point at rest. The generated trajectory must be feasible with respect to the quadrotor dynamics (6) and (7) and the control input constraints (2)–(5). It should also reach the target point as quickly as possible, and its generation should be computationally inexpensive.

IV. Feasibility Conditions and Decoupling

This section describes how the trajectory generation problem statement from the previous section is converted into three separate more tractable trajectory generation problems.

The derivation of the control inputs for an arbitrary vehicle trajectory $(x(t),\ y(t),\ z(t))$ is presented first (see Section IV-A). Using this derivation, the general conditions under which trajectories are feasible are then obtained (see Section IV-B). From these general trajectory constraints, a set of decoupled trajectory constraints are then generated through approximations that preserve feasibility. These decoupled constraints differ from the general trajectory constraints in which the motion constraints of one degree of freedom (e.g., $x(t)$) do not depend on the other two degrees of freedom (e.g., $y(t)$ and $z(t)$).

A. Control Inputs for a Given Trajectory

Let $(x(t),\ y(t),\ z(t))$ denote a candidate vehicle trajectory. For notational convenience, the time dependence will hereafter be omitted unless specific times are considered. Taking the second derivative of the trajectory and combining it with the translational equation of motion (6), the vector f is defined to represent the total mass-normalized forces required by the quadrocopter

to follow the trajectory

$$f := \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \mathrm{g} \end{bmatrix} = {}^{\mathrm{o}}_{\mathrm{v}}R \begin{bmatrix} 0 \\ 0 \\ a \end{bmatrix}. \tag{8}$$

Using the two-norm (denoted by $\|\cdot\|$), the thrust a required to follow the trajectory can be calculated by using the property that a pure rotation matrix does not change the two-norm of a vector [30]

$$\|f\| = \left\| {}^{\mathrm{o}}_{\mathrm{v}}R \begin{bmatrix} 0 \\ 0 \\ a \end{bmatrix} \right\| = a\,. \tag{9}$$

The direction of thrust is the unit vector in the direction of f

$$\bar{f} := \frac{f}{\|f\|} \tag{10}$$

and can be seen to define the third column of the rotation matrix by substituting $\bar{f}$ back into (8)

$${}^{\mathrm{o}}_{\mathrm{v}}R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \bar{f}\,. \tag{11}$$

Taking the derivative of the above equation and combining it with (7) gives two of the vehicle body rates as functions of the current attitude and $\dot{\bar{f}}$ as

$$\begin{bmatrix} \omega_y \\ -\omega_x \\ 0 \end{bmatrix} = {}^{\mathrm{v}}_{\mathrm{o}}R\dot{\bar{f}}\,. \tag{12}$$

Equation (9) and the first two rows of (12) provide three equations for four unknown control inputs. To fully constrain the control input trajectories for the given position trajectory, a fourth constraint must be additionally specified, which can be taken to be a user-defined constraint on ω_z ($\omega_z = 0$, for example).

B. Coupled Feasibility Conditions

Feasibility constraints for trajectories can now be calculated from the initial state and control input constraints as follows.

1) Collective Thrust: The collective thrust calculated from (9) must lie between the minimum and maximum thrust defined by (5), i.e.,

$$\|f\| = \sqrt{\ddot{x}^2 + \ddot{y}^2 + (\ddot{z} + \mathrm{g})^2} \geq a_{\min} \tag{13}$$

$$\|f\| \leq a_{\max}\,. \tag{14}$$

2) Rotational Rates: The actual body rate control inputs ω_x and ω_y can only be computed if a constraint on ω_z is specified in addition to the trajectory. However, they can be bounded using the unit norm property of the rotation matrix [30] with (12)

$$\omega_{x,y} \leq \left\| \dot{\bar{f}} \right\| \tag{15}$$

where $\omega_{x,y}$ is used to denote that the above holds for both ω_x and ω_y. It follows that a trajectory is guaranteed to be feasible if

$$\left\| \dot{\bar{f}} \right\| \leq \omega_{xy,\max}\,. \tag{16}$$

The above constraint is independent of the specification of ω_z, and, therefore, holds for a translational trajectory irrespective of the rotational motion defined through ω_z.

3) Initial State: The trajectory must satisfy the constraints arising from the initial state of the vehicle. Specifically, the position and velocity must coincide with the initial values

$$x(t=0) = x_0, \quad y(t=0) = y_0, \quad z(t=0) = z_0 \tag{17}$$

$$\dot{x}(t=0) = \dot{x}_0, \quad \dot{y}(t=0) = \dot{y}_0, \quad \dot{z}(t=0) = \dot{z}_0 \tag{18}$$

and the acceleration at the start of the trajectory must be such that (11) is satisfied with the given attitude, that is

$${}^{\mathrm{o}}_{\mathrm{v}}R_0 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \bar{f}(t=0)\,. \tag{19}$$

C. Decoupling of the Feasibility Conditions

To simplify the 3-D planning problem, it is desirable to perform the planning separately for each of the coordinates x, y, and z. This requires the definition of independent motion constraints for each degree of freedom. In terms of trajectory feasibility, the three axes are coupled in the acceleration constraints (13) and (14), in the mixed acceleration and jerk constraint (16), and in the initial attitude constraint (19). By conservatively approximating these constraints, it is possible to reformulate the trajectory generation problem such that the three degrees of freedom are entirely decoupled, implying that the overall trajectory will be guaranteed to be feasible if the three degrees of freedom satisfy their respective independent constraints.

1) Minimum Thrust Constraint: In order to satisfy the minimum thrust constraint (13), the vertical acceleration is constrained to be

$$\ddot{z} \geq \ddot{z}_{\min} \tag{20}$$

where the minimum permissible vertical acceleration $\ddot{z}_{\min}$ must satisfy

$$\ddot{z}_{\min} \geq a_{\min} - \mathrm{g}. \tag{21}$$

It is then straightforward to verify that (13) holds independently of the trajectories in x and y because the thrust is no smaller than $a_{\min}$. While, at this point, it may seem beneficial to choose the smallest possible value of $\ddot{z}_{\min}$, the tradeoffs involved in the choice of this parameter will become obvious in the following section.

2) Rotational Rate Constraint: The rotational rate control inputs nonlinearly couple the jerk and acceleration of the three degrees of freedom according to (12), and their limitations (2) and (3), therefore, couple the permissible values of the trajectory jerks and accelerations.

The trajectory jerk $\dot{f}$ is related to the actual control inputs through (12) by explicitly calculating $\dot{\bar{f}}$ as

$$\begin{bmatrix} \omega_y \\ -\omega_x \\ 0 \end{bmatrix} = {}^{\mathrm{v}}_{\mathrm{o}}R \left(\frac{\dot{f}}{\|f\|} - \frac{f f^T \dot{f}}{\|f\|^3} \right). \tag{22}$$

The above expression is at most

$$\omega_{x,y} \leq \left\| \frac{\dot{f}}{\|f\|} - \frac{f f^T \dot{f}}{\|f\|^3} \right\| \tag{23}$$

$$\omega_{x,y} \leq \left\| \left(I - \frac{f f^T}{\|f\|^2} \right) \frac{\dot{f}}{\|f\|} \right\|. \tag{24}$$

Noting that the term in brackets in the above bound represents the projection to the plane orthogonal to f, $\omega_{x,y}$ is bounded by

$$\omega_{x,y} \leq \frac{\left\| \dot{f} \right\|}{\|f\|}. \tag{25}$$

As the minimum vertical acceleration constraint (20) provides a lower bound for $\|f\|$, feasibility with respect to the body rate input constraints (2) and (3) is guaranteed if

$$\left\| \dot{f} \right\| \leq (\ddot{z}_{\min} + \mathrm{g}) \omega_{xy,\max} \tag{26}$$

holds. For the decoupled system, this can be ensured by limiting the permissible per-axis jerk of a planned trajectory by

$$|\dddot{x}| \leq \dddot{x}_{\max}, \qquad |\dddot{y}| \leq \dddot{y}_{\max}, \qquad |\dddot{z}| \leq \dddot{z}_{\max} \tag{27}$$

and choosing the limiting values such that (26) is satisfied

$$\dddot{x}_{\max}^2 + \dddot{y}_{\max}^2 + \dddot{z}_{\max}^2 \leq (\ddot{z}_{\min} + \mathrm{g})^2 \omega_{xy,\max}^2. \tag{28}$$

It can now be seen that the lower bound of the vertical acceleration $\ddot{z}_{\min}$ can be used as a design parameter in order to tradeoff higher allowable jerk magnitudes against achievable negative accelerations in the vertical (see (20) and (28)).

For the purpose of this paper, the jerk limit is chosen to be identical for the three degrees of freedom, i.e.,

$$\dddot{w}_{\max} := \dddot{x}_{\max} = \dddot{y}_{\max} = \dddot{z}_{\max} = \frac{(\ddot{z}_{\min} + \mathrm{g})}{\sqrt{3}} \omega_{xy,\max}. \tag{29}$$

3) Maximum Thrust Constraint: The maximum allowable thrust (14) imposes a constraint on the two-norm of the accelerations in the three degrees of freedom. This constraint is satisfied if

$$|\ddot{x}| \leq \ddot{x}_{\max}, \qquad |\ddot{y}| \leq \ddot{y}_{\max}, \qquad \ddot{z} \leq \ddot{z}_{\max} \tag{30}$$

where $\ddot{z}$ is already lower bounded by (20). The acceleration bounds of the three degrees of freedom are parameterized as functions of the maximum thrust constraint (5) using the maximum acceleration tradeoff parameters $\alpha_x, \alpha_z \in (0\ 1)$ as

$$\alpha_z := \frac{\ddot{z}_{\max}}{(a_{\max} - \mathrm{g})} \tag{31}$$

$$\alpha_x := \frac{\ddot{x}_{\max}}{\sqrt{a_{\max}^2 - (\ddot{z}_{\max} + \mathrm{g})^2}}. \tag{32}$$

The two parameters specify the maximum accelerations $\ddot{x}_{\max}$ and $\ddot{z}_{\max}$. The permissible acceleration $\ddot{y}_{\max}$ is then chosen to be as large as possible while fulfilling constraint (14)

$$\ddot{y}_{\max} = \sqrt{a_{\max}^2 - \ddot{x}_{\max}^2 - (\ddot{z}_{\max} + \mathrm{g})^2}. \tag{33}$$

If the constraints (30) defined through (31)–(33) with $\alpha_x, \alpha_z \in (0\ 1)$ are satisfied, then the maximum thrust constraint (14) is satisfied by design.

4) Initial Acceleration: To decouple the initial acceleration constraint (19), it is replaced by individual initial acceleration constraints for the three degrees of freedom. The initial accelerations $\ddot{x}_0, \ddot{y}_0, \ddot{z}_0$ are defined through the initial attitude ${}^{\mathrm{o}}_{\mathrm{v}}R_0$ using (6)

$$\begin{bmatrix} \ddot{x}_0 \\ \ddot{y}_0 \\ \ddot{z}_0 \end{bmatrix} := {}^{\mathrm{o}}_{\mathrm{v}}R_0 \begin{bmatrix} 0 \\ 0 \\ a_0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -\mathrm{g} \end{bmatrix} \tag{34}$$

with the initial thrust control input a_0 remaining a free degree of freedom. For applications where the trajectory generation algorithm is applied recursively (e.g., when it is used as a feedback law), the last value of the thrust a is typically available and can readily be used to compute the initial acceleration; in other usage scenarios, the initial value a_0 may be considered a design variable.

It is assumed that the initial state $\ddot{z}_0$ defined by (34) satisfies the minimum acceleration constraint (20), i.e., the initial acceleration must satisfy $\ddot{z}_0 \geq \ddot{z}_{\min}$ for the chosen value a_0. This condition limits the range of permissible initial attitudes as can be seen in (34).

Because the allowable acceleration magnitudes ($\ddot{x}_{\max}, \ddot{y}_{\max}, \ddot{z}_{\max}$) can be traded off freely through (31) and (32), the initial accelerations $(\ddot{x}_0, \ddot{y}_0, \ddot{z}_0)$ may not satisfy the constraints (30). For nonzero initial accelerations, the acceleration constraints (30) are, therefore, replaced by the time-varying constraints

$$|\ddot{x}| \leq \overline{\ddot{x}}(t), \qquad |\ddot{y}| \leq \overline{\ddot{y}}(t), \qquad \ddot{z} \leq \overline{\ddot{z}}(t). \tag{35}$$

These time-varying constraints are constructed as follows, here explained for the example of the first degree of freedom: the maximum allowable acceleration magnitude $\overline{\ddot{x}}(t)$ starts at the magnitude of the initial acceleration ($|\ddot{x}_0|$). It then has constant slope (c_x) until it reaches the design value ($\ddot{x}_{\max}$)

$$\overline{\ddot{x}}(t) = \begin{cases} |\ddot{x}_0| + c_x t, & \text{for } 0 \leq t < (\ddot{x}_{\max} - |\ddot{x}_0|)/c_x \\ \ddot{x}_{\max}, & \text{for } t \geq (\ddot{x}_{\max} - |\ddot{x}_0|)/c_x \end{cases}. \tag{36}$$

It now remains to choose the slopes c_x, c_y, c_z such that the maximum thrust constraint (14) remains satisfied for all times given the initial acceleration $(\ddot{x}_0, \ddot{y}_0, \ddot{z}_0)$. This is done by differentiating between degrees of freedom where the constraints are increasing (i.e., $c_x > 0$) and those where they are decreasing ($c_x < 0$).

For degrees of freedom where the initial acceleration exceeds the allowable acceleration (e.g., $|\ddot{x}_0| > \ddot{x}_{\max}$), the slope is chosen to be the maximum jerk $c_x = -\dddot{w}_{\max}$, as given in (29).

This corresponds to the fastest possible transition into the designed maximum magnitude that is permissible under the jerk constraint (27).

Degrees of freedom where the initial acceleration is smaller than the allowable acceleration (e.g., $|\ddot{x}_0| < \ddot{x}_{\max}$) cannot increase their acceleration at the maximum permissible jerk value without violating the maximum thrust constraint (14). Intuitively speaking, they must "wait" for other degrees of freedom to reduce their respective acceleration before using the full acceleration. Given that the constraints are always reduced in magnitude at the maximum rate, the longest duration for any degree of freedom to reduce to zero acceleration is

$$\Delta T_0 := \max\left(\frac{|\ddot{x}_0|}{\dddot{w}_{\max}}, \frac{|\ddot{y}_0|}{\dddot{w}_{\max}}, \frac{|\ddot{z}_0|}{\dddot{w}_{\max}}\right). \tag{37}$$

The slope for acceleration bounds exceeding the initial acceleration magnitude are then chosen to reach their allowable acceleration after ΔT_0, given here, for example, the x degree of freedom

$$c_x = \frac{\ddot{x}_{\max} - |\ddot{x}_0|}{\Delta T_0}. \tag{38}$$

The proof that this choice of slopes c_x, c_y, c_z satisfies the maximum thrust constraint (14) is omitted here but may be found in Appendix.

5) Overview: This completes the decoupling of the system. In conclusion, if the following conditions are satisfied:

1) the initial attitude is converted to an initial acceleration according to (34) and satisfies the minimum vertical acceleration constraint (20);
2) the jerk in each axis is constrained in magnitude according to (27); and
3) the acceleration of the individual degrees of freedom is constrained by (20) and (35);

then the 3-D overall trajectory will be feasible with respect to the system dynamics (6) and (7), the input constraints (2)–(5), and the initial conditions. It is then possible to design the trajectory of the three translational degrees of freedom independently.

The parameters used in the decoupling of the three degrees of freedom are the maximum acceleration tradeoff parameters α_x and α_z, the minimum vertical acceleration $\ddot{z}_{\min}$, and the initial thrust a_0.

V. Decoupled Trajectory Planning

The basic trajectory generation algorithm will be introduced in this section. The decoupled feasibility constraints (see Section IV-C) allow the planning to be carried out independently for each degree of freedom while guaranteeing feasibility. This makes the trajectory generation problem significantly more tractable and allows it to be solved in a computationally efficient way.

A. Time-Optimal Trajectories of the Decoupled System

Each degree of freedom is represented by a triple integrator under acceleration and jerk constraints (the only difference is that the acceleration constraints for x and y are symmetric, while they may be asymmetric for z). As the planning is identical for all coordinates, it is presented here for a general degree of freedom w. In order to achieve fast motion, we plan time-optimal trajectories for each axis.

1) Problem Statement: Let $s = (s_1,\ s_2,\ s_3) = (w,\ \dot{w},\ \ddot{w})$ be the state. The time-optimal planning problem can then be stated as: find the planning input $u = \dddot{w}$ minimizing the final time $t_{f,w}$

$$u^* = \arg\min_u t_{f,w} \tag{39}$$

subject to the system dynamics

$$\dot{s}_1 = s_2 \tag{40}$$

$$\dot{s}_2 = s_3 \tag{41}$$

$$\dot{s}_3 = u \tag{42}$$

the initial and final conditions

$$s_1(t=0) = w_0 \tag{43}$$

$$s_2(t=0) = \dot{w}_0 \tag{44}$$

$$s_3(t=0) = \ddot{w}_0 \tag{45}$$

$$s(t=t_{f,w}) = 0 \tag{46}$$

and the state and input constraints

$$\underline{\ddot{w}} \le s_3 \le \overline{\ddot{w}} \tag{47}$$

$$|u| \le \dddot{w}_{\max}. \tag{48}$$

In this formulation, the initial vehicle attitude (34) is enforced by the initial condition (45), the jerk constraint (27) by the input constraint (48), and the acceleration constraints (20) and (35) by the state constraint (47). Because the translational dynamics (6) contain no velocity-dependent aerodynamic effects, no velocity constraints are considered.

2) Necessary Optimality Conditions: Using Pontryagin's minimum principle (see, for example, [31]), necessary conditions for optimal input trajectories will now be derived. The methodology used to handle the state constraint (47) is the *direct adjoining approach* [32], in which the augmented Hamiltonian function is defined by

$$\begin{aligned} H(s,u,\lambda,\eta) = {} & \lambda_1 s_2 + \lambda_2 s_3 + \lambda_3 u \\ & + \eta_1 (s_3 - \underline{\ddot{w}}) + \eta_2 \left(\overline{\ddot{w}} - s_3\right) \end{aligned} \tag{49}$$

where λ are the adjoint variables and η are state constraint multipliers that fulfill

$$\eta \ge 0 \tag{50}$$

$$\eta_1 = 0, \quad \text{if } s_3 > \underline{\ddot{w}} \tag{51}$$

$$\eta_2 = 0, \quad \text{if } s_3 < \overline{\ddot{w}}. \tag{52}$$

The adjoint variables must fulfill

$$\dot{\lambda} = -\nabla_s H(s,u,\lambda,\eta) \tag{53}$$

which results in

$$\dot{\lambda}_1 = 0 \tag{54}$$

$$\dot{\lambda}_2 = \lambda_1 \tag{55}$$

$$\dot{\lambda}_3 = \lambda_2 + \eta_1 - \eta_2 \,. \tag{56}$$

The optimal control u^* is the control input that minimizes the Hamiltonian function

$$u^* = \arg\min H(s, u, \lambda, \eta) \tag{57}$$

$$= \arg\min \lambda_3 u \,. \tag{58}$$

For this problem structure, it can be shown that the adjoint variables λ are continuous [32]. Furthermore, $\lambda_3 = 0$ must hold when a state constraint is active. The optimal control input u^* consists of *interior arcs* and *boundary arcs*. An interior arc is characterized by $\eta_1 = \eta_2 = 0$ (i.e., the state constraint (47) is not active), $\lambda_3 \neq 0$, and, therefore, $u^* = \pm\dddot{w}_{\max}$. On boundary arcs, the state constraint is active (i.e., $s_3 = \underline{\ddot{w}}$ or $s_3 = \overline{\ddot{w}}$), and it follows that $\lambda_3 = 0$ and u^* is chosen such that the state constraint remains active [that is, the control input is the derivative of the acceleration constraint (36)].

3) Structure of Time-Optimal Trajectories: Using the properties of interior arcs and boundary arcs, it is straightforward to characterize the solution of the costate dynamics (56) further: During an interior arc, λ_3 is a second-order polynomial since $\dddot{\lambda}_3 = \dot{\lambda}_1 = 0$. Since λ_3 is continuous, a boundary arc can only begin at a point where $\lambda_3 = 0$ holds, and the solution of λ_3 is then given by the constraint that $\lambda_3 = 0$ must hold for the duration of the boundary arc.

It can be verified from the above constraints that the trajectory consists of at most five sections as follows:

1) $[0\ t_1]$ is an interior arc, with $u^* = \pm\dddot{w}_{\max}$.
2) $[t_1\ t_2]$ is a boundary arc, with u^* such that $\ddot{w} = \overline{\ddot{w}}$ or $\ddot{w} = \underline{\ddot{w}}$.
3) $[t_2\ t_3]$ is an interior arc, with $u^* = \mp\dddot{w}_{\max}$.
4) $[t_3\ t_4]$ is a boundary arc, with u^* such that $\ddot{w} = \overline{\ddot{w}}$ or $\ddot{w} = \underline{\ddot{w}}$.
5) $[t_4\ t_f]$ is an interior arc, with $u^* = \pm\dddot{w}_{\max}$.

Furthermore, each boundary arc induces one additional constraint, in which one or the other of the following conditions must hold:

1) If the duration of the boundary arc is nonzero ($t_2 - t_1 > 0$ or $t_4 - t_3 > 0$, respectively), then s_3 must coincide with the upper or lower acceleration constraint at the start point of the boundary arc (t_1 or t_3).
2) If the above condition does not hold, the corresponding boundary arc must vanish ($t_2 = t_1$ or $t_4 = t_3$, respectively).

In the following, the solution to the optimal control problem (39)–(48) will be denoted by $w^*(t)$. An example of the solution of the optimal control problem is illustrated in Fig. 2. This example depicts the case where all five sections are of nonzero duration.

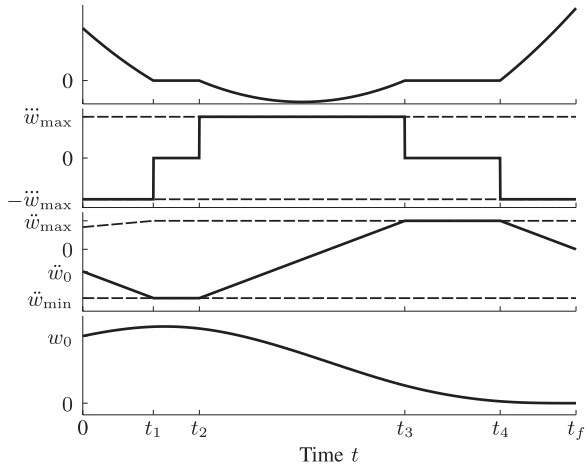

Fig. 2. Example of a solution to the 1-D planning problem introduced in Section V-A1. From top to bottom, the plots show the trajectories of: 1) the third adjoint variable λ_3; 2) the optimal control input u^* (solid) along with the maximum allowable input magnitude $\dddot{w}_{\max}$ (dashed); 3) the acceleration trajectory $\ddot{w}^*$ (solid) along with the upper and lower acceleration bounds $\underline{\ddot{w}}$, $\overline{\ddot{w}}$ (dashed); and 4) the position trajectory w^*. Note that this sketch shows asymmetrical acceleration bounds $\underline{\ddot{w}}$, $\overline{\ddot{w}}$, as used for the vertical degree of freedom.

B. Computation of Solutions

The control trajectory $u^*(t)$ (and, therefore, $w^*(t)$) is fully specified by the five times t_1, t_2, t_3, t_4, and t_f and the initial control input. The trajectory is constrained by the three terminal state conditions (46) and the two boundary arc constraints. Analytically integrating the equations of motion (40)–(42) is straightforward and results in three equations of first, second, and third order in the switch times $t_1, \ldots, t_f$, respectively. The two boundary arc constraints yield two additional first order equations in the switch times.

The solution is computed by applying a bisection algorithm [33] to the final time t_f. For a given value of t_f, it is straightforward to compute the remaining times $t_1, \ldots, t_4$ and, thereby the resulting final position $w(t_f)$, which is a strict monotonic function of t_f for a given initial control input [21].

C. Overview

The basic trajectory generation algorithm has herewith been described. The total 3-D maneuver is given by

$$x(t) = \begin{cases} x^*(t), & \text{for } 0 \leq t \leq t_{f,x} \\ 0, & \text{for } t > t_{f,x} \end{cases} \tag{59}$$

$$y(t) = \begin{cases} y^*(t), & \text{for } 0 \leq t \leq t_{f,y} \\ 0, & \text{for } t > t_{f,y} \end{cases} \tag{60}$$

$$z(t) = \begin{cases} z^*(t), & \text{for } 0 \leq t \leq t_{f,z} \\ 0, & \text{for } t > t_{f,z} \,. \end{cases} \tag{61}$$

The maneuver satisfies the initial conditions, the quadrocopter dynamics, and the input constraints. It ends at rest at the origin

at time

$$t_f := \max(t_{f,x}, t_{f,y}, t_{f,z})\,. \tag{62}$$

It is, therefore, a valid solution to the trajectory generation problem stated in Section III.

VI. Choice of Design Parameters

The algorithm presented in the previous chapter computes a feasible trajectory that satisfies the trajectory planning problem. However, the decoupling approach involved the choice of the design parameters $\alpha_x, \alpha_z, \ddot{z}_{\min}, a_0$ in order to allow the separate planning in each degree of freedom. This section presents and discusses the properties and influence of the individual parameters, and proposes possible iterative improvement schemes that permit the computation of better trajectories[2] through their variation.

For any choice of the decoupling parameters satisfying the conditions outlined in Section IV-C, the solution of the decoupled trajectory planning (59)–(61) is a feasible solution to the trajectory generation problem. It is, therefore, possible to apply the iterative improvement schemes to only a subset of parameters, or to not apply them at all.

A consequence of this property is that as long as the basic trajectory generation algorithm from the previous chapter has been executed at least once, a feasible solution is available at any time, and further executions then improve on previous ones. This implies that computational constraints can always be abided by aborting computation when the permissible computation duration is reached. If the computation time is not sufficient for the iteration schemes to converge to a user-defined level of accuracy, then the available solution differs from the converged solution in that the maneuver duration may be longer, but is still guaranteed to be feasible with respect to the dynamics and constraints of the problem.

A. Acceleration Tradeoff Parameters

The acceleration tradeoff parameters α_x and α_z control how much of the vehicle thrust capability is allocated to each of the three degrees of freedom according to the respective acceleration bounds (47).

To plan a 3-D trajectory, the time-optimal trajectory for each of the three decoupled systems is computed. The execution times of the three degrees of freedom will generally depend on the choice of the respective acceleration bounds (47); the maneuver durations $t_{f,x}, t_{f,y}, t_{f,z}$ are likely to differ from each other. To improve overall performance, the allowable per-axis acceleration may be varied such that if one degree of freedom has a longer execution time than the others, it is allocated more acceleration in order to reduce the execution time.

An important property of the decoupling parameters α_x and α_z is that the duration of the time-optimal maneuver of each single degree of freedom depends on them monotonically. This can be shown as follows:

[2]Because the objective of the trajectory generation is the minimization of the maneuver duration, trajectories are considered better if they are of shorter duration. The descriptions "shorter" and "better" are therefore used interchangeably.

The acceleration boundaries $(\ddot{x}_{\max}, \ddot{y}_{\max}, \ddot{z}_{\max})$ are strictly monotonic with respect to the control effort tradeoff parameters (31) and (32). By construction, the time-varying acceleration constraints (36) monotonically depend on the respective acceleration boundary $\ddot{w}_{\max}$. That is, if $\ddot{w}_{\max,1} \geq \ddot{w}_{\max,2}$ then

$$\overline{\ddot{w}}_1(t) \geq \overline{\ddot{w}}_2(t), \quad \text{for all } t \geq 0\,. \tag{63}$$

Because the computed trajectories are optimal with respect to the maneuver duration, an increase of the constraints cannot increase the terminal time of the maneuver [31], and the maneuver duration is, therefore, monotonic with respect to the decoupling parameters α_x and α_z.

From this property, it follows that a maneuver synchronizing the maneuver durations of the three degrees of freedom minimizes t_f. This can be shown as follows: Assume that the three maneuver durations are synchronized. Because the maneuver durations of the three degrees of freedom are monotonic with respect to the control effort tradeoff parameters, no change of the parameters can reduce all three maneuver durations [and, thereby the total maneuver duration t_f as given by (62)].

The monotonicity of the maneuver durations with respect to the optimization parameters allows the use of a large number of optimization algorithms to find their optimal value. A straightforward implementation consists of two loops: an inner loop that synchronizes the two horizontal degrees of freedom (i.e., $t_{f,x}$ and $t_{f,y}$) by varying α_x, and an outer loop that synchronizes the vertical motion (i.e., $t_{f,z}$) to the two horizontal motions by varying α_z. Each of these loops can be implemented in a simple manner by a bisection algorithm, which assures linear convergence. Specifically, this implies that the complexity of the two nested bisections finding the optimal values of α_x and α_z to within a tolerance ϵ_{xz} is [33]

$$\mathcal{O}\left(\left(\log_2\left(\epsilon_{xz}\right)\right)^2\right)\,. \tag{64}$$

In addition, the computation of the inner loop can be aborted early if it is found that the maneuver duration of both horizontal degrees of freedom is either shorter or longer than the duration of the vertical degree of freedom. Due to the maneuver duration monotonicity with respect to α_x, it is not possible to increase or decrease the maneuver duration of both horizontal degrees of freedom, and synchronization with the vertical degree of freedom is, therefore, not possible. By using this property, it is not necessary to perform the bisection of α_x to full accuracy for each bisection step in α_z.

The two nested bisection algorithms provide a straightforward way to compute the optimal values of the two control tradeoff parameters α_x and α_z. Furthermore, the maneuver duration monotonicity and linear convergence of the bisection method allow the *a priori* determination of hard constraints on the number of required executions. However, more sophisticated and multivariate optimization methods (see e.g., [34]) could provide higher computational efficiency in the finding of the optimal values.

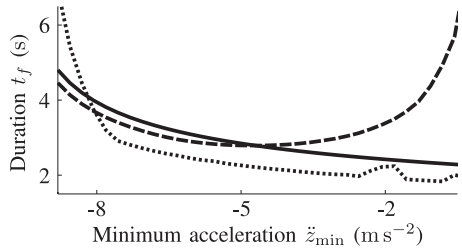

Fig. 3. Influence of the design parameter $\ddot{z}_{\min}$ on the maneuver duration t_f with optimized tradeoff parameters α_x, α_z for three different maneuvers (solid, dashed, and dotted). The optimization presented in Section VI-B finds the minimum of this function through an exhaustive search.

B. Minimum Vertical Acceleration

The minimum vertical acceleration $\ddot{z}_{\min}$ directly affects the acceleration constraint (47) for the decoupled motion planning problem of the vertical degree of freedom. Through (29), it also influences the allowable control input (48) for all the three degrees of freedom. Fig. 3 shows the maneuver duration (after optimization of α_x and α_z to a tolerance of $\epsilon_{xz} = 10^{-3}$ as described in the previous section) over varying values of $\ddot{z}_{\min}$ for three different maneuvers. It can be seen that the function may have multiple local minima. This complicates the application of standard optimization methods.

In order to find the global minimum maneuver time during the on-line optimization of the minimum vertical acceleration, it is necessary to use an optimization algorithm that is not sensitive to local minima. A straightforward way to achieve this is to grid the search space at a user-defined grid size, and evaluate the maneuver duration at each point. While such a naive approach would be computationally prohibitive for multivariate optimizations, it can be performed to a satisfactory tolerance in the case of only one optimization variable.

Like in the optimization of the acceleration tradeoff parameters, the use of more sophisticated optimization algorithms could further reduce the computational complexity. While it may be difficult to guarantee the convergence to the global optimum in all cases, it may be acceptable to provide only local convergence since feasibility remains guaranteed.

C. Overview

This chapter presented schemes for finding the optimal values of the design parameters $\alpha_x, \alpha_z, \ddot{z}_{\min}$ such that the maneuver duration is minimized. The application of these schemes is optional since the feasibility of the trajectory generation algorithm does not depend on them. Depending on the computational constraints and performance requirements, one may choose to optimize all design parameters, or to fix some or all of them ahead of time.

The schemes presented herein are focused on simplicity and assured convergence, and optimize each of the design parameters individually in nested loops. Through the use of more advanced optimization algorithms, and by directly considering the multivariate optimization problem, the computation time may be reduced further.

The decoupling process also involved the choice of the initial thrust a_0 in (34). While this can be considered a design parameter and therefore optimized, it is assumed herein that it is chosen based on the thrust applied previously. This results in continuous accelerations and, therefore, continuous thrust commands [as computed by (9)], a favourable property with respect to the underlying thrust dynamics.

VII. Use as a Feedback Law

By solving the trajectory generation problem repeatedly at a high frequency, it is possible to use the trajectory generation algorithm as an implicit feedback law. The control law then consists of updating the initial conditions according to an updated state estimate (typically obtained using new measurements), replanning the trajectory to the target point, and applying the control inputs of the first control interval (that is, for the duration until the trajectory is replanned).

To apply the feedback law, it is necessary to compute the control inputs from the planned trajectory. Whereas the control inputs are usually assumed piecewise constant in MPC, the trajectory planned here typically results in continuously varying control inputs (the trajectory jerk is piecewise constant, but the control inputs vary over time). For any given time, the thrust input a can be computed according to (9), and the body rates ω_x, ω_y can be computed by numerically integrating the rotation matrix derivative (7) with the intermediate control inputs computed through (12). In addition, the specification of the body rate ω_z is required, which is not defined by the planned translational trajectory (see Section IV-B). Because the trajectory is feasible for arbitrary choices of ω_z and the control inputs can be computed under consideration of any given trajectory of ω_z, this additional degree of freedom can be chosen independently. Examples of possible choices are to simply set $\omega_z = 0$ in order to reduce control effort, or to choose ω_z such that the vehicle heading remains constant or follows a specified trajectory.

The feedback control law resulting from the repeated trajectory generation using the presented algorithm is closely related to MPC methods [6] in that an updated optimal trajectory is generated at each time step. The presented algorithm always plans a full trajectory to the final state, similar to infinite-horizon MPC formulations. The planning of the full to rest trajectory results in a control law that is known to be stable if the decoupling parameters (presented in Section VI) are chosen to be constant [35]. If the decoupling parameters are allowed to vary (for example through online optimizations), then the stability of the resulting feedback law generally depends on the variations of the decoupling parameters. A potential direction of future research would be to find conditions on the way the decoupling parameters are varied in order to guarantee a stable closed-loop system, and to investigate the robustness of the feedback law to parameter uncertainty.

VIII. Performance Evaluation

The overall trajectory generation algorithm, consisting of the basic decoupled trajectory generator (see Section V) and the iterative improvement schemes (see Section VI), has now been described. Generated trajectories satisfy the feasibility constraints imposed by the dynamics of the quadrocopter by construction. This section will characterize the performance of the presented

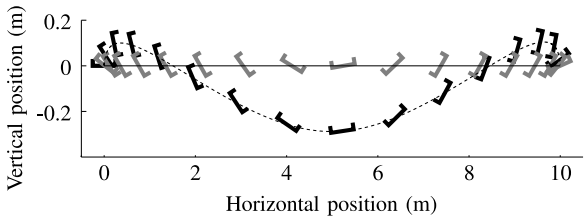

Fig. 4. Comparison of a horizontal translation maneuver of 10 m generated using the algorithm presented herein (solid line, gray snapshots; total maneuver duration 1.76 s) and a time-optimal translation [18] (dashed line, black snapshots; total duration 1.56 s). Snapshots are shown at an interval of 0.1 s. The decoupling presented in Section IV-C results in the vehicle remaining at a constant altitude throughout the maneuver. Note that the vertical axis is not to scale.

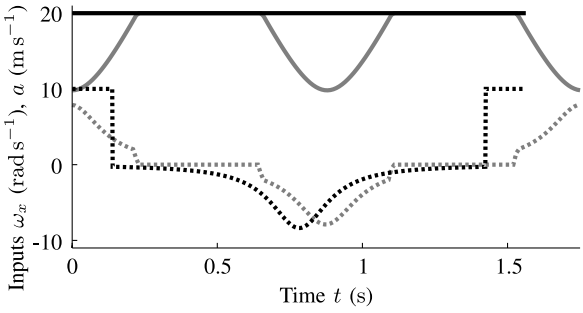

Fig. 5. Control inputs corresponding to the comparison of the purely horizontal maneuver, as shown in Fig. 4. The solid lines represent the thrust input a, and the dotted lines the rotational rate input ω_x. Black lines correspond to the time-optimal translation [18], and gray lines are the inputs of the trajectory generated by the algorithm presented herein. Note that the decoupling yields non-maximal thrust inputs when the vehicle rotates (beginning, center, and end of the maneuver), and the conservatism introduced in the decoupled jerk feasibility constraints leads to a less efficient use of the rotational rate control input.

algorithm with respect to the design objectives of achieving high flight performance (short maneuver durations) and low computational complexity (short computation times).

A. Comparison with Time-Optimal Trajectories

To determine the performance of maneuvers computed by the presented method and to characterize the influence of the decoupling assumptions (presented in Section IV-C), the generated trajectories are compared with time-optimal trajectories.

A method using Pontryagin's minimum principle to derive optimality conditions when computing time-optimal trajectories for quadrocopters was presented previously in [18]. The method uses a reduced version of the quadrocopter model (2)–(7) to make the problem computationally tractable. The reduced version consists of two degrees of freedom (one horizontal and one vertical), resulting in five system states (the 2-D position, the 2-D velocity, and the scalar tilt angle). While computation times on the order of several hours and required user interaction make the method difficult to use in practical applications, the explicit consideration of the minimum principle optimality conditions makes the resulting trajectories strong candidates for truly time-optimal trajectories. They, thus, provide useful benchmarks for comparing against other methods (such as the one presented in this paper) in order to analyze relative performance.

Time-optimal reference trajectories were computed for hover-to-hover translations with the input saturations chosen to be $a_{\max} = 20$ m s^{-2}, $a_{\min} = 1$ m s^{-2}, and $\omega_{xy,\max} = 10$ rad s^{-1}. The same saturations were used for the real-time trajectory generation algorithm. Both iterative performance improvement schemes presented in Section VI were applied, with the tolerance of the acceleration tradeoff parameter optimization being $\epsilon_{xz} = 10^{-3}$ and a minimum vertical acceleration grid size of 0.25 m s^{-2}. While the time-optimal reference trajectories were computed using a 2-D model, no such simplifications were assumed in the trajectory generation algorithm. The initial acceleration [see (34)] was set to $a_0 = \mathrm{g}$.

Consider, as a first example, a purely horizontal translation of 10 m. The planned trajectories of both the time-optimal trajectory generator and the real-time trajectory generation algorithm are shown in Fig. 4, and the control inputs along the trajectory can be seen in Fig. 5. The duration of the time-optimal maneuver is 1.56 s, whereas the trajectory generated with the algorithm presented herein takes 1.76 s, which amounts to an increase of 13% . Considering the shape of the generated trajectories, two significant factors contributing to the increased duration can be identified as follows.

1) The decoupling assumption, along with identical initial and final altitudes and the maneuver starting at rest, leads to the generated trajectory being entirely at constant altitude. In comparison, the time-optimal maneuver exploits any available thrust during attitude changes to accelerate upward, thereby allowing the vehicle to tilt further and increase its horizontal acceleration. This effect is further enabled because the time-optimal maneuver is planned using the true control inputs (a, ω_x), enabling instantaneous changes in thrust. The decoupling presented herein results in the trajectory being planned in jerk, meaning that vehicle acceleration and, therefore, thrust [as computed by (9)] is always continuous throughout the maneuver. Although this lack of discontinuities in the thrust control input results in lower performance, it may be desirable due to the underlying actuator dynamics of the quadrocopter.
2) A further noticeable effect is highlighted in the control inputs corresponding to the two maneuvers (see Fig. 5): While the time-optimal maneuver uses the maximum rotational rate input for significant durations at the beginning and end of the maneuver, the motion generated using the algorithm presented herein results in lower rotational rate control inputs. This is mostly caused by the conservative approximations used to decouple the rotational rate feasibility conditions (29).

As a second example, consider a translation of 8 m in the horizontal and 8 m in the vertical. The time-optimal trajectory and the generated trajectory are shown in Fig. 6. In this case, the duration of the generated trajectory is 44% longer than the time-optimal trajectory (2.53 versus 1.76 s). Two contributing factors can be identified as follows.

1) The arguments about the use of jerk as a control input also apply in this case, leading for example, to a slower initial acceleration than the time-optimal motion provides.

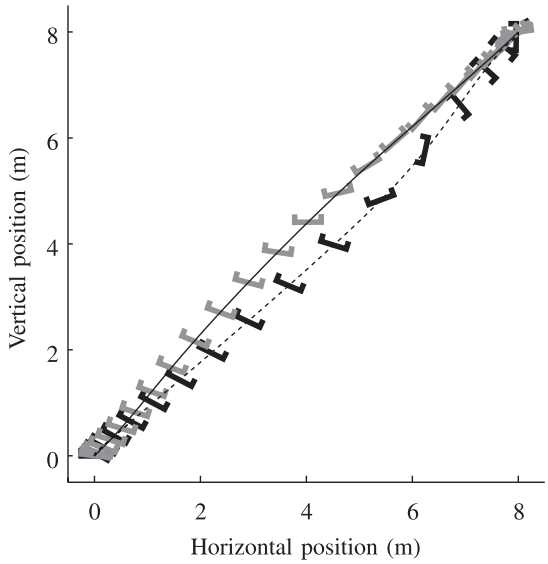

Fig. 6. Comparison of a diagonal translation maneuver of 8 m in the horizontal and 8 m in the vertical. The solid line and gray snapshots show the trajectory generated using the algorithm presented herein (total maneuver duration 2.53 s). The dashed line and black snapshots represent a time-optimal translation [18] (total duration 1.76 s). Snapshots are shown at an interval of 0.1 s. Note that the time-optimal maneuver involves the vehicle rotating to a pitch angle of approx. 140° in the final deceleration phase, whereas the minimum vertical acceleration constraint of the trajectory generation algorithm presented herein limits the pitch angle to approximately 50°.

2) A significant difference can be seen toward the end of the motion: During the final deceleration phase, the generated trajectory results in the vehicle pitching to an angle of approximately 50°, applying the optimized minimal vertical acceleration $\ddot{z}_{\min}$ and the minimal horizontal acceleration $-\ddot{x}_{\max}$. The time-optimal trajectory results in the vehicle being pitched to an upside-down attitude at an angle of approximately 140°, applying full thrust and, thereby, achieving significantly lower vertical accelerations. This difference is caused by the decoupling of the three degrees of freedom, wherein the minimum vertical acceleration constraint (20) was introduced. The value of the constraint was then optimized to tradeoff the permissible jerk against allowable vertical deceleration (see Section VI-B).

B. Computational Performance

In the following, computation times will be given for a laptop computer with an Intel Core i7-2620M processor running at 2.7 GHz. While a laptop computer would typically be used as an off-board computation platform, there are multirotor computation platforms that provide comparable computation capabilities (e.g., the Ascending Technologies Mastermind onboard PC). As will be obvious from the following results, it is also straightforward to deploy the algorithm on platforms with less available computation power.

1) Minimal Computational Requirements: The minimal implementation of the trajectory generation algorithm consists of the solution of the decoupled planning problem (presented in Section V) for each of the three degrees of freedom. The decoupling parameters $\alpha_x, \alpha_z, \ddot{z}_{\min}$ are chosen *a priori*. This minimal implementation results in low computational requirements at the price of reduced flight performance.

The computation time was evaluated through the computation of several million trajectories from randomized initial conditions. The average computation time of a 3-D flight trajectory was 24.6 μs. This short computation time is particularly suited for implementations on low-power platforms: Based on typical feedback rates on the order of 50 to 100 Hz [7], [8], a current-generation microcontroller (as used in open-source flight control units [7] and costing less than 5US$) would suffice to use the trajectory generation algorithm as an implicit feedback law.[3]

2) Iterative Improvement Performance: After the execution of the minimal implementation of the algorithm, remaining computation time can be used to iteratively improve the trajectory performance, as presented in Section VI. For example, it can be shown that the optimization of the acceleration tradeoff parameters α_x and α_z to a tolerance of 1% requires at most 171 calls of the 1-D decoupled trajectory generator, therefore requiring approximately 1.4 ms of computation time. For a tolerance of 10% in the tradeoff parameters, a maximum of 78 calls suffice (resulting in approximately 0.6-ms computation time). This optimization can then be repeated for varying values of the minimum vertical acceleration $\ddot{z}_{\min}$, where the number of evaluated points may be chosen based on computational constraints.

IX. Experimental Validation

To verify the trajectory generation algorithm experimentally, it was implemented in the Flying Machine Arena, an indoor aerial vehicle development platform at ETH Zurich [37]. The vehicles used for the experiments are custom-built quadrocopters that are based on Ascending Technologies "Hummingbird" vehicles [38]. The electronics mounted on board each vehicle provide inertial measurements and implement body rate feedback loops. The control inputs ($\omega_x, \omega_y, \omega_z, a$ as defined in Section II-A) are communicated to the vehicle from a desktop computer through a low-latency wireless communication channel at a rate of 50 Hz. A commercial motion capture system provides position and attitude information, which is filtered by a Luenberger observer.

For the experiments presented herein, the trajectory generation algorithm is run recursively and used as a feedback law as presented in Section VII. Every 20 ms, the updated vehicle state (computed by the Luenberger observer) is used as an initial condition to solve for the trajectory to the target point. The control inputs are then computed and sent to the vehicle.

A. Constant Altitude Experiment

As a first test case, trajectories are planned with nonchanging altitude target points, as shown in Fig. 7. The experiment starts

[3]Consider, as an example, the STM32F3 microcontroller by STMicroelectronics, which is based on the ARM Cortex M4F architecture. At 72 MHz, it achieves a CoreMark [36] score of approximately 245 compared with the laptop computer's CoreMark score of 13 986. Using the CoreMark scores as a rough performance measure, replanning trajectories at 100 Hz using the minimal implementation would, therefore, require about 15% of the available computation time on the microcontroller.

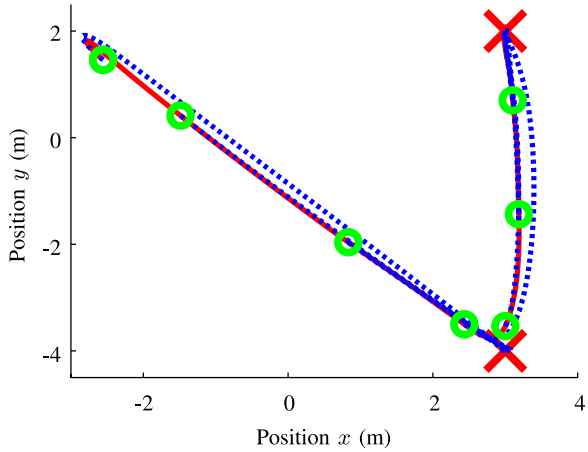

Fig. 7. Selection of replanned trajectories during experiment at constant altitude. The trajectory is replanned every 20 ms, and used as an implicit feedback law by applying the initial control inputs. The figure shows the flight path (solid red) and a small selection of planned trajectories (dotted blue, starting at green circles and ending at red crosses). The vehicle starts at the top-left green circle, moving toward the top left, and the target point is set to the bottom right. As the vehicle approaches the bottom right target point, the target is switched to the top right. Because the trajectory is replanned at every time step, the new target point is accounted for almost instantaneously.

with the vehicle at approximately $(x = -2.6\ \text{m}, y = 1.5\ \text{m})$, with the target point being $(x = 3\,\text{m}, y = -4\,\text{m})$, and the vehicle moving away from the target point at a speed of approximately 3 m s^{-1}. When the vehicle is within 1 m of the target point, the target point is switched to $(x = 3\ \text{m}, y = 2\ \text{m})$. During the experiment, the vehicle reached a maximum speed of 7 m s^{-1}.

While the first target point does not induce an overshoot of the planned trajectory in the positive x-direction, the switch in target point induces an overshoot of up to 40 cm. This overshoot is caused because the acceleration tradeoff parameter α_x (see Section VI-A) is adjusted by the optimization when the trajectory to the new target point is planned, and changes from a value of $\alpha_x = 0.69$ to $\alpha_x = 0.07$ when the target point switches. The trajectory generation's planning objective is to minimize the maneuver duration; almost all available acceleration is allocated to the second degree of freedom after the target point changes because the largest motion is required in that direction.

Note that the implicit control law does not track a previously planned trajectory, but a new trajectory to the target point is computed at every controller update. This replanning behavior can be observed particularly well when significant deviations from a previously planned trajectory occur, for example during the initial deceleration phase and when the vehicle changes direction after the target point is changed. While the initially planned trajectory contains a significant amount of overshoot, the actual overshoot during execution of the maneuver is smaller. Potential explanations for this behavior are the unmodeled effects of drag (which can cause more deceleration) [28] and translational lift (which can cause increased propeller efficiency) [39]; both of these are known to have significant influence at the maneuvering speeds encountered in this experiment. As the higher-than-planned deceleration allows for a more direct flight path, the trajectories planned later on contain almost no overshoot.

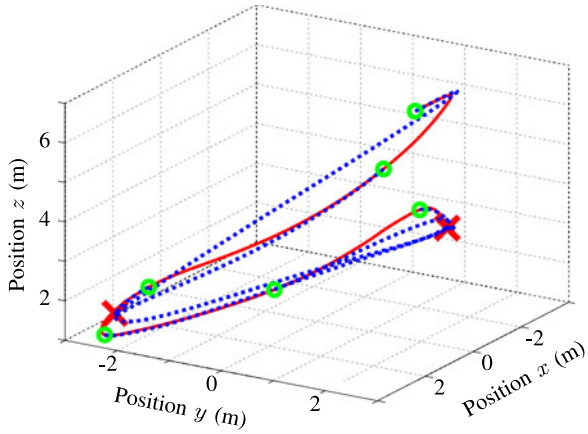

Fig. 8. Three-dimensional motion experiment. The vehicle starts at the top right (green circle) with the target point being at the bottom left (left red cross). As the vehicle approaches the target point, the target is switched to the center-right target point (right red cross). The solid red line shows the flown trajectory, and the dotted blue lines (starting at the green circles) show some of the planned trajectories.

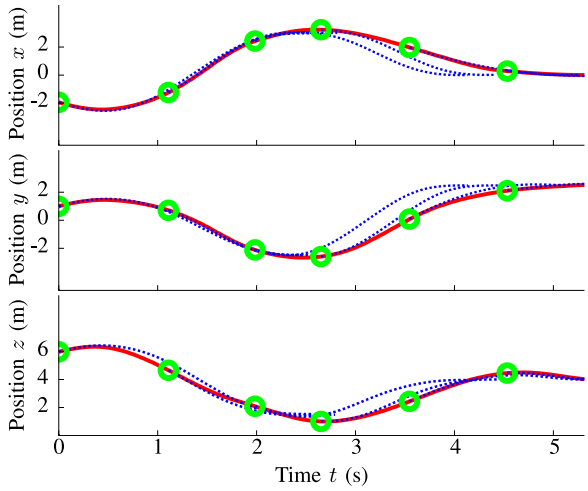

Fig. 9. Flown (solid red) and a selection of planned (dashed blue, starting at green circles) trajectories over time for the experiment shown in Fig. 8. The set point switches from $(x = 3$ m, $y = -2.5$ m, $z = 1.5$ m) to $(x = 0$ m, $y = 2.5$ m, $z = 4$ m) at $t = 1.94$ s.

B. Three-Dimensional Motion

In a second experiment, the set points are varied in all three dimensions. As can be seen in Fig. 8, the vehicle starts at approximately $(x = -2\ \text{m}, y = 1\ \text{m}, z = 6\ \text{m})$, with the target point being $(x = 3\ \text{m}, y = -2.5\ \text{m}, z = 1.5\ \text{m})$. Like in the previous experiment, the target point is changed when the vehicle is within 1 m of the target point, and the new target point is then $(x = 0\,\text{m}, y = 2.5\,\text{m}, z = 4\,\text{m})$. The vehicle starts at a speed of 3.6 m s^{-1}, and reaches a peak speed of 7.2 m s^{-1}.

Fig. 9 shows the flight trajectory over time along with the same planned trajectories that are seen in Fig. 8. Note that particularly during high-speed flight phases, the vehicle lags behind the planned trajectories. This behavior is again consistent with aerodynamic effects [40], and is excarberated (in comparison with the constant altitude experiment) by the vertical speed of

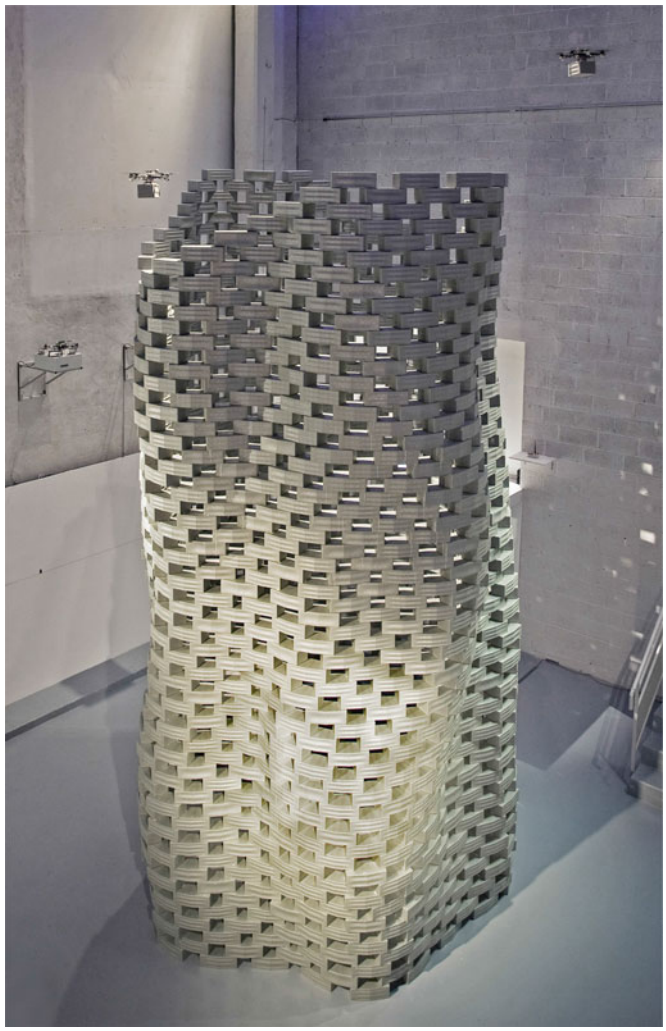

Fig. 10. Tower made of foam bricks, assembled by quadrocopters during the Flight Assembled Architecture project [41]. Two of the quadrocopters can be seen above the tower to the left and right; a third is seen resting on a charging station on the wall to the left. The flight paths used by the quadrocopters for this project were generated using the trajectory generation algorithm presented in this paper. Photograph: François Lauginie, with permission.

the vehicle ranging from -3.8 to $2.8\,\mathrm{ms}^{-1}$. The propeller inflow velocity thus changes considerably.

The deviation of the flown trajectory from the trajectories planned at various points in time demonstrate that the dynamic model, where negligible aerodynamic effects were assumed (discussed in Section II-B), does not fully capture the dynamics during fast maneuvering. However, it can be seen that the continuous replanning of the trajectory naturally takes the deviations from the planned trajectory into account. This approach of using a less accurate model to reduce the computational burden and allow for fast replanning is similar to approaches used in MPC (see, for example, [29]).

C. Other Uses

This trajectory generation algorithm has also been used extensively as a building block in higher-level tasks. An example of this is its use in the Flight Assembled Architecture project [41], in which a fleet of four quadrocopters assembled a 6-m tall tower out of 1500 foam bricks (see Fig. 10) in front of a live audience. To achieve this task, the minimal implementation of the trajectory generation algorithm (that is, using fixed decoupling parameters $\alpha_x, \alpha_z, \ddot{z}_{\min}$) was used to plan the motions of quadrocopters between battery charging stations, the pick-up points of the foam bricks, and their respective placement point. The ability to plan trajectories from nonrest conditions was used in conjunction with way points in order to guide vehicles around obstacles without stopping. Furthermore, the planned trajectories were used as an input to a space reservation system (inspired by [42]) in order to ensure that no collisions occur. To ensure that vehicles comply with space reservations, trajectories were not planned at every time step, but generated once and then tracked by a feedback controller. Over the duration of the project, tens of thousands of trajectories were generated.

X. Conclusion

This paper presented and analyzed a method for generating quadrocopter flight trajectories. The method efficiently computes trajectories that both allow for fast motion and are guaranteed to be feasible.

The decoupling of the trajectory generation problem under conservative approximation of the feasibility constraints allows the problem to be simplified considerably, and the computation of time-optimal trajectories for the decoupled system reduces to determining the switching times of the bang-singular solution trajectory. A particular strength of the method is that it very quickly allows the computation of a provably feasible trajectory (that is, within few tens of microseconds on a laptop computer), but can also iteratively provide higher quality solutions if more computation time is available. The iterative improvement allows optimal design parameters to be determined and can be carried out until convergence criteria are met, or aborted at any time if no more computation time is available.

Analysis of the computed trajectories showed structural differences when compared with time-optimal trajectories. These differences are mostly caused by the decoupling approach. The trajectory generation algorithm presented herein trades off the full use of the dynamic capabilities of quadrocopters for the ability to compute tens of thousands of trajectories every second. This makes it particularly suitable to applications that require the planning of large numbers of trajectory candidates (such as, for example, sampling-based path planners) or fast replanning due to rapidly changing or *a priori* unknown environments or task objectives.

The approach was experimentally verified for quadrocopters maneuvering at moderately high speeds (up to $7\,\mathrm{ms}^{-1}$), and was shown to cope well with disturbances due to unmodeled dynamic effects and changing target points. Its robustness and applicability as a building block in more complex systems has been demonstrated by its use in the assembly of a large structure by quadrocopters, in the process of which tens of thousands of trajectories were generated.

While this paper has demonstrated the validity of the decoupling approach and the possibility to design algorithms of

sufficiently low computational efficiency, a number of potential improvements remain to be investigated.

First, the algorithm presented herein is aimed at applications where relatively small spaces are to be navigated. This allows the quadrotor dynamics to be simplified by neglecting aerodynamic effects, and leads to the planned trajectories not being subject to velocity constraints. To broaden the scope of the algorithm, an interesting extension would be to include velocity constraints, which could be used to guarantee that aerodynamic effects remain sufficiently small. Such constraints may also be imposed by the sensing modalities used for the state estimation of the quadrocopter.

Second, in the design of the decoupled feasibility constraints, the per-axis jerk constraint [see (29)] was chosen to be symmetric for all three axes. The tradeoff of the three jerk values could also be considered to be design parameters, and could be optimized for each maneuver.

Furthermore, the iterative improvement schemes used to adapt the decoupling parameters to the specific problem data could be improved, thereby increasing performance. The optimal choice of the decoupling parameters forms an optimization problem that contains strong structure, and should, therefore, lend itself to the application of many more advanced optimization methods. When the trajectory generator is called repeatedly (for example, because it is used as a feedback law), the use of the previous solution to initialize the optimization algorithm could further reduce computational cost. When the trajectory generator is used as a feedback law, a particular point of interest in the optimization of the decoupling parameters is the influence of the optimization on the stability of the resulting feedback law.

An alternative approach to the optimization of the decoupling parameters would be the application of machine learning algorithms to predict the optimal choice of the decoupling parameters from the initial conditions. This could allow moving the computational burden to before the real time use of the algorithm without compromising performance.

APPENDIX
FEASIBILITY OF TIME-VARYING ACCELERATION CONSTRAINTS

This appendix contains the proof that the time-varying acceleration constraints (35) (used to account for nonzero initial accelerations) satisfy the maximum thrust constraint (14). For the purpose of simplicity, the proof will first be presented for a related set of time-varying constraints (referred to as *constant time-bound change*). The findings of this related acceleration change will then be applied to the actual time-varying acceleration constraints (35).

A. Constant Time-Bound Change

Consider the set of time-varying acceleration constraints $\overline{\ddot{x}}_{\mathrm{ct}}, \overline{\ddot{y}}_{\mathrm{ct}}, \overline{\ddot{z}}_{\mathrm{ct}}$, defined according to (36), where the slope values c_x, c_y, c_z are chosen such that all degrees of freedom reach their respective bounds $(\ddot{x}_{\max}, \ddot{y}_{\max}, \ddot{z}_{\max})$ in constant time ΔT_0, i.e.,

$$|\ddot{x}_0| + c_x \Delta T_0 = \ddot{x}_{\max} \tag{65}$$

$$|\ddot{y}_0| + c_y \Delta T_0 = \ddot{y}_{\max} \tag{66}$$

$$|\ddot{z}_0| + c_z \Delta T_0 = \ddot{z}_{\max} \tag{67}$$

with ΔT_0 defined in (37). Then the maximum acceleration bound (14) remains satisfied for all $t \in (0 \;\; \Delta T)$

$$\begin{aligned} &\overline{\ddot{x}}_{\mathrm{ct}}^2 + \overline{\ddot{y}}_{\mathrm{ct}}^2 + \left(\overline{\ddot{z}}_{\mathrm{ct}} + \mathrm{g}\right)^2 = \\ &\left(|\ddot{x}_0| + \frac{c_x}{\Delta T} t\right)^2 + \left(|\ddot{y}_0| + \frac{c_y}{\Delta T} t\right)^2 \\ &+ \left(|\ddot{z}_0| + \mathrm{g} + \frac{c_z}{\Delta T} t\right)^2 \leq a_{\max}. \end{aligned} \tag{68}$$

Proof: Without loss of generality, let the normalized time be $\tilde{t}$ such that $\Delta \tilde{T} = 1$. Rewriting (68) in terms of $(\ddot{x}_{\max}, \ddot{y}_{\max}, \ddot{z}_{\max})$ using (65)–(67) results in the condition

$$\begin{aligned} &\left(\ddot{x}_{\max} + c_x \left(\tilde{t} - 1\right)\right)^2 + \left(\ddot{y}_{\max} + c_y \left(\tilde{t} - 1\right)\right)^2 \\ &\quad + \left(\ddot{z}_{\max} + \mathrm{g} + c_z \left(\tilde{t} - 1\right)\right)^2 \leq a_{\max} \end{aligned} \tag{69}$$

which can be rewritten using (33) to be

$$\begin{aligned} &2\left(\ddot{x}_{\max} c_x + \ddot{y}_{\max} c_y + \left(\ddot{z}_{\max} + \mathrm{g}\right) c_z\right) \geq \\ &\quad \left(1 - \tilde{t}\right)\left(c_x^2 + c_y^2 + c_z^2\right). \end{aligned} \tag{70}$$

Because $0 \leq \left(1 - \tilde{t}\right) \leq 1$, a sufficient condition for the inequality to hold is

$$\left(c_x^2 + c_y^2 + c_z^2\right) \leq 2\left(\ddot{x}_{\max} c_x + \ddot{y}_{\max} c_y + \left(\ddot{z}_{\max} + \mathrm{g}\right) c_z\right). \tag{71}$$

The initial acceleration (34) can be rewritten using (65)–(67) to yield

$$\left(\ddot{x}_{\max} - c_x\right)^2 + \left(\ddot{y}_{\max} - c_y\right)^2 + \left(\ddot{z}_{\max} + \mathrm{g} - c_z\right)^2 = a_0 \tag{72}$$

with $a_0 \leq a_{\max}$ by design. Expanding the above yields the required inequality (71).

B. Time-Varying Acceleration Constraints

The time-varying acceleration constraints $\overline{\ddot{x}}, \overline{\ddot{y}}, \overline{\ddot{z}}$ presented in Section IV-C4 differ from the constant time-bound change $\overline{\ddot{x}}_{\mathrm{ct}}, \overline{\ddot{y}}_{\mathrm{ct}}, \overline{\ddot{z}}_{\mathrm{ct}}$ in the choice of c_x, c_y, c_z in the case that the initial acceleration exceeds the allowed acceleration magnitude (e.g., $|\ddot{x}_0| > \ddot{x}_{\max}$). In this case, the actual time-varying bounds decrease at the minimum jerk $-\dddot{w}_{\max}$ until the allowable acceleration (e.g., $\ddot{x}_{\max}$) is reached. To show that the inequality (68) remains satisfied, it suffices to note that through the construction of c_x, c_y, c_z in (65)–(67)

$$c_x \geq -\dddot{w}_{\max}, \qquad c_y \geq -\dddot{w}_{\max}, \qquad c_z \geq -\dddot{w}_{\max} \tag{73}$$

holds, and therefore

$$\overline{\ddot{x}}_{\mathrm{ct}} \geq \overline{\ddot{x}}, \qquad \overline{\ddot{y}}_{\mathrm{ct}} \geq \overline{\ddot{y}}, \qquad \overline{\ddot{z}}_{\mathrm{ct}} \geq \overline{\ddot{z}} \tag{74}$$

holds for all times t. The time-varying acceleration constraints must, therefore, be feasible with respect to the maximum acceleration bound (14) since the constant time-bound change is feasible.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful comments on the original version of this manuscript, especially the anonymous reviewer who highlighted (25), for which the original version of the manuscript contained a worse-performing bound.

REFERENCES

[1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Autonomous Mobile Robots*, 2nd ed. Cambridge, MA, USA: MIT Press, 2011.

[2] J. P. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *IEEE Control Syst. Mag*, vol. 28, no. 2, *28(2)*. pp. 51–64, 2008.

[3] M. B. Milam, K. Mushambi, and R. M. Murray, "A new computational approach to real-time trajectory generation for constrained mechanical systems," in *Proc. Conf. Decision Control*, 2000, pp. 845–851.

[4] T. Kroger and F. M. Wahl, "Online trajectory generation: Basic concepts for instantaneous reactions to unforeseen events," *IEEE Trans. Robot*, vol. 26, no. 1, pp. 94–111, 2010.

[5] A. Kelly and B. Nagy, "Reactive nonholonomic trajectory generation via parametric optimal control," *Int. J. Robot. Res, vol. 22, no. 7/8*, pp. 583–601, 2003.

[6] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—A survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[7] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Frandorfer, and M. Pollefeys, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Auton. Robots*, vol. 33, no. 1/2, pp. 21–39, 2012.

[8] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro UAV testbed," *IEEE Robot. Autom. Mag.*, vol. 17, no. 3, pp. 56–65, 2010.

[9] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res*, vol. 30, no. 7, pp. 846–894, 2011.

[10] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter trajectory tracking control," in *Proc. AIAA Guidance, Navigation, and Control Conf.*, 2008.

[11] I. D. Cowling, O. A. Yakimenko, and J. F. Whidborne, "A prototype of an autonomous controller for a quadrotor UAV," in *Proc. Eur. Control Conf.*, 2007, pp. 4001–4008.

[12] Y. Bouktir, M. Haddad, and T. Chettibi, "Trajectory planning for a quadrotor helicopter," in *Proc. Med. Conf. Control Autom.*, 2008, pp. 1258-1263.

[13] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. Int. Conf. Robot. Autom.*, 2011, pp. 2520–2525.

[14] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Proc. Int. Conf. Robot. Research*, 2013.

[15] P. Bouffard, A. Aswani, and C. J. Tomlin, "Learning-based model predictive control on a quadrotor: Onboard implementation and experimental results," in *Proc. Int. Conf. Robot. Autom.*, 2012.

[16] M. W. Mueller and R. D'Andrea, "A model predictive controller for quadrocopter state interception," in *Proc. Eur. Control Conf.*, 2013, pp. 1383–1389.

[17] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification," in *Proc. Int. Conf. Intell. Robot. Syst.*, 2013, pp. 3480–3486.

[18] M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control," *Auton. Robots*, vol. 33, no. 1/2, pp. 69–88, 2012.

[19] W. Van Loock, G. Pipeleers, and J. Swevers, "Time-optimal quadrotor flight," in *Proc. Eur. Control Conf.*, 2013, pp. 1788–1792.

[20] H. Hermes and G. Haynes, "On the nonlinear control problem with control appearling linearly," *J. Soc. Ind. Appl. Math. Ser. A Control*, vol. 1, no. 2, pp. 85–108, 1963.

[21] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control*. New York, NY, USA: Taylor & Francis, 1975.

[22] C. Guarino, L. Bianco, and F. Ghilardelli, "Third order system for the generation of minimum-time trajectories with asymmetric bounds on velocity, acceleration , and jerk," in *Proc. Int. Conf. Intell. Robot. Syst. Workshop Robot Motion Planning, Online, Reactive Real-time*, 2012, pp. 137–143.

[23] O. Purwin and R. D'Andrea, "Trajectory generation and control for four wheeled omnidirectional vehicles," *Robot. Auton. Syst.*, vol. 54, pp. 13–22, 2006.

[24] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," in *Proc. IFAC World Congr.*, 2011, pp. 1485–1491.

[25] Q. Jiang, D. Mellinger, C. Kappeyne, and V. Kumar, "Analysis and synthesis of multi-rotor aerial vehicles," in *Proc. Int. Des. Eng. Tech. Conf.*, 2011, pp. 711–720.

[26] P. C. Hughes, *Spacecraft Attitude Dynamics*. New York, NY, USA: Wiley, 1986.

[27] P. Pounds, R. Mahony, and P. Corke, "Modelling and control of a quadrotor robot," in *Proc. Aust. Conf. Robot. Autom.*, 2006.

[28] M. Bangura and R. Mahony, "Nonlinear dynamic modeling for high performance control of a quadrotor," in *Proc. Aust. Conf. Robot. Autom.*, 2012.

[29] J. Hahn and T. F. Edgar, "An improved method for nonlinear model reduction using balancing of empirical gramians," *Comput. Chem. Eng.*, vol. 26, no. 10, pp. 1379–1397, 2002.

[30] D. S. Bernstein, *Matrix Mathematics*. Princeton, NJ, USA: Princeton Univ. Press, 2005.

[31] D. P. Bertsekas, *Dynamic Programming and Optimal Control.* 3rd ed. Belmont, MA, USA: Athena Scientific. vol. 1, 2005.

[32] R. F. Hartl, S. P. Sethi, and R. G. Vickson, "A survey of the maximum principles for optimal control problems with state constraints," *SIAM Rev.*, vol. 37, no. 2, pp. 181–218.

[33] R. L. Burden and J. D. Faires, *Numerical Analysis*, 9th ed. Pacific Grove, CA, USA: Brooks/Cole, 2011.

[34] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*. New York, NY, USA: Academic, 2003.

[35] M. Morari and J. H. Lee, "Model predictive control: Past, present and future," *Comput. Chem. Eng.*, vol. 23, no. 4/5, pp. 667–682, 1999.

[36] S. Gal-on and M. Levy. (2012). Exploring CoreMark—A benchmark maximizing simplicity and efficacy. *The Embedded Microprocessor Benchmark Consortium*. [Online]. Available: www.eembc.org

[37] S. Lupashin, M. Hehn, M. W. Mueller, A. P. Schoellig, M. Sherback, and R. D'Andrea, "A platform for aerial robotics research and demonstration: The flying machine arena," *Mechatronics*, vol. 24, no. 1, pp. 41–54, 2014.

[38] D. Gurdan, J. Stumpf, M. Achtelik, K.-M. Doth, G. Hirzinger, and D. Rus. "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz," in *Proc. Int Conf. Robot. Autom.*, 2007, pp. 361–366.

[39] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," in *Proc. Int Conf. Robot. Autom.*, 2009, pp. 3277–3282.

[40] C. Powers, D. Mellinger, A. Kushleyev, and B. Kothmann, "Influence of aerodynamics and proximity effects in quadrotor flight," in *Proc. Int. Symp. Exp. Robot.*, 2012, pp. 289–302.

[41] F. Augugliaro, S. Lupashin, M. Hamer, C. Male, M. Hehn, M. W. Mueller, J. Willmann, F. Gramazio, M. Kohler, and R. D'Andrea, "The flight assembled architecture installation: Cooperative construction with flying machines," *IEEE Control Syst. Mag.*, vol. 34, no. 4, pp. 46–64, 2014.

[42] O. Purwin, R. D'Andrea, and J.-W. Lee, "Theory and implementation of path planning by negotiation for decentralized agents," *Robot. Auton. Syst*, vol. 56, no. 5, pp. 422–436, 2008.

Markus Hehn (S'10–M'14) received the Diplom-Ingenieur degree in mechatronics from TU Darmstadt, Darmstadt, Germany, in 2009 and the Doctor of Sciences degree from ETH Zurich, Zurich, Switzerland, in 2014.

He received a scholarship from Robert Bosch GmbH for his graduate studies and received the Jakob Ackeret prize from the Swiss Association of Aeronautical Sciences for his doctoral thesis. He has worked on optimizing the operating strategy of diesel engines to reduce emissions and fuel consumption on the characterization of component load profiles for axle-split hybrid vehicle drivetrains, and on the performance development of Formula One racing engines. His main research interests include the control and trajectory generation for multirotor vehicles during fast maneuvering, optimality-based maneuvers, multivehicle coordination, and learning algorithms.

Raffaello D'Andrea (F'10) received the B.Sc. degree in engineering science from University of Toronto, Toronto, ON, Canada, in 1991, and the M.S. and Ph.D. degrees in electrical engineering from California Institute of Technology, Pasadena, CA, USA, in 1992 and 1997, respectively.

He was an Assistant and then an Associate Professor with Cornell University, Ithaca, NY, USA, from 1997 to 2007. While on leave from Cornell University, from 2003 to 2007, he cofounded Kiva Systems, North Reading, MA, USA, where he led the systems architecture, robot design, robot navigation and coordination, and control algorithms efforts. He is currently a Professor of Dynamic Systems and Control with ETH Zurich, Zurich, Switzerland.