Minimum-Time Trajectory Generation for Quadrotors in Constrained Environments

This is the post peer-review accepted manuscript of:

S. Spedicato and G. Notarstefano, "Minimum-Time Trajectory Generation for Quadrotors in Constrained Environments," in IEEE Transactions on Control Systems Technology, vol. 26, no. 4, pp. 1335-1344, July 2018.

The published version is available online at:

https://doi.org/10.1109/ TCST.2017.2709268

Minimum-time trajectory generation for quadrotors in constrained environments

Sara Spedicato, and Giuseppe Notarstefano, *Member, IEEE*

***Abstract*—In this paper, we present a novel strategy to compute minimum-time trajectories for quadrotors in constrained environments. In particular, we consider the motion in a given flying region with obstacles and take into account the physical limitations of the vehicle. Instead of approaching the optimization problem in its standard time-parameterized formulation, the proposed strategy is based on an appealing re-formulation. Transverse coordinates, expressing the distance from a frame path, are used to parameterize the vehicle position and a spatial parameter is used as independent variable. This re-formulation allows us to (i) obtain a fixed horizon problem and (ii) easily formulate (fairly complex) position constraints. The effectiveness of the proposed strategy is proven by numerical computations on two different illustrative scenarios. Moreover, the optimal trajectory generated in the second scenario is experimentally executed with a real nano-quadrotor in order to show its feasibility.**

***Index Terms*—Minimum-time, nonlinear optimal control, aerial vehicles, trajectory optimization**

I. Introduction

Numerous applications involving Unmanned Aerial Vehicles (UAVs), and in particular quadrotors, require them to move inside areas characterized by physical boundaries, obstacles and even tight space constraints (as e.g., urban environments) in order to accomplish their robotics tasks. Such applications are, for example, structural inspections, transportation tasks, surveillance and search and rescue missions. Trajectory generation, a core step for physical task realization [1], becomes extremely challenging in this scenario. A physically realizable trajectory must satisfy (i) the (nonlinear) system dynamics, (ii) the physical limits of the vehicle, such as the maximum thrust, and (iii) the position constraints. Although safety (ensured by a feasible trajectory) is the primary requirement for all applications, trajectory optimization is becoming necessary in different application domanis. The cost to minimize can be, for example, the time to execute a maneuver (in a search and rescue scenario), the energy consumption (during long endurance missions), or the "distance" from a desired unfeasible state-input curve (during inspections). The further requirement of performance optimization poses an additional challenge in the trajectory generation problem.

The problem of computing optimal paths (or trajectories) for UAVs (e.g., [2] and [3]) has received significant attention and a number of algorithms for quadrotors have been proposed to accomplish complex tasks, e.g., landing on a moving target [4] and blind navigation in unknown populated environments [5]. Focusing on collision avoidance, two different approaches, namely reacting or planning, can be applied. The reactive approach is based on navigation laws preventing from possible collisions. It can be performed, e.g., modulating the velocity reference [6], selecting ad-hoc reference way-points [7] and defining an harmonic potential field [8]. On the contrary, the planned approach deals with a problem involving dynamics and state-input constraints with (possibly) a performance criterion to optimize. The majority of the planning algorithms regarding quadrotors, such as [9], [10], [11], [12], [13], takes advantage of the differential flatness property and relies on approximations via motion primitives. When dealing with obstacle dense environments, trajectory generation is often performed using a decoupled approach ([14], [15], [16]). In a first stage, a collision-free path is generated by sampling-based path planning algorithms, such as the Rapidly-exploring Random Tree (RRT) in [14], [15] or the Probabilistic Roadmap (PRM) in [16], and without the dynamics constraint. In a second stage, an optimal trajectory (satisfying the system dynamics) is generated from the collision-free path. Optimization techniques such as [9], [10], [11] can be used at this stage. In order to overcome the limitations due to the decoupled approach, a variant of the RRT algorithm is developed in [17], an approximated dynamics with an a-posteriori correction is used in [18] and a space-parameterized problem reformulation, suitable for modeling complex flight scenarios, is adopted in [12]. Differently from the previous works, in [19] the structure of the minimum-time trajectories is found by the Pontryagin's minimum principle. Nevertheless, position constraints are not considered. Finally, in [20] a discretized simple point-mass dynamics and approximated convex constraints are considered. The approximation of non-convex constraints into convex ones is also used in [21], in which a sequential convex programming approach is used to achieve a collision free motion for dancing quadrotors.

Our main contribution is the design of an optimization framework to generate feasible minimum-time quadrotor trajectories in structured environments as, e.g., rooms, corridors, passages, or urban areas. Our strategy computes optimal trajectories that satisfy the quadrotor nonlinear dynamics. The strategy can be applied to general models, which may be

S. Spedicato and G. Notarstefano are with the Department of Engineering, Università del Salento, Via per Monteroni, 73100 Lecce, Italy, e-mail: name.lastname@unisalento.it.

This result is part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 638992 - OPT4SMART).

A short, preliminary version of this work was presented at the IEEE Conference on Decision and Control 2016. Differences with that work include: (i) a more comprehensive treatment of the proposed strategy, (ii) an extended version of the strategy involving the computation of collision free regions shaped by obstacles, (iii) numerical computations for more general scenarios and (iv) an experimental test on a nano-quadrotor.

more complicated than the differentially flat ones. Instead of addressing the minimum-time problem in its standard free-horizon formulation, we derive a fixed-horizon reformulation in which transverse coordinates, expressing the "transverse" distance from a frame path, are used to parameterize the vehicle position. The resulting problem, having a spatial parameter as independent variable, is easier to solve than the time-parametrized one. Position constraints can be easily added into the reformulated problem by defining the constraint boundaries as a function of the spatial parameter and shaping them according to the presence of obstacles. Approximate solutions to the infinite-dimensional optimization problem are numerically computed by combining the Projection Operator Newton method for Trajectory Optimization (PRONTO) [22] with a barrier function approach [23]. This method generates trajectories in a numerically stable manner and guarantees recursive feasibility during the algorithm evolution, i.e., at each algorithm iteration a system trajectory is available. Moreover the approximated solution always satisfies the constraints since the barrier function approach is an interior function method. As an additional contribution, we present numerical computations to show the effectiveness of the proposed strategy on two challenging scenarios. In the first one, the moving space is delimited by rooms with obstacles of different shapes. In the second scenario, the constrained environment is a tubular region delimited by hula hoops. The optimal minimum-time trajectory related to this second scenario is experimentally performed on our nano-quadrotor testbed.

Our algorithm compares to the literature in the following way. The majority of works, such as [9], [10], [11], [12], [13], uses the differential flatness to avoid the integration of nonlinear differential equations, to reduce the order of the problem and to simplify the definition of constraints [9]. On the contrary, our strategy does not rely on the differential flatness hypothesis and thus it can be applied to more complex models. In the previously cited works, the optimization problem is posed in the flat output space, where outputs are approximated using motion primitives, such as polynomial functions [9], [10], [13], B-splines [11], or "convex combinations of feasible paths" [12]. The optimization variables are thus the parameters of the motion primitives. Differently from these works, we do not rely on motion primitives: the state-input trajectory is the optimization variable in our problem formulation. Similarly to the problem formulation in [24], our reformulated minimum-time problem has a spatial parameter, instead of time, as independent variable. While in [24] the maximum velocity profile (for a given path) is computed for a motorcycle model by using a quasi-static approximation of the dynamics, we optimize the whole state-input trajectory and we consider the full nonlinear dynamics of the quadrotor. Finally, other optimization strategies using the PRONTO method are [25] and [26], which aim to compute respectively minimum-energy trajectories for two-wheeled mobile robots and minimum-distance trajectories (from an unfeasible desired maneuver) for UAVs. Differently from these works, we consider a more general three-dimensional space with position constraints and we reformulate the minimum-time problem by using the transverse coordinates.

The paper is organized as follows. In Section II we present the standard formulation of the optimization problem we aim to solve. In Section III our trajectory generation strategy, based on an appealing reformulation of the problem, is described. Finally, in Section IV, we provide numerical computations and experiments, and discuss interesting features of the computed minimum-time trajectories.

II. The quadrotor minimum-time problem

We first briefly introduce the quadrotor model used in the paper and then recall the standard problem formulation.

A. Quadrotor model

The quadrotor dynamics can be described by the so called vectored-thrust dynamical model in [27], where the gravity is the only external force and the generated torque does not influence the translational dynamics, i.e.,

$$\dot{\boldsymbol{p}} = \mathbf{v} \tag{1}$$

$$\dot{\mathbf{v}} = g\boldsymbol{e}_3 - \frac{F}{m} R(\boldsymbol{\Phi})\boldsymbol{e}_3 \tag{2}$$

$$\dot{\boldsymbol{\Phi}} = J(\boldsymbol{\Phi})\boldsymbol{\omega} \tag{3}$$

$$\dot{\boldsymbol{\omega}} = -I^{-1}\hat{\boldsymbol{\omega}} I \boldsymbol{\omega} + I^{-1}\boldsymbol{\gamma}. \tag{4}$$

with $\boldsymbol{p} = [p_1\ p_2\ p_3]^T$, $\mathbf{v} = [\mathrm{v}_1\ \mathrm{v}_2\ \mathrm{v}_3]^T$, $\boldsymbol{\Phi} = [\varphi\ \theta\ \psi]^T$, where φ, θ, ψ are respectively the roll, pitch and yaw angles, and $\boldsymbol{\omega} = [p\ q\ r]^T$. The symbols in equations (1-4) are defined in the following table, where $\mathcal{F}_i$ and $\mathcal{F}_b$ respectively denote the inertial and the body frame.

TABLE I: Nomenclature

$\boldsymbol{p} \in \mathbb{R}^3$	position vector expressed in $\mathcal{F}_i$
$\mathbf{v} \in \mathbb{R}^3$	velocity vector expressed in $\mathcal{F}_i$
$\boldsymbol{\Phi} \in \mathbb{R}^3$	vector of angles (yaw-pitch-roll w.r.t. current frame)
$R(\boldsymbol{\Phi}) \in SO(3)$	rotation matrix to map vectors in $\mathcal{F}_b$ into vectors in $\mathcal{F}_i$
$\boldsymbol{\omega} \in \mathbb{R}^3$	angular rate vector expressed in $\mathcal{F}_b$
$\hat{\boldsymbol{\omega}} \in so(3)$	skew-symmetric matrix associated to $\boldsymbol{\omega}$
$J(\boldsymbol{\Phi}) \in \mathbb{R}^{3\times 3}$	matrix mapping $\boldsymbol{\omega}$ into $\dot{\boldsymbol{\Phi}}$
$m \in \mathbb{R}$	vehicle mass
$I \in \mathbb{R}^{3\times 3}$	inertia matrix
$g \in \mathbb{R}$	gravity constant
$\boldsymbol{e}_3 \in \mathbb{R}^3$	vector defined as $\boldsymbol{e}_3 := [0\ 0\ 1]^T$
$F \in \mathbb{R}$	thrust
$\boldsymbol{\gamma} \in \mathbb{R}^3$	torque vector

For the vehicle maneuvering, we adopt a cascade control scheme with an off-board position/attitude control loop and an on-board angular rate controller. Assuming that the *virtual* control input $\boldsymbol{\omega}$ is tracked by the on-board angular rate controller, we restrict our trajectory generation problem on the position/attitude subsystem (1-3), which can be written in state-space form as

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{5}$$

with state $\boldsymbol{x} = [\boldsymbol{p}^T\ \mathbf{v}^T\ \boldsymbol{\Phi}^T]^T$, input $\boldsymbol{u} = [\boldsymbol{\omega}^T\ F]^T$ and suitably defined f.

B. Quadrotor minimum-time problem: standard formulation

We deal with the following optimal control problem:

$$\begin{aligned}
\min_{\boldsymbol{x}(\cdot),\boldsymbol{u}(\cdot),T} \quad & T \\
\text{subj. to} \quad & \dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t),\boldsymbol{u}(t)), \quad \boldsymbol{x}(0) = \boldsymbol{x_0} \textit{ (dynamics)} \\
& \boldsymbol{x}(T) \in X_T \textit{ (final constraint)} \\
& |p(t)| \leq p_{max} \textit{ (roll rate)} \\
& |q(t)| \leq q_{max} \textit{ (pitch rate)} \\
& |r(t)| \leq r_{max} \textit{ (yaw rate)} \\
& 0 < F_{min} \leq F(t) \leq F_{max} \textit{ (thrust)} \\
& |\varphi(t)| \leq \varphi_{max}(t) \textit{ (roll angle)} \\
& |\theta(t)| \leq \theta_{max}(t) \textit{ (pitch angle)} \\
& |\psi(t)| \leq \psi_{max}(t) \textit{ (yaw angle)} \\
& c_{obs}(\boldsymbol{p}(t)) \leq 0 \textit{ (position constraints)},
\end{aligned} \tag{6}$$

where $X_T \subset \mathbb{R}^9$ is a desired final region, p_{max}, q_{max} and r_{max} are bounds on roll, pitch and yaw rate, respectively, F_{min} and F_{max} are lower and upper bounds on thrust, $\varphi_{max}(\cdot)$, $\theta_{max}(\cdot)$ and $\psi_{max}(\cdot)$ are bounds on roll-pitch-yaw angles, and $c_{obs} : \mathbb{R}^3 \to \mathbb{R}$ represents position constraints. The t-dependent constraints in (6) hold for all $t \in [0, T]$, with the exception of $c_{obs}(\boldsymbol{p}(t)) \leq 0$, which holds for all $t \in [0, T)$. The bounds on the angular rates avoid fast solutions. The vehicle thrust is also limited: quadrotor vehicles can only generate positive thrust and the maximum rotor speed is limited. Furthermore, constraints on roll and pitch angles are imposed into the optimization problem in order to avoid acrobatic vehicle configurations and to satisfy $\theta \neq \pm\frac{\pi}{2}$ (which makes the matrix $J(\boldsymbol{\Phi})$ in (3) always well defined). Time dependent boundaries can be used for roll and pitch constraints when the vehicle has to move through small passages. The constraint on the yaw angle may be useful in applicative scenarios in which a sensor, e.g., a camera, is provided onboard the vehicle and needs to be pointed toward a target region. The position constraints take into account physical boundaries (possibly shaped by the presence of obstacles) and may also represent GPS denied areas or spaces with limited communication.

III. Minimum-Time Trajectory Generation Strategy

In this section, we describe our strategy to compute minimum-time trajectories.

Minimum-time problem (6) is difficult to solve, since it is a constrained, free-horizon problem (time T is an optimization variable). For this reason, instead of directly designing an algorithm to solve problem (6), we provide a strategy to obtain an equivalent, but computationally more appealing, fixed-horizon formulation. In the following, we give an informal idea of the strategy steps to derive the new problem formulation. First, we define a *frame path* as a (purely geometric) curve in $\mathbb{R}^3$ used to express the quadrotor position in terms of new coordinates. That is, as depicted in Figure 1, the position is identified by the arc-length of the point on the path at minimum distance and by two transverse coordinates expressing how far the quadrotor position is from the curve. Second, we rewrite the dynamics in terms of the transverse coordinates and show that it depends on time only through the arc-length time-evolution. Thus, by using the arc-length as independent variable, we obtain a "space-dependent" transverse dynamics. Third, the time T can be expressed itself as a function of the arc-length over a fixed "spatial" horizon $[0, L]$, with L being the total length of the frame path. Thus, minimizing T can be rewritten as minimizing an integral function over the fixed spatial interval $[0, L]$. Similarly, pointwise constraints can be written in terms of the transverse coordinates and as function of the arc-length.

The resulting fixed-horizon optimal control problem is solved by using the Projection Operator Newton method (PRONTO), [22], combined with a barrier function approach to handle the constraints, [23].

We provide a detailed and formal explanation of the strategy steps in the following subsections.

A. Frame path

The first step of the strategy is the generation of an arc-length parameterized frame path $\bar{\boldsymbol{p}}_f(s)$, $\forall s \in [0, L]$, where s is the arc-length of the path and L is its total length. In the following, we denote the arc-length parameterized functions with a bar, and the derivative with respect to the arc-length with a prime, i.e., $\bar{\boldsymbol{p}}'_f(s) := d\bar{\boldsymbol{p}}_f(s)/ds$. The frame path $\bar{\boldsymbol{p}}_f(\cdot)$ has to be locally a non-intersecting C^2 curve with non-vanishing $\bar{\boldsymbol{p}}'_f(\cdot)$. Note that the frame path is only a geometric path and it is not required to satisfy the position constraints. A possibility is the computation of the frame path as a C^∞ geometric curve, e.g., using arctangent functions as in our numerical computations. More details on the frame path used for our numerical computations will be given in Section IV.

The frame path is used to parameterize the inertial position of the vehicle in the new transverse coordinates, as will be clear later. In order to define the transverse coordinates, we consider the Serret-Frenet frame, whose origin has $\bar{\boldsymbol{p}}_f(s)$ as coordinates, and defined $\forall s \in [0, L]$. In particular, the tangent, normal and bi-normal vectors, respectively $\bar{\boldsymbol{t}}(s), \bar{\boldsymbol{n}}(s), \bar{\boldsymbol{b}}(s)$, are defined, with components in the inertial frame, as

$$\bar{\boldsymbol{t}}(s) := \bar{\boldsymbol{p}}'_f(s), \tag{7}$$

$$\bar{\boldsymbol{n}}(s) := \frac{\bar{\boldsymbol{p}}''_f(s)}{\bar{k}(s)}, \tag{8}$$

$$\bar{\boldsymbol{b}}(s) := \bar{\boldsymbol{t}}(s) \times \bar{\boldsymbol{n}}(s), \tag{9}$$

where $\bar{k}(s) := \|\bar{\boldsymbol{p}}''_f(s)\|_2$ is the curvature of $\bar{\boldsymbol{p}}_f(\cdot)$ at s. Moreover, we define the rotation matrix

$$\bar{R}_{SF} := [\, \bar{\boldsymbol{t}}\ \bar{\boldsymbol{n}}\ \bar{\boldsymbol{b}}\,] \tag{10}$$

mapping vectors with components in the Serret-Frenet frame into vectors with components in the inertial frame. According to the Serret-Frenet formulas [28], the arc-length derivative of the Serret-Frenet rotation matrix is

$$\bar{R}'_{SF}(s) = \bar{R}_{SF}(s) \begin{bmatrix} 0 & -\bar{k}(s) & 0 \\ \bar{k}(s) & 0 & -\bar{\tau}(s) \\ 0 & \bar{\tau}(s) & 0 \end{bmatrix}, \tag{11}$$

where $\bar{\tau}(s) := \bar{\boldsymbol{n}}(s)\, \bar{\boldsymbol{b}}'(s)$ is the torsion of $\bar{\boldsymbol{p}}_f(\cdot)$ at s.

B. Transverse dynamics

The second step of the strategy is the derivation of the transverse dynamics by using the transverse coordinates defined with respect to the frame path $\bar{\boldsymbol{p}}_f(\cdot)$. In order to rewrite the standard dynamics (1-3) into the transverse dynamics, we proceed as follows.

First, we design a change of coordinates from the inertial position $\boldsymbol{p} \in \mathbb{R}^3$ to the transverse coordinate vector $\boldsymbol{w} \in \mathbb{R}^2$, such that $\boldsymbol{w} = [w_1\ w_2]^T$, where w_1 and w_2 are the transverse coordinates. Let us consider the quadrotor center of mass with position $\boldsymbol{p}(t)$. As depicted in Figure 1, its orthogonal projection on the frame path identifies a point with position $\bar{\boldsymbol{p}}_f(s_f(t))$, where the function $s_f : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ is

$$s_f(t) := \arg\min_{s \in \mathbb{R}_0^+} \|\boldsymbol{p}(t) - \bar{\boldsymbol{p}}_f(s)\|^2. \tag{12}$$

For simplicity, in the following we use $s_f^t := s_f(t)$ and $\dot{s}_f^t := \dot{s}_f(t)$. Note that, the minimizing arc-length is unique provided that $\bar{\boldsymbol{p}}_f(\cdot)$ is locally a non-intersecting C^2 curve with non-vanishing $\bar{\boldsymbol{p}}_f'(\cdot)$. By mapping $\boldsymbol{p} - \bar{\boldsymbol{p}}_f(s_f^t)$ into a vector with components in the Serret-Frenet frame attached to $\bar{\boldsymbol{p}}_f(s_f^t)$, we obtain

$$\boldsymbol{d} := \bar{R}_{SF}(s_f^t)^T(\boldsymbol{p} - \bar{\boldsymbol{p}}_f(s_f^t)). \tag{13}$$

Noticing that the component related to the tangent vector is always zero by construction, we define the components w_1 and w_2 of the transverse error vector $\boldsymbol{w}$ as, respectively, the second and third components of $\boldsymbol{d}$, i.e.,

$$\begin{aligned} w_1 &:= \bar{\boldsymbol{n}}(s_f^t)^T(\boldsymbol{p} - \bar{\boldsymbol{p}}_f(s_f^t)), \\ w_2 &:= \bar{\boldsymbol{b}}(s_f^t)^T(\boldsymbol{p} - \bar{\boldsymbol{p}}_f(s_f^t)), \end{aligned} \tag{14}$$

and thus obtaining

$$\boldsymbol{d} = [0\ w_1\ w_2]^T. \tag{15}$$

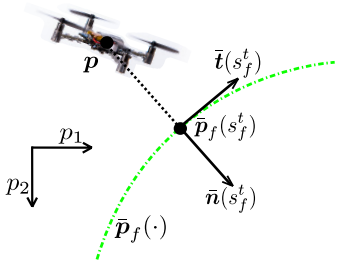


Fig. 1: Selection of the arc-length s identifying the point on the frame path at minimum distance from the quadrotor position at the time instant t.

Second, we rewrite equation (1) using $\boldsymbol{w}$ instead of $\boldsymbol{p}$. We note that, the invertible function $s_f(\cdot)$ provides a change of variables from the time t to the arc-length s. A generic arc-length function $\bar{\alpha}(\cdot)$ can be expressed as the time function $\bar{\alpha}(s_f(\cdot))$ and its time derivative is $\frac{d\bar{\alpha}(s_f(t))}{dt} = \bar{\alpha}'(s_f^t)\ \dot{s}_f^t$. Let us rewrite equation (1). By using equation (13), the position of the quadrotor center of mass $\boldsymbol{p}(t)$, at time instant t, can be written as

$$\boldsymbol{p}(t) = \bar{\boldsymbol{p}}_f(s_f^t) + \bar{R}_{SF}(s_f^t)\ \boldsymbol{d}(t). \tag{16}$$

Differentiating (16) with respect to time, since (1) holds, we get

$$\mathbf{v}(t) = \bar{\boldsymbol{p}}_f'(s_f^t)\ \dot{s}_f^t + \bar{R}_{SF}'(s_f^t)\ \dot{s}_f^t\ \boldsymbol{d}(t) + \bar{R}_{SF}(s_f^t)\ \dot{\boldsymbol{d}}(t). \tag{17}$$

Multiplying both sides of equation (17) by $\bar{R}_{SF}^T$, using (11), (15) and $\bar{\boldsymbol{p}}_f'(s_f^t) = \bar{R}_{SF}(s_f^t)[1\ 0\ 0]^T$, we get

$$\begin{bmatrix} 0 \\ \dot{w}_1(t) \\ \dot{w}_2(t) \end{bmatrix} + \dot{s}_f^t \begin{bmatrix} 1 - \bar{k}(s_f^t)w_1(t) \\ -\bar{\tau}(s_f^t)w_2(t) \\ \bar{\tau}(s_f^t)w_1(t) \end{bmatrix} - \bar{R}_{SF}^T(s_f^t)\mathbf{v}(t) = 0,$$

i.e., using (10),

$$\dot{s}_f^t = \frac{\bar{\boldsymbol{t}}(s_f^t)^T\mathbf{v}(t)}{1 - \bar{k}(s_f^t)w_1(t)} \tag{18}$$

$$\dot{w}_1(t) = \bar{\boldsymbol{n}}(s_f^t)^T\mathbf{v}(t) + \bar{\tau}(s_f^t)\dot{s}_f^t w_2(t) \tag{19}$$

$$\dot{w}_2(t) = \bar{\boldsymbol{b}}(s_f^t)^T\mathbf{v}(t) - \bar{\tau}(s_f^t)\dot{s}_f^t w_1(t). \tag{20}$$

Third and final, we rewrite equations (19), (20), (2), (3), by using the arc-length s as independent variable. Let us denote by $\bar{t}_f : \mathbb{R}_0^+ \mapsto \mathbb{R}_0^+$ the inverse function of $s_f : \mathbb{R}_0^+ \mapsto \mathbb{R}_0^+$, satisfying $t = \bar{t}_f(s_f^t)$. Due to the invertibility of $s_f(\cdot)$, a generic time function $\alpha(\cdot)$ can be expressed as the arc-length function $\alpha(\bar{t}_f(\cdot))$ and, defining $\bar{\alpha} := \alpha \circ \bar{t}_f$, we have $\alpha(t) = \bar{\alpha}(s_f^t)$. In particular,

$$\boldsymbol{w}(t) = \bar{\boldsymbol{w}}(s_f^t), \quad \mathbf{v}(t) = \bar{\mathbf{v}}(s_f^t), \quad \boldsymbol{\Phi}(t) = \bar{\boldsymbol{\Phi}}(s_f^t), \tag{21}$$

$$\boldsymbol{\omega}(t) = \bar{\boldsymbol{\omega}}(s_f^t), \quad F(t) = \bar{F}(s_f^t). \tag{22}$$

Deriving with respect to time equations (21), we get

$$\dot{\boldsymbol{w}}(t) = \bar{\boldsymbol{w}}'(s_f^t)\dot{s}_f^t, \quad \dot{\mathbf{v}}(t) = \bar{\mathbf{v}}'(s_f^t)\dot{s}_f^t, \quad \dot{\boldsymbol{\Phi}}(t) = \bar{\boldsymbol{\Phi}}'(s_f^t)\dot{s}_f^t,$$

and equations (19),(20),(2),(3) become

$$\begin{aligned} \bar{w}_1'(s_f^t) &= \bar{\boldsymbol{n}}(s_f^t)^T\mathbf{v}(t)\ \frac{1}{\dot{s}_f^t} + \bar{\tau}(s_f^t)w_2(t), \\ \bar{w}_2'(s_f^t) &= \bar{\boldsymbol{b}}(s_f^t)^T\mathbf{v}(t)\ \frac{1}{\dot{s}_f^t} - \bar{\tau}(s_f^t)w_1(t), \\ \bar{\mathbf{v}}'(s_f^t) &= (g\boldsymbol{e}_3 - \frac{F(t)}{m}R(\boldsymbol{\Phi}(t))\boldsymbol{e}_3)\ \frac{1}{\dot{s}_f^t}, \\ \bar{\boldsymbol{\Phi}}'(s_f^t) &= J(\boldsymbol{\Phi}(t))\boldsymbol{\omega}(t)\ \frac{1}{\dot{s}_f^t}. \end{aligned} \tag{23}$$

Using (18), (21) and (22), equations (23) depend on time only through the variable s_f^t. Thus, we can rewrite the dynamics in the arc-length, $s \in [0, L]$, domain. Formally, considering s as the independent variable, we get the *transverse dynamics*

$$\begin{aligned} \bar{w}_1' &= \bar{\boldsymbol{n}}^T\bar{\mathbf{v}}\ \frac{1 - \bar{k}\bar{w}_1}{\bar{\boldsymbol{t}}^T\bar{\mathbf{v}}} + \bar{\tau}\bar{w}_2, \\ \bar{w}_2' &= \bar{\boldsymbol{b}}^T\bar{\mathbf{v}}\ \frac{1 - \bar{k}\bar{w}_1}{\bar{\boldsymbol{t}}^T\bar{\mathbf{v}}} - \bar{\tau}\bar{w}_1, \\ \bar{\mathbf{v}}' &= (g\boldsymbol{e}_3 - \frac{\bar{F}}{m}R(\bar{\boldsymbol{\Phi}})\boldsymbol{e}_3)\ \frac{1 - \bar{k}\bar{w}_1}{\bar{\boldsymbol{t}}^T\bar{\mathbf{v}}}, \\ \bar{\boldsymbol{\Phi}}' &= J(\bar{\boldsymbol{\Phi}})\bar{\boldsymbol{\omega}}\ \frac{1 - \bar{k}\bar{w}_1}{\bar{\boldsymbol{t}}^T\bar{\mathbf{v}}}. \end{aligned} \tag{24}$$

Note that the dependence by s is omitted for simplicity. Equations (24) can be written in state-space form as

$$\bar{\boldsymbol{x}}_w'(s) = \bar{f}(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s)), \tag{25}$$

with state $\bar{\boldsymbol{x}}_w = [\bar{\boldsymbol{w}}^T \ \bar{\mathbf{v}}^T \ \bar{\boldsymbol{\Phi}}^T]^T$, input $\bar{\boldsymbol{u}} = [\bar{\boldsymbol{\omega}}^T \ \bar{F}]^T$ and suitable $\bar{f}$.

Remark 1: The general theory regarding the transverse coordinates is introduced in [29] and used to design a maneuver regulation controller for a bi-dimensional case in [30]. Differently from [30], we use the transverse coordinates in a more general three-dimensional case and in order to develop a trajectory optimization strategy rather than a controller.

C. Arc-length parameterization of cost and constraints

The third step of the strategy consists into the reformulation of cost and constraints in problem (6) by using the new (arc-length dependent) variables $\bar{\boldsymbol{x}}_w$ and $\bar{\boldsymbol{u}}$.

The cost functional in (6), i.e., $T = \int_0^T 1 \ dt$, is rewritten into an arc-length parameterization by considering the change of variable from t to s, i.e.,

$$\int_0^T 1 \ dt = \int_{s_f(0)}^{s_f(T)} \bar{t}_f^{'}(s) \ ds.$$

Since $\frac{d\bar{t}_f(s_f(t))}{dt} = \bar{t}_f^{'}(s_f^t)\dot{s}_f^t$ and $\bar{t}_f(s_f^t) = t$, we get

$$\bar{t}_f^{'}(s_f^t) = 1/\dot{s}_f^t \tag{26}$$

with $\dot{s}_f^t$ as in (18). Since $w_1(t) = \bar{w}_1(s_f^t)$ and $\mathbf{v}(t) = \bar{\mathbf{v}}(s_f^t)$, as in (21), equation (26) can be written as

$$\bar{t}_f^{'}(s_f^t) = \frac{1 - \bar{k}(s_f^t)\bar{w}_1(s_f^t)}{\bar{\boldsymbol{t}}(s_f^t)^T\bar{\mathbf{v}}(s_f^t)}, \tag{27}$$

where all the variables depend on time only through s_f^t. Thus, we can rewrite (27) in the arc-length, $s \in [0, L]$, domain, obtaining

$$\bar{t}_f^{'}(s) = \frac{1 - \bar{k}(s)\bar{w}_1(s)}{\bar{\boldsymbol{t}}(s)^T\bar{\mathbf{v}}(s)}. \tag{28}$$

Finally, since $s_f(0) = 0$, $s_f(T) = L$, and (28) holds, we rewrite the cost functional in (6) as

$$\int_0^L \frac{1 - \bar{k}(s)\bar{w}_1(s)}{\bar{\boldsymbol{t}}(s)^T\bar{\mathbf{v}}(s)} \ ds. \tag{29}$$

Notice that, according to (29), the hypothesis $\bar{\boldsymbol{t}}(s)^T\bar{\mathbf{v}}(s) \neq 0$, has to be satisfied $\forall s \in [0, L]$, i.e., the velocity projected on the tangent vector of the frame path has to be not null.

The constraints in (6) are rewritten into an arc-length parameterization suitable to apply the barrier function approach [23]. The constraint $\boldsymbol{x}(T) \in \boldsymbol{X}_T$ is written in the form

$$c_f(\bar{\boldsymbol{x}}_w(L)) \leq 0, \tag{30}$$

with scalar components

$$c_{f,i}(\bar{x}_{w_i}(L)) = \Big(\frac{2\,\bar{x}_{w_i}(L) - (\bar{x}_{w_i,max} + \bar{x}_{w_i,min})}{(\bar{x}_{w_i,max} - \bar{x}_{w_i,min})}\Big)^2 - 1, \tag{31}$$

$\forall i = 1, ..., 8$, where $\bar{x}_{w_i}$ is the i-th component of $\bar{\boldsymbol{x}}_w$, and $\bar{x}_{w_i,min}$ and $\bar{x}_{w_i,max}$ are the bounds on the final states. The constraints on the angular rates, thrust and roll-pitch-yaw angles are rewritten by using equations (21), (22) and reparameterizing the time-dependent bounds $\varphi_{max}(t)$, $\theta_{max}(t)$ and $\psi_{max}(t)$, $\forall t \in [0, T]$, by the arc-length s. Thus, we have

$$\begin{aligned}
&\Big(\frac{\bar{p}(s)}{p_{max}}\Big)^2 - 1 \leq 0, \quad \Big(\frac{\bar{q}(s)}{q_{max}}\Big)^2 - 1 \leq 0, \quad \Big(\frac{\bar{r}(s)}{r_{max}}\Big)^2 - 1 \leq 0, \\
&\Big(\frac{\bar{\varphi}(s)}{\bar{\varphi}_{max}(s)}\Big)^2 - 1 \leq 0, \Big(\frac{\bar{\theta}(s)}{\bar{\theta}_{max}(s)}\Big)^2 - 1 \leq 0, \Big(\frac{\bar{\psi}(s)}{\bar{\psi}_{max}(s)}\Big)^2 - 1 \leq 0, \\
&\Big(\frac{2\bar{F}(s) - (F_{max} + F_{min})}{(F_{max} - F_{min})}\Big)^2 - 1 \leq 0.
\end{aligned} \tag{32}$$

As regards the position constraints $c_{obs}(\boldsymbol{p}(t)) \leq 0$, they are written in the generic form

$$\bar{c}_{obs}(\bar{w}_1(s), \bar{w}_2(s)) \leq 0, \tag{33}$$

which can be particularized according to the shape of the flying region. For environments with circular sections, the inequality (33) becomes

$$\Big(\frac{\sqrt{\bar{w}_1^2(s) + \bar{w}_2^2(s)}}{\bar{r}_{obs}(s)}\Big)^2 - 1 \leq 0, \tag{34}$$

where $\bar{r}_{obs}(s)$ identifies the radius of the circular boundary at a given arc-length s. For environments with rectangular sections, the inequality (33) becomes

$$\Big(\frac{2\bar{w}_i(s) - (\bar{w}_{i,max}(s) + \bar{w}_{i,min}(s))}{(\bar{w}_{i,max}(s) - \bar{w}_{i,min}(s))}\Big)^2 - 1 \leq 0, \tag{35}$$

$\forall i = 1, 2$, where $\bar{w}_{i,min}(s)$ and $\bar{w}_{i,max}(s)$ are the lower and upper bounds at a given arc-length s, defining the boundaries of the region. The constraint boundaries are arc-length functions suitable to model fairly complex regions. They represent the physical boundary of a region and they can be shaped in order to take into account the presence of obstacles attached to the boundary. As an illustrative example, let us consider the environment with rectangular sections depicted in Figure 2. An obstacle restricts the collision-free space inside the physical boundary of the region.

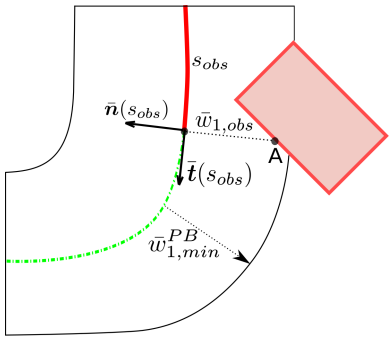


Fig. 2: Representation of $\bar{w}_{1,obs}$ related to a point A on the obstacle boundary. The frame path is depicted in red (portion identifying s_{obs}) and dot-dashed green.

Let us denote by $\bar{w}^{PB}_{i,max}(s)$ and $\bar{w}^{PB}_{i,min}(s)$ the (respectively) positive and negative distance of the physical boundary from the frame path at a given arc-length s. We first set $\bar{w}_{i,min}(s) = \bar{w}^{PB}_{i,min}(s)$ and $\bar{w}_{i,max}(s) = \bar{w}^{PB}_{i,max}(s)$, $\forall s \in [0, L]$. Then, in

order to take into account the obstacle, we suitably restrict the bounds as follows. Let us consider a point A on the boundary surface of the obstacle. Let $\boldsymbol{p}_{obs}$ be the position of point A with components in the inertial frame. We map $\boldsymbol{p}_{obs}$ in the transverse coordinate vector $\bar{\boldsymbol{w}}_{obs}$. First, we select the arc-length on the frame path, identifying the point at minimum distance from A, as

$$s_{obs} := \arg\min_{s\in\mathbb{R}_0^+} \|\boldsymbol{p}_{obs} - \bar{\boldsymbol{p}}_f(s)\|^2. \tag{36}$$

Second, we map $\boldsymbol{p}_{obs} - \bar{\boldsymbol{p}}_f(s_{obs})$ into a vector with components in the Serret-Frenet frame attached to the point identified by s_{obs}, obtaining

$$\bar{w}_{1,obs} = \bar{\boldsymbol{n}}^T(s_{obs})(\boldsymbol{p}_{obs} - \bar{\boldsymbol{p}}_f(s_{obs})), \tag{37}$$

$$\bar{w}_{2,obs} = \bar{\boldsymbol{b}}^T(s_{obs})(\boldsymbol{p}_{obs} - \bar{\boldsymbol{p}}_f(s_{obs})). \tag{38}$$

Since, according to the particular scenario, the obstacle only affects the function $\bar{w}_{1,min}(\cdot)$, we update

$$\bar{w}_{1,min}(s_{obs}) = \max\{\bar{w}^{PB}_{1,min}(s_{obs}), \bar{w}_{1,obs}\}.$$

D. Equivalent minimum-time formulation and optimal control solver

The minimum-time problem (6) is reformulated in the new (arc-length dependent) variables $\bar{\boldsymbol{x}}_w$ and $\bar{\boldsymbol{u}}$, by using the cost (29), the transverse dynamics (25) and the constraints (30), (32), and (33). Denoting by $c(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s)) \le 0,\ \forall s \in [0, L]$, the constraints (32) and (33) in vectorial form, the reformulated problem is

$$\begin{aligned} \min_{\bar{\boldsymbol{x}}_w(\cdot),\bar{\boldsymbol{u}}(\cdot)} & \int_0^L \frac{1-\bar{k}(s)\bar{w}_1(s)}{\bar{\boldsymbol{t}}(s)^T\bar{\boldsymbol{v}}(s)}\, ds, \\ \text{subj. to } & \bar{\boldsymbol{x}}_w'(s) = \bar{f}(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s)), \quad \bar{\boldsymbol{x}}_w(0) = \boldsymbol{x}_{w0}, \\ & c_f(\bar{\boldsymbol{x}}_w(L)) \le 0, \\ & c(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s)) \le 0,\ \forall s \in [0, L]. \end{aligned} \tag{39}$$

Note that the fixed horizon problem (39) is equivalent to (6) since trajectories solving (39) can be mapped into trajectories solving (6).

In order to solve problem (39), we use a combination of the PRojection Operator based Newton method for Trajectory Optimization (PRONTO) [22] with a barrier function approach [23]. We relax state-input constraints by adding them in the cost functional, i.e., we consider the problem

$$\begin{aligned} \min_{\bar{\boldsymbol{x}}_w(\cdot),\bar{\boldsymbol{u}}(\cdot)} & \int_0^L \Big(\frac{1-\bar{k}(s)\bar{w}_1(s)}{\bar{\boldsymbol{t}}(s)^T\bar{\boldsymbol{v}}(s)} + \epsilon \sum_j \beta_\nu(-c_j(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s))) \Big) ds \\ & + \ \epsilon_f \sum_i \beta_{\nu_f}(-c_{f,i}(\bar{\boldsymbol{x}}_w(L))), \\ \text{subj. to } & \bar{\boldsymbol{x}}_w'(s) = \bar{f}(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s)), \quad \forall s \in [0, L], \\ & \bar{\boldsymbol{x}}_w(0) = \boldsymbol{x}_{w0}. \end{aligned} \tag{40}$$

where ϵ and ϵ_f are positive parameters and $\beta_\ell(\cdot)$, $\ell \in \{\nu, \nu_f\}$, is a function depending on the parameter ℓ and defined as

$$\beta_\ell(x) := \begin{cases} -\log(x) & x > \ell, \\ -\log(\ell) + \frac{1}{2}\left[(\frac{x-2\ell}{\ell})^2 - 1\right] & x \le \ell. \end{cases}$$

Let an initial trajectory for the initialization of the algorithm be given. The strategy to find an approximated solution to (39) can be summarized as follows. Problem (40) is iteratively solved by reducing the parameters $\epsilon, \nu, \epsilon_f$ and ν_f at each iteration, and thus pushing the trajectory towards the constraint boundaries. Each instance of problem (40) is solved by means of the PRONTO algorithm described in Appendix A.

E. Summary of the strategy

A pseudo code of the whole strategy to compute minimum time trajectories is reported in the following table (Algorithm 1). We denote with $(\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^0$ the initial trajectory to initialize the algorithm and with `PRONTO` the PRojection Operator based Newton method for Trajectory Optimization routine that, given a trajectory $(\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^{i-1}$, computes the solution $(\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^i$ to problem (40), i.e., $(\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^i = \texttt{PRONTO}((\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^{i-1})$.

Algorithm 1 Minimum-time strategy

Given: initial condition $\boldsymbol{x}_0$, final desired region X_T, bounds $p_{max}, q_{max}, r_{max}, f_{min}, f_{max}, \varphi_{max}(\cdot), \theta_{max}(\cdot), \psi_{max}(\cdot)$, and the dynamic model (5)

A. Frame path

generate $\bar{\boldsymbol{p}}_f(s),\ \forall s \in [0, L]$

compute

- tangent, normal and binormal vectors $\bar{\boldsymbol{t}}(s) = \bar{\boldsymbol{p}}_f'(s), \quad \bar{\boldsymbol{n}}(s) = \frac{\bar{\boldsymbol{p}}_f''(s)}{\|\bar{\boldsymbol{p}}_f'(s)\|_2}, \quad \bar{\boldsymbol{b}}(s) = \bar{\boldsymbol{t}}(s) \times \bar{\boldsymbol{n}}(s)$
- curvature $\bar{k}(s) = \|\bar{\boldsymbol{p}}_f''(s)\|_2$
- torsion $\bar{\tau}(s) = \bar{\boldsymbol{n}}(s)\, \bar{\boldsymbol{b}}'(s)$

B. Transverse dynamics

set-up transverse dynamics (24)

C. Cost and constraints

set-up cost $\int_0^L \frac{1-\bar{k}(s)\bar{w}_1(s)}{\bar{\boldsymbol{t}}(s)^T\bar{\boldsymbol{v}}(s)}\, ds$

set-up constraints (30) and (32)

define boundary constraints by using (34) and/or (35)

E. Numerical solution to (39)

compute initial trajectory $(\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^0$

set $\epsilon = 1$, $\epsilon_f = 1$, $\nu = 1$, $\nu_f = 1$

for $i = 1, 2 \ldots$ **do**

compute: $(\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^i = \texttt{PRONTO}((\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^{i-1})$

update: ϵ, ϵ_f, ν, ν_f

end for

Output: $(\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))_{opt} = (\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))^i$

IV. Numerical computations

In this section, we present numerical computations and experimental tests on a nano-quadrotor with mass $m = 0.0325$ kg, in order to show the effectiveness of the proposed strategy. First, we consider a scenario with two obstacles: a parallelepiped and a cylinder, as depicted in Figure 3a. Second, we consider an experimental scenario and we show the results related to the execution of the optimal trajectory using our maneuver regulation control scheme [31].

A. Rooms with obstacles

The first scenario is as follows. The vehicle has to move from one room to another through a narrow corridor. There is a parallelepiped in the first room and a cylinder in the second room. As an additional requirement, the quadrotor must reach a neighborhood of $\boldsymbol{x}_{w0}$ at the end of its motion. In order to fulfil this objective, we consider the final constraint (30) with $c_{f,i}(\bar{x}_{w_i}(L))$ as in (31), where $\bar{x}_{w_i,min} = x_{w_i,0} - \text{tol}_i$, $\bar{x}_{w_i,max} = x_{w_i,0} + \text{tol}_i$, tol_i is a given tolerance and $x_{w_i,0}$ is the i-th component of $\boldsymbol{x}_{w0}$. Results are depicted in Figures 3, 4, 5. The initial trajectory is depicted in dot-dashed green, intermediate trajectories in dotted black and the minimum-time trajectory in solid blue. Collision-free boundaries are depicted in grey and remaining state-input constraints in dashed red.

We choose as frame path a C^∞ curve on the $\bar{p}_1 - \bar{p}_2$ plane with constant binormal vector $\bar{\boldsymbol{b}} = [0\ 0\ 1]^T$ and curvature

$$\bar{k}(s) = \frac{1}{5} \frac{\tanh(s-5) - \tanh(s-5(1+\frac{\pi}{2}))}{\max(\tanh(s-5) - \tanh(s-5(1+\frac{\pi}{2})))}.$$

The collision free region is defined by constraint (35) where obstacle boundaries $\bar{w}_{i,min}(\cdot)$ and $\bar{w}_{i,max}(\cdot)$, $i = 1, 2$, are chosen as follows. Functions $\bar{w}_{1,max}$ and $\bar{w}_{2,max}$ are not affected by obstacles. As depicted in Figures 3b and 3c, $\bar{w}_{1,max}(\cdot)$ and $\bar{w}_{2,max}(\cdot)$ are obtained using sigmoid functions with values varying from 2 m to 0.25 m. Functions $\bar{w}_{1,min}$ and $\bar{w}_{2,min}$ are affected by obstacles. In order to model the obstacles, we consider the position of the obstacle boundary as a function of its arc-length. We choose 10^{-3} as discretization step for the arc-length and for every value of the boundary position $\boldsymbol{p}_{obs}$, we compute s_{obs} and $\bar{w}_{1,obs}, \bar{w}_{2,obs}$ by using equations (36), (37) and (38), respectively. Thus, in order to define $\bar{w}_{1,min}$ and $\bar{w}_{2,min}$, we first set $\bar{w}_{1,min}(s) = -\bar{w}_{1,max}(s)$ and $\bar{w}_{2,min}(s) = -\bar{w}_{2,max}(s)$, $\forall s \in [0, L]$. Second, for each s^R_{obs} and $\bar{w}^R_{1,obs}$ related to a point R on the parallelepiped, we update

$$\bar{w}_{1,min}(s^R_{obs}) = \max\{-\bar{w}_{1,max}(s^R_{obs}), \bar{w}^R_{1,obs}\}.$$

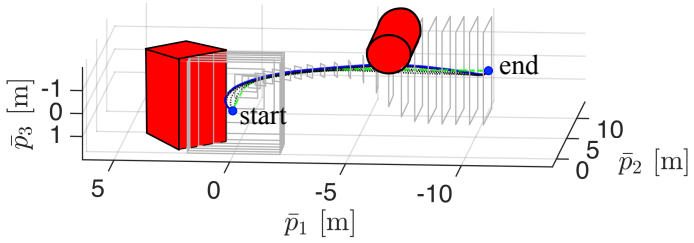
Third and final, for each s^C_{obs} and $\bar{w}^C_{2,obs}$ related to a point C on the cylinder, we update

$$\bar{w}_{2,min}(s^C_{obs}) = \max\{-\bar{w}_{2,max}(s^C_{obs}), \bar{w}^C_{2,obs}\}.$$

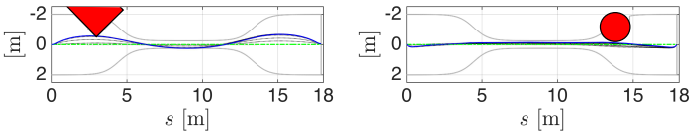
We choose the initial trajectory as follows. We set the frame path as the position, a velocity module of 0.5 m/s along the curve and a zero yaw angle. The remaining initial states and inputs are computed by using the differential flatness of the quadrotor dynamics [10]. It is worth noting that the position part of the initial trajectory does not have to necessarily match the frame path. Also, the initial trajectory could be alternatively computed through the projection of a state-input curve by using the projection operator (41) described in Appendix A, instead of using the differential flatness.

Having the initial trajectory in hand, we run the algorithm to numerically compute solutions. Note that the PRONTO method (described in Appendix A) is designed considering an s-dependent continuous dynamics. In order to implement it by using a numerical toolbox (Matlab), we consider a suitable tolerance. We choose 10^{-3} as discretization step on s and we use the tolerance of the Matlab solver to integrate the differential equations. Each intermediate optimal trajectory is computed by solving the optimization problem (40) with constant values of the parameters ϵ, ν, ϵ_f and ν_f. We start with $\epsilon = 1$, $\nu = 1$, $\epsilon_f = 1$, $\nu_f = 1$ and, following a suitable heuristic, we decrease them at each iteration. Since the algorithm operates in an interior point fashion, intermediate trajectories are all feasible and are pushed to the constraint boundaries when $\epsilon, \nu, \epsilon_f, \nu_f$ are decreased.

As regards the minimum-time trajectory, the maneuver is performed in 3.57 s and the path touches the constraint boundaries when the vehicle is inside the corridor and in the proximity of obstacles (Figures 3b and 3c). The velocity $\bar{\boldsymbol{t}}^T \bar{\mathbf{v}}$ (Figure 4a) reaches a peak of about 8.5 m/s in the middle of the path and approaches the final desired value at the end. Velocities $\bar{\boldsymbol{n}}^T \bar{\mathbf{v}}$ and $\bar{\boldsymbol{b}}^T \bar{\mathbf{v}}$ (Figures 4c and 4e, respectively) are between -2.0 m/s and 2.0 m/s. Roll and pitch angles (Figures 4b, 4d, respectively) do not touch constraint boundaries and alternate positive and negative values between -50 deg and 50 deg. The yaw angle has values between -20 deg and 50 deg (Figure 4f). As regards the inputs, while constraints on roll and pitch rates (Figures 5a, 5b, respectively) are always active, yaw rate and thrust (Figures 5c and 5d, respectively) alternate intervals with active and inactive constraints. Furthermore, note that the final state reaches a neighborhood of the initial state, satisfying $||\bar{\boldsymbol{x}}_w(L) - \boldsymbol{x}_{w0}|| < 0.07$.



(a) Path: 3D view

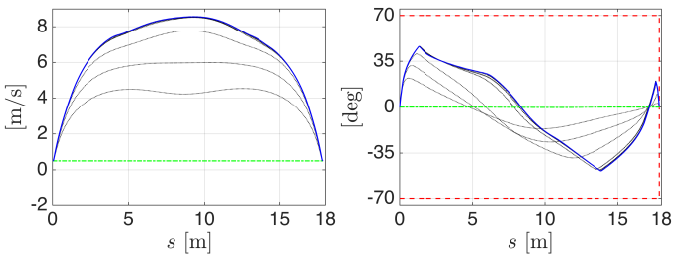


(b) Transverse coordinate $\bar{w}_1$ (c) Transverse coordinate $\bar{w}_2$

Fig. 3: Path and transverse coordinates. Initial (dot-dashed green), intermediate (dotted black) and minimum-time (solid blue) trajectory. Constraint boundaries are depicted in grey.

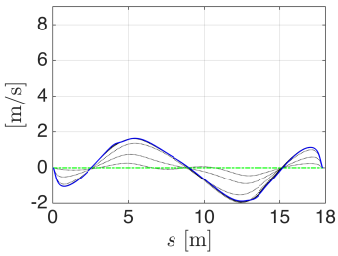
B. Tubular passage

As a second test, we consider a region delimited by hula hoops as constrained environment. First, we compute a minimum-time trajectory through our optimization strategy and second, we experimentally execute the minimum-time trajectory on the CrazyFlie nano-quadrotor (https://www.bitcraze.io/crazyflie/), by using a suitable controller. We invite the reader to watch the attached video related to this experiment.

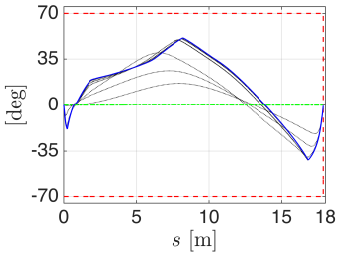


(a) Velocity $\bar{\boldsymbol{t}}^T\bar{\mathbf{v}}$

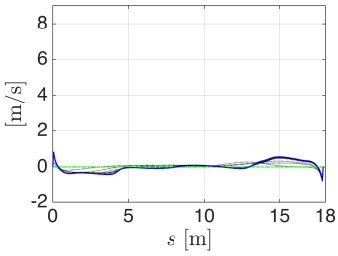
(b) Roll angle $\bar{\varphi}$



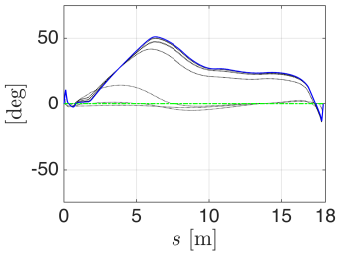
(c) Velocity $\bar{\boldsymbol{n}}^T\bar{\mathbf{v}}$



(d) Pitch angle $\bar{\theta}$

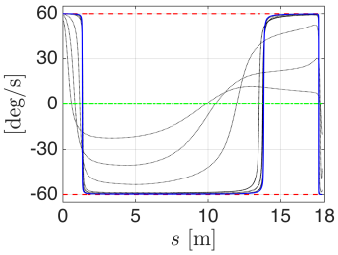


(e) Velocity $\bar{\boldsymbol{b}}^T\bar{\mathbf{v}}$

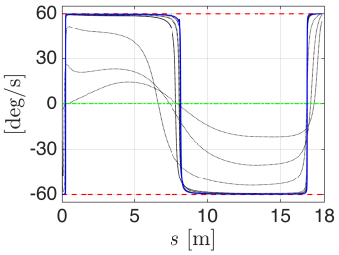


(f) Yaw angle $\bar{\psi}$

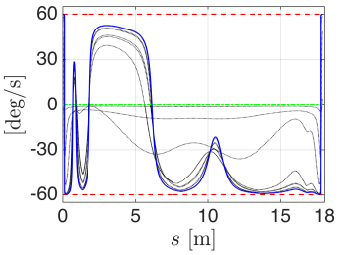
Fig. 4: Velocities and angles. Initial (dot-dashed green), intermediate (dotted black) and minimum-time (solid blue) trajectory. Constraint boundaries are depicted in red.



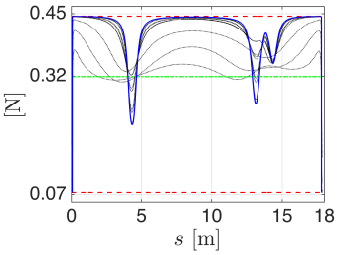
(a) Roll rate $\bar{p}$



(b) Pitch rate $\bar{q}$



(c) Yaw rate $\bar{r}$



(d) Thrust $\bar{f}$

Fig. 5: Inputs. Initial (dot-dashed green), intermediate (dotted black) and minimum-time (solid blue) trajectory. Constraint boundaries are depicted in dashed red.

We set-up the optimization algorithm as follows. We approximate the collision free region as a tube with circular section. We choose as frame path a curve on the $\bar{p}_2 - \bar{p}_3$ plane with constant binormal vector $\bar{\boldsymbol{b}} = [1\ 0\ 0]^T$ and curvature

$$\bar{k}(s) = \frac{\frac{1}{1+e^{-8(s-2.27)}} - \frac{1}{1+e^{-8(s-3.67)}}}{\max(\frac{1}{1+e^{-8(s-2.27)}} - \frac{1}{1+e^{-8(s-3.67)}})}.$$

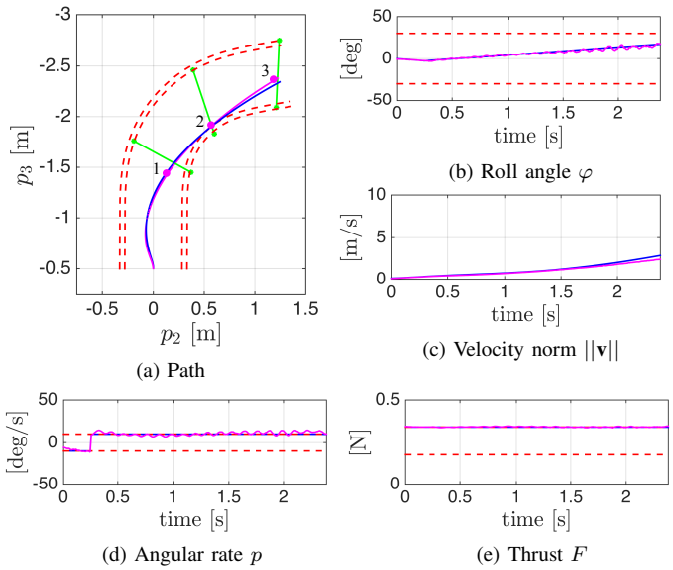
Moreover, we consider the constraint (34) with constant $\bar{r}_{obs} = r_{hh} - l - e_p$, where $r_{hh} = 0.33$ m is the hula hoop radius, $l = 0.04$ m is the distance between the quadrotor center of mass and its propellers and $e_p = 0.01$ m is the estimated position error arising during control.

As regards input constraints, we impose, for safety reasons, more severe bounds than the ones required by the physical vehicle limitations. In this way, we also ensure that the "experimental" trajectory remains feasible although the imperfect tracking of desired inputs by actual values (naturally arising during control). We choose $p_{min} = -15$ deg/s and $p_{max} = 15$ deg/s for the roll rate and $F_{min} = 0.1779$ N and $F_{max} = 0.3411$ N for thrust.

By using our minimum-time strategy, we obtain the following result. The optimal trajectory, performed in 2.38 s, is depicted in Figure 6 (solid blue). Constraint boundaries are depicted in dashed red and the hula hoops are depicted in solid green. The optimal path (blue line in Figure 6a) first takes negative values of p_2 until changing direction toward positive p_2 values, touching the constraint boundary in the proximity of the maximum curvature of the tube, and staying in the middle of the feasibility region at the end. The roll angle (blue line in Figure 6b) decreases in order to push the vehicle to negative p_2 values and then it monotonically increases during the remaining time interval. The velocity module (blue line in Figure 6c) always increases, as we expect for a minimum-time trajectory. As regards the inputs, in the beginning, the angular rate p (blue line in Figure 6d) stays on the lower bound and then it switches to the upper bound. The thrust F (blue line in Figure 6e) always takes the upper bound.

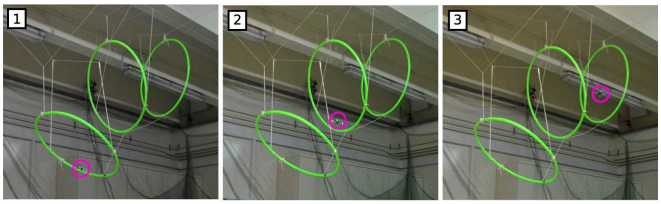
We execute the computed minimum-time trajectory on the CrazyFlie nano-quadrotor by using the closed-loop, maneuver regulation controller developed in [31], in which the minimum-time trajectory is used for the desired maneuver. The maneuver regulation controller computes thrust and angular rate virtual inputs, which are tracked by the standard off-the-shelf angular rate controller provided on board the CrazyFlie. The actual (experimental) trajectory performed using our maneuver regulation controller is depicted in Figure 6 in solid magenta. Snapshots of the experiment are depicted in Figure 6f. As expected, the quadrotor passes close to the second hula hoop maintaining the distance imposed by the restrictive constraints in the optimization problem. The actual velocity does not perfectly match (at higher velocities) the desired one, due to the unmodeled drag effect. Since the vehicle is asked to follow the desired thrust, the actual velocity becomes lower than the desired one because of the opposing aerodynamic force. The experiment shows the actual feasibility of the optimal trajec-

tory and also reveals that a more accurate model including aerodynamic effects would improve the control performance.



(a) Path (b) Roll angle φ (c) Velocity norm $||\mathbf{v}||$

(d) Angular rate p (e) Thrust F



(f) Experiment snapshots

Fig. 6: Experimental test. Desired trajectory (blue) and actual trajectory (magenta). Constraint boundaries are depicted in dashed red. Hula hoops are depicted in solid green.

V. Conclusion

In this paper, we have presented a strategy to address the minimum-time problem for quadrotors in constrained environments. Our approach consists of: (i) generating a frame path, (ii) expressing the quadrotor dynamics in a new set of coordinates "transverse" with respect to that path, and (iii) redefining cost and constraints in the new coordinates. Thus, we obtain a reformulation of the problem, which we solve by combining the PRONTO algorithm with a barrier function approach. Numerical computations on two challenging scenarios prove the effectiveness of the strategy and allow us to show interesting dynamic capabilities of the vehicle. Moreover, the experimental test of the second scenario shows the feasibility of the computed trajectory. As a future work, we aim at extending our strategy to a scenario with moving obstacles. Challenges to be addressed include how to combine trajectory generation and control, and how to take into account a fast integration of the dynamics for realtime computation.

Appendix A
Projection Operator Newton Method

Here, we provide a brief description of the PRONTO algorithm [22]. The PRONTO algorithm is based on a properly designed *projection operator* $\mathcal{P} : \xi_c \to \xi$, mapping a state-control curve $\xi_c = (\bar{\boldsymbol{x}}_{w,c}(\cdot), \bar{\boldsymbol{u}}_c(\cdot))$ into a system trajectory $\xi = (\bar{\boldsymbol{x}}_w(\cdot), \bar{\boldsymbol{u}}(\cdot))$, by the nonlinear feedback system

$$\begin{aligned} \bar{\boldsymbol{x}}'_w(s) &= \bar{f}(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s)), \quad \bar{\boldsymbol{x}}_w(0) = \boldsymbol{x}_{w0}, \\ \bar{\boldsymbol{u}}(s) &= \bar{\boldsymbol{u}}_c(s) + \bar{K}(s)(\bar{\boldsymbol{x}}_{w,c}(s) - \bar{\boldsymbol{x}}_w(s)), \end{aligned} \tag{41}$$

where the feedback gain $\bar{K}(\cdot)$ is designed by solving a suitable linear quadratic optimal control problem on the linearized dynamics of (25) about the trajectory ξ. Note that the feedback gain $\bar{K}(\cdot)$ is only used to define the projection operator and it is not related to the controller used to execute the optimal trajectory in our experimental test. The projection operator is used to convert the dynamically constrained optimization problem (40) into the unconstrained problem

$$\min_{\xi} g(\xi; \bar{k}), \tag{42}$$

where $g(\xi; \bar{k}) = h(\mathcal{P}(\xi); \bar{k})$, and $h(\xi; \bar{k}) := \int_0^L (\frac{1-\bar{k}(s)\bar{w}_1(s)}{\bar{\boldsymbol{t}}^T(s)\bar{\boldsymbol{v}}(s)} + \epsilon \sum_j \beta_\nu(-c_j(\bar{\boldsymbol{x}}_w(s), \bar{\boldsymbol{u}}(s))))ds + \epsilon_f \sum_i \beta_{\nu_f}(-c_{f,i}(\bar{\boldsymbol{x}}_w(L)))$. Then, using an (infinite dimensional) Newton descent method, a local minimizer of (42) is computed iteratively. Given the current trajectory iterate ξ_i, the search direction ζ_i is obtained by solving a linear quadratic optimal control problem with cost $Dg(\xi_i; \bar{k}) \cdot \zeta + \frac{1}{2} D^2 g(\xi_i; \bar{k})(\zeta, \zeta)$, where $\zeta \mapsto Dg(\xi_i; \bar{k}) \cdot \zeta$ and $\zeta \mapsto D^2 g(\xi_i; \bar{k})(\zeta, \zeta)$ are respectively the first and second Fréchet differentials of the functional $g(\xi, \bar{k})$ at ξ_i. Then, the curve $\xi_i + \gamma_i \zeta_i$, where γ_i is a step size obtained through a standard backtracking line search, is projected, by means of the projection operator, in order to get a new trajectory ξ_{i+1}.

The strength of this approach is that the local minimizer of (42) is obtained as the limit of a sequence of trajectories, i.e., curves satisfying the dynamics. Furthermore, the feedback system (41), defining the projection operator, allows us to generate trajectories in a numerically stable manner.

Remark 2: An elegant extension of the PRONTO method to Lie groups is developed in [32] and could be alternatively used in our strategy.

References

[1] N. Dadkhah and B. Mettler, "Survey of motion planning literature in the presence of uncertainty: considerations for UAV guidance," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1-4, pp. 233–246, 2012.

[2] C. L. Bottasso, D. Leonello, and B. Savini, "Path Planning for Autonomous Vehicles by Trajectory Smoothing Using Motion Primitives," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 6, pp. 1152–1168, 2008.

[3] G. Ambrosino, M. Ariola, U. Ciniglio, F. Corraro, E. D. Lellis, and A. Pironti, "Path Generation and Tracking in 3-D for UAVs," *IEEE Transactions on Control Systems Technology*, vol. 17, no. 4, pp. 980–988, 2009.

[4] B. Herissé, T. Hamel, R. Mahony, and F.-X. Russotto, "Landing a VTOL unmanned aerial vehicle on a moving platform using optical flow," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 77–89, 2012.

[5] R. Naldi, A. Torre, and L. Marconi, "Robust control of a miniature ducted-fan aerial robot for blind navigation in unknown populated environments," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 1, pp. 64–79, 2015.

[6] X. Hou and R. Mahony, "Dynamic Kinesthetic Boundary for Haptic Teleoperation of VTOL Aerial Robots in Complex Environments," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 5, pp. 694–705, 2016.

[7] M. Furci, R. Naldi, S. Karaman, and L. Marconi, "A Combined Planning and Control Strategy for Mobile Robots Navigation in Populated Environments," in *IEEE Conference on Decision and Control*, 2015.

[8] A. Masoud and A. Al-Shaikhi, "Time-sensitive, sensor-based, joint planning and control of mobile robots in cluttered spaces: A harmonic potential approach," in *IEEE Conference on Decision and Control*, 2015.

[9] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "Direct method based control system for an autonomous quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 60, no. 2, pp. 285–316, 2010.

[10] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation*, 2011.

[11] Y. Bouktir, M. Haddad, and T. Chettibi, "Trajectory planning for a quadrotor helicopter," in *Mediterranean Conference on Control and Automation*, 2008.

[12] W. Van Loock, G. Pipeleers, and J. Swevers, "Time-optimal quadrotor flight," in *European Control Conference*, 2013.

[13] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *IEEE International Conference on Robotics and Automation*, 2016.

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

[15] E. Koyuncu and G. Inalhan, "A probabilistic b-spline motion planning algorithm for unmanned helicopters flying in dense 3d environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[16] P. M. Bouffard and S. L. Waslander, "A hybrid randomized/nonlinear programming technique for small aerial vehicle trajectory planning in 3d," *Planning, Perception and Navigation for Intelligent Vehicles*, vol. 63, 2009.

[17] D. Devaurs, T. Siméon, and J. Cortés, "Optimal Path Planning in Complex Cost Spaces With Sampling-Based Algorithms," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 415–424, 2016.

[18] R. Allen and M. Pavone, "A Real-Time Framework for Kinodynamic Planning with Application to Quadrotor Obstacle Avoidance," in *AIAA Conf. on Guidance, Navigation and Control, San Diego, CA*, 2016.

[19] M. Hehn and R. D'Andrea, "Real-Time Trajectory Generation for Quadrocopters," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 877–892, 2015.

[20] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[21] ——, "Dance of the flying machines: Methods for designing and executing an aerial dance choreography," *IEEE Robotics & Automation Magazine*, vol. 20, no. 4, pp. 96–104, 2013.

[22] J. Hauser, "A projection operator approach to the optimization of trajectory functionals," in *IFAC world congress*, 2002.

[23] J. Hauser and A. Saccon, "A barrier function method for the optimization of trajectory functionals with constraints," in *IEEE Conference on Decision and Control*, 2006.

[24] ——, "Motorcycle modeling for high-performance maneuvering," *IEEE Control Systems*, vol. 26, no. 5, pp. 89–105, 2006.

[25] A. J. Hausler, A. Saccon, A. P. Aguiar, J. Hauser, and A. M. Pascoal, "Energy-Optimal Motion Planning for Multiple Robotic Vehicles With Collision Avoidance," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 3, pp. 867–883, 2015.

[26] A. Rucco, A. P. Aguiar, and J. Hauser, "A Virtual Target Approach for Trajectory Optimization of a General Class of Constrained Vehicles," in *IEEE Conference on Decision and Control*, 2015.

[27] M. D. Hua, T. Hamel, P. Morin, and C. Samson, "Introduction to feedback control of underactuated VTOL vehicles: A review of basic control design ideas and principles," *IEEE Control Systems*, vol. 33, no. 1, pp. 61–75, 2013.

[28] M. P. Setterlund, "Geometric-based Spatial Path Planning," PhD dissertation, University of Texas at Austin, 2008.

[29] A. Banaszuk and J. Hauser, "Feedback linearization of transverse dynamics for periodic orbits," in *IEEE Conference on Decision and Control*, 1994.

[30] A. Saccon, J. Hauser, and A. Beghi, "A Virtual Rider for Motorcycles: Maneuver Regulation of a Multi-Body Vehicle Model," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 2, pp. 332–346, March 2013.

[31] S. Spedicato, G. Notarstefano, H. H. Bülthoff, and A. Franchi, "Aggressive Maneuver Regulation of a Quadrotor UAV," in *The 16th International Symposium on Robotics Research*, 2013.

[32] A. Saccon, J. Hauser, and A. P. Aguiar, "Optimal control on Lie groups: The projection operator approach," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2230–2245, 2013.

Sara Spedicato is a Post-Doctoral Researcher at the Università del Salento (Lecce, Italy), where she received the Laurea degree "summa cum laude" in Mechanical Engineering (curriculum "Servomechanisms and industrial automation") in 2012 and the Ph.D. degree in Information Engineering in 2016. She carried out an internship activity at the ETH Zürich (Zürich, Switzerland) from June to September 2012. She was a visiting graduate student at the Max Planck Institute for Biological Cybernetics (Tübingen, Germany) from January to August 2013. Her research activity involves nonlinear optimal control, distributed optimization, trajectory optimization and maneuvering for autonomous aerial vehicles.

Giuseppe Notarstefano (M'11) is Associate Professor at the Università del Salento (Lecce, Italy), where he was Assistant Professor (Ricercatore) from February 2007 to May 2016. He received the Laurea degree "summa cum laude" in Electronics Engineering from the Università di Pisa in 2003 and the Ph.D. degree in Automation and Operation Research from the Università di Padova in 2007. He has been visiting scholar at the University of Stuttgart, University of California Santa Barbara and University of Colorado Boulder. His research interests include distributed optimization, cooperative control in complex networks, applied nonlinear optimal control, and trajectory optimization and maneuvering of aerial and car vehicles. He serves as an Associate Editor for IEEE Transactions on Control Systems Technology, for IEEE Control Systems Letters, for the Conference Editorial Board of the IEEE Control Systems Society and for other IEEE and IFAC conferences. He coordinated the VI-RTUS team winning the International Student Competition Virtual Formula 2012. He is recipient of an ERC Starting Grant 2014.