NAVIGATION

Time-optimal planning for quadrotor waypoint flight

Philipp Foehn*, Angel Romero, Davide Scaramuzza

Quadrotors are among the most agile flying robots. However, planning time-optimal trajectories at the actuation limit through multiple waypoints remains an open problem. This is crucial for applications such as inspection, delivery, search and rescue, and drone racing. Early works used polynomial trajectory formulations, which do not exploit the full actuator potential because of their inherent smoothness. Recent works resorted to numerical optimization but require waypoints to be allocated as costs or constraints at specific discrete times. However, this time allocation is a priori unknown and renders previous works incapable of producing truly time-optimal trajectories. To generate truly time-optimal trajectories, we propose a solution to the time allocation problem while exploiting the full quadrotor's actuator potential. We achieve this by introducing a formulation of progress along the trajectory, which enables the simultaneous optimization of the time allocation and the trajectory itself. We compare our method against related approaches and validate it in real-world flights in one of the world's largest motion-capture systems, where we outperform human expert drone pilots in a drone-racing task.

INTRODUCTION

Autonomous drones are nowadays used for inspection, delivery, cinematography, search and rescue, and entertainment such as drone racing (*1*). The most prominent aerial system is the quadrotor, thanks to its simplicity and versatility ranging from smooth maneuvers to extremely aggressive trajectories. This renders quadrotors among the most agile and maneuverable aerial robots (*2*, *3*).

However, quadrotors have limited flight range, dictated by their battery capacity, which limits how much time can be spent on a specific task. If the task consists of visiting multiple waypoints [delivery, inspection, and drone racing (*4*–*6*)], then doing so in minimal time is often desired, and, in the context of search and rescue or drone racing (Fig. 1), even the ultimate goal. Expert human drone racing pilots accomplish this with astonishing performance, guiding their quadrotors through race tracks at speeds so far unreached by any autonomous system. This begs the question of how close human pilots fly to the theoretical limit of a quadrotor and whether planning algorithms could find and execute such theoretical optima.

For simple point-mass systems, time-optimal trajectories can be computed in closed form, resulting in bang-bang acceleration trajectories (*7*), which can be sampled over multiple waypoints (*8*). However, quadrotors are underactuated systems that need to rotate to adjust their actuated acceleration direction, which always lies in the body z axis (*9*, *10*). Both the linear and rotational acceleration are controlled through the rotor thrusts, which are physically limited by the actuators. This introduces a coupling in the achievable linear and rotational accelerations. Therefore, time-optimal planning becomes the search for the optimal trade-off between maximizing these accelerations.

Two common approaches for planning quadrotor trajectories exist: continuous-time polynomials and discrete-time state space representations. The first option is the widely used polynomial formulation (*10*–*12*) exploiting the quadrotor's differentially flat output states with high computational efficiency. However, these polynomials are inherently smooth and therefore cannot represent rapid state or input changes [e.g., bang-bang (*7*)] at reasonable order and only reach the input limits for infinitesimal short durations or constantly for the full trajectory time. This renders polynomials suboptimal because they cannot exploit the full actuator potential. Both problems are visualized and further explained in the "Preface: Time-optimal quadrotor trajectory" section in the Supplementary Materials.

The second option includes all approaches using time-discretized trajectories that can be found using search and sampling-based methods (*13*–*16*) or optimization-based methods (*17*–*20*). However, sampling the four-dimensional (4D) continuous input space over many discrete time steps with sufficient resolution quickly becomes computationally intractable, which is why prior work (*13*–*16*) restores to point-mass, polynomial, or differential-flatness approximations and therefore does not handle single-rotor thrust constraints. Therefore, planning time-discretized trajectories with optimization-based methods is the only viable solution in the short-medium term. In such methods, the system dynamics and input boundaries are enforced as constraints. In contrast to the polynomial formulation, this allows the optimization to pick any input within bounds for each discrete time step. For a time-optimal solution, the trajectory time t_N is part of the optimization variables and is the sole term in the cost function. However, if multiple waypoints must be passed, then these must be allocated as constraints to specific nodes on the trajectory. This time allocation is a priori undefined, because the time spent between any two waypoints is unknown, which renders traditional discretized state space formulations ineffective for time-optimal trajectory generation through multiple waypoints.

We investigate this problem and provide a solution that allows simultaneously optimizing the trajectory and waypoint allocation in a given sequence, exploiting the full actuator potential of a quadrotor. Our approach formulates a progress measure for each waypoint along the trajectory, indicating completion of a waypoint (see Fig. 5). We then introduce a "complementary progress constraint" (CPC) that allows completion only in proximity to a waypoint. Intuitively, proximity and progress must complement each other, enforcing completion of all waypoints without specifying their time allocation.

There already exists a number of works toward time-optimal quadrotor flight (*21*–*24*), which, however, all suffer from severe limitations, such as limiting the collective thrust and body rates, rather than the actual constraint of limited single-rotor thrusts. The

Robotics and Perception Group, University of Zurich, Zurich, Switzerland.
*Corresponding author. Email: foehn@ifi.uzh.ch

Fig. 1. A time-optimal trajectory. This time-optimal flight path was computed using the proposed CPCs and executed in a motion capture system, outperforming the best human expert.

Fig. 2. The quadrotor vehicle. The autonomous platform used for the real-world experiments with a theoretical TWR of ~4 at 0.8-kg weight, equipped with a Jetson TX2, a Laird communication module, off-the-shelf drone racing components, and infrared-reflective markers for motion capture.

two earlier works (*21*, *22*) are based on the aforementioned bang-bang approaches extended through numerical optimization of the switching times (*21*) and a trajectory representation using a convex combination of multiple analytical path functions (*22*). However, both are restricted to 2D maneuvers, whereas our approach generalizes to arbitrary 3D waypoint sequences.

Another approach is taken in (*23*), where a change of variables along an analytic reference path is used to put the vehicle state space into a traverse-dynamics formulation. This allows using the arc length along the reference path as a progress measure and enables the formulation of costs and constraints independent of the time variable. However, as in the previous works, they simplify the platform limits to collective thrust and body rates, neglecting realistic actuator saturation. Furthermore, because of the use of Euler angles, their orientation space only covers a subset of the feasible attitudes and limits the solutions to a, possibly suboptimal, subspace.

Last, Ryou *et al.* (*24*) use a completely different approach, where the segment times of a polynomial trajectory are refined on the basis of learning a Gaussian classification model predicting feasibility. The classification is trained on analytic models, simulation, and real flight data. Although emphasizing real-world applicability, this approach is still constrained to polynomials and requires real-world data specifically collected for the given vehicle. Furthermore, it is an approximate method that only refines the execution speed of a predefined trajectory, rather than modifying the trajectory itself to a time-optimal solution, as opposed to our method.

In contrast to existing methods, our approach resolves these problems by taking inspiration from optimization under contacts (*25*), proposing the formulation of CPCs, where we introduce a measure of progress and complement (*26*) it with waypoint proximity. More specifically, we formulate two factors that must complement each other, where, in our case, one factor is the completion of a waypoint (progress), whereas the other factor is the local proximity to a waypoint. Intuitively, a waypoint can only be marked as completed when the quadrotor is within a certain tolerance of the waypoint, allowing simultaneous optimization of the state and input trajectory and the waypoint time allocation.

We demonstrate how our formulation can generate trajectories that are faster than human expert flights and evaluate it against two professional human drone racing pilots, outperforming them in terms of lap time and consistency on a 3D race track in a large-scale motion capture system (Fig. 1). Because our proposed optimization problem is highly nonconvex, we also provoke nonconvexity effects in simulation experiments in the Supplementary Materials (the "Simulation experiments" section).

Our method not only can serve as a baseline for time-optimal quadrotor flight but also might find applications in other fields, such as (multi-) target interception and orbital maneuvers, avoiding mixed-integer formulations (*27*) and any problem where a sequence of task goals of unknown duration must be optimized under complex dynamic constraints.

RESULTS

We chose drone racing as a demonstrator for our method because in racing, the ultimate goal is to fully exploit the actuator potential to accomplish a task in minimal time. In our experiment, we set up a human baseline on a 3D race track with seven gates (Fig. 1 and Movie 1) in a motion capture environment with two professional expert drone racing pilots. We planned a time-optimal trajectory through the same race track and used an in-house developed drone platform and software stack to execute the trajectory in the same motion capture environment. We generated the trajectory at a slightly lower thrust bound than what the platform can deliver, to maintain controllability under disturbances as mentioned in the "Real-world restrictions" section in the Supplementary Materials and discussed in the "Real-world deployment" section. Our results show that we can outperform the humans and consistently beat their best lap time.

Experimental drone platform

The experiments were flown with an in-house developed drone platform (Fig. 2) based on off-the-shelf drone-racing components such as a carbon-fiber frame, BLDC (brush-less direct current) motors, 5-inch propellers, and a BetaFlight flight controller. The quadrotor was equipped with an NVIDIA Jetson TX2 computing unit with Wi-Fi and a Laird RM024 module for wireless low-latency communication. The static maximum thrust of a single rotor was measured using a load cell and verified in-flight, marking the platform's real maximum limit at a thrust-to-weight ratio (TWR) of ~4. Other specifications can be taken from table S1 under the race quadrotor configuration.

For state estimation (pose, linear, and angular velocities), we used a VICON system with 36 cameras. For control, we deployed a

Table 1. Timing statistics.

Timing	Mean (s)	Median (s)	Min (s)	Max (s)	Std (s)
Human expert 1	6.794	6.740	6.389	7.530	0.2556
Human expert 2	6.987	6.960	6.450	7.780	0.2859
Ours with TWR: 3.15	6.272	6.272	6.190	6.354	0.0280
Ours with TWR: 3.3	6.120	6.116	6.048	6.215	0.278

model predictive controller (MPC), similar to (*28*) but on the basis of the quadrotor dynamics from Eqs. 16 to 18, with the state space $\mathbf{x} = [\mathbf{p}_{IB}, \mathbf{q}_{IB}, \mathbf{v}_{IB}, \boldsymbol{\omega}_B]^\top$ and input space $\mathbf{u} = [T_1, T_2, T_3, T_4]$. The MPC operated over a horizon of $N_{\mathrm{MPC}} = 20$ time steps of $\delta t = 0.05$ s, with a quadratic cost function, and also accounted for the single-rotor thrust constraints. The implementation was done using the ACADO (*29*) toolkit and qpOASES (*30*) as solver. We executed the MPC in a real-time iteration scheme (*31*) at a feedback rate of 100 Hz. The low-level BetaFlight controller has access to high-frequency IMU (inertial measurement unit) measurements, which allows precise tracking of body rate and collective thrust commands. These commands were extracted from the MPC controller and were guaranteed to stay within the platform capabilities because of the single-rotor thrust constraints.

Human expert pilot baseline

Because human pilots so far outperformed autonomous vehicles, we established a baseline by inviting two professional expert drone racing pilots, M. Isler and T. Trowbridge, both of whom compete in professional drone racing competitions. A list of their participation and rankings can be found in table S2. We created a 3D racing track in a VICON motion capture environment spanning roughly 25 m by 30 m by 8 m and let the humans train on this track for hours. We captured multiple races, each consisting of multiple laps, from which we evaluated the one with the overall best lap time (Table 1 and Fig. 3), according to our timing strategy described in the "Timing analysis" section. The quadrotor platform used for human flights has the same TWR as the autonomous platform. This provides a fair baseline because (i) both the humans and the autonomous drone have the same limitations; (ii) the humans were given enough training time to adapt to the track, as is the case at a real drone racing event; and (iii) we compared against the race including the absolute best lap time from all runs. Especially, the latter point gives a substantial advantage to the human competitors, because they are typically not capable of reproducing the absolute best lap time reliably and would therefore fall behind in any multilap evaluation. The evaluated best human runs each contain seven laps.

The human platform was also tracked in the VICON system and resembled the autonomous drone described in the "Experimental drone platform" section but dropped the Jetson and Laird modules for a remote control receiver and a first-person-view camera system, with no weight difference. To keep a fair baseline, the human platform was restricted to the same maximum thrust-to-weight limit as the autonomous drone. The track, time-optimal reference

Movie 1. Time-optimal planning for quadrotors. This movie depicts the real-world experiments evaluating our time-optimal quadrotor trajectories against professional human drone racing pilots.

(TWR of 3.3), and the human expert 1 trajectory are visualized in Fig. 4, with their speed and acceleration profile colorized.

Trajectory generation

To generate the time-optimal trajectories that will be executed by our platform, we set the gate positions of the track shown in Fig. 4 as waypoint constraints. We generated 2.5 laps to ensure that our experiments were not affected by start and end transient effects or large drops in battery voltage and to ensure at least two full laps at maximum speed. For optimal results, the laps were generated by concatenating the waypoints for a single lap multiple times and solving the full multilap problem at once. Therefore, the optimization passes through 18 waypoints and can be solved on a normal desktop computer in ~40 min. We used $N = 720$ nodes with a tolerance of $d_{\mathrm{tol}} = 0.3$ m.

Because a time-optimal trajectory exploits the full actuator potential, it is extremely aggressive from start to end. To ensure safe execution on a real drone, we linearly ramped up the thrust limit for the trajectory generation from hover to the full thrust limit, guaranteeing a smooth start of the trajectory. Because we planned over 2.5 laps and excluded the start and end from the timing, this had no notable impact on the reported timing results. The same start and end exclusion were done for the human timings (in the "Timing analysis" section).

In addition, we planned the trajectory for a multitude of TWRs reaching from 2.5 with a lap time of 7.14 s to a maximum of 3.6, resulting in a lap time of 5.81 s. As expected, the time shrinks with higher thrust-to-weight capabilities. We evaluated two configurations with TWRs of 3.15 (6.27 s) and 3.3 (6.10 s) in our real-world experiments, reported in the following section. Both of these configurations were consistently faster than the human trajectory while staying within a safe margin of the quadrotor absolute limit (TWR of ~4), which allows for robust control even under disturbances, noise, and model imperfections.

Timing analysis

First of all, our real-world experiments should provide a proof of concept that time-optimal trajectories planned below the drone's

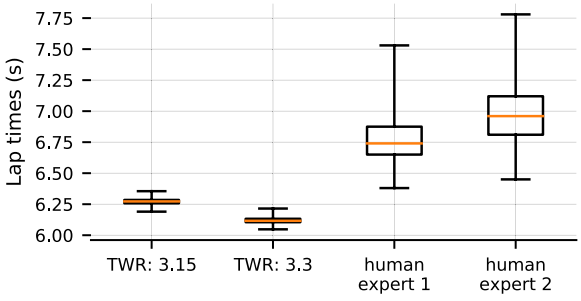

Fig. 3. Timing analysis. Lap time box plot of our time-optimal trajectory executed in real world for two configurations (two laps) and two human races (seven laps). Note that even with a 5% reduced TWR of 3.15, the proposed time-optimal trajectories are faster than the best human lap time, with substantially lower variance.

actual thrust limit are a feasible and viable solution. We refer the reader to the accompanying Movie 1.

Second, we point out that it is possible to generate and execute trajectories that can outperform the human baseline. For this, we provide a statistical lap time analysis in Table 1 and Fig. 3, indicating the superior performance on both configurations.

To compute reliable lap times, we first defined a full lap as each lap that is not affected by any start (takeoff) or end (landing) segment. We could then pick a set of timing points $\mathcal{S}_{pt} = \{\mathbf{p}_{t0}, \mathbf{p}_{t1}, \ldots\}$ along the trajectory and define the timing as the time needed to visit one such point twice, i.e., the time of a segment starting and ending at the same point. This measure allows us to extract a statistically valid timing of a single closed lap, which does not depend on the location of timing start and stop.

Effectively, we timed the best laps of the human flights and two full laps of both our race trajectories, each flown twice. Figure 3 shows the obtained timing results, where it is clearly visible that the autonomous drone outperforms the human pilots both in absolute time and consistency. The latter is expected because once the trajectory is generated, it can be repeated multiple times without variation.

DISCUSSION

Velocity and acceleration distribution

Furthermore, we evaluate the velocity and acceleration distribution over the different human and time-optimal flights. We depict the trajectories colored by their speed and acceleration profile in Fig. 4 with the time-optimal reference with TWR of 3.3. Inspecting the hotspots of darker colors in the velocity plots (Fig. 4, top row) indicating higher speeds, we can see that the velocity distribution is rather similar for the autonomous time-optimal and the human trajectory. However, comparing the acceleration of the human and the time-optimal trajectory (Fig. 4, bottom row), we observe that the acceleration of the human varies considerably more and often dips to lower values than the time-optimal ones. This corresponds to sections where the human pilots do not use the full actuation spectrum of the platform and lose substantial performance over the time-optimal trajectory. This is especially visible in the right-most section of the track, where the flight path has relatively low curvature. The time-optimal trajectory exploits the full acceleration capabilities, whereas the human notably reduces the acceleration between the right-most gates, leading to lower speeds in the following bottom section of the track.

Furthermore, we can identify the high-speed region in the sections of low curvature, where both human and autonomous platforms spend more acceleration in the velocity direction (accelerating and braking) than perpendicular to the velocity (direction change). Although the platforms have equal TWR, and our time-optimal trajectory is planned with a substantial margin to the platform's TWR limit, it exceeds the human speed profile.

Human performance comparison

From our findings in the "Timing analysis" and "Velocity and acceleration distribution" sections, we conclude that human pilots are not far away from our time-optimal trajectory, because they reach similar but slower velocity distribution and lap times. However, humans struggle to consistently exploit the full actuation spectrum of the vehicle, resulting in suboptimal performance compared to our approach. One possible reason for this can be found in (*32*), where we analyzed eye gaze fixations of human pilots during racing and found that they fixate their eyes on upcoming gates well before passing the next gate, indicating that humans use a receding planning horizon, whereas our approach optimizes the full trajectory at once.

Tracking error considerations

We want to point out that although we achieved successful deployment on a real quadrotor system, we experience substantial tracking errors in doing so, as visible in Fig. 4. Although this work is not about improving the tracking performance but should rather serve as a feasibility study given our method, we still feel responsible for pointing out the encountered difficulties. A number of effects lead to this tracking error of ~0.7 positional root mean square error:

1) We did not account for any latency correction of the whole pipeline, including motion capture and pose filtering, data transmission to the drone, MPC execution time, and communication to the flight controller.

2) Our simple linear-drag aerodynamic model was verified in (*33*) at speeds up to 5 m s^{-1}. However, in the vastly higher speed regimes that we reach during our real-world experiments, the model seems to be inaccurate, especially in terms of drag at higher speeds and in body z direction. The effect of this is visible in Fig. 4, where there is a disturbance toward the inside of the curvature for most of the time.

3) The used BetaFlight controller is a reliable system for human drone pilots. However, as such, it includes many filtering, feed-forward, and control strategies that are tuned for a consistent human flight feeling. Unfortunately, this does not translate to accurate closed-loop control or desirable control-loop shaping and causes more harm than good when used with closed-loop high-level controllers such as our MPC. Not only is it not possible to command the single-rotor thrusts from the MPC to BetaFlight but also the provided body rate tracking performed poorly. This, however, could be solved with a custom flight controller and might be part of further studies.

Convexity and optimality

Although the problem of trajectory optimization quickly becomes nonconvex when using complex and/or nonlinear dynamic models, constraints, or even cost formulations (e.g., obstacle avoidance), it is often a valid approach to generate feasible (in terms of model

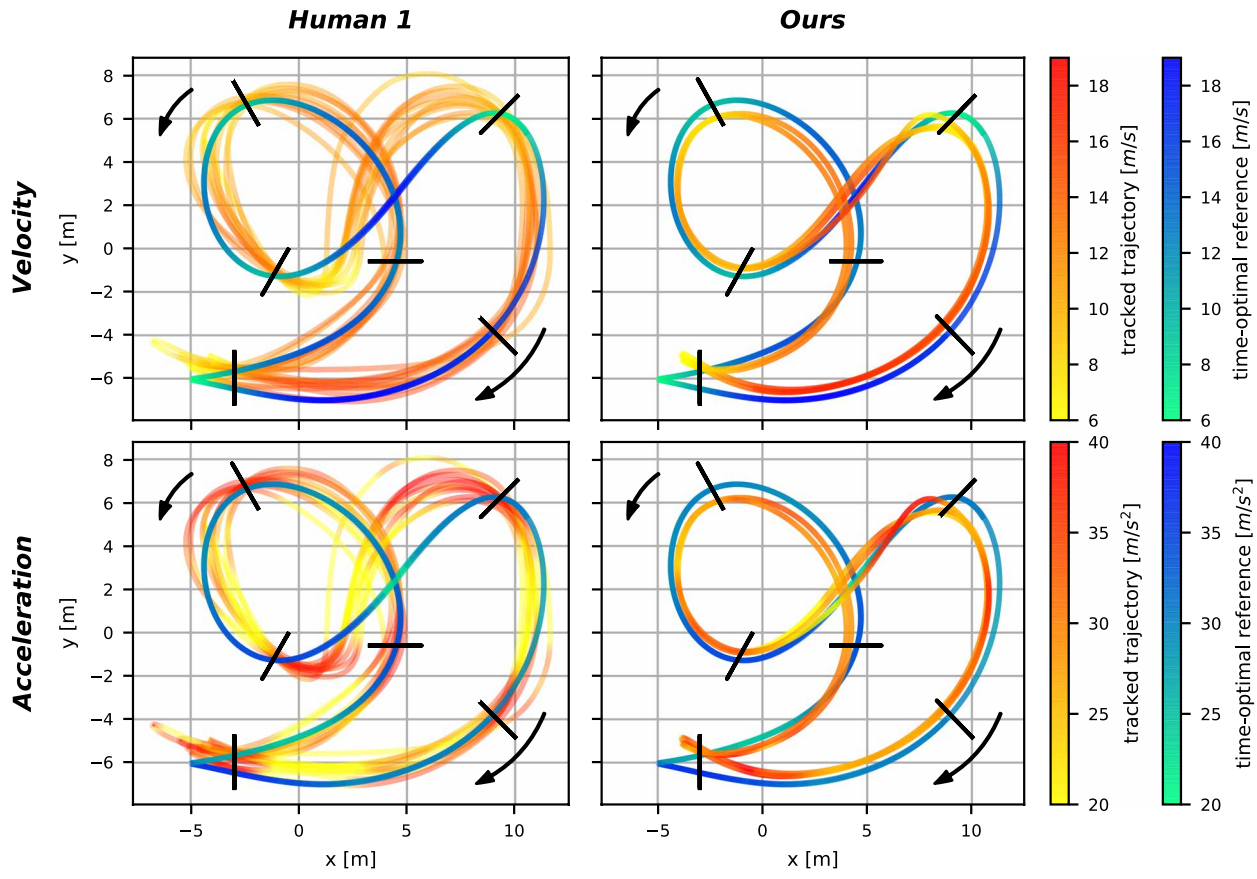

Fig. 4. Comparison against a human pilot. The race track with seven gates visualized with the yellow-red real-world trajectories of the humans (**left**; all seven laps of the race including the overall best lap time) and the autonomous drone (**right**; two laps visualized), and the green-blue time-optimal reference trajectory with a TWR of 3.3. The trajectories are colored by their speed profile (top row) and acceleration profile (bottom row) to indicate the hotspots of highest velocity and acceleration along the track. The two black arrows indicate the direction of flight. The human pilots vary their acceleration substantially more than the autonomous drone and spend more time at suboptimal acceleration (yellow coldspots on the lower left). Note that the gate in the lower left corner consists of two gates stacked vertically.

dynamics) trajectories. Given a nonconvex problem, solution schemes such as the used interior point method can only guarantee local optimality and global feasibility but not global optimality. Within our approach, we can summarize the following nonconvex properties:

1) The quadrotor dynamics (Eq. 16) are nonlinear and therefore nonconvex in the context of Eq. 1.

2) The acceleration space of a quadrotor (Eq. 18) is nonconvex given nonzero minimal thrust $T_{\min} > 0.0$.

3) Equation 14 is nonconvex because of the norm on distance and the bilinearity of the progress change $\boldsymbol{\mu}$ and tolerance slack $\boldsymbol{\nu}$.

In our experiments in the "Provoking nonconvexity issues" and "Local-versus-global optimum" sections in the Supplementary Materials, we provoke such nonconvex properties and explain how the optimization can be supported with an advanced initialization scheme to start close to the global optimum.

This can be achieved by reducing the nonlinear quadrotor model into a linear point-mass model with bounded 3D acceleration input $\mathbf{u} = \mathbf{a}$ where $\| \mathbf{a} \| \leq a_{\max}$. The linear model removes the prominent nonconvex dynamics and allows us to find a solution from which the original problem with the quadrotor model can be initialized, both in terms of translational trajectory, and also with a noncontinuous orientation guess based on the point-mass acceleration direction. Although this initial guess is not yet dynamically feasible for quadrotors because of the absent rotational dynamics, it serves as a valid initial guess in the convex region of the global-optimal translation space. Last, the rotational nonconvexity can be resolved by solving multiple problems of different initializations, as demonstrated in the "Provoking nonconvexity issues" section in the Supplementary Materials.

Last but not least, the reader should note that even in the case of a local (but not global) optimal solution, the vehicle dynamics are satisfied and the trajectory is dynamically feasible.

Real-world deployment

There are three challenges when deploying our approach in real-world scenarios.

The first problem is posed by the nature of time-optimal trajectories themselves, because the true solution for a given platform is nearly always at the actuator constraints and leaves no control authority. This means that even the smallest disturbance could potentially have damaging consequences for the drone and render the remainder of the trajectory unreachable. One has to define a margin lowering the actuator constraints used for the trajectory generation to add control authority and therefore robustness against disturbances. However, this also leads to a slower solution, which is no longer the platform-specific time-optimal one. In the context of a

competition, this effectively becomes a risk-management problem with interesting connections to game theory.

Second, our method is computationally demanding, ranging from a few minutes (<20 min) for scenarios as in the "Time-optimal hover-to-hover trajectories" and "Provoking nonconvexity issues" sections in the Supplementary Materials toward an hour or more for larger scenarios such as in Results with ~40 min and in the "Microsoft AirSim, NeurIPS 2019 Qualification 1" in the Supplementary Materials with ~65 min on a normal desktop computer. However, this is highly implementation dependent and could be vastly broken down to usable times or precomputed for static race tracks and other nondynamic environments. Furthermore, our approach provides a method for finding the theoretical upper bound on performance, as a benchmark for other methods.

Third, we use a motion capture system to deploy our method. However, in real-world scenarios, such high-performance off-board localization systems are barely ever available. This necessitates the deployment using on-board state-estimation systems, such as visual-inertial odometry. Unfortunately, these systems can suffer from high motion blur in such fast flight scenarios and therefore need substantial further research and development to be of sufficient robustness for the purpose of time-optimal flight (*4*, *5*). Despite those difficulties, we have demonstrated that our method can be deployed and is substantially faster than human experts.

MATERIALS AND METHODS

General trajectory optimization

The general optimization problem of finding the minimizer $\mathbf{x}^*$ for cost $L(\mathbf{x})$ in the state space $\mathbf{x} \in \mathbb{R}^n$ can be stated as

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} L(\mathbf{x}) \text{ subject to } \mathbf{g}(\mathbf{x}) = 0 \text{ and } \mathbf{h}(\mathbf{x}) \leq 0$$

where $\mathbf{g}(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ contain all equality and inequality constraints, respectively. The full state space $\mathbf{x}$ is used equivalently to the term optimization variables. The cost $L(\mathbf{x})$ typically contains one or multiple quadratic costs on the deviation from a reference, costs on the systems actuation inputs, or other costs describing any desired behaviors.

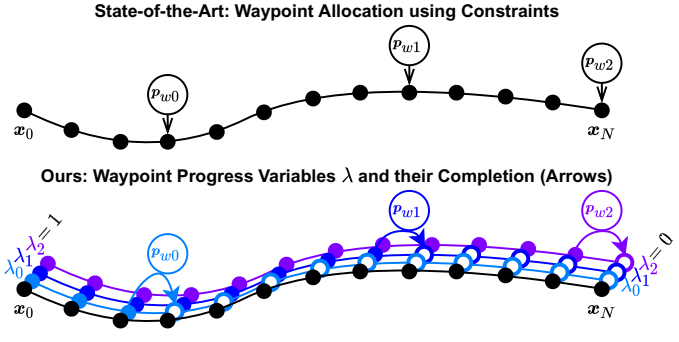

Fig. 5. Progress variables. (**Top**) State-of-the-art fixed allocation of positional waypoints $\mathbf{p}_{wj}$ to specific nodes $\mathbf{x}_i$. (**Bottom**) Our method of defining one progress variable λ_j per waypoint. The progress variable can switch from 1 (incomplete) to 0 (completed) only when in the proximity of the relevant waypoint, implemented as a complementary constraint.

Multiple-shooting method

To represent a dynamic system in the state space, the system state $\mathbf{x}_k$ is described at discrete times $t_k = dt \cdot k$ at $k \in [0, N]$, also called nodes, where its actuation inputs between two nodes are $\mathbf{u}_k$ at t_k with $k \in [0, N)$. The systems evolution is defined by the dynamics $\dot{\mathbf{x}} = \mathbf{f}_{\text{dyn}}(\mathbf{x}, \mathbf{u})$, anchored at $\mathbf{x}_0 = \mathbf{x}_{\text{init}}$, and implemented as an equality constraint of the fourth-order Runge-Kutta integration scheme (RK4)

$$\mathbf{x}_{k+1} - \mathbf{x}_k - \text{dt} \cdot \mathbf{f}_{\text{RK4}}(\mathbf{x}_k, \mathbf{u}_k) = 0 \tag{1}$$

which is part of $\mathbf{g}(\mathbf{x}) = 0$ in the general formulation. Both $\mathbf{x}_k$ and $\mathbf{u}_k$ are part of the state space and can be summarized as the vehicle's dynamic states $\mathbf{x}_{\text{dyn},k}$ at node k. Note that this renders the problem formulation nonconvex for nonlinear system dynamics.

Time-optimal trajectory optimization

Optimizing for a time-optimal trajectory means that the only cost term is the overall trajectory time $L(\mathbf{x}) = t_N$. Therefore, t_N needs to be in the optimization variables $\mathbf{x} = [t_N, \ldots]^\top$ and must be positive $t_N > 0$. The integration scheme can then be adapted to use $dt = t_N/N$.

Passing waypoints through optimization

To generate trajectories passing through a sequence of waypoints $\mathbf{p}_{wj}$ with $j \in [0, \ldots, M]$, one would typically define a distance cost or constraint and allocate it to a specific state $\mathbf{x}_{\text{dyn},k}$ at node k with time t_k. For cost-based formulations, quadratic distance costs are robust in terms of convergence and implemented as

$$L_{\text{dist},j} = (\mathbf{p}_k - \mathbf{p}_{wj})^\top (\mathbf{p}_k - \mathbf{p}_{wj}) \tag{2}$$

where $\mathbf{p}_k$, part of $\mathbf{x}$, is the position state at a user defined time t_k. However, such a cost-based formulation is only a soft requirement and if summed with other cost terms does not imply that the waypoint is actually passed within a certain tolerance. To guarantee to pass within a tolerance, constraint-based formulations can be used, such as

$$(\mathbf{p}_k - \mathbf{p}_{wj})^\top (\mathbf{p}_k - \mathbf{p}_{wj}) \leq \tau_j^2 \tag{3}$$

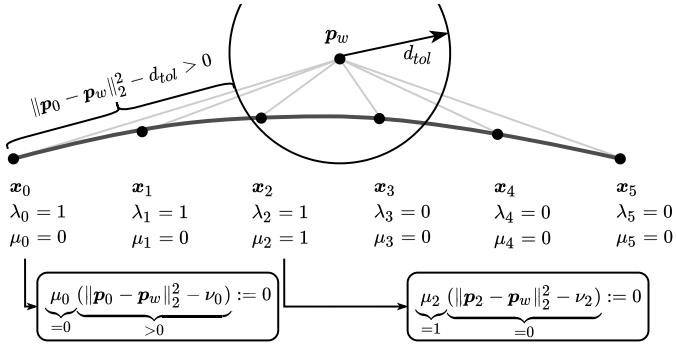

Fig. 6. Complementary progress constraint. The progress change μ can only be nonzero if the distance to the waypoint $\mathbf{p}_w$ is less than the tolerance d_{tol}. This is not the case for $\mathbf{x}_0$, but for $\mathbf{x}_1$, and allowing the progress variable to switch to 0 (complete).

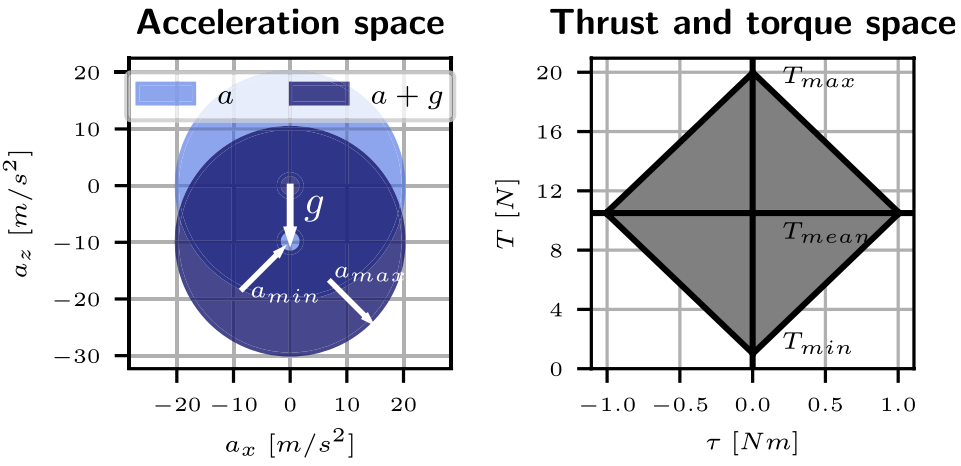

Fig. 7. Quadrotor input space. Acceleration (**left**) and thrust/torque space (**right**) of a standard quadrotor configuration. Note that the acceleration space is nonconvex because of the minimum acceleration (the idle thrust at minimum motor speed) $a_{\min} > 0$ being nonzero, and the thrust and torque limits are dependent on each other.

which, in the general problem, is part of $\mathbf{h}(\mathbf{x}) \leq 0$ and requires the trajectory to pass by waypoint j at position $\mathbf{p}_{wj}$ within tolerance τ_j at time t_k.

Progress measure variables

To describe the progress throughout a track, we want a measure that fulfills the following requirements: (i) it starts at a defined value, (ii) it must reach a different value by the end of the trajectory, and (iii) it can only change when a waypoint is passed within a certain tolerance. To achieve this, let the vector $\boldsymbol{\lambda}_k \in \mathbb{R}^M$ define the progress variables λ_k^j at time step t_k for all M waypoints indexed by j. All progress variables start at 1 as in $\boldsymbol{\lambda}_0 = \mathbf{1}$ and must reach 0 at the end of the trajectory as in $\boldsymbol{\lambda}_N = \mathbf{0}$. The progress variables $\boldsymbol{\lambda}$ are chained together and their evolution is defined by

$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k - \boldsymbol{\mu}_k \tag{4}$$

where the vector $\boldsymbol{\mu}_k \in \mathbb{R}^M$ indicates the progress change at every time step (Fig. 5). Note that the progress can only be positive, and therefore, $\mu_k^j \geq 0$. Both $\boldsymbol{\lambda}_k$ and $\boldsymbol{\mu}_k$ for every time step are part of the optimization variables $\mathbf{x}$, which replicates the multiple-shooting scheme for the progress variables. To define when and how the progress variables can change, we now imply a vector of constraints $\mathbf{f}_{\text{prog}}$ on $\boldsymbol{\mu}_k$, in its general form as

$$\boldsymbol{\epsilon}_k^- \leq \mathbf{f}_{\text{prog}}(\mathbf{x}_k, \boldsymbol{\mu}_k) \leq \boldsymbol{\epsilon}_k^+ \tag{5}$$

where $\boldsymbol{\epsilon}^-$ and $\boldsymbol{\epsilon}^+$ can form equality or inequality constraints. Last, to ensure that the waypoints are passed in the given sequence, we enforce subsequent progress variables to be bigger than their prequel at each time step by

$$\lambda_k^j \leq \lambda_k^{j+1} \forall k \in [0, N], j \in [0, M) \tag{6}$$

Note that the last waypoint $\mathbf{p}_{wM}$ is always reached at the last node at t_N and, therefore, could be implemented as a fixed positional constraint on $\mathbf{x}_N$, without loss of generality.

Complementary progress constraints

In the context of waypoint following, the goal is to allow $\boldsymbol{\mu}_k$ to only be nonzero at the time of passing a waypoint. Therefore, $\mathbf{f}_{\text{prog}}$ and $\boldsymbol{\epsilon}^- = \boldsymbol{\epsilon}^+ = 0$ are chosen to represent a complementarity constraint (*25*) as

$$f_{\text{prog},j}(\mathbf{x}_k, \boldsymbol{\mu}_k) = \mu_k^j \cdot \| \mathbf{p}_k - \mathbf{p}_{wj} \|_2^2 \coloneqq 0 \forall j \in [0, M] \tag{7}$$

which can be interpreted as a mathematical NAND (not and) function, because either μ_k^j or $\| \mathbf{p}_k - \mathbf{p}_{wj} \|$ must be 0. Intuitively, the two elements complement each other (Fig. 6).

Tolerance relaxation

With Eq. 7, the trajectory is forced to pass exactly through a waypoint. Not only is this impractical, because, often, a certain tolerance is admitted or even wanted, but also it negatively affects the convergence behavior and time optimality, because the system dynamics are discretized and one of the discrete time steps must coincide with the waypoint. Therefore, it is desirable to relax a waypoint constraint by a certain tolerance, which is achieved by extending Eq. 7 to

$$f_{\text{prog},j}(\mathbf{x}_k, \boldsymbol{\mu}_k) = \mu_k^j \cdot (\| \mathbf{p}_k - \mathbf{p}_{wj} \|_2^2 - \nu_k^j) \coloneqq 0 \; \forall j \in [0, M] \tag{8}$$

$$\text{subject to } 0 \leq \nu_k^j \leq d_{\text{tol}}^2$$

where ν_k^j is a slack variable to allow the distance to the waypoint to be relaxed to zero when it is smaller than d_{tol}, the maximum distance tolerance. This now enforces that the progress variables cannot change, except for the time steps at which the system is within tolerance to the waypoint. Furthermore, please note that the spatial discretization δs depends on the number of nodes N and the speed profile. It should hold that $\delta s < d_{\text{tol}}$, to always allow at least one node to lie within the tolerance, as visualized in Fig. 7. This can be verified after the optimization and approximated beforehand by $\delta s \approx D/N$, where D is the cumulative distance between all waypoints.

Optimization problem summary

The full space of optimization variables $\mathbf{x}$ consists of the overall time and all variables assigned to nodes k as $\mathbf{x}_k$. All nodes k include the robot's dynamic state $\mathbf{x}_{\text{dyn},k}$, its inputs $\mathbf{u}_k$, and all progress variables, $\mathbf{x} = [t_N, \mathbf{x}_0, \ldots, \mathbf{x}_N]$, where

$$\mathbf{x}_k = \begin{pmatrix} [\mathbf{x}_{\text{dyn},k}, \mathbf{u}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k, \boldsymbol{\nu}_k] & \text{for } k \in [0, N) \\ & \text{for } k = N \end{pmatrix} \tag{9}$$

On the basis of this representation, we write the full problem as

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} t_N \tag{10}$$

subject to the system dynamics and initial constraint

$$\mathbf{x}_{k+1} - \mathbf{x}_k - dt \cdot \mathbf{f}_{\text{RK4}}(\mathbf{x}_k, \mathbf{u}_k) = 0 \, \mathbf{x}_0 = \mathbf{x}_{\text{init}} \tag{11}$$

the input constraints

$$\mathbf{u}_{\min} - \mathbf{u}_k \leq 0 \, \mathbf{u}_k - \mathbf{u}_{\max} \leq 0 \tag{12}$$

the progress evolution, boundary, and sequence constraints

$$\begin{gathered} \boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda}_k + \boldsymbol{\mu}_k = 0 \\ \boldsymbol{\lambda}_0 - 1 = 0 \quad \boldsymbol{\lambda}_N = 0 \\ \boldsymbol{\mu}_k \geq 0 \quad \lambda_k^j - \lambda_k^{j+1} \leq 0 \quad \forall k \in [0, N), \quad j \in [0, M) \end{gathered} \tag{13}$$

and the CPC with tolerance

$$\mu_k^j \cdot (\| \mathbf{p}_k - \mathbf{p}_{wj} \|_2^2 - \nu_k^j) = 0 \tag{14}$$

$$-\nu_k^j \leq 0 \quad \nu_k^j - d_{\text{tol}}^2 \leq 0 \tag{15}$$

Note that constraints 14 and 15 are nonlinear because of the norm on distance and the bilinearity of the progress change $\boldsymbol{\mu}$ and tolerance slack $\boldsymbol{\nu}$.

Quadrotor dynamics

The quadrotor's state space is described between the inertial frame I and body frame B, as $\mathbf{x} = [\mathbf{p}_{IB}, \mathbf{q}_{IB}, \mathbf{v}_{IB}, \boldsymbol{\omega}_B]^\top$ corresponding to position $\mathbf{p}_{IB} \in \mathbb{R}^3$, unit quaternion rotation on the rotation group $\mathbf{q}_{\text{IB}} \in \mathbb{SO}(3)$ given $\|\mathbf{q}_{IB}\| = 1$, velocity $\mathbf{v}_{IB} \in \mathbb{R}^3$, and body rate $\boldsymbol{\omega}_B \in \mathbb{R}^3$. The input modality is on the level of collective thrust $\mathbf{T}_B = [0 \quad 0 \quad T_{\text{Bz}}]^\top$ and body torque $\boldsymbol{\tau}_B$. From here on, we drop the frame indices because they are consistent throughout the description. The dynamic equations are

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{v} & \dot{\mathbf{q}} &= \frac{1}{2} \boldsymbol{\Lambda}(\mathbf{q}) \begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix} \\ \dot{\mathbf{v}} &= \mathbf{g} + \frac{1}{m} \mathbf{R}(\mathbf{q}) \mathbf{T} & \dot{\boldsymbol{\omega}} &= \mathbf{J}^{-1}(\boldsymbol{\tau} - \boldsymbol{\omega} \times \mathbf{J}\omega) \end{aligned} \tag{16}$$

where $\boldsymbol{\Lambda}$ represents a quaternion multiplication, $\mathbf{R}(\mathbf{q})$ the quaternion rotation, m the quadrotor's mass, and $\mathbf{J}$ its inertia.

Quadrotor inputs

The input space given by $\mathbf{T}$ and $\boldsymbol{\tau}$ is further decomposed into the single-rotor thrusts $\mathbf{u} = [T_1, T_2, T_3, T_4]$, where T_i is the thrust at rotor $i \in \{1,2,3,4\}$.

$$\mathbf{T} = \begin{bmatrix} 0 \\ 0 \\ \sum T_i \end{bmatrix} \text{ and } \boldsymbol{\tau} = \begin{bmatrix} l/\sqrt{2}(T_1 + T_2 - T_3 - T_4) \\ l/\sqrt{2}(-T_1 + T_2 + T_3 - T_4) \\ c_\tau(T_1 - T_2 + T_3 - T_4) \end{bmatrix} \tag{17}$$

with the quadrotor's arm length l and the rotor's torque constant c_τ. The quadrotor's actuators limit the applicable thrust for each rotor, effectively constraining T_i as

$$0 \leq T_{\min} \leq T_i \leq T_{\max} \tag{18}$$

In Fig. 7, we visualize the acceleration space and the thrust torque space of a quadrotor in the xz plane. Note that the acceleration space in Fig. 7 is nonconvex because of $T_{\min} > 0$ for the depicted model parameters from the standard configuration of table S1. The torque space is visualized in Fig. 7, where the coupling between the achievable thrust and torque is visible.

Approximative linear aerodynamic drag

Last, we extend the quadrotor's dynamics to include a linear drag model (*33*), to approximate the most dominant aerodynamic effects with diagonal matrix $\mathbf{D}$ by

$$\dot{\mathbf{v}} = \mathbf{g} + \frac{1}{m} \mathbf{R}(\mathbf{q}) \mathbf{T} - \mathbf{R}(\mathbf{q}) \mathbf{D} \mathbf{R}^\top(\mathbf{q}) \cdot \mathbf{v} \tag{19}$$

where we approximate $\mathbf{D} = \text{diag}\,(d_x, d_y, d_z)$ in this work.

SUPPLEMENTARY MATERIALS

robotics.sciencemag.org/cgi/content/full/6/56/eabh1221/DC1
Tables S1 to S3
Figs. S1 to S8
Preface: Time-optimal quadrotor trajectory
Simulation experiments
References (*34*–*38*)

REFERENCES AND NOTES

1. G. Loianno, D. Scaramuzza, Special issue on future challenges and opportunities in vision-based drone navigation. *J. Field Robot.* **37**, 495–496 (2020).
2. E. Ackermann, Ai-powered drone learns extreme acrobatics, https://spectrum.ieee.org/automaton/robotics/drones/ai-powered-drone-extreme-acrobatics (2020) [accessed 21 June 2021].
3. J. Verbeke, J. D. Schutter, Experimental maneuverability and agility quantification for rotary unmanned aerial vehicle. *Int. J. Micro Air Veh.* **2018**, 3–11 (2018).
4. H. Moon, J. Martinez-Carranza, T. Cieslewski, M. Faessler, D. Falanga, A. Simovic, D. Scaramuzza, S. Li, M. Ozo, C. De Wagter, G. de Croon, S. Hwang, S. Jung, H. Shim, H. Kim, M. Park, T. -C. Au, S. J. Kim, Challenges and implemented technologies used in autonomous drone racing. *J. Intel. Service Robot.* **12**, 137–148 (2019).
5. P. Foehn, D. Brescianini, E. Kaufmann, T. Cieslewski, M. Gehrig, M. Muglikar, D. Scaramuzza, AlphaPilot: Autonomous drone racing, in *Proceedings of Robotics: Science and Systems* (RSS, 2020), pp. 1–9.
6. A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, D. Scaramuzza, Deep drone racing: From simulation to reality with domain randomization. *IEEE Transact. Robot.* **36**, 1–14 (2019).
7. S. M. LaValle, *Planning Algorithms* (Cambridge Univ. Press, 2006).
8. L. S. Pontryagin, E. Mishchenko, V. Boltyanskii, R. Gamkrelidze, *The Mathematical Theory of Optimal Processes* (Wiley, 1962).
9. S. Bouabdallah, R. Siegwart, Full control of a quadrotor, in *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2007), pp. 153–158.
10. R. Mahony, V. Kumar, P. Corke, Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robot. Autom. Mag.* **19**, 20–32 (2012).
11. D. Mellinger, N. Michael, V. Kumar, Trajectory generation and control for precise aggressive maneuvers with quadrotors. *Int. J. Robot. Res.* **31**, 664–674 (2012).
12. M. W. Mueller, M. Hehn, R. D'Andrea, A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification, in *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2013), pp. 3480–3486.
13. D. J. Webb, J. van den Berg, Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics, in *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2013), pp. 5054–5061.
14. R. Allen, M. Pavone, A real-time framework for kinodynamic planning with application to quadrotor obstacle avoidance, in *AIAA Guidance, Navigation, and Control Conference*, 4 to 8 January 2016, San Diego, CA (2016).
15. S. Liu, K. Mohta, N. Atanasov, V. Kumar, Search-based motion planning for aggressive flight in SE(3). *IEEE Robot. Autom. Lett.* **3**, 2439–2446 (2018).
16. B. Zhou, F. Gao, L. Wang, C. Liu, S. Shen, Robust and efficient quadrotor trajectory generation for fast autonomous flight. *IEEE Robot. Autom. Lett.* **4**, 3529–3536 (2019).
17. C. R. Hargraves, S. W. Paris, Direct trajectory optimization using nonlinear programming and collocation. *J. Guid. Control Dynam.* **10**, 338–342 (1987).
18. M. Geisert, N. Mansard, Trajectory generation for quadrotor based systems using numerical optimal control, in *2016 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2016), pp. 2958–2964.
19. F. Augugliaro, A. P. Schoellig, R. D'Andrea, Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2012), pp. 1917–1922.

20. B. Penin, R. Spica, P. Robuffo Giordano, F. Chaumette, Vision-based minimum-time trajectory generation for a quadrotor UAV, in *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2017), pp. 6199–6206.
21. M. Hehn, R. Ritz, R. D'Andrea, Performance benchmarking of quadrotor systems using time-optimal control. *Auton. Robot.* **33**, 69–88 (2012).
22. W. V. Loock, G. Pipeleers, J. Swevers, Time-optimal quadrotor flight, in *Proceedings of the 2013 European Control Conference (ECC)* (IEEE, 2013), pp. 1788–1792.
23. S. Spedicato, G. Notarstefano, Minimum-time trajectory generation for quadrotors in constrained environments. *IEEE Trans. Control Syst. Technol.* **26**, 1335–1344 (2018).
24. G. Ryou, E. Tal, S. Karaman, Multi-fidelity black-box optimization for time-optimal quadrotor maneuvers. arXiv:2006.02513 [cs.RO] (3 June 2020).
25. M. Posa, C. Cantu, R. Tedrake, A direct method for trajectory optimization of rigid bodies through contact. *Int. J. Robot. Res.* **33**, 69–81 (2014).
26. R. W. Cottle, G. B. Dantzig, Complementary pivot theory of mathematical programming. *Linear Algebra Appl.* **1**, 103–125 (1968).
27. A. Richards, J. P. How, Aircraft trajectory planning with collision avoidance using mixed integer linear programming, in *Proceedings of the 2002 American Control Conference (ACC)* (IEEE, 2002), pp. 1936–1941.
28. D. Falanga, P. Foehn, P. Lu, D. Scaramuzza, PAMPC: Perception-aware model predictive control for quadrotors, in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2018), pp. 1–8.
29. B. Houska, H. J. Ferreau, M. Diehl, ACADO toolkit—An open-source framework for automatic control and dynamic optimization. *Optimal Control Appl. Methods* **32**, 298–312 (2011).
30. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, M. Diehl, qpOASES: A parametric active-set algorithm for quadratic programming. *Math. Program. Comput.* **6**, 327–363 (2014).
31. M. Diehl, H. G. Bock, H. Diedam, P. B. Wieber, *Fast Motions in Biomechanics and Robotics* (Springer, 2006).
32. C. Pfeiffer, D. Scaramuzza, Human-piloted drone racing: Visual processing and control. *IEEE Robot. Automat. Lett.* **6**, 3467–3474 (2021).
33. M. Faessler, A. Franchi, D. Scaramuzza, Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robot. Autom. Lett.* **3**, 620–626 (2018).
34. D. Mellinger, V. Kumar, Minimum snap trajectory generation and control for quadrotors, in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2011), pp. 2520–2525.
35. J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, M. Diehl, CasADi: A software framework for nonlinear optimization and optimal control. *Math. Program. Comput.* **11**, 1–36 (2019).
36. A. Wächter, L. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.* **106**, 25–57 (2006).
37. S. Shah, D. Dey, C. Lovett, A. Kapoor, in *Field and Service Robotics* (Springer, 2017), pp. 621–635.
38. R. Madaan, N. Gyde, S. Vemprala, M. Brown, K. Nagami, T. Taubner, E. Cristofalo, D. Scaramuzza, M. Schwager, A. Kapoor, AirSim Drone Racing Lab, in *Proceedings of the NeurIPS 2019 Competition and Demonstration Track*, Proceedings of Machine Learning Research Series, vol. 123, pp. 177–191 (2019).

Acknowledgments: We thank the two professional human pilots, M. Isler and T. Trowbridge, for their participation. Without the said pilots' incredible skills, establishing a fair human baseline would not have been possible. **Funding:** This work was supported by the National Centre of Competence in Research (NCCR) Robotics through the Swiss National Science Foundation (SNSF) and the European Union's Horizon 2020 Research and Innovation Program under grant agreement no. 871479 (AERIAL-CORE) and the European Research Council (ERC) under grant agreement no. 864042 (AGILEFLIGHT). **Ethics statement:** The study protocol has been approved by the local ethics committee of the University of Zurich. The participants gave their written informed consent before participating in the study and gave their written consent to be mentioned by name and listed with their competition participation record. **Author contributions:** P.F. developed and implemented the problem formulation using CPCs and designed and performed all simulation experiments. P.F. and A.R. performed all real-world experiments and wrote the manuscript. D.S. provided funding, contributed to the design and analysis of the experiments, and revised the manuscript. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** All materials are described within this paper and completed by the dataset and code necessary to reproduce our results, which can be found at https://datadryad.org/stash/dataset/doi:10.5061/dryad.9kd51c5h7. All data needed to evaluate the conclusions of this paper are available in the paper or the Supplementary Materials.

Submitted 17 February 2021
Accepted 23 June 2021
Published 21 July 2021
10.1126/scirobotics.abh1221

Citation: P. Foehn, A. Romero, D. Scaramuzza, Time-optimal planning for quadrotor waypoint flight. *Sci. Robot.* **6**, eabh1221 (2021).

Science Robotics

Time-optimal planning for quadrotor waypoint flight

Philipp Foehn, Angel Romero, and Davide Scaramuzza

Sci. Robot. **6** (56), eabh1221. DOI: 10.1126/scirobotics.abh1221

Science Robotics (ISSN 2470-9476) is published by the American Association for the Advancement of Science. 1200 New York Avenue NW, Washington, DC 20005. The title *Science Robotics* is a registered trademark of AAAS.