ANNUAL REVIEWS

Annual Review of Control, Robotics, and Autonomous Systems

Autonomous Flight

Sarah Tang and Vijay Kumar

General Robotics, Automation, Sensing, and Perception (GRASP) Laboratory, University of Pennsylvania, Philadelphia, Pennsylvania 19104, USA; email: sytang@seas.upenn.edu, kumar@seas.upenn.edu

Annu. Rev. Control Robot. Auton. Syst. 2018. 1:29–52

First published as a Review in Advance on January 12, 2018

The *Annual Review of Control, Robotics, and Autonomous Systems* is online at control.annualreviews.org

https://doi.org/10.1146/annurev-control-060117-105149

Keywords

quadrotors, quadrocopters, aerial robots, unmanned aerial vehicles, UAVs

Abstract

This review surveys the current state of the art in the development of unmanned aerial vehicles, focusing on algorithms for quadrotors. Tremendous progress has been made across both industry and academia, and full vehicle autonomy is now well within reach. We begin by presenting recent successes in control, estimation, and trajectory planning that have enabled agile, high-speed flight using low-cost onboard sensors. We then examine new research trends in learning and multirobot systems and conclude with a discussion of open challenges and directions for future research.

1. INTRODUCTION

In the past decade, the field of aerial robotics has exploded. Tremendous progress has been made in the design and autonomy of flight systems, which can roughly be categorized as either man-made or bio-inspired. In the former class, fixed-pitch rotorcrafts have become particularly popular for their mechanical simplicity, but several more complex vehicles have also been developed. For example, decoupled rotational and translational degrees of freedom can be achieved with variable-pitch (1, 2) or omnidirectional (3) rotor configurations. Passive vehicle stability—the ability to fly stably without active attitude control—can be accomplished with the addition of appropriately designed cages (4), lowering vehicles' energy consumption and increasing their robustness to failures of inertial sensors.

Nonetheless, these mechanical designs are ultimately structurally rigid. Winged animals, by contrast, can dynamically change their body shapes to switch between flapping and gliding flight and to perform complex maneuvers such as perching and diving. As a result, designing bio-inspired ornithopters has also become an active research area. Characterizing the complex, unsteady aerodynamic flows around the wings of these vehicles, particularly during flapping flight, is a challenging task. Such flows are especially difficult to characterize at insect scales, where robot components and force variations can be on the orders of millimeters and micronewtons, respectively. Nonetheless, several at-scale experiments studying fluid flow around small wings (5) and characterizing various wing parameters (6) have quantified some of the effects, and these results have enabled the successful development of insect-scale flying robots (7, 8).

For avian-scale vehicles, emulating animals' complex wing structures presents an additional challenge. For example, bats' wings have more than 40 degrees of freedom. As mechanically recreating each degree is impractical, principal component analysis has been used to identify lower-dimensional vector spaces in which flapping actions can be represented and guide the design of systems with fewer degrees of freedom (9). This technique has been used to design the Bat Bot (B2) and to automate flying and diving maneuvers (10). Other platforms at similar scales have emulated hummingbirds (11), ravens (12), and flies (13).

Fixed-wing aircrafts are mechanically simpler than ornithopters but are still able to glide and execute birdlike maneuvers, such as perching (14). Compared with ornithopters, they are easier to model and are typically better able to carry onboard sensors, allowing for the successful development of planning (15), perception (16), and formation control (17) algorithms.

Multirotor aircrafts, particularly quadrotors, have been the most capable vehicles in terms of accessibility, maneuverability, capacity for onboard sensors, and applicability to a breadth of applications. Great research progress has been made across multiple areas, including the control of agile maneuvers; planning and perception in unknown, unstructured environments; and collaboration in multiagent teams, sparking a surge of industry investment. The first half of 2017 saw \$216 million in venture capital funding for drone companies (18), spanning such applications as infrastructure inspection, rapid package delivery, precision agriculture, disaster response, and choreographed performances.

Most significantly, there has been a major increase in the availability of off-the-shelf quadrotor platforms. **Figure 1** illustrates several popular platforms and their relative sizes with respect to large-scale industrial unmanned aerial vehicles. These commercial platforms are available in a range of sizes, allowing for different sensor and computation payloads and operation in different indoor and outdoor environments, but they are all orders of magnitude smaller, lighter, and more agile than industrial vehicles, which in turn has allowed researchers to develop novel algorithms for autonomy. This progress motivated the 2015 launch of the Defense Advanced Research Projects Agency (DARPA) Fast Lightweight Autonomy program, which sponsors research

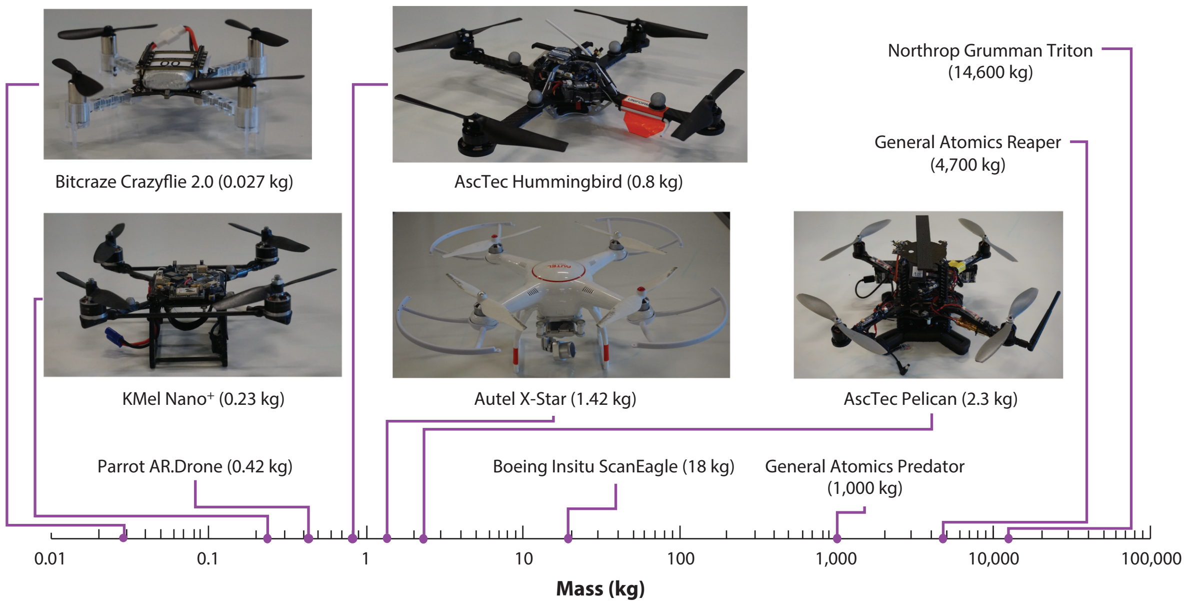

Figure 1

Examples of commercial and government unmanned aerial vehicle platforms.

teams to work toward fully autonomous quadrotor navigation at high speeds using only onboard resources.

The research efforts of the government, industry, and academia have positioned quadrotors to become one of the first flying platforms with full end-to-end autonomy between high-level task specification and motor actuation. This review discusses research progress toward this vision, focusing on advancements in the past five years (for a survey of earlier work, see 19). The first part of the review focuses on high-impact successes in model-based techniques for single-quadrotor systems: Section 2 describes mathematical modeling and control algorithms; Section 3 provides an overview of methods for state estimation and perception; Section 4 presents computationally efficient, real-time planning techniques; and Section 5 discusses applications of these techniques to agile, high-speed flight. The remainder of the review discusses new research frontiers: Section 6 presents methods for learning and adaptation, and Section 7 discusses the incorporation of quadrotors into multiagent teams. Finally, Section 8 concludes with a discussion of open problems and future challenges.

2. DYNAMICS AND CONTROL

The quadrotor (illustrated in **Figure 2**) is a vehicle with four propellers, each rigidly mounted at a distance L from the center of mass. The front and back rotors rotate anticlockwise, while the left and right rotors rotate clockwise. Modulating the rotor thrusts allows the vehicle to roll, pitch, and yaw, which in turn induces translational motion.

Let $\mathcal{I} = \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ be the inertial world frame. Let $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3\}$ be an intermediate frame after yaw rotation ψ and $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$ be a body frame fixed to the vehicle. Let $\mathbf{x}_Q \in \mathbb{R}^3$ represent the position of the vehicle's center of mass relative to $\mathcal{I}$, $R \in \mathrm{SO}(3)$ represent the body-to world-frame rotation matrix (the body-frame components of vector ${}^{\mathcal{B}}\mathbf{v}$ can be translated into

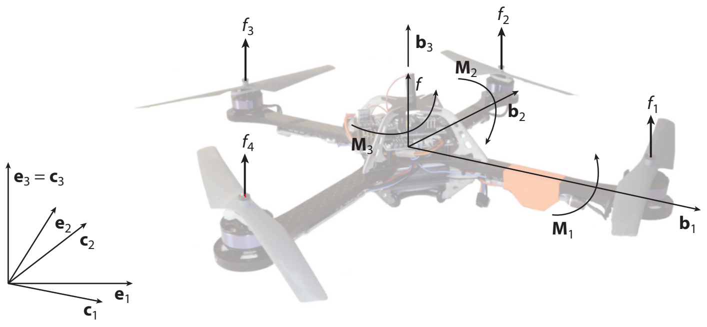

Figure 2

Mathematical model of a quadrotor. The front propeller is on the right.

world-frame components ${}^{\mathcal{I}}\mathbf{v}$, with ${}^{\mathcal{I}}\mathbf{v} = R^{\mathcal{B}}\mathbf{v}$), and $\Omega \in \mathbb{R}^3$ represent the angular velocity expressed in $\mathcal{B}$. The system's dynamics evolve on SE(3), with state

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_Q^\top & \dot{\mathbf{x}}_Q^\top & R & \Omega^\top \end{bmatrix}^\top, \qquad 1.$$

and the input is

$$\mathbf{u} = \begin{bmatrix} f & \mathbf{M}^\top \end{bmatrix}^\top \in \mathbb{R}^4. \qquad 2.$$

The thrust magnitude, f, acts in the $\mathbf{b}_3$ direction, and the moment, $\mathbf{M}$, acts about the body-frame axes. These result from individual thrust forces from each rotor, f_i:

$$\begin{bmatrix} f \\ \mathbf{M} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & L & 0 & -L \\ -L & 0 & L & 0 \\ -\mu & \mu & -\mu & \mu \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, \qquad 3.$$

where μ is a constant representing the ratio of drag to lift force produced by each propeller.

Let m_Q represent the vehicle's mass, J represent its inertia tensor in $\mathcal{B}$, and g represent the positive gravity constant. The quadrotor's equations of motion can then be derived:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{x}}_Q \\ \frac{f}{m_Q} R\mathbf{e}_3 - g\mathbf{e}_3 \\ R\hat{\Omega} \\ J^{-1}\left(\mathbf{M} - \Omega \times J\Omega\right) \end{bmatrix}. \qquad 4.$$

The hat operator, $\hat{\cdot}$, maps a vector to a skew-symmetric matrix such that $\hat{\mathbf{x}}\mathbf{y} = \mathbf{x} \times \mathbf{y}$.

During autonomous operation, the quadrotor must track desired trajectories $\mathbf{x}^{\text{des}}(t)$. A significant control challenge is handling the quadrotor's nonlinear and underactuated dynamics. Fortunately, it can be shown that the dynamics are differentially flat; that is, there exists a set of flat outputs, $\mathbf{x}^f$, such that the model's states and inputs can be written as functions of $\mathbf{x}^f$ and its higher derivatives (20). Mellinger & Kumar (21) showed that there exists a diffeomorphism between Equations 1 and 2 and the vector:

$$\begin{bmatrix} \mathbf{x}_Q^\top & \dot{\mathbf{x}}_Q^\top & \ddot{\mathbf{x}}_Q^\top & \dddot{\mathbf{x}}_Q^\top & \ddddot{\mathbf{x}}_Q^\top & \psi & \dot{\psi} & \ddot{\psi} \end{bmatrix}^\top. \qquad 5.$$

Thus, the flat outputs are

$$\mathbf{x}^f = \begin{bmatrix} \mathbf{x}_Q^\top & \psi \end{bmatrix}^\top \in \mathbb{R}^4. \qquad 6.$$

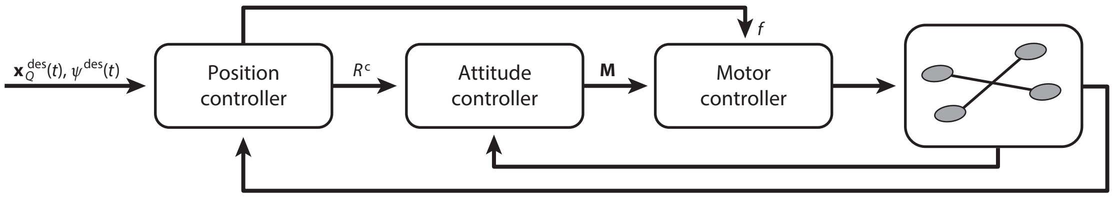

Figure 3

Architecture of a hierarchical quadrotor controller.

This is a powerful observation. When planning trajectories in the full state space $\mathbf{x}$, Equation 4 must be included as nonlinear constraints. However, any sufficiently continuous trajectories in $\mathbf{x}^f$ can always be associated with a full state trajectory that satisfies Equation 4.

This facilitates the design of a hierarchical controller, illustrated in **Figure 3**. The superscript "des" references desired values calculated by a trajectory planner, while the superscript "c" (for "commanded") references intermediate values calculated by the controller. The outermost position controller is typically a traditional proportional–derivative controller:

$$\mathbf{F} = -k_x(\mathbf{x}_Q - \mathbf{x}_Q^{\text{des}}) - k_v(\dot{\mathbf{x}}_Q - \dot{\mathbf{x}}_Q^{\text{des}}) + m_Q(\ddot{\mathbf{x}}_Q^{\text{des}} + g\mathbf{e}_3), \tag{7.}$$

where k_x and k_v are positive gain matrices. From this,

$$f = \mathbf{F} \cdot R\mathbf{e}_3. \tag{8.}$$

To derive R^c, the third body-frame vector can be defined as

$$\mathbf{b}_3^c = \frac{\mathbf{F}}{\|\mathbf{F}\|_2}, \tag{9.}$$

and a desired direction for the intermediate frame axis, $\mathbf{c}_1^c$, can be defined with

$$\mathbf{c}_1^c = [\cos(\psi^{\text{des}}) \;\; \sin(\psi^{\text{des}}) \;\; 0]^\top. \tag{10.}$$

Assuming R is a z–x–y rotation matrix, axes $\mathbf{b}_1$ and $\mathbf{c}_1$ are coplanar, and the remaining body-frame vectors are

$$\mathbf{b}_2^c = \frac{\mathbf{b}_3^c \times \mathbf{c}_1^c}{\|\mathbf{b}_3^c \times \mathbf{c}_1^c\|_2}, \tag{11.}$$

$$\mathbf{b}_1^c = \mathbf{b}_2^c \times \mathbf{b}_3^c. \tag{12.}$$

Note that $\mathbf{x}b_3^c \times \mathbf{c}_1^c \neq 0$; that is, the thrust vector cannot be parallel to $\mathbf{c}_1$. The desired orientation is then

$$R^c = \left[\mathbf{b}_1^c \;\; \mathbf{b}_2^c \;\; \mathbf{b}_3^c\right]. \tag{13.}$$

From here, an attitude controller commands an input moment, $\mathbf{M}$, to track R^c, and an innermost controller calculates individual motor commands. In practice, each inner loop must run significantly faster than its previous layer. Thus, the position control loop can optionally be executed on an external base station, but the speeds demanded of the attitude and motor controllers require them to run on onboard processors.

Proposed attitude controllers have commonly taken the form

$$\mathbf{M} = -k_R e_R(R, R^c) - k_\Omega e_\Omega(\Omega, \Omega^c) + \mathbf{M}_{\text{ff}}, \tag{14.}$$

where k_R and k_Ω are positive gain matrices, e_R and e_Ω are error functions, and $\mathbf{M}_{\text{ff}}$ is a feedforward moment. In earlier works, the dynamics were linearized about the hover equilibrium and the attitude controller was implemented in Euler angles (22). This linearization performs well for roll and pitch angles up to approximately 30°, corresponding to linear velocities of approximately 1.5–2 m/s; however, nonlinear controllers have largely eliminated these limitations.

For example, in a proposed geometric controller (23),

$$e_R(R, R_c) = \frac{1}{2}((R^c)^\top R - R^\top R^c)^\vee, \tag{15.}$$

where $\vee$ is the map from a skew-symmetric matrix to its vector representation. This defines the attitude error directly on the manifold SO(3) (i.e., Equation 15 returns the axis-angle representation of the rotation from R to R^c). Similarly, the angular velocity error is

$$e_\Omega(\Omega, \Omega^c) = \Omega - R^\top R^c \Omega^c, \tag{16.}$$

where

$$\Omega^c = (R^c)^\top \dot{R}^c. \tag{17.}$$

$\dot{R}^c$ can be found by differentiating Equation 13 and substituting values calculated from differential flatness equations. Finally, the feedforward moment is defined as

$$\mathbf{M}_{\text{ff}} = \Omega \times J\Omega - J\left(\Omega \times R^\top R^c \Omega^c - R^\top R^c \dot{\Omega}^c\right), \tag{18.}$$

where $\dot{\Omega}^c$ can again be found through differential flatness. By expressing the control equations in a coordinate-free, singularity-free manner, this controller can guarantee almost global exponential attractiveness (23), and in practice, it can control the quadrotor at almost inverted orientations (24). Yu et al. (25) presented a similar formulation but decomposed the attitude controller to track the fast-response roll and pitch angles before the slow-response yaw angle, and Brescianini et al. (26) presented a quaternion-based controller that also has singularity-free, globally asymptotic stability properties.

The design of the innermost motor controller is also crucial. This controller typically involves two steps. In the first step, the calculated inputs f and $\mathbf{M}$ must be translated into individual thrusts f_i. Assuming that f_i can be achieved almost instantaneously, Equation 3 can simply be inverted. This assumption has proven practical for slow-moving vehicles; for high-speed flight, however, incorporating a first-order dynamic model for f_i can noticeably improve performance (27). Furthermore, in many works, when the calculated f_i exceeds a set maximum, the thrust is simply clipped to $f_i = f_{\max}$. However, this can result in unpredictable tracking errors. Thrust-mapping methods that deliberately reassign f_i according to set criteria (e.g., prioritizing achieving the desired roll and pitch over yaw) can result in more robust behavior (27, 28). Faessler et al. (27) further showed that modeling μ as a variable dependent on f_i also improves performance. Alternatively, nonlinear model predictive control schemes can be used to directly obtain outputs f_i to track the desired R^c and Ω^c (29). Here, an optimal control problem is solved in a receding-horizon fashion to yield the motor thrusts at each time step. This formulation has the advantage that thrust and angular velocity limits can be directly incorporated into the optimization, but it is more computationally expensive.

In the second step, motor commands $\omega_{r,i}$ must be computed from forces f_i. A basic model relates

$$f_i = k_F \omega_{r,i}, \tag{19.}$$

where k_F is a motor constant found through calibration. Recently, however, researchers have begun to adopt a polynomial model (27, 30):

$$f_i = k_2 \omega_{r,i}^2 + k_1 \omega_{r,i} + k_0. \tag{20.}$$

Bangura & Mahony (31) further proposed a hierarchical motor controller that explicitly models aerodynamic effects.

A final challenge is maintaining safety and stability in the presence of unexpected disturbances. Small aerodynamic effects that are usually rejected as unmodeled disturbances become significant near surfaces (e.g., the ground or obstacle walls) and other vehicles (32). In these cases, the effects must be explicitly modeled through techniques such as experimental identification of drag constants (33) or numerical calculation using a computational fluid dynamics solver (34). Alternatively, Yao et al. (35) predicted future disturbances from past observations in flight, and Bartholomew et al. (36) predicted aerodynamic effects around obstacles using observed depth images and prior training data. For safety in the presence of mechanical failures, Mueller & D'Andrea (37) experimentally demonstrated controllers that enable stable flight even following the loss of one or two propellers.

These sophisticated models and nonlinear control strategies have enabled quadrotors to perform maneuvers that fully utilize the systems' entire range of motion, such as perching on vertical surfaces (38), a feat that would not be feasible with previous linearized controllers.

3. STATE ESTIMATION AND PERCEPTION

Early implementations of aggressive maneuvers with nonlinear controllers were demonstrated solely in motion-capture spaces. Enabling robots to execute such motions in real-world environments additionally requires accurate and robust state estimators, and in cluttered environments, the robots must also construct maps of their surroundings. These tasks are particularly challenging for aerial robots. Unlike ground vehicles, which are constrained to a plane, quadrotors operate in a full three-dimensional workspace. To achieve high accelerations, vehicles need high thrust-to-weight ratios, and to guarantee safe stopping from high speeds, they need large perception ranges. Their maximum velocities are therefore limited by the weight, range, and accuracy of their sensors and the computational speed of their perception and estimation algorithms. Unfortunately, information-rich, three-dimensional sensors have relatively limited fields of view, as in the case of RGB-D (red, green, and blue plus depth) sensors, or are excessively heavy, as in the case of 3-D lidars.

In this light, the combination of one or more cameras with an inertial measurement unit (IMU) has become an attractive low-cost, lightweight, low-power sensor suite for quadrotors. These two sensors offer several complementary capabilities. Camera images are typically used to extract position and velocity information, while IMU measurements report rotational velocity and linear acceleration; image processing offers accurate measurements at slower rates, while IMU readings provide high-frequency measurements; and vision-based tracking is more precise at lower velocities, while inertial measurements have lower uncertainties at high velocities.

The development of visual odometry and visual–inertial odometry methods has progressed in both the computer vision and robotics communities. In this section, we focus on algorithms validated on quadrotor platforms. We refer readers to References 39 and 40 for a tutorial on visual odometry, References 41 and 42 for an introduction to visual–inertial odometry, Reference 43 for a comparison of current open-source visual–inertial odometry implementations, and References 44 and 45 for details on related algorithms.

A formal observability analysis shows that, under proper vehicle motions, a single camera and IMU are sufficient to estimate a vehicle's position, velocity, and roll and pitch orientation (46). Many works have achieved accurate estimation with this minimal sensor suite. Feature-based visual odometry estimates position by tracking chosen features between image frames; Troiani et al. (47) showed that pose estimates can be derived from a single camera frame viewing only three features and that outlier rejection can be achieved by viewing only one or two features. However, feature tracking becomes difficult in the presence of motion blur and in textureless environments, causing other researchers to turn to direct methods that operate on image pixel intensities. Such techniques are typically computationally expensive, but advances in processor capabilities have allowed successful onboard implementations of more robust direct visual odometry methods (48). Hybrid algorithms that leverage the advantages of both approaches have also been successful (49, 50).

Vision data can be fused with IMU measurements using graph-based (51–54) or filtering-based approaches (55). For Gaussian filters, the quadrotor's non-Euclidean configuration space and nonlinear dynamics must be handled carefully. Goodarzi & Li (56) showed that tracking performance in an extended Kalman filter can be improved by defining variations on the appropriate configuration manifold during linearization. An alternative to the Taylor series approximation of the extended Kalman filter is the unscented Kalman filter (57), which uses the unscented transform to approximate the nonlinear transformation of the belief state between updates. This transform and the unscented Kalman filter itself are generalizable to Riemannian manifolds (58) and Lie groups (59). Loianno et al. (60) successfully used an unscented Kalman filter in the configuration space SE(3) for accurate, singularity-free state estimation. In particular, they demonstrated that, unlike Euler angle–based parameterizations, this formulation on the manifold is robust to extreme motions, such as a 360° rotation about the camera axis.

A major drawback of monocular camera configurations, however, is the need for an initialization phase before key parameters (such as scale) can be reliably estimated. This becomes particularly problematic when visual tracking is lost midflight. Faessler et al. (61) proposed a fallback method with which a quadrotor that has lost tracking can still self-stabilize and reinitialize its estimator. B-splines have been proposed for robust initialization at high altitudes (62). Alternatively, References 52 and 63–65 showed that integrating the terms of the IMU propagation equations in the appropriate frame (often referred to as IMU preintegration) allows the initialization phase to be eliminated. Stereo (66) and other multicamera configurations (67), which do not require an explicit initialization step, have also been proposed.

These state estimation advances represent significant progress toward maneuvering in obstacle-free, GPS-denied environments. However, to navigate cluttered spaces, robots must further maintain, update, and orient themselves with respect to a map. Recent work has shown that obstacle detection for reactive collision avoidance is possible at high speeds on a CPU (68) running onboard a fixed-wing aircraft traveling at up to 14 m/s. However, maintaining a complete map for longer-range planning is computationally expensive, and as a result, most mapping algorithms have been executed using either offboard resources (50) or onboard specialized hardware, such as GPUs (53) or field-programmable gate arrays (69). These mapping algorithms can be combined with trajectory-generation methods to enable autonomous navigation through cluttered environments (49, 53, 68).

Several challenges remain. While autonomous navigation to operator-assigned goals has been successful, autonomous exploration, in which robots intelligently determine new goals to navigate to, has not been achieved. In addition, knowledge of the relative transformations between mounted cameras and inertial sensors is often required, motivating the development of self-calibration methods (63, 70–72). Furthermore, the performance of estimation algorithms depends heavily on robots' motions. For example, for monocular sensor suites, hovering and constant-velocity flight do not allow observation of metric scale (46). Thus, recent works have proposed trajectory-generation algorithms that incorporate measures of system observability (73, 74). As another example, robots must often keep obstacles in their sensors' fields of view (75). This has led to the emergence of active perception, or active vision, where perception constraints are explicitly considered in planning strategies. Section 4 further discusses trajectory-generation techniques.

4. TRAJECTORY GENERATION

The trajectory-generation problem can be stated as follows: Find a trajectory $\mathbf{x}^{\text{des}}(t)$ that brings the vehicle from its current state, $\mathbf{x}^0$, to a desired goal state, $\mathbf{x}^1$. This problem poses three main challenges. First, the trajectory must be dynamically feasible (i.e., there must be a solution to Equation 4) and satisfy motor and actuator constraints (i.e., the upper and lower thrust bounds and maximum angular velocity). Second, the trajectory-generation method must be real time, that is, able to be replan trajectories in milliseconds to respond to events. Finally, the trajectory must be safe and not lead to collisions with obstacles in the environment.

Traditional search (e.g., A* or D* search) and sampling-based planners (e.g., rapidly exploring random trees) have generally performed poorly for quadrotors, while optimization-based approaches have proven to be well suited to handling the vehicle's complex, nonlinear constraints and high-dimensional state space. The trajectory-generation problem can be formulated as an optimal control problem (76):

$$\begin{aligned}
\min_{\mathbf{u}\in\mathbf{U}_{\text{fes}}} \quad & \int_0^T \mathcal{L}(t,\mathbf{x}(t),\mathbf{u}(t))\mathrm{d}t + \mathcal{K}(T,\mathbf{x}^1)\\
\text{subject to} \quad & \text{Dynamic constraints: } \dot{\mathbf{x}} = f(\mathbf{x}(t),\mathbf{u}(t)),\\
& \text{State constraints: } \mathbf{x}(t) \in \mathbf{X}_{\text{fes}} \quad \forall t \in [0,T],\\
& \text{Initial constraints: } \mathbf{x}(0) = \mathbf{x}^0,\\
& \text{Final constraints: } \mathbf{x}(T) = \mathbf{x}^1.
\end{aligned} \qquad 21.$$

$f(\mathbf{x},\mathbf{u})$ is given by Equation 4. $\mathbf{X}_{\text{fes}}$ represents the space of feasible states where the robot is not in collision with an obstacle or in violation of the angular velocity maximum. $\mathbf{U}_{\text{fes}}$ represents the space of feasible thrust inputs. The trajectory end time, T, can be fixed or free. Equation 21 is not directly solvable, and the remainder of this article focuses on approximations and abstractions that reformulate Equation 21 to facilitate real-time computation of solutions.

4.1. Computation of Dynamically Feasible Trajectories

To begin, assume an obstacle-free space. A common abstraction assumes a fast attitude controller and models the system input as $\tilde{\mathbf{u}} = \left[f \;\; (\Omega^c)^\top\right]^\top$. This abstracted system has states $\tilde{\mathbf{x}} = \left[\mathbf{x}_Q^\top \;\; \dot{\mathbf{x}}_Q^\top \;\; R\right]^\top$. For agile flight, the goal is often to find time-optimal trajectories:

$$\begin{aligned}
\min_{\tilde{\mathbf{u}}\in\tilde{\mathbf{U}}_{\text{fes}}} \quad & \int_0^T \mathrm{d}t\\
\text{subject to} \quad & \text{Dynamic constraints: } \dot{\tilde{\mathbf{x}}} = f(\tilde{\mathbf{x}}(t),\tilde{\mathbf{u}}(t)),\\
& \text{Initial constraints: } \tilde{\mathbf{x}}(0) = \tilde{\mathbf{x}}^0,\\
& \text{Final constraints: } \tilde{\mathbf{x}}(T) = \tilde{\mathbf{x}}^1.
\end{aligned} \qquad 22.$$

While the Hamilton–Jacobi–Bellman equation gives a partial differential equation for the globally optimal solution, it typically cannot be solved analytically. As a result, Pontryagin's minimum principle—a necessary (but not sufficient) condition for optimality posed as an ordinary differential equation—is often used instead. Using Roxin's theorem, Hehn et al. (77) showed that, in the absence of obstacles, a unique solution to the minimum-principle conditions always exists between two desired states. The optimal input trajectories can be found numerically with a switching-time optimization technique. Unfortunately, executing this optimization for even a two-dimensional dynamic model requires significant computation time (77) and is impractical for real-time implementation.

In this light, Hehn & D'Andrea (78) proposed an approximation to decouple the translational degrees of freedom. The dynamics in each dimension are approximated as simple third-order systems with states $\mathbf{s} = [s \;\; \dot{s} \;\; \ddot{s}]^\top$ and inputs u_s, where $s \in \{x, y, z\}$. The problem in each dimension becomes

$$\begin{aligned} \min_{u_s \leq u_{\max,s}} \quad & \int_0^T \mathrm{d}t \\ \text{subject to} \quad & \text{Dynamic constraints: } \dot{\mathbf{s}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{s} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_s, \\ & \text{State constraints: } a_{\min,s} \leq \ddot{s} \leq a_{\max,s}, \\ & \text{Initial constraints: } \mathbf{s}(0) = \begin{bmatrix} s^0 & \dot{s}^0 \, f_s(R^0) \end{bmatrix}^\top, \\ & \text{Final constraints: } \mathbf{s}(T) = \mathbf{s}^1. \end{aligned} \qquad 23.$$

The function $f_s(R^0)$ maps the initial orientation to an initial acceleration in each dimension. Acceleration and jerk constraints approximate the original thrust and angular velocity constraints, respectively. Equation 23 is a linear, one-dimensional optimal control problem that can be solved quickly. Hehn & D'Andrea (78) demonstrated that, for computing trajectories to rest (i.e., $\mathbf{s}^1 = \mathbf{0}$), Equation 23 reduces the computation time to microseconds. This method can replan trajectories at high frequencies and can even be used as a model predictive control–like position controller.

Mueller et al. (79) further considered a modified cost functional with predefined end time T:

$$\min_{u_s} \int_0^T u_s^2 \, \mathrm{d}t. \qquad 24.$$

In the absence of state and input constraints, the differential equation posed by the minimum principle can be solved analytically to yield a fifth-order polynomial, which can then be quickly checked for conformity to state and input constraints. This analytical solution can be computed and checked up to 20 times faster than Equation 23 can be solved.

Equation 24 can alternatively be interpreted as a calculus of variations problem. Recall that the quadrotor's differential flatness guarantees that sufficiently continuous trajectories for $\mathbf{x}_Q(t)$ and $\psi(t)$ can be translated into a dynamically feasible state trajectory $\mathbf{x}(t)$. Thus, it is possible to optimize the position and yaw trajectories without consideration of dynamic constraints (21). The original $\mathbf{x}^0$ and $\mathbf{x}^1$ can be related to initial and final states for the set $\{\frac{\mathrm{d}^{k_1}}{\mathrm{d}t^{k_1}}\mathbf{x}_Q, \frac{\mathrm{d}^{k_2}}{\mathrm{d}t^{k_2}}\psi \mid k_1 \in [0,3], k_2 \in [0,1]\}$ through differential flatness. We can then construct the optimization problem:

$$\begin{aligned} \min_{\mathbf{x}_Q,\psi} \quad & \int_0^T \left(\|\mathbf{x}_Q^{(n_1)}\|_2^2 + \|\psi^{(n_2)}\|^2 \right) \mathrm{d}t \\ \text{subject to} \quad & \\ \text{Initial constraints: } & \frac{\mathrm{d}^{k_1}}{\mathrm{d}t_1^k}\mathbf{x}_Q(0) = \frac{\mathrm{d}^{k_1}}{\mathrm{d}t_1^k}\mathbf{x}_Q^0 \;\; \forall k_1 \in [0,3], \; \frac{\mathrm{d}^{k_2}}{\mathrm{d}t^{k_2}}\psi(0) = \frac{\mathrm{d}^{k_2}}{\mathrm{d}t^{k_2}}\psi^0 \;\; \forall k_2 \in [0,1], \\ \text{Final constraints: } & \frac{\mathrm{d}^{k_1}}{\mathrm{d}t^{k_1}}\mathbf{x}_Q(T) = \frac{\mathrm{d}^{k_1}}{\mathrm{d}t^{k_1}}\mathbf{x}_Q^1 \;\; \forall k_1 \in [0,3], \; \frac{\mathrm{d}^{k_2}}{\mathrm{d}t^{k_2}}\psi(T) = \frac{\mathrm{d}^{k_2}}{\mathrm{d}t_{k_2}}\psi^1 \;\; \forall k_2 \in [0,1]. \end{aligned} \qquad 25.$$

Equation 25 can be solved using variational calculus techniques. The Euler–Lagrange equations give the following necessary conditions:

$$\mathbf{x}_Q^{(2n_1)} = 0, \psi^{(2n_2)} = 0. \qquad 26.$$

Integrating Equation 26 shows that the optimal position and yaw trajectories are polynomials of orders $2n_1 - 1$ and $2n_2 - 1$, respectively. Note that when $n_1 = 3$, $\mathbf{x}_Q$ becomes a fifth-order polynomial, corroborating the analytical result for Equation 24. This substantiates the popular choice of using polynomial trajectories for quadrotors.

Like Equation 24, Equation 25 does not explicitly incorporate input or state constraints. However, these constraints can be checked by calculating the relevant values from differential flatness equations. If any constraints are violated, a coordinate change $t' = \frac{T't}{T}$, where T' is a new trajectory duration $T' > T$, will yield trajectories $\mathbf{x}_Q(t')$ and $\psi(t')$, which retain the same path but have smaller higher-derivative values. The trajectory time T' can be iteratively varied until a suitable solution is found.

Using differential flatness, the moment input, $\mathbf{M}$, can be expressed as a function of $\mathbf{x}_Q^{(4)}$ and $\ddot{\psi}$ (21). Thus, minimum-snap trajectories ($n_1 = 4$) are often chosen for $\mathbf{x}_Q(t)$, and minimum-acceleration trajectories ($n_2 = 2$) are chosen for $\psi(t)$. Conceptually, this approximates the cost functional of Equation 24 by seeking to indirectly minimize the system input. Furthermore, the minimum-snap and minimum-acceleration trajectories are of degrees 7 and 3, respectively, and the number of specified boundary constraints allows their coefficients to be found analytically with a simple matrix inversion. In practice, lower values of n_1 and n_2 also produce reasonable trajectories. However, to ensure a continuous state trajectory $\mathbf{x}(t)$, $\mathbf{x}_Q(t)$ should be at least four times and ψ at least twice differentiable; this is fulfilled for any $n_1 \geq 3$ and $n_2 \geq 2$.

While single trajectory segments are sufficient for short, motion-primitive-like paths, they cannot adequately span large spatial distances. To address this, many works use piecewise-polynomial trajectories. In this formulation, $\mathbf{x}_Q(t)$ is given by

$$\mathbf{x}_Q(t) = \begin{cases} \sum_{i=0}^{N} c_{i,0}\mathcal{P}_i(t) & t_0 \leq t \leq t_1 \\ \sum_{i=0}^{N} c_{i,1}\mathcal{P}_i(t) & t_1 < t \leq t_2 \\ \ldots & \\ \sum_{i=0}^{N} c_{i,m-1}\mathcal{P}_i(t) & t_{m-1} < t \leq t_m \end{cases}, \qquad 27.$$

where N is the polynomial degree, m is the number of segments, and $\mathcal{P}_i$ is any set of polynomial basis functions (e.g., Legendre polynomials). Times $t_0, t_1, \ldots, t_m$ are preselected break times. To ensure sufficient differentiability, continuity constraints are added to Equation 25:

$$\frac{\mathrm{d}^{k_1}}{\mathrm{d}t^{k_1}}\left(\sum_{i=0}^{N} c_{i,j}\mathcal{P}_i(t_{j+1})\right) = \frac{\mathrm{d}^{k_1}}{\mathrm{d}t^{k_1}}\left(\sum_{i=0}^{N} c_{i,j+1}\mathcal{P}_i(t_{j+1})\right) \quad \forall j \in [1, m-1], k_1 \in [0, 3]. \qquad 28.$$

To solve this optimization problem, consider the following vector of coefficients:

$$\mathbf{c} = \begin{bmatrix} c_{0,0} & c_{1,0} & c_{2,0} & \ldots & c_{N,m-1} \end{bmatrix}^\top. \qquad 29.$$

The position term of the cost functional in Equation 25 is positive semidefinite and quadratic with respect to $\mathbf{c}$, while all constraints, including Equation 28, are linear with respect to $\mathbf{c}$. The trajectory optimization can be reformulated as

$$\begin{aligned} &\min_{\mathbf{c}} \ \mathbf{c}^T Q\mathbf{c} \\ &\text{subject to} \quad A\mathbf{c} = b. \end{aligned} \qquad 30.$$

Equation 30 is a quadratic program with linear constraints, which can be solved analytically. Exact constraints on the quadrotor's position or higher derivatives can be placed instead of the continuity constraints. An analogous process can solve for the yaw trajectory.

For large N and m, Equation 30 can become numerically unstable. This instability can be mitigated by parameterizing trajectory segments in terms of a nondimensionalized time $\tau \in [0, 1]$ (80). Furthermore, Richter et al. (81) showed that the quadratic program becomes more stable when reformulated using the polynomial's position and higher-derivative values as decision variables. Time allocation along the trajectory (i.e., choosing the break times) is also a crucial design decision. If the break times are chosen poorly, then the optimal trajectory will display large excursions between waypoints (80). Mellinger & Kumar (21) and Richter et al. (81) proposed numerical methods for break-time optimization, but break-time optimization remains an open challenge.

To summarize, trajectory-generation problems can be formulated to solve for the optimal maneuver time and control inputs or optimal flat output trajectories. The latter formulation results in a large gain in computational efficiency at the expense of time optimality, although many works have shown that heuristically choosing trajectory times works well enough for practical applications. We also note that, while optimal control formulations typically yield inputs f and Ω^c, a differential flatness formulation yields flat variable inputs to the position controller, allowing trajectories to be replanned at a lower rate (1–10 Hz). Finally, Section 4.2 shows that Equation 25 allows for the incorporation of convex obstacle-avoidance constraints.

4.2. Convex Optimization for Safe Navigation in Cluttered Environments

Most successes in real-time planning among obstacles have introduced additional obstacle-avoidance constraints to the quadratic-program formulation given in Equation 25. Note that the quadrotor is symmetric about the $\mathbf{b}_3$ axis and that a change in ψ along a trajectory will not induce new collisions with obstacles. Thus, for the remainder of this section, we assume without lost of generality that $\psi(t)$ is held constant.

Consider the navigation problem presented in **Figure 4*a***, where the robot must navigate from one point to another while avoiding multiple obstacles. Constraining a three-dimensional point inside a convex polytope can be done with a set of linear inequality constraints (82). Leveraging this observation, the method described by Landry et al. (83) first identifies a set of convex polytopes inside of which the quadrotor is guaranteed to be safe (**Figure 4*b***). A subset of overlapping polytopes is then chosen using a mixed-integer second-order cone program as a flight corridor (**Figure 4*c***). Finally, a semidefinite program that restricts the trajectory to the inside of this corridor can be solved to obtain the quadrotor trajectory.

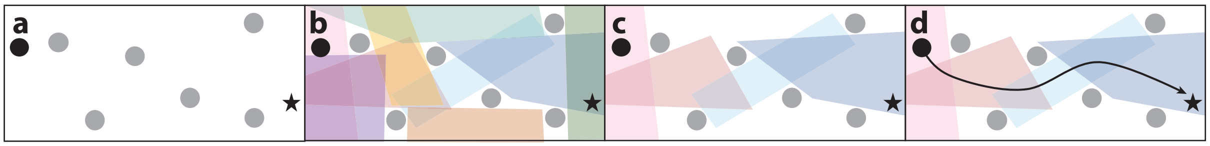

Figure 4

Formulation of the trajectory-generation problem as a quadratic program on an example problem. (*a*) Problem statement. The robot must navigate from a starting point (*black circle*) to an end point (*black star*) while avoiding multiple obstacles (*gray circles*). (*b*) Convex decomposition of the free space into polytopes inside of which the quadrotor will be safe (*colored regions*). (*c*) Identification of a subset of overlapping polytopes that create a flight corridor between the start and end points. (*d*) Generation of the specific trajectory within the chosen flight corridor (*black line*).

While this method guarantees safety, it is not fast enough for real-time computation, even using commercial software. Many works have thus focused on speeding up the steps for free-space decomposition and flight-corridor identification. A breakthrough idea was to use search- or sampling-based planners to first find a nominal path from the start to the goal. Rather than decomposing the entire free space, this method enables overlapping convex polytopes to be quickly found in the immediate vicinity of the path (84, 85). Chen et al. (86) showed that an A* planner can be used instead of an optimization problem to identify a flight corridor through octo-tree cells. Similarly, Gao & Shen (87) used a sampling-based search algorithm to identify a flight corridor through cells of a *k*-D tree. Many works also define time allocation (84) or polytope adjustment (88) heuristics to further improve the trajectory.

These techniques allow trajectories to be replanned in fractions of a second in a receding-horizon manner as the maps are updated. It is particularly important for algorithms to account for failures to find quadratic-program solutions, as a failure to replan when the quadrotor is already operating at a high velocity can lead to a collision. To address this, Watterson & Kumar (89) proposed a dual-horizon planning scheme. In a short-range planning policy, two trajectories are planned at each replanning iteration. The first trajectory makes incremental progress toward the desired goal, while the second safely brings the robot to a stop from its current state. Executing the latter trajectory in emergencies allows the robot to return to hover and replan using a more exhaustive long-range planning policy. This paradigm is safe and complete and has enabled many successes in high-speed quadrotor flight (84, 85, 89).

5. APPLICATIONS FOR AGILE, HIGH-SPEED FLIGHT

In this section, we review representative applications demonstrating the impact of the research presented in Sections 2–4. Many works have successfully demonstrated autonomous navigation in unknown workspaces using completely onboard mapping, estimation, planning, and control pipelines (53, 66, 85, 86, 90). Notably, Ling et al. (66) demonstrated flight with translational velocities of up to 4.2 m/s and rotational velocities of up to 245°/s using a stereo camera and IMU in an obstacle-free environment. Lin et al. (53) presented a quadrotor equipped with a monocular camera and IMU that can reach average speeds of 1.1 m/s and top speeds of 2.2 m/s in cluttered indoor and outdoor spaces. Schmid et al. (90) demonstrated similar capabilities with a stereo camera configuration. The University of Pennsylvania's platform for the DARPA Fast Lightweight Autonomy challenge (49, 85), which uses a stereo camera and IMU for estimation and a laser sensor for mapping, can reach speeds of 7 m/s. **Figure 5** illustrates identified safe trajectories from a single start position to multiple goals throughout a space.

In these methods, quadrotors encounter mainly naturally occurring, vertically oriented obstacles (e.g., walls). Falanga et al. (75) and Loianno et al. (91) further demonstrated the ability to detect, plan, and fly through a narrow window oriented at various pitch and roll angles using a single camera and IMU as primary sensors. This task requires robots to reach angular velocities of 400°/s and recover from losses of state estimation and large attitude angles (up to 45–90°).

Following these successes, many researchers have been interested in enabling quadrotors to move beyond navigation to further interact with their environments. Thomas et al. (38) demonstrated perching on surfaces inclined at up to 90°, where a nonlinear controller (23) stably controls the quadrotor even at almost vertical orientations. Other maneuvers have included landing (92), hanging from branch-like structures (93), and object catching (94). Aerial manipulation has been an area of particular interest. Thomas et al. (95) proposed a geometric controller and trajectory-generation algorithm for high-speed avian-inspired grasping of stationary objects. In these maneuvers, a quadrotor grabs an object at 3 m/s in a single swooping motion (**Figure 6*a***). Similarly,

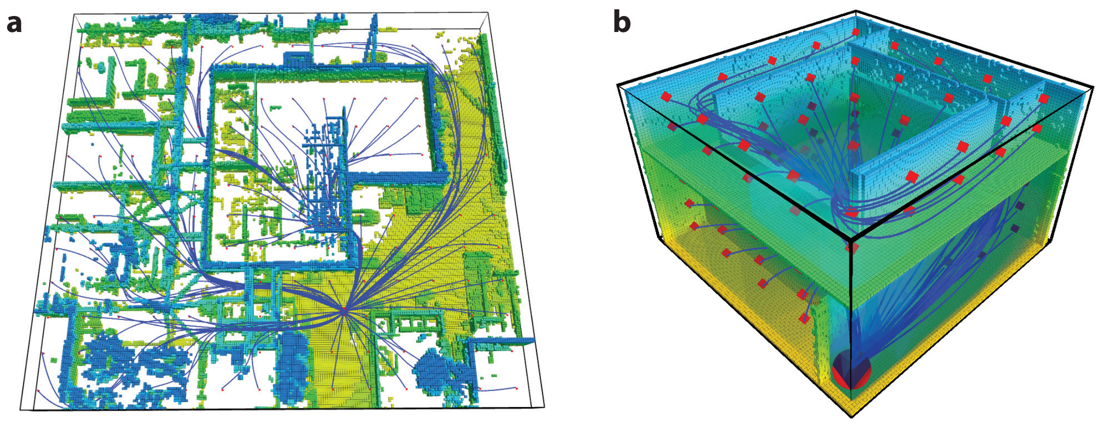

Figure 5

Optimized safe trajectories through (*a*) an outdoor and (*b*) an indoor environment. Adapted from Reference 85 with permission.

quadrotors can manipulate payloads via a cable-suspension mechanism (**Figure 6*b***). In contrast to traditional strategies that aim to eliminate payload motions, a more optimal strategy deliberately plans safe payload swings for energy optimality. An almost globally exponentially attractive controller allows control of the payload position even at large angles from the vertical configuration (96). Trajectory generation for the system's flat variables—the payload position and robot yaw—can be formulated as a mixed-integer quadratic program (97). Notably, the optimization is able to generate trajectories that would not be possible if the system were constrained to be swing free, such as swinging a payload through a narrow window with a height shorter than the cable length or performing pickup and release maneuvers where the vehicle is not directly above the payload. An alternate formulation without differential flatness yields a nonlinear optimization problem (98). In these works, however, trajectories are planned using global information about obstacle locations and executed using motion-capture feedback. The implementation of fully

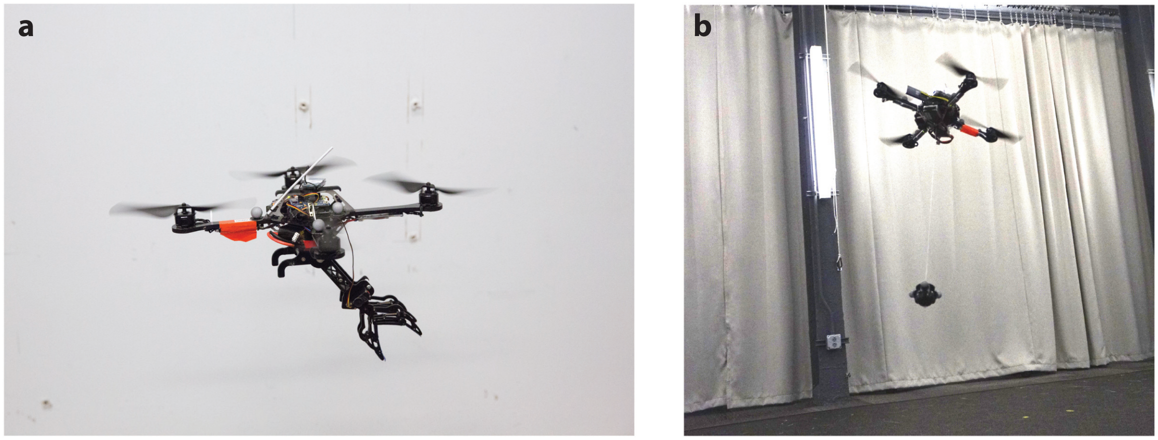

Figure 6

Aerial manipulation via (*a*) an attached gripper (e.g., 95) and (*b*) a cable-suspension mechanism (e.g., 97).

onboard perception, planning, and control for maneuvers of this increased level of complexity remains an active research area.

6. LEARNING

Implementing the model-based algorithms described above presents several challenges. First, algorithms often involve many parameters, from control gains to noise covariance matrices to trajectory break times. Tuning these variables is time consuming, and it is practically impossible for humans to find optimal parameters. Second, models are idealized approximations, and many significant effects, such as drag forces and wind disturbances, are often omitted or cannot be accurately modeled.

As a result, many works have proposed learning and adaptation methods. For parameter turning, a well-posed Bayesian optimization problem can be used to find locally optimal controller gains (99) with a relatively small number of iterations. To account for modeling inaccuracies, iterative learning control (ILC) can refine the reference or input signals of a desired maneuver based on data from previous executions. This mimics the way that humans refine their skills through practice. Given planned trajectory inputs u_0 and assuming that unmodeled dynamics are a linear additive term to the state equation d, each ILC iteration can be decomposed into two steps: disturbance estimation (where a Kalman filter finds the current estimate of the disturbance, $\hat{d}_k$) and input update (where an improved quadrotor input, u_{k+1}, is found). The input can be abstracted at any level, from robot thrust and angular velocities (100) to the trajectory's position commands (101, 102). Experiments show that even a small number of iterations (on the order of 10–20) is enough to characterize repeatable disturbances and improve tracking performance.

A major drawback of ILC is that the controller is retrained for each desired maneuver. Several efforts have therefore been made to generalize learned knowledge by executing high-speed trajectories after learning at slower speeds (102) or navigating through arbitrary obstacle courses after learning from a single configuration (103). Robots can also potentially use training data generated by other identical agents (104). However, this is more difficult in practice, as vehicles are never completely identical, and overestimating robots' similarities when sharing training data can counterproductively degrade performance (105).

Alternatively, some works have proposed eliminating explicit dynamic models altogether. For example, rather than imposing a linear structure on the disturbance term, as in ILC, a neural network can be trained to directly map from observed tracking performance to a modified desired trajectory (106). This idea can be applied to other modules as well. Instead of a traditional proportional–integral–derivative (PID) control design, a network can map from a robot's state to motor commands (107). More ambitiously, some works have mapped directly from camera inputs to motor commands (108–110) and demonstrated navigation through indoor hallways and forests. These results suggest that robots could potentially learn general, end-to-end policies for autonomy without relying on task- or application-specific modeling techniques.

However, learning end-to-end policies is extremely difficult. Model-free methods require many more training iterations (on the order of thousands) before reasonable performance can be achieved, and many iterations yield inputs that would lead to vehicle crashes. As a result, training is typically done completely or partially on recorded data from expert-piloted flights (108, 110) or in simulated environments (109). Training onboard real vehicles is complicated by the fact that quadrotors are inherently unstable, and a characterization of the set of safe inputs or a fallback model-based controller is required to avoid damaging the vehicle. As a result, many researchers are studying methods to use dynamic models to guide learning algorithms (111). These types of end-to-end learning approaches have been able to achieve impressive results in vehicle stabilization from aggressive throws (107), navigation through indoor environments (109), and reactive collision avoidance through forests (108, 110) and are an exciting new research frontier.

7. MULTIROBOT SYSTEMS

The successes in single-robot autonomy have led to growing interest in the development of cooperative multirobot systems. To this end, several academic test beds have been constructed from either custom-built vehicles (112) or commercial platforms, such as the AscTec Hummingbird (113) and the Bitcraze Crazyflie (114). Currently, the largest academic test bed consists of 49 Crazyflie robots (114), and the largest industry system consists of 500 Intel Shooting Star drones (115).

While research on multirobot planning dates back to the 1980s, much progress is still needed to bridge the gap between the planning and quadrotor communities. The literature on multirobot planning traditionally models robots with first-order kinematic models and allows trajectories with instantaneous velocity changes. Many methods further discretize the workspace into graph-based representations and return paths with arbitrarily sharp turns between vertices. Such trajectories are not dynamically feasible for quadrotors, and extending existing multirobot planning algorithms to high-order, high-dimensional, underactuated systems is nontrivial.

In some applications, robots must travel in formation to selected goal positions. Early works incorporated interrobot distance constraints into a mixed-integer quadratic program for trajectory generation (80). This approach allows for the simultaneous optimization of the team's formation and individual robot trajectories. Unfortunately, because this method concatenates all robots' trajectory coefficients into a single decision vector, it is impractical for large teams (more than 20 robots). This problem can be made more computationally tractable by requiring the team to select from a set of predefined formations (116).

In other instances, each vehicle must independently navigate to a desired goal position. In unlabeled problems, robots are considered identical; each goal in a set must be reached by some robot, but it does not matter which one. This abstraction can be used in surveillance tasks, where robots must distribute themselves among given locations from which to gather information (117, 118). Here, the task-allocation and trajectory-generation problems must be simultaneously solved. Paradoxically, the ability to assign goals reduces the problem's computational complexity, and a cleverly chosen cost function for goal assignment can directly guarantee trajectory safety (119, 120). In labeled problems, by contrast, robots must visit predefined, noninterchangeable goal positions. This collision-avoidance problem can be solved with reactive approaches (121) or optimization-based methods (122, 123). Recent work has further considered the case in which each quadrotor is carrying a suspended payload and must safely navigate through a shared workspace for a delivery task (124).

Many open questions remain in this area. Many planning works assume the existence of a central base station with global information, limiting the teams' range of operation and presenting a single point of failure. It is much more beneficial to adopt a decentralized architecture, where robots require only communication with (or sensing of) their most immediate neighbors. Decentralized methods have been successful for small teams of robots operating at low speeds (121, 125); however, a safe, complete, scalable method for high-speed flight has not yet been found.

Modules other than trajectory generation still need to be developed for multirobot scenarios. Higher-level planning methods are needed to derive individual robot tasks from team-level task specifications. Perception modules must be able to accurately sense the states of other moving vehicles. In some scenarios, robots might even become physically coupled, and as a result, they require novel control schemes. For example, Oung & D'Andrea (126) described a distributed flight array in which individual single-rotor hexagonal modules can interconnect to create arbitrarily large multirotor vehicles. Ritz & D'Andrea (127) and Sreenath & Kumar (128) addressed cooperative carrying of suspended payloads. These systems are significantly more complex than the first- or second-order models typically assumed in multirobot planning and pose challenging new research questions.

8. CONCLUSIONS

There have been many tremendous successes in the development of autonomous quadrotors. Nonlinear controllers have enabled precise control of maneuvers that exploit the vehicle's full range of motion. Computationally efficient computer vision and filtering techniques have allowed onboard perception and state estimation with low-cost, lightweight sensors. And techniques for approximating the system's dynamics and geometrically representing the environment have allowed on-the-fly trajectory planning.

However, many compelling research avenues remain. Robots are often still dependent on humans to provide task specifications, such as desired goal positions, and further work needs to be done to realize autonomy from high-level task commands (such as "monitor this area" or "deliver these packages"). In the multirobot domain, estimation, perception, and real-time planning schemes must be extended to account for teammates operating in a shared workspace.

There are also many challenges in developing vehicles that can quickly adapt to complex, changing environments. Currently, learning algorithms such as ILC that converge within a few iterations improve only a limited aspect of the vehicle's behavior (e.g., reducing the tracking error for a specific trajectory). End-to-end solutions that offer more generalized capabilities are often difficult to train, tune, and characterize. In particular, while algorithms for image or text processing can easily be trained on large quantities of samples given enough computational resources, algorithms for quadrotors must be trained in a way that guarantees the stability and safety of the physical system. This typically requires a nominal model of the system for simulation-based training or a characterization of unsafe system inputs. The best way to combine traditional model-based methods with learning techniques in order to train robots to operate robustly across a breadth of real-world environments (e.g., in various daylight and weather conditions) remains an open problem.

Finally, quadrotor platforms used in research often incorporate expensive sensors and processors and can cost on the order of tens of thousands of dollars. Bringing the same level of agility and autonomy to vehicles with consumer-grade electronics is difficult. Fortunately, the rapid development of sensor-rich and computationally powerful communication devices has opened up the potential for plug-and-play interfaces between smartphones, such as the Google Tango (129) and the Samsung Galaxy S5 (130), and off-the-shelf quadrotor frames. This would allow the development of algorithms on standardized sensor suites that are easily accessible to consumers. A related goal is to develop small yet capable quadrotor systems. Smaller vehicles have the potential to be more agile and safer for use in close proximity to humans, but they also have lower payload capacities for sensing and computation resources. Several works have proposed designs for small vehicles (131) and rapid, repeatable fabrication methods for their production (132); however, these prototypes rely on offboard resources, and many challenges remain before platforms that are small, lightweight, and affordable can be produced.

We believe that continued research in these areas will lead to the development of fully autonomous, intelligent quadrotor systems. This will in turn affect a breadth of industries, including manufacturing and construction, entertainment, inspection and surveillance, and precision agriculture, and will undoubtedly be transformative for our technological future.

DISCLOSURE STATEMENT

The authors are not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

Sarah Tang is supported by National Science Foundation Research Fellowship grant DGE-1321851.

LITERATURE CITED

1. Rajappa S, Ryll M, Bülthoff HH, Franchi A. 2015. Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4006–13. New York: IEEE
2. Ryll M, Bülthoff HH, Giordano PR. 2013. First flight tests for a quadrotor UAV with tilting propellers. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 295–302. New York: IEEE
3. Brescianini D, D'Andrea R. 2016. Design, modeling and control of an omni-directional aerial vehicle. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3261–66. New York: IEEE
4. Piccoli M, Yim M. 2015. Passive stability of vehicles without angular momentum including quadrotors and ornithopters. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1716–21. New York: IEEE
5. Chen Y, Gravish N, Desbiens AL, Malka R, Wood RJ. 2016. Experimental and computational studies of the aerodynamic performance of a flapping and passively rotating insect wing. *J. Fluid Mech.* 791:1–33
6. Chen Y, Ma K, Wood RJ. 2016. Influence of wing morphological and inertial parameters on flapping flight performance. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2329–36. New York: IEEE
7. Rosen MH, le Pivain G, Sahai R, Jafferis NT, Wood RJ. 2016. Development of a 3.2g untethered flapping-wing platform for flight energetics and control experiments. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3227–33. New York: IEEE
8. Chirarattananon P, Ma KY, Wood RJ. 2016. Perching with a robotic insect using adaptive tracking control and iterative learning control. *Int. J. Robot. Res.* 35:1185–206
9. Hoff J, Ramezani A, Chun SJ, Hutchinson S. 2016. Synergistic design of a bio-inspired micro aerial vehicle with articulated wings. In *Robotics: Science and Systems XII*, ed. D Hsu, N Amato, S Berman, S Jacobs, chap. 9. N.p.: Robot. Sci. Syst. Found.
10. Ramezani A, Chung SJ, Hutchinson S. 2017. A biomimetic robotic platform to study flight specializations of bats. *Sci. Robot.* 2:eaal2505
11. Keennon M, Klingebiel K, Won H, Andriukov A. 2012. Development of the nano hummingbird: a tailless flapping wing micro air vehicle. In *50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*, chap. 2012-0588. Reston, VA: Am. Inst. Aeronaut. Astronaut.
12. Gerdes J, Holness A, Perez-Rosado A, Roberts L, Greisinger A, et al. 2014. Robo Raven: a flapping-wing air vehicle with highly compliant and independently controlled wings. *Soft Robot.* 1:275–88
13. de Croon GCHE, Groen MA, Wagter CD, Remes B, Ruijsink R, van Oudheusden BW. 2012. Design, aerodynamics and autonomy of the DelFly. *Bioinspir. Biomimet.* 7:025003
14. Moore J, Cory R, Tedrake R. 2014. Robust post-stall perching with a simple fixed-wing glider using LQR-trees. *Bioinspir. Biomimet.* 9:025013
15. Majumdar A, Tedrake R. 2017. Funnel libraries for real-time robust feedback motion planning. *Int. J. Robot. Res.* 36:947–82
16. Barry AJ, Tedrake R. 2015. Pushbroom stereo for high-speed navigation in cluttered environments. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3046–52. New York: IEEE
17. Whitzer M, Keller J, Bhattacharya S, Kumar V, Sands T, et al. 2016. In-flight formation control for a team of fixed-wing aerial vehicles. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 372–80. New York: IEEE

18. CB Insights. 2017. *Still soaring: deals to drone startups on track for 6th annual high*. Res. Brief, CB Insights, New York. **https://www.cbinsights.com/research/drones-startup-funding**
19. Kumar V, Michael N. 2012. Opportunities and challenges with autonomous micro aerial vehicles. *Int. J. Robot. Res.* 31:1279–91
20. Murray RM, Rathinam M, Sluis W. 1995. Differential flatness of mechanical control systems: a catalog of prototype systems. In *Proceedings of the 1995 ASME International Mechanical Engineering Congress and Exposition (IMECE)*. New York: ASME. Preprint version available at **http://www.cds.caltech.edu/~murray/preprints/mrs95-imece.pdf**
21. Mellinger D, Kumar V. 2011. Minimum snap trajectory generation and control for quadrotors. *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2520–25. New York: IEEE
22. Michael N, Mellinger D, Lindsey Q, Kumar V. 2010. The GRASP multiple micro-UAV testbed. *IEEE Robot. Autom. Mag.* 17:56–65
23. Lee T, Leok M, McClamroch NH. 2012. Nonlinear robust tracking control of a quadrotor UAV on SE(3). In *2012 American Control Conference (ACC)*, pp. 4649–54. New York: IEEE
24. Watterson M, Kumar V. 2018. Control of quadrotors using the HOPF fibration on SO(3). In *Proceedings of the 2017 International Symposium on Robotics Research*. Forthcoming
25. Yu Y, Yang S, Wang M, Li C, Li Z. 2015. High performance full attitude control of a quadrotor on SO(3). In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1698–703. New York: IEEE
26. Brescianini D, Hehn M, D'Andrea R. 2013. *Nonlinear quadrocopter attitude control*. Tech. Rep., ETH Zürich, Zürich, Switz.
27. Faessler M, Falanga D, Scaramuzza D. 2017. Thrust mixing, saturation, and body-rate control for accurate aggressive quadrotor flight. *IEEE Robot. Autom. Lett.* 2:476–82
28. Monteiro JC, Lizarralde F, Hsu L. 2016. Optimal control allocation of quadrotor UAVs subject to actuator constraints. In *2016 American Control Conference (ACC)*, pp. 500–5. New York: IEEE
29. Kamel M, Alexis K, Achtelik M, Siegwart R. 2015. Fast nonlinear model predictive control for multicopter attitude tracking on SO(3). In *2015 IEEE Conference on Control Applications (CCA)*, pp. 1160–66. New York: IEEE
30. Bangura M, Mahony R. 2014. Real-time model predictive control for quadrotors. *IFAC Proc. Vol.* 47:11773–80
31. Bangura M, Mahony R. 2017. Thrust control for multirotor aerial vehicles. *IEEE Trans. Robot.* 33:390–405
32. Powers C, Mellinger D, Kushleyev A, Kothmann B, Kumar V. 2013. Influence of aerodynamics and proximity effects in quadrotor flight. In *Experimental Robotics: The 13th International Symposium on Experimental Robotics*, ed. JP Desai, G Dudek, O Khatib, V Kumar, pp. 289–302. Heidelberg, Ger.: Springer
33. Schiano F, Alonso-Mora J, Rudin K, Beardsley P, Siegwar R, Siciliano B. 2014. Towards estimation and correction of wind effects on a quadrotor UAV. In *IMAV 2014: International Micro Air Vehicle Conference and Competition 2014*, pp. 134–41. Delft, Neth.: Delft Univ. Technol.
34. Ware J, Roy N. 2016. An analysis of wind field estimation and exploitation for quadrotor flight in the urban canopy layer. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1507–14. New York: IEEE
35. Yao JW, Desaraju VR, Michael N. 2017. Experience-based models of surface proximal aerial robot flight performance in wind. In *Experimental Robotics: The 15th International Symposium on Experimental Robotics*, ed. D Kulić, Y Nakamura, O Khatib, G Venture, pp. 563–73. Cham, Switz.: Springer
36. Bartholomew J, Calway A, Mayol-Cuevas W. 2014. Learning to predict obstacle aerodynamics from depth images for micro air vehicles. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4967–73. New York: IEEE
37. Mueller MW, D'Andrea R. 2016. Relaxed hover solutions for multicopters: application to algorithmic redundancy and novel vehicles. *Int. J. Robot. Res.* 35:873–89
38. Thomas J, Loianno G, Pope M, Hawkes EW, Estrada MA, et al. 2015. Planning and control of aggressive maneuvers for perching on inclined and vertical surfaces. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE)*, pap. DETC2015-47710. New York: ASME

39. Scaramuzza D, Fraundorfer F. 2011. Visual odometry: part I – the first 30 years and fundamentals. *IEEE Robot. Autom. Mag.* 18:80–92
40. Scaramuzza D, Fraundorfer F. 2012. Visual odometry: part II – matching, robustness, and applications. *IEEE Robot. Autom. Mag.* 19:78–90
41. Corke P, Lobo J, Dias J. 2007. An introduction to inertial and visual sensing. *Int. J. Robot. Res.* 26:519–35
42. Gui J, Gu D, Wang S, Hu H. 2015. A review of visual inertial odometry from filtering and optimisation perspectives. *Adv. Robot.* 29:1289–301
43. Li AQ, Coskun A, Doherty SM, Ghasemlou S, Jagtap AS, et al. 2017. Experimental comparison of open source vision based state estimation algorithms. In *Experimental Robotics: The 15th International Symposium on Experimental Robotics*, ed. D Kulić, Y Nakamura, O Khatib, G Venture, pp. 775–85. Cham, Switz.: Springer
44. Hartley RI, Zisserman A. 2004. *Multiple View Geometry in Computer Vision*. Cambridge, UK: Cambridge Univ. Press. 2nd ed.
45. Thrun S, Burgard W, Fox D. 2005. *Probabilistic Robotics*. Cambridge, MA: MIT Press
46. Martinelli A. 2013. Visual-inertial structure from motion: observability and resolvability. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4235–42. New York: IEEE
47. Troiani C, Martinelli A, Laugier C, Scaramuzza D. 2015. Low computational-complexity algorithms for vision-aided inertial navigation of micro aerial vehicles. *Robot. Auton. Syst.* 69:80–97
48. Bloesch M, Omari S, Hutter M, Siegwart R. 2015. Robust visual inertial odometry using a direct EKF-based approach. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 298–304. New York: IEEE
49. Mohta K, Watterson M, Mulgaonkar Y, Liu S, Qu C, et al. 2018. Fast, autonomous flight in GPS-denied and cluttered environments. *J. Field Robot.* 35:101–20
50. Faessler M, Fontana F, Forster C, Mueggler E, Pizzoli M, Scaramuzza D. 2016. Autonomous, vision-based flight and live dense 3D mapping with a quadrotor micro aerial vehicle. *J. Field Robot.* 33:431–50
51. Achtelik M, Achtelik M, Weiss S, Siegwart R. 2011. Onboard IMU and monocular vision based control for MAVs in unknown in- and outdoor environments. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3056–63. New York: IEEE
52. Shen S, Mulgaonkar Y, Michael N, Kumar V. 2016. Initialization-free monocular visual-inertial state estimation with application to autonomous MAVs. In *Experimental Robotics: The 14th International Symposium on Experimental Robotics*, ed. MA Hsieh, O Khatib, V Kumar, pp. 211–27. Cham, Switz.: Springer
53. Lin Y, Gao F, Qin T, Gao W, Liu T, et al. 2018. Autonomous aerial navigation using monocular visual-inertial fusion. *J. Field Robot.* 35:23–51
54. Shen S, Michael N, Kumar V. 2015. Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5303–10. New York: IEEE
55. Li M, Mourikis AI. 2013. High-precision, consistent EKF-based visual-inertial odometry. *Int. J. Robot. Res.* 32:690–711
56. Goodarzi FA, Lee T. 2016. Extended Kalman filter on SE(3) for geometric control of a quadrotor UAV. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1371–80. New York: IEEE
57. Wan EA, Merwe RVD. 2000. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pp. 153–58. New York: IEEE
58. Hauberg S, Lauze F, Pedersen KS. 2013. Unscented Kalman filtering on Riemannian manifolds. *J. Math. Imaging Vis.* 46:103–20
59. Brossard M, Bonnabel S, Condomines JP. 2017. Unscented Kalman filtering on Lie groups. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2485–91. New York: IEEE
60. Loianno G, Watterson M, Kumar V. 2016. Visual inertial odometry for quadrotors on SE(3). In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1544–51. New York: IEEE
61. Faessler M, Fontana F, Forster C, Scaramuzza D. 2015. Automatic re-initialization and failure recovery for aggressive flight with a monocular vision-based quadrotor. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1722–29. New York: IEEE

62. Liu T, Shen S. 2017. Spline-based initialization of monocular visual-inertial state estimators at high altitude. *IEEE Robot. Autom. Lett.* 2:2224–31
63. Yang Z, Shen S. 2017. Monocular visual-inertial state estimation with online initialization and camera-IMU extrinsic calibration. *IEEE Trans. Autom. Sci. Eng.* 14:39–51
64. Lupton T, Sukkarieh S. 2012. Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Trans. Robot.* 28:61–76
65. Forster C, Carlone L, Dellaert F, Scaramuzza D. 2017. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Trans. Robot.* 33:1–21
66. Ling Y, Liu T, Shen S. 2016. Aggressive quadrotor flight using dense visual-inertial fusion. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1499–506. New York: IEEE
67. Shen S, Mulgaonkar Y, Michael N, Kumar V. 2013. Vision-based state estimation for autonomous rotorcraft MAVs in complex environments. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1758–64. New York: IEEE
68. Barry AJ, Florence PR, Tedrake R. 2017. High-speed autonomous obstacle avoidance with pushbroom stereo. *J. Field Robot.* 35:52–68
69. Oleynikova H, Honegger D, Pollefeys M. 2015. Reactive avoidance using embedded stereo vision for MAV flight. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 50–56. New York: IEEE
70. Yang Z, Liu T, Shen S. 2016. Self-calibrating multi-camera visual-inertial fusion for autonomous MAVs. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4984–91. New York: IEEE
71. Kelly J, Sukhatme GS. 2011. Visual-inertial sensor fusion: localization, mapping and sensor-to-sensor self-calibration. *Int. J. Robot. Res.* 30:56–79
72. Kaiser J, Martinelli A, Fontana F, Scaramuzza D. 2017. Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation. *IEEE Robot. Autom. Lett.* 2:18–25
73. Hausman K, Preiss J, Sukhatme GS, Weiss S. 2017. Observability-aware trajectory optimization for self-calibration with application to UAVs. *IEEE Robot. Autom. Lett.* 2:1770–77
74. Preiss J, Hausman K, Sukhatme G, Weiss S. 2017. Trajectory optimization for self-calibration and navigation. In *Robotics: Science and Systems XIII*, ed. N Amato, S Srinivasa, N Ayanian, S Kuindersma, chap. 13. N.p.: Robot. Sci. Syst. Found.
75. Falanga D, Mueggler E, Faessler M, Scaramuzza D. 2017. Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5774–81. New York: IEEE
76. Liberzon D. 2012. *Calculus of Variations and Optimal Control Theory: A Concise Introduction*. Princeton, NJ: Princeton Univ. Press
77. Hehn M, Ritz R, D'Andrea R. 2012. Performance benchmarking of quadrotor systems using time-optimal control. *Auton. Robots* 33:69–88
78. Hehn M, D'Andrea R. 2015. Real-time trajectory generation for quadrocopters. *IEEE Trans. Robot.* 31:877–92
79. Mueller MW, Hehn M, D'Andrea R. 2015. A computationally efficient motion primitive for quadrocopter trajectory generation. *IEEE Trans. Robot.* 31:1294–310
80. Mellinger D, Kushleyev A, Kumar V. 2012. Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 477–83. New York: IEEE
81. Richter C, Bry A, Roy N. 2016. Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. In *Robotics Research: The 16th International Symposium ISRR*, ed. M Inaba, P Corke, pp. 649–66. Cham, Switz.: Springer
82. Flores ME. 2008. *Real-time trajectory generation for constrained nonlinear dynamical systems using non-uniform rational B-spline basis functions.* PhD Thesis, Div. Eng. Appl. Sci., Calif. Inst. Technol., Pasadena
83. Landry B, Deits R, Florence PR, Tedrake R. 2016. Aggressive quadrotor flight through cluttered environments using mixed integer programming. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1469–75. New York: IEEE

84. Liu S, Watterson M, Tang S, Kumar V. 2016. High speed navigation for quadrotors with limited onboard sensing. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1484–91. New York: IEEE

85. Liu S, Watterson M, Mohta K, Sun K, Bhattacharya S, et al. 2017. Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments. *IEEE Robot. Autom. Lett.* 2:1688–95

86. Chen J, Liu T, Shen S. 2016. Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1476–83. New York: IEEE

87. Gao F, Shen S. 2016. Online quadrotor trajectory generation and autonomous navigation on point clouds. In *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pp. 139–46. New York: IEEE

88. Chen J, Su K, Shen S. 2015. Real-time safe trajectory generation for quadrotor flight in cluttered environments. In *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1678–85. New York: IEEE

89. Watterson M, Kumar V. 2015. Safe receding horizon control for aggressive MAV flight with limited range sensing. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3235–40. New York: IEEE

90. Schmid K, Lutz P, Tomić T, Mair E, Hirschmüller H. 2014. Autonomous vision-based micro air vehicle for indoor and outdoor navigation. *J. Field Robot.* 31:537–70

91. Loianno G, Brunner C, McGrath G, Kumar V. 2017. Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and IMU. *IEEE Robot. Autom. Lett.* 2:404–11

92. Mohta K, Kumar V, Daniilidis K. 2014. Vision-based control of a quadrotor for perching on lines. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3130–36. New York: IEEE

93. Thomas J, Loianno G, Daniilidis K, Kumar V. 2016. Visual servoing of quadrotors for perching by hanging from cylindrical objects. *IEEE Robot. Autom. Lett.* 1:57–64

94. Su K, Shen S. 2017. Catching a flying ball with a vision-based quadrotor. In *Experimental Robotics: The 15th International Symposium on Experimental Robotics*, ed. D Kulić, Y Nakamura, O Khatib, G Venture, pp. 550–62. Cham, Switz.: Springer

95. Thomas J, Loianno G, Polin J, Sreenath K, Kumar V. 2014. Toward autonomous avian-inspired grasping for micro aerial vehicles. *Bioinspir. Biomimet.* 9:025010

96. Sreenath K, Lee T, Kumar V. 2013. Geometric control and differential flatness of a quadrotor UAV with a cable-suspended load. In *52nd IEEE Conference on Decision and Control (CDC)*, pp. 2269–74. New York: IEEE

97. Tang S, Kumar V. 2015. Mixed integer quadratic program trajectory generation for a quadrotor with a cable-suspended payload. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2215–22. New York: IEEE

98. Foehn P, Falanga D, Kuppuswamy N, Tedrake R, Scaramuzza D. 2017. Fast trajectory optimization for agile quadrotor maneuvers with a cable-suspended payload. In *Robotics: Science and Systems XIII*, ed. N Amato, S Srinivasa, N Ayanian, S Kuindersma, chap. 30. N.p.: Robot. Sci. Syst. Found.

99. Berkenkamp F, Schöllig AP, Krause A. 2015. Safe controller optimization for quadrotors with Gaussian processes. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 491–96. New York: IEEE

100. Schöllig A, Mueller F, D'Andrea R. 2012. Optimization-based iterative learning for precise quadrocopter trajectory tracking. *Auton. Robots* 33:103–27

101. Mueller FL, Schöllig AP, D'Andrea R. 2012. Iterative learning of feed-forward corrections for high-performance tracking. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3276–81. New York: IEEE

102. Hehn M, D'Andrea R. 2013. A frequency domain iterative feed-forward learning scheme for high performance periodic quadrocopter maneuvers. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2445–51. New York: IEEE

103. Hamer M, Waibel M, D'Andrea R. 2013. Knowledge transfer for high-performance quadrocopter maneuvers. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1714–19. New York: IEEE
104. Schöllig AP, Alonso-Mora J, D'Andrea R. 2012. Limited benefit of joint estimation in multi-agent iterative learning. *Asian J. Control* 14:613–23
105. Schöllig A, D'Andrea R. 2011. Sensitivity of joint estimation in multi-agent iterative learning control. *IFAC Proc. Vol.* 44:1204–12
106. Li Q, Qian J, Zhu Z, Bao X, Helwa MK, Schoellig AP. 2017. Deep neural networks for improved, impromptu trajectory tracking of quadrotors. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5183–89. New York: IEEE
107. Hwangbo J, Sa I, Siegwart R, Hutter M. 2017. Control of a quadrotor with reinforcement learning. *IEEE Robot. Autom. Lett.* 2:2096–103
108. Ross S, Melik-Barkhudarov N, Shankar KS, Wendel A, Dey D, et al. 2013. Learning monocular reactive UAV control in cluttered natural environments. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1765–72. New York: IEEE
109. Sadeghi F, Levine S. 2017. CAD^{2}RL: real single-image flight without a single real image. In *Robotics: Science and Systems XIII*, ed. N Amato, S Srinivasa, N Ayanian, S Kuindersma, chap. 34. N.p.: Robot. Sci. Syst. Found.
110. Daftry S, Bagnell JA, Hebert M. 2017. Learning transferable policies for monocular reactive MAV control. In *Experimental Robotics: The 15th International Symposium on Experimental Robotics*, ed. D Kulić, Y Nakamura, O Khatib, G Venture, pp. 3–11. Cham, Switz.: Springer
111. Kahn G, Zhang T, Levine S, Abbeel P. 2017. PLATO: policy learning using adaptive trajectory optimization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3342–49. New York: IEEE
112. Kushleyev A, Mellinger D, Powers C, Kumar V. 2013. Towards a swarm of agile micro quadrotors. *Auton. Robots* 35:287–300
113. Lupashin S, Hehn M, Mueller MW, Schoellig AP, Sherback M, D'Andrea R. 2014. A platform for aerial robotics research and demonstration: the flying machine arena. *Mechatronics* 24:41–54
114. Preiss JA, Hoenig W, Sukhatme GS, Ayanian N. 2017. Crazyswarm: a large nano-quadcopter swarm. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3299–304. New York: IEEE
115. Kaplan K. 2016. 500 drones light night sky to set record. *iQ*, Nov. 4. **https://iq.intel.com/500-drones-light-show-sets-record**
116. Alonso-Mora J, Montijano E, Schwager M, Rus D. 2016. Distributed multi-robot formation control among obstacles: a geometric and optimization approach with consensus. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5356–63. New York: IEEE
117. Mohta K, Turpin M, Kushleyev A, Mellinger D, Michael N, Kumar V. 2016. Quadcloud: a rapid response force with quadrotor teams. In *Experimental Robotics: The 14th International Symposium on Experimental Robotics*, ed. MA Hsieh, O Khatib, V Kumar, pp. 577–90. Cham, Switz.: Springer
118. Scaramuzza D, Achtelik MC, Doitsidis L, Friedrich F, Kosmatopoulos E, et al. 2014. Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in GPS-denied environments. *IEEE Robot. Autom. Mag.* 21:26–40
119. Turpin M, Michael N, Kumar V. 2014. CAPT: concurrent assignment and planning of trajectories for multiple robots. *Int. J. Robot. Res.* 33:98–112
120. Turpin M, Mohta K, Michael N, Kumar V. 2014. Goal assignment and trajectory planning for large teams of interchangeable robots. *Auton. Robots* 37:401–15
121. Alonso-Mora J, Naegeli T, Siegwart R, Beardsley P. 2015. Collision avoidance for aerial vehicles in multi-agent scenarios. *Auton. Robots* 39:101–21
122. Tang S, Kumar V. 2016. Safe and complete trajectory generation for robot teams with higher-order dynamics. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1894–901. New York: IEEE
123. Tang S, Thomas J, Kumar V. 2017. Safe navigation of quadrotor teams to labeled goals in limited workspaces. In *Experimental Robotics: The 15th International Symposium on Experimental Robotics*, ed. D Kulić, Y Nakamura, O Khatib, G Venture, pp. 586–98. Cham, Switz.: Springer

124. Tang S, Sreenath K, Kumar V. 2018. Multi-robot trajectory generation for an aerial payload delivery system. In *Proceedings of the 2017 International Symposium on Robotics Research*. Forthcoming
125. Zhou D, Wang Z, Schwager M. 2017. Fast, on-line collision avoidance for dynamic vehicles using buffered Voronoi cells. *IEEE Robot. Autom. Lett.* 2:1047–54
126. Oung R, D'Andrea R. 2014. The distributed flight array: design, implementation, and analysis of a modular vertical take-off and landing vehicle. *Int. J. Robot. Res.* 33:375–400
127. Ritz R, D'Andrea R. 2013. Carrying a flexible payload with multiple flying vehicles. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3465–71. New York: IEEE
128. Sreenath K, Kumar V. 2013. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. In *Robotics: Science and Systems IX*, ed. P Newman, D Fox, D Hsu, chap. 11. N.p.: Robot. Sci. Syst. Found.
129. Loianno G, Cross G, Qu C, Mulgaonkar Y, Hesch JA, Kumar V. 2015. Flying smartphones: automated flight enabled by consumer electronics. *IEEE Robot. Autom. Mag.* 22:24–32
130. Loianno G, Mulgaonkar Y, Brunner C, Ahuja D, Ramanandan A, et al. 2016. A swarm of flying smartphones. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1681–88. New York: IEEE
131. Mulgaonkar Y, Whitzer M, Morgan B, Kroninger CM, Harrington AM, Kumar V. 2014. Power and weight considerations in small, agile quadrotors. *Proc. SPIE* 9083:90831Q
132. Mehta AM, Rus D, Mohta K, Mulgaonkar Y, Piccoli M, Kumar V. 2016. A scripted printable quadrotor: rapid design and fabrication of a folded MAV. In *Robotics Research: The 16th International Symposium ISRR*, ed. M Inaba, P Corke, pp. 203–19. Cham, Switz.: Springer