Auton Robot (2013) 35:287–300
DOI 10.1007/s10514-013-9349-9

Towards a swarm of agile micro quadrotors

Alex Kushleyev · Daniel Mellinger · Caitlin Powers · Vijay Kumar

Received: 12 October 2012 / Accepted: 18 June 2013 / Published online: 10 July 2013

Abstract We describe a prototype 75 g micro quadrotor with onboard attitude estimation and control that operates autonomously with an external localization system. The motivation for designing quadrotors at this scale comes from two observations. First, the agility of the robot increases with a reduction in size, a fact that is supported by experimental results in this paper. Second, smaller robots are able to operate in tight formations in constrained, indoor environments. We describe the hardware and software used to operate the vehicle as well our dynamic model. We also discuss the aerodynamics of vertical flight and the contribution of ground effect to the vehicle performance. Finally, we discuss architecture and algorithms to coordinate a team of these quadrotors, and provide experimental results for a team of 20 micro quadrotors.

Keywords Micro aerial vehicles · Quadrotors · Trajectory generation

1 Introduction

The last decade has seen rapid progress in micro aerial robots, autonomous aerial vehicles that are smaller than 1 m in scale and 1 kg or less in mass. Winged aircrafts range from fixed-wing vehicles (Floreano et al. 2011) to flapping-wing vehicles (Aeroenvironment 2011), the latter mostly inspired by insect flight. Rotor crafts, including helicopters, coaxial rotor crafts (Bermes 2010), ducted fans (Pines and Bohorquez 2006), quadrotors (Bouabdallah 2007) and hexarotors, have become a mature technology (Kumar and Michael 2011) with quadrotors being the most commonly used aerial platform in robotics research labs. In this class, the Hummingbird quadrotor sold by Ascending Technologies, GmbH (2012), with a tip-to-tip wingspan of 55 cm, a height of 8 cm, mass of about 500 g including a lithium polymer battery and consuming about 75 W is a remarkably capable and robust platform as shown in Lupashin et al. (2010) and Mellinger and Kumar (2011). Our interest in this paper is scaling down the quadrotor platform to develop a truly small micro UAV. We describe the design of a prototype 75 g micro quadrotor. We also present the architecture and algorithms to coordinate a team of micro quadrotors in three-dimensional formation flight. Much of the material is drawn from two conference papers (Kushleyev et al. 2012; Powers et al. 2012).

The most important and obvious benefit of scaling down in size is the ability of the quadrotor to operate in tightly constrained environments in tight formations. In addition, these micro quadrotors are less expensive to manufacture than larger designs. Another interesting benefit of scaling down is agility. As argued later and illustrated with experimental results, smaller quadrotors exhibit higher accelerations allowing more rapid adaptation to disturbances and higher stability. Finally, their small size allows them to sustain less damage during crashes than their larger counterparts.

While the payload capacity of the quadrotor falls dramatically, it is possible to deploy multiple quadrotors that cooperate to overcome this limitation. Another disadvantage is the reduced flight time due to their limited capacity to carry batteries. However, this can be overcome with autonomous recharging schemes as illustrated by Valenti et al. (2006) and Mulgaonkar (2012). We investigate the aerodynamics of

A. Kushleyev · D. Mellinger · C. Powers (✉) · V. Kumar
GRASP Laboratory, 3330 Walnut Street, Philadelphia, PA, USA
e-mail: cpow@seas.upenn.edu

A. Kushleyev
e-mail: akushley@seas.upenn.edu

D. Mellinger
e-mail: dmel@seas.upenn.edu

flight in the vertical direction and flight along a horizontal trajectory close to the ground.

In the next section, we provide some of the scaling arguments that motivate the work in this paper. In Sect. 3, we discuss the system design, while the dynamics and control of the micro quadrotor are described in Sect. 4. The architecture and algorithms for formation flight are explained in Sect. 5 with experimental results in Sect. 6 and concluding remarks in Sect. 7.

Fig. 1 A formation of 20 micro quadrotors in flight. See video at http://youtu.be/50Fdi7712KQ

2 Agility of micro quadrotors

It is useful to develop a simple physics model to analyze a quadrotor's ability to produce linear and angular accelerations from a hover state. If the characteristic length is L, the rotor radius R scales linearly with L. The mass scales as L^3 and the moments of inertia as L^5. On the other hand the lift or thrust, F, and drag, D, from the rotors scale with the cross-sectional area and the square of the blade-tip velocity, v. If the angular speed of the blades is defined by $\omega = \frac{v}{L}$, $F \sim \omega^2 L^4$ and $D \sim \omega^2 L^4$. The linear acceleration a scales as $a \sim \frac{\omega^2 L^4}{L^3} = \omega^2 L$. Thrusts from the rotors produce a moment with a moment arm L. Thus the angular acceleration $\alpha \sim \frac{\omega^2 L^5}{L^5} = \omega^2$.

The rotor speed, ω, also scales with length since smaller motors produce less torque which limits their peak speed because of the drag resistance, D. There are two commonly accepted approaches to study scaling in aerial vehicles (Wolowicz et al. 1979). Mach scaling is used for compressible flows and essentially assumes that the tip velocities are constant leading to $\omega \sim \frac{1}{R}$. Froude scaling is used for incompressible flows and assumes that for similar aircraft configurations, the Froude number, $\frac{v^2}{Lg}$, is constant. Here g is the acceleration due to gravity. This yields $\omega \sim \frac{1}{\sqrt{R}}$. However, neither Froude or Mach number similitudes take motor characteristics nor battery properties into account. While motor torque increases with length, the operating speed for the rotors is determined by matching the torque-speed characteristics of the motor to the drag versus speed characteristics of the propellers (Kumar and Michael 2012). Further, the motor torque depends on the ability of the battery to source the required current. All these variables are tightly coupled for smaller designs since there are fewer choices of motors available at smaller length scales. Finally, the assumption that propeller blades are rigid may be wrong, and the aerodynamics of the blades at small scales may be different, so the quadratic scaling of the lift with speed may not be accurate. Nevertheless these two cases are meaningful since they provide some insight into the physics underlying the maneuverability of the craft (Fig. 1).

Froude scaling suggests that the acceleration is independent of length while the angular acceleration $\alpha \sim L^{-1}$. On the other hand, Mach scaling leads to the conclusion that $a \sim L$ while $\alpha \sim L^{-2}$. Since quadrotors must rotate in order to translate, the maximum allowable angular acceleration is the most critical factor for maneuverability, and therefore, smaller quadrotors are much more agile.

There are two design points that are illustrative of the quadrotor configuration. The Pelican quadrotor from Ascending Technologies (2012) equipped with sensors (approx. 2 kg gross weight, 0.75 m diameter, and 5,400 rpm nominal rotor speed at hover), consumes approximately 400 W of power (Shen et al. 2011). The Hummingbird quadrotor from Ascending Technologies (500 g gross weight, approximately 0.5 m diameter, and 5,000 rpm nominal rotor speed at hover) without additional sensors consumes about 75 W. The micro quadrotor described here is approximately 40 % of the size of the Hummingbird, 15 % of its mass, and consuming approximately 20 % of the power for hovering.

3 The micro quadrotor

3.1 The vehicle

The prototype quadrotor is shown in Fig. 2. Its booms are made of carbon fiber rods which are sandwiched between a custom motor controller board on the bottom and the main controller board on the top. To produce lift the vehicle uses four fixed-pitch propellers with diameters of 8 cm. The vehicle propeller-tip-to-propeller-tip distance is 21 cm and its weight without a battery is 50 g. The hover time is approximately 11 min with a 2-cell 400 mAh Li-Po battery that weighs 25 g.

3.2 Electronics

Despite its small size this vehicle contains a full suite of onboard sensors. An ARM Cortex-M3 processor, running at 72 MHz, serves as the main processor. The vehicle contains

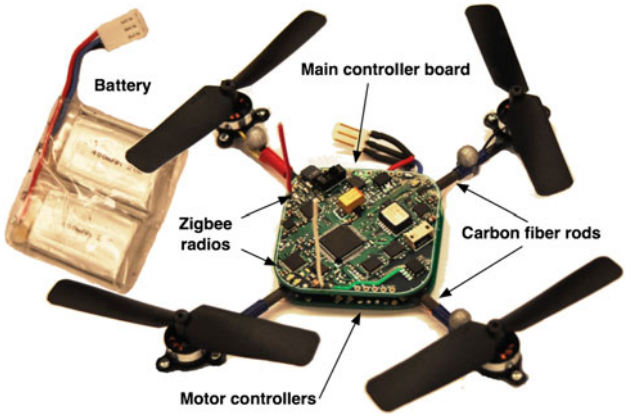


Fig. 2 A prototype micro quadrotor

a 3-axis magnetometer, a 3-axis accelerometer, a 2-axis 2,000 deg/s rate gyro for the roll and pitch axes, and a single-axis 500 deg/s rate gyro for the yaw axis. The vehicle also contains a barometer that can be used to sense a change in altitude. For communication the vehicle contains two Zigbee transceivers that can operate at either 900 MHz or 2.4 GHz.

3.3 Software infrastructure

The work in this paper uses a Vicon motion capture system 2012 to sense the position of each vehicle at 100 Hz as shown in Fig. 3. This data is streamed over a gigabit ethernet network to a desktop base station. High-level control and planning is done in MATLAB on the base station which sends commands to each quadrotor at 100 Hz. The software infrastructure for controlling a large team of quadrotors is described later in Sect. 6.1 (see Fig. 10). Low-level estimation and control loops run on the onboard microprocessor at a rate of 600 Hz.

Each quadrotor has two independent radio transceivers, operating at 900 MHz and 2.4 GHz. The base station sends, via custom radio modules, the desired commands, containing orientation, thrust, angular rates and attitude controller gains to the individual quadrotors. The onboard rate gyros and accelerometer are used to estimate the orientation and angular velocity of the craft. The main microprocessor runs the attitude controller described in Sect. 4 and sends the desired propeller speeds to each of the four motor controllers at full rate (600 Hz).

4 Dynamics and control

The dynamic model and control for the micro quadrotor is based on the approach in Mellinger and Kumar (2011). The analysis of the rotor aerodynamics is taken from the approach in Powers et al. (2012).

4.1 Rigid body dynamics

As shown in Fig. 4, we consider a body-fixed frame $\mathcal{B}$ aligned with the principal axes of the quadrotor (unit vectors $\mathbf{b}_i$) and an inertial frame $\mathcal{A}$ with unit vectors $\mathbf{a}_i$. $\mathcal{B}$ is described in $\mathcal{A}$ by a position vector $\mathbf{r}$ to the center of mass C and a rotation matrix R. In order to avoid singularities associated with parameterization, we use the full rotation matrix to describe orientations. The angular velocity of the quadrotor in the body frame, ω, is given by $\Omega = R^T \dot{R}$.

As shown in Fig. 4, the four rotors are numbered 1–4, with odd numbered rotors having a pitch that is opposite to

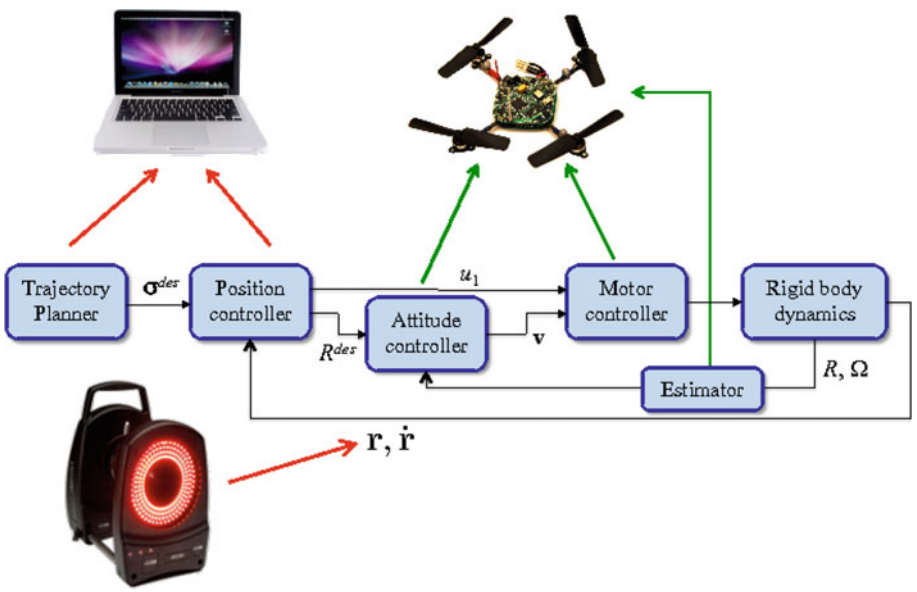


Fig. 3 The attitude estimation and control for roll and pitch is done on board while the position feedback, control and planning are done off board in MATLAB

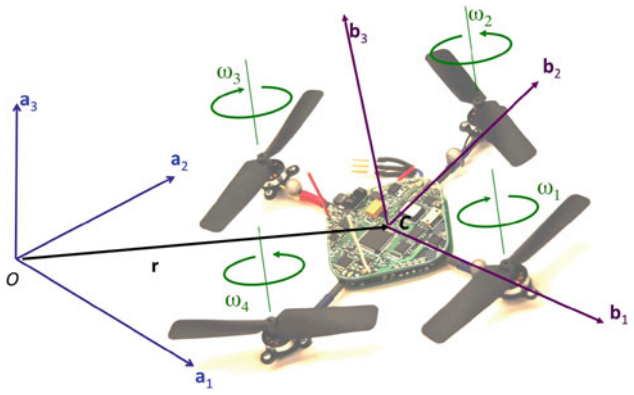


Fig. 4 The reference frames and propeller numbering convention

the even numbered rotors. The angular speed of the rotor is ω_i. The resulting lift, F_i, and the reaction moment, M_i, are given by:

$$F_i = k_F \omega_i^2, \quad M_i = k_M \omega_i^2. \tag{1}$$

where k_F and k_M are empirically determined functions of the rotor geometry and aerodynamics as discussed in Sect. 4.3, which we model as constants for the purpose of control. For our micro quadrotor, the motor dynamics have a time constant less than 10 msec and are much faster than the time scale of rigid body dynamics and aerodynamics. Thus we neglect the dynamics and assume F_i and M_i can be instantaneously changed. Including the motor dynamics would increase the order of the system. Therefore the control inputs to the system, $\mathbf{u}$, consists of the net thrust in the $\mathbf{b}_3$ direction, $u_1 = \Sigma_{i=1}^4 F_i$, and the moments in $\mathcal{B}$, $\mathbf{v} = [u_2, u_3, u_4]^T$, given by:

$$\mathbf{u} = \begin{bmatrix} k_F & k_F & k_F & k_F \\ 0 & k_F L & 0 & -k_F L \\ -k_F L & 0 & k_F L & 0 \\ k_M & -k_M & k_M & -k_M \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}, \tag{2}$$

where L is the distance from the axis of rotation of the propellers to the center of the quadrotor.

The Newton–Euler equations of motion are given by:

$$m\ddot{\mathbf{r}} = -mg\mathbf{a}_3 + u_1 \mathbf{b}_3 \tag{3}$$

$$\dot{\omega} = \mathcal{I}^{-1} \left[-\omega \times \mathcal{I}\omega + \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} \right] \tag{4}$$

where $\mathcal{I}$ is the moment of inertia matrix along $\mathbf{b}_i$.

We specify the desired trajectory using a time-parameterized position vector and yaw angle. Given a trajectory, $\sigma(t) : [0, t_f] \to \mathbb{R}^3 \times SO(2)$, the controller derives the input u_1 based on position and velocity errors:

$$u_1 = (-K_p \mathbf{e}_p - K_v \mathbf{e}_v + mg\mathbf{a}_3) \cdot \mathbf{b}_3 \tag{5}$$

where $\mathbf{e}_p = \mathbf{r} - \mathbf{r}_T$ and $\mathbf{e}_v = \dot{\mathbf{r}} - \dot{\mathbf{r}}_T$. The other three inputs are determined by computing the desired rotation matrix. We want to align the thrust vector $u_1 \mathbf{b}_3$ with $(-K_p \mathbf{e}_p - K_v \mathbf{e}_v + mg\mathbf{a}_3)$ in (5). Second, we want the yaw angle to follow the specified yaw $\psi_T(t)$. From these two pieces of information we can compute R_{des} and the error in rotation according to:

$$\mathbf{e}_R = \frac{1}{2}(R_{\text{des}}^T R - R^T R_{\text{des}})^\vee$$

where $^\vee$ represents the *vee map* which takes elements of $so(3)$ to $\mathbb{R}^3$. The *vee map* takes A onto a such that $A^\vee b = a \times b \; \forall A \in so(3), b \in R^3$. The desired angular velocity is computed by differentiating the expression for R and the desired moments can be expressed as a function of the orientation error, $\mathbf{e}_R$, and the angular velocity error, $\mathbf{e}_\omega$:

$$[u_2, u_3, u_4]^T = -K_R \mathbf{e}_R - K_\omega \mathbf{e}_\omega, \tag{6}$$

where K_R and K_ω are diagonal gain matrices. Finally we compute the desired rotor speeds to achieve the desired $\mathbf{u}$ by inverting (2).

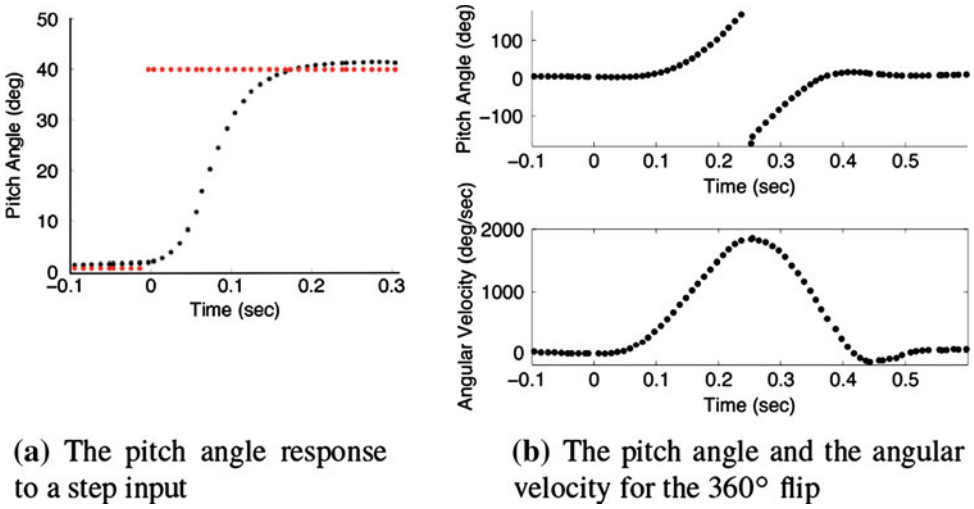
4.2 Performance

Experimental results from the onboard attitude controller are shown in Fig. 5. The small moments of inertia of the vehicle enable the vehicle to create large angular accelerations. As shown in Fig. 5a the attitude control is designed to be approximately critically damped with a settling time of less than 0.2 s. Note that this is twice as fast as the settling time for the attitude controller for the AscTec Hummingbird reported in (Mellinger et al. 2010). Data for a 360° flip is presented in 5b. Here the vehicle completes a complete flip about its y axis in about 0.4 s and reaches a maximum angular velocity of 1,850 deg/s.

The position controller described in Sect. 4.1 uses the roll and pitch angles to control the x and y position of the vehicle. For this reason, a stiff attitude controller is a required for stiff position control. Response to step inputs in the lateral and vertical directions are shown in Fig. 6b. For the hovering performance data shown in Fig. 6a the standard deviations of the error for x and y are about 0.75 cm and about 0.2 cm for z.

4.3 Aerodynamics of forward and vertical flight

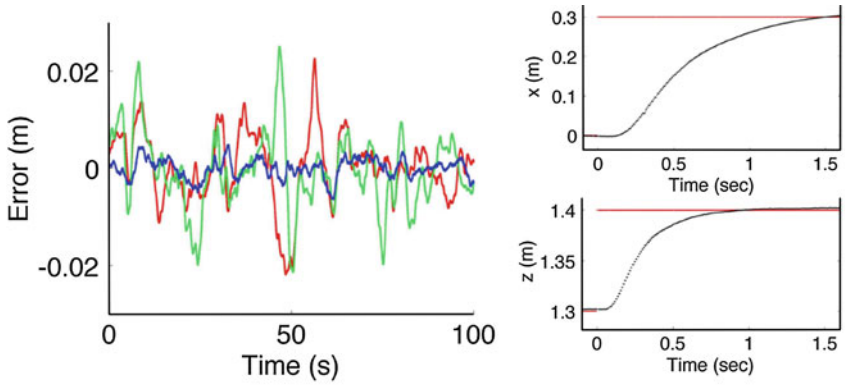
In the preceding section, we assume that the force, or thrust, produced by the rotors is a function only of the square of their speed, ω^2. However, in reality there are various aerodynamic phenomena which can cause the produced thrust to vary. Rotor blades can encounter blade flapping or ground effect, and the thrust can also vary with the speed and angle of the rotor relative to the air surrounding it (Johnson 1980). The effect of blade flapping on quadrotor flight has been



(a) The pitch angle response to a step input

(b) The pitch angle and the angular velocity for the 360° flip

Fig. 5 Attitude controller performance data



(a) Position error

(b) Position step input response

Fig. 6 The *red*, *green*, and *blue lines* in **a** represent the x, y, and z errors while hovering. **b** The step response for the position controller in x (*top*) and z (*bottom*) (Color figure online)

investigated in Huang et al. (2009), and flight near surfaces has been studied in Kim et al. (2012). In this section, we present a simple discussion of the aerodynamics of vertical flight ($\alpha = 0$) as well as ground effect. Details about forward flight ($\alpha \neq 0$) can be found in Powers et al. (2012).

4.3.1 Vertical flight

Two methods are commonly used to derive the total thrust generated from a rotor (Johnson 1980). The first, momentum theory, relies on conservation of momentum in the fluid surrounding the rotor. As the column of air surrounding the rotor travels through the plane of the rotor, the velocity of the fluid increases. This additional velocity is called the induced velocity, v. The thrust in vertical flight according to momentum theory can be expressed as

$$F = 2\rho A v|V + v| \tag{7}$$

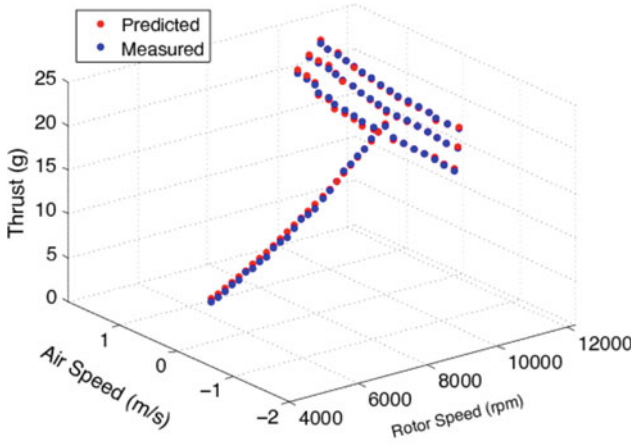
where F is the thrust as in the rigid body dynamics, ρ is the density of the surrounding medium (air, for quadrotors), v is the induced velocity, and V is the relative velocity between the rotor and the surrounding air.

The second, blade element theory, integrates the differential forces (thrust, drag) along the blade. This analysis leads to the relationship

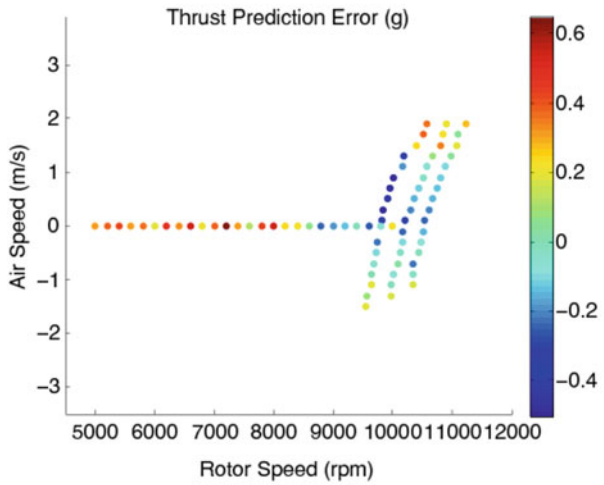
$$F = \frac{\rho abc\omega^2 R^3}{4}\left(\theta_{\text{tip}} - \frac{V + v}{\omega R}\right) \tag{8}$$

where R is the radius of the rotor and a, b, c and θ_{tip} are functions of the blade geometry which remain fixed for a specific vehicle.

Since the induced velocity appears in both (8) and (7) and can only be known via measurement, we combine both of these approaches to find the rotor thrust at a given velocity V. This is done by experimentally finding the thrust produced at several combinations of V and ω, shown in Fig. 7a, fitting a model to the data, and then solving the system of two nonlinear equations for both F and v. Measurements were made on an experimental apparatus commanding a fixed rotor to various speeds without extraneous air flow, which form the line of measurements at zero air speed. Additional



(a) Thrust Data and Predictions



(b) Prediction Errors

Fig. 7 Thrust data and predictions. Thrust data was collected for several rotor speeds and air speeds and used to fit a model of thrust dependence on air speed. The prediction errors are less than 1 g while the thrust produced ranges over 20 g

measurements were made by commanding a quadrotor to fly at a certain fixed velocity with various payloads and observing the resulting rotor speeds, forming the measurements at constant thrust. Insight into the rotor aerodynamics can help us design better controllers for aggressive trajectories. However, for moderate speeds ($\leq$2 m/s), it is not necessary to incorporate this high fidelity model. For the remainder of the paper, we neglect these effects and use the simple model in (1).

4.3.2 Ground effect

When a rotor is operating near the ground, more thrust is produced at a given rotor speed ω. This phenomenon is extensively documented in helicopter literature (Johnson 1980; Leishman 2000). We demonstrate in Fig. 8 that ground effect is present for the micro UAVs we describe. In addition, we show results indicating that a similar phenomenon is observed when hovering underneath large surfaces, which we call "ceiling effect". It is proposed in (Johnson 1980) that the ratio of force produced by a rotor in ground effect to the force produced far from the ground (F_∞) can be modeled as

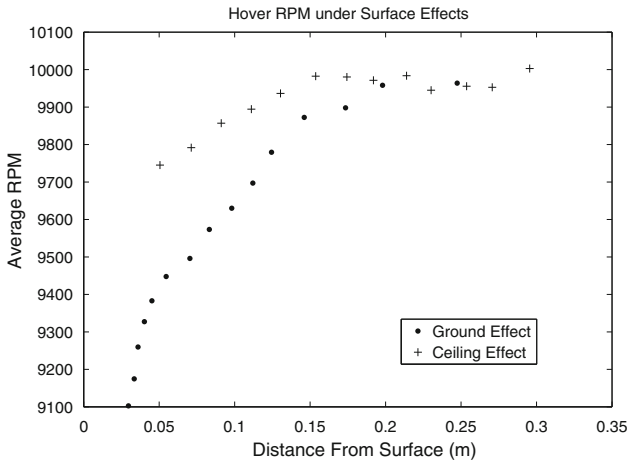


Fig. 8 Flight near surfaces. The RPM required to hover decreases as the vehicle becomes closer to the surface

$$\frac{F}{F_\infty} = \frac{1}{1 - \left(\frac{R}{4z}\right)^2} \tag{9}$$

where R is the radius of the rotor and z is the height off of the ground. To experimentally test this relationship for our quadrotor platform, a micro UAV was commanded to hover above the ground at heights ranging from 3 to 25 cm. We then recorded the steady state rotor speed required to hover, which is shown in Fig. 8. The same experiment was performed for "ceiling effect", using a large clear acrylic sheet as the top surface. Our data shows that the magnitude of the "ceiling effect" is much smaller than the ground effect. Quantifying the effects of flight near surface is very important for maneuvers such as take off and landing. However, for the purposes of this paper these effects are easily avoided by ensuring a spacing of at least 0.5 m between any horizontal surface.

5 Control and planning for groups

5.1 Architecture

We are primarily interested in the challenge of coordinating a large team of quadrotors. To manage the complexity that results from growth of the state space dimensionality and limit the combinatorial explosion arising from interactions

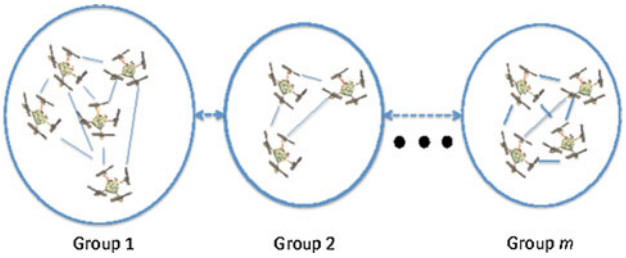


Fig. 9 The team of quadrotors is organized into m groups. While vehicles within the group are tightly coordinated and centralized control and planning is possible, the inter-group coordination need not be centralized

between labeled vehicles, we consider a team architecture in which the team is organized into labeled groups, each with labeled vehicles. Formally, we can define a *group* of agents as a collection of agents which work simultaneously to complete a single task. Two or more groups act in a *team* to complete a task which requires completing multiple parallel subtasks (Anderson and Franks 2001). We assume that vehicles within a group can communicate at high data rates with low latencies while the communication requirements for coordination across groups are much less stringent. Most importantly, vehicles within a group are labeled. The small group size allows us to design controllers and planners that provide global guarantees on shapes, communication topology, and relative positions of individual, agile robots.

Our approach is in contrast to truly decentralized approaches which are necessary in swarms with hundreds and thousands of agents (Parrish and Hamner 1997). While models of leaderless aggregation and swarming with aerial robots are discussed in the robotics community (Bullo et al. 2009; Tanner et al. 2007; Michael and Kumar 2011), here the challenge of enumerating labeled interactions between robots is circumvented by controlling such aggregate descriptors of formation as statistical distributions. These methods cannot provide guarantees on shape or topology. Reciprocal collision avoidance algorithms (van den Berg et al. 2011) have the potential to navigate robots to goal destinations but no guarantees are available for transient performance and no proof of convergence is available.

On the other hand, the problem of designing decentralized controllers for trajectory tracking for three dimensional rigid structures is now fairly well understood (Desai et al. 2001; Egerstedt and Hu 2001; Beard et al. 2001), although few experimental results are available for aerial robots. Our framework allows the maintenance of such rigid structures in groups (Fig. 9).

5.2 Formation flight

Flying in formation reduces the complex task of generating trajectories for individuals in a large team of vehicles to that of generating a trajectory for a small number of groups. If the controllers are well designed, there is no need to explicitly incorporate collision avoidance between vehicles within a group. The position error for quadrotor q at time t can be written as

$$\mathbf{e}_{pq}(t) = \mathbf{e}_f(t) + \mathbf{e}_{lq}(t) \tag{10}$$

where $\mathbf{e}_f(t) \in \mathbb{R}^3 \times SO(2)$ is the gross *formation error* describing the error of position of the group from the prescribed trajectory, and $\mathbf{e}_{lq}(t) \in \mathbb{R}^3 \times SO(2)$ is the local error of quadrotor q within the formation of the group. In our implementation, the position of the formation is taken to be the mean of the positions of each vehicle. Any formation can be defined this way without loss of generality and it provides a convenient way to derive formation and local errors from measurements of vehicle positions. As we will show in Sect. 6.2 the local error is typically quite small even for aggressive trajectories although the formation error can be quite large.

A major disadvantage of formation flight is that the effective dimensions of the group increase with the size of the group and a rigid group formation can no longer fly through narrow gaps. This means that the shape of the formation of the team must be changed on the fly or the team must be divided into smaller groups as needed, allowing each group to negotiate the gap independently. The latter increases the complexity of the planning problem as the number of groups increase. However, if we constrain the groups to follow the same path, it is possible to reduce the complexity of the trajectory generation problem. This is discussed in Sect. 5.3.

5.3 Time-separated trajectory following

To reduce the complexity of the trajectory generation problem we require all vehicles to follow the same path but be separated by a specified time interval. Here we let the trajectory for quadrotor q be defined as

$$\mathbf{r}_{Tq}(t) = \mathbf{r}_{TT}(t + \Delta t_q) \tag{11}$$

where $\mathbf{r}_{TT}$ is the team trajectory and Δt_q is the specified time shift for quadrotor q from some common clock time, t. If the team path does not intersect or does not bend around so that robots can come within an unsafe distance of other robots then the robots simply need to follow each other at a safe time separation. Large numbers of vehicles can follow team paths that intersect themselves if the time separations, Δt_q, are chosen carefully so that no two vehicles are at any of the intersection points at the same time. An experiment for an intersecting team trajectory is shown in Sect. 6.2.

5.4 Trajectory generation with MIQPs

The generation of smooth, safe trajectories through known 3-D environments satisfying specifications on intermediate

waypoints for multiple vehicles builds on our previous work in Mellinger and Kumar (2011). Integer constraints are used to enforce collision constraints with obstacles and other vehicles and also to optimally assign goal positions. This method draws from the extensive literature on mixed-integer linear programs and their application to trajectory planning from Schouwenaars et al. (2001, 2006).

5.4.1 Basic method

As described in Mellinger and Kumar (2011) an optimization program can be used to generate trajectories that smoothly transition through n_w desired waypoints at specified times, t_w. The optimization program to solve this problem while minimizing the integral of the k_rth derivative of position squared for n_q quadrotors is shown below.

$$\min \sum_{q=1}^{n_q} \int_{t_0}^{t_{n_w}} \left\| \frac{d^{k_r} \mathbf{r_{Tq}}}{dt^{k_r}} \right\|^2 dt \tag{12}$$

$$\text{s.t.} \quad \mathbf{r}_{Tq}(t_w) = \mathbf{r}_{wq}, \quad w = 0, \ldots, n_w; \quad \forall q$$

$$\frac{d^j x_{Tq}}{dt^j}|_{t=t_w} = 0 \text{ or free}, \quad w = 0, n_w; \quad j = 1, \ldots, k_r; \quad \forall q$$

$$\frac{d^j y_{Tq}}{dt^j}|_{t=t_w} = 0 \text{ or free}, \quad w = 0, n_w; \quad j = 1, \ldots, k_r; \quad \forall q$$

$$\frac{d^j z_{Tq}}{dt^j}|_{t=t_w} = 0 \text{ or free}, \quad w = 0, n_w; \quad j = 1, \ldots, k_r; \quad \forall q$$

Here $\mathbf{r}_{Tq} = [x_{Tq}, y_{Tq}, z_{Tq}]$ represents the trajectory for quadrotor q and $\mathbf{r}_{wq}$ represents the desired waypoints for quadrotor q. We enforce continuity of the first k_r derivatives of $\mathbf{r}_{Tq}$ at $t_1, \ldots, t_{n_w-1}$. As shown in Mellinger and Kumar (2011) writing the trajectories as piecewise polynomial functions allows (12) to be written as a quadratic program (or QP) in which the decision variables are the coefficients of the polynomials.

For quadrotors, since the inputs u_2 and u_3 appear as functions of the fourth derivatives of the positions, we generate trajectories that minimize the integral of the square of the norm of the snap (the second derivative of acceleration, $k_r = 4$). Large order polynomials are used to satisfy such additional trajectory constraints as obstacle avoidance that are not explicitly specified by intermediate waypoints. This QP has $3n_w n_q n_p$ variables and $3n_q n_w + 6n_q k_r$ constraints, where n_p is the number of polynomial coefficients. As an example, consider a situation with 5 quadrotors directed to two waypoints and nine coefficients, which leads to at most 270 variables and 150 constraints. However, the problem grows as coupling is introduced between the quadrotors in the form of constraints for collision avoidance. The computational complexity of the optimization problem is discussed further in Mellinger et al. (2012).

5.4.2 Integer constraints for collision avoidance

For collision avoidance we model the quadrotors as a rectangular prisms oriented with the world frame with side lengths l_x, l_y, and l_z. These lengths are large enough so that the quadrotor can roll, pitch, and yaw to any angle and stay within the prism. We consider navigating this prism through an environment with n_o convex obstacles. Each convex obstacle o can be represented by a convex region in configuration space with $n_f(o)$ faces. For each face f the condition that the quadrotor's desired position at time t_k, $\mathbf{r}_{Tq}(t_k)$, be outside of obstacle o can be written as

$$\mathbf{n}_{of} \cdot \mathbf{r}_{Tq}(t_k) \leq s_{of}, \tag{13}$$

where $\mathbf{n}_{of}$ is the normal vector to face f of obstacle o in configuration space and s_{of} is a scalar that determines the location of the plane. If (13) is satisfied for *at least* one of the faces then the rectangular prism, and hence the quadrotor, is not in collision with the obstacle. The condition that quadrotor q does not collide with an obstacle o at time t_k can be enforced with binary variables, b_{qofk}, as

$$\begin{aligned} \mathbf{n}_{of} \cdot \mathbf{r}_{Tq}(t_k) &\leq s_{of} + M b_{qofk} \forall f = 1, \ldots, n_f(o) \\ b_{qofk} &= 0 \text{ or } 1 \forall f = 1, \ldots, n_f(o) \\ \sum_{f=1}^{n_f(o)} b_{qofk} &\leq n_f(o) - 1 \end{aligned} \tag{14}$$

where M is a large positive number (Schouwenaars et al. 2001). Note that if b_{qofk} is 1 then the inequality for face f is always satisfied. The last inequality in (14) requires that the non-collision constraint be satisfied for at least one face of the obstacle which implies that the prism does not collide with the obstacle. We can then introduce (14) into (12) for all n_q quadrotors for all n_o obstacles at n_k intermediate time steps between waypoints. The addition of the integer variables into the QP causes this optimization problem to become a mixed-integer quadratic program (MIQP).

5.4.3 Inter-quadrotor collision avoidance

When transitioning between waypoints quadrotors must stay a safe distance away from each other. We enforce this constraint at n_k intermediate time steps between waypoints which can be represented mathematically for quadrotors 1 and 2 by the following set of constraints:

$$\begin{aligned} \forall t_k \; &: x_{T1}(t_k) - x_{T2}(t_k) \leq d_{x12} \\ &\text{or} x_{T2}(t_k) - x_{T1}(t_k) \leq d_{x21} \\ &\text{or} y_{T1}(t_k) - y_{T2}(t_k) \leq d_{y12} \\ &\text{or} y_{T2}(t_k) - y_{T1}(t_k) \leq d_{y21} \end{aligned} \tag{15}$$

Here the d terms represent safety distances. For axially symmetric vehicles $d_{x12} = d_{x21} = d_{y12} = d_{y21}$. Experimentally

we have found that quadrotors must avoid flying in each other's downwash because of a decrease in tracking performance and even instability in the worst cases. Therefore we do not allow vehicles to fly underneath each other here. Finally, we incorporate constraints (15) between all n_q quadrotors in the same manner as in (14) into (12).

5.4.4 Integer constraints for optimal goal assignment

In many cases one might not care that a certain quadrotor goes to a certain goal but rather that any vehicle does. Here we describe a method for using integer constraints to find the optimal goal assignments for the vehicles. This results in a lower total cost compared to fixed-goal assignment and often a faster planning time because there are more degrees of freedom in the optimization problem. For each quadrotor q and goal g we introduce the integer constraints:

$$\begin{aligned} x_{Tq}(t_{n_w}) &\leq x_g + M\beta_{qg} \\ x_{Tq}(t_{n_w}) &\geq x_g - M\beta_{qg} \\ y_{Tq}(t_{n_w}) &\leq y_g + M\beta_{qg} \\ y_{Tq}(t_{n_w}) &\geq y_g - M\beta_{qg} \\ z_{Tq}(t_{n_w}) &\leq z_g + M\beta_{qg} \\ z_{Tq}(t_{n_w}) &\geq z_g - M\beta_{qg} \end{aligned} \tag{16}$$

Here β_{qg} is a binary variable used to enforce the optimal goal assignment. If β_{qg} is 0 then quadrotor q must be at goal g at t_{n_w}. If β_{qg} is 1 then these constraints are satisfied for any final position of quadrotor q. In order to guarantee that at least n_g quadrotors reach the desired goals we introduce the following constraint.

$$\sum_{q=1}^{n_q}\sum_{g=1}^{n_g}\beta_{qg} \leq n_g n_q - n_g \tag{17}$$

Note that this approach can be easily adapted if there are more quadrotors than goals or vice versa.

5.4.5 Relaxations for large teams

The solving time of the MIQP grows exponentially with the number of binary variables that are introduced into the MIQP. Therefore, the direct use of this method does not scale well for large teams. Here we present two relaxations that enable this approach to be used for large teams of vehicles.

Planning for groups within a team A large team of vehicles can be divided into smaller groups. We can then use the MIQP method to generate trajectories to transition groups of vehicles to group goal locations. This reduces the complexity of the MIQP because instead of planning trajectories for all n_q vehicles we simply plan trajectories for the groups. Of course we are making a sacrifice here by not allowing the quadrotors to have the flexibility to move independently.

Planning for sub-regions In many cases the environment can be partitioned into n_r convex sub-regions where each sub-region contains the same number of quadrotor start and goal positions. After partitioning the environment the MIQP trajectory generation method can be used for the vehicles inside each region. Here we require quadrotors to stay inside their own regions using linear constraints on the positions of the vehicles. This approach guarantees collision-free trajectories and allows quadrotors the flexibility to move independently. We are gaining tractability at the expense of optimality since the true optimal solution might actually require quadrotors to cross region boundaries while this relaxed version does not. Also, it is possible that no feasible trajectories exist inside a sub-region but feasible trajectories do exist which cross region boundaries. Nonetheless, this approach works well in many scenarios and we show its application to formation transitions for teams of 16 vehicles in Sect. 6.5.2.

6 Experimental results

6.1 Software infrastructure for groups

Our architecture (Sect. 5.1) is important for a very practical reason. For a large team of quadrotors it is impossible to run a single loop that can receive all the Vicon data, compute the commands, and communicate with each quadrotor at a fast enough rate. As shown in Fig. 10, each group is controlled by a dedicated software node, running in an independent thread. These control nodes receive vehicle pose data from a special Vicon node via shared memory. The node connects to the Vicon tracking system, receives marker positions for each subject, performs a 6D pose fit to the marker data and does additional processing for velocity estimation. Finally, the processed pose estimates are published to the shared memory using the Boost C++ library 2012. Shared memory is the fastest method of inter-process communica-

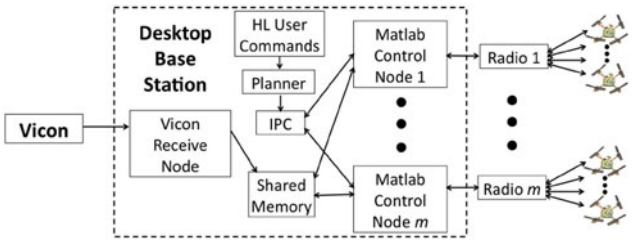


Fig. 10 Software Infrastructure

tion, which ensures the lowest latency of the time-critical data.

The control nodes, implemented in MATLAB, read the pose data directly from shared memory and compute the commanded orientation and net thrusts for several quadrotors based on the controller described in Sect. 4.1. For non-time-critical data sharing we use Inter Process Communication (IPC) 2012. For example, high-level user commands such as desired vehicle positions are sent to a planner which computes the trajectories for the vehicles which are sent to the MATLAB control nodes via IPC. IPC provides flexible message passing and uses TCP/IP sockets to send data between processes.

Each MATLAB control node is associated with a radio module containing a 900 MHz and 2.4 GHz Zigbee transceivers, which is used to communicate with all the vehicles in its group. The radio module sends control commands to several vehicles, up to five in this work. Each vehicle operates on a separate channel and the radio module hops between the frequencies for each quadrotor, sending out commands to each vehicle at 100 Hz. The radio modules can also simultaneously receive high bandwidth feedback from the vehicles, making use of the two independent transceivers.

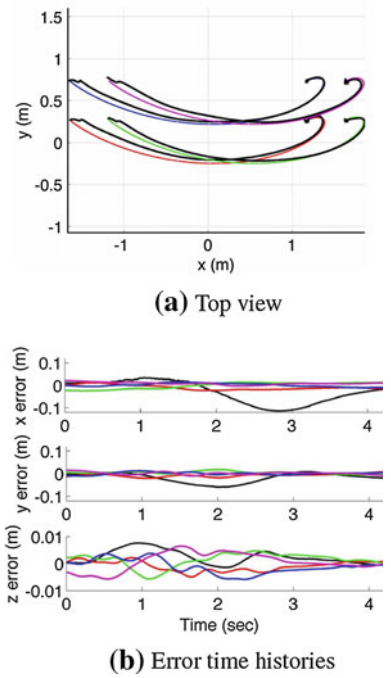


Fig. 11 Formation following for a four quadrotor trajectory. In **a** the *colored lines* represent the desired trajectories for each of the four vehicles and the *black lines* represent the actual trajectories. The errors are shown in (**b**). Here the *black line* represents the formation error, $\mathbf{e}_f(t)$, from the desired trajectory and the *colored lines* represent the local errors, $\mathbf{e}_{li}(t)$, for each quadrotor (Color figure online)

6.2 Formation flight

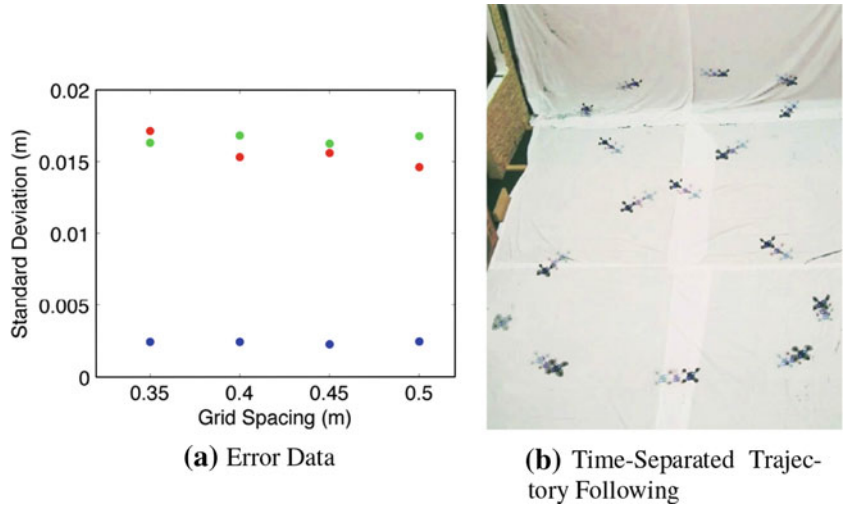
In Fig. 11 we present data for a team of four quadrotors following a trajectory as a formation. The group formation error is significantly larger than the local error. The local x and y errors are always less than 3 cm while the formation x error is as large 11 cm. This data is representative of all formation trajectory following data because all vehicles are nominally the same and are running the same controller with the same gains. Therefore, even though the deviation from the desired trajectory may be large, the relative position error within the group is small. Taking into account the magnitude of the formation and local errors when specifying the rectangular prism for obstacle avoidance will decrease the likelihood of collisions.

In Fig. 12a we show average error data for 20 vehicles flying in the grid formation shown in Fig. 1. For this experiment the vehicles were controlled to hover at a height of 1.3 meters for at least 30 s at several quadrotor-center-to-quadrotor-center grid spacing distances. The air disturbance created from the downwash of all 20 vehicles is significant and causes the tracking performance for any vehicle in this formation to be worse than for an individual vehicle in still air as presented in Fig. 6. However, as shown in Fig. 12a, the separation distance did not have any affect on the hovering performance. Note that at 35 cm grid spacing the nominal distance between propeller tips is about 14 cm.

6.3 Proximity effects

We would like to experimentally verify that the micro UAVs are able to operate in tight formations. In order to test the effect of neighboring vehicles on the flight performance of the quadrotors, we flew groups of up to three vehicles in three formations, shown in Fig. 13a. The first formation is a control experiment of one vehicle flying in a line along the x axis. The second formation is comprised of three vehicles flying along the same line. This tests the interaction of vehicles flying along the same path in close proximity, such as the time separated trajectory following described in Sect. 6.4. The final formation involves three vehicles flying along parallel but offset lines in opposite directions. This will test interactions between vehicles whose paths are oriented differently from each other.

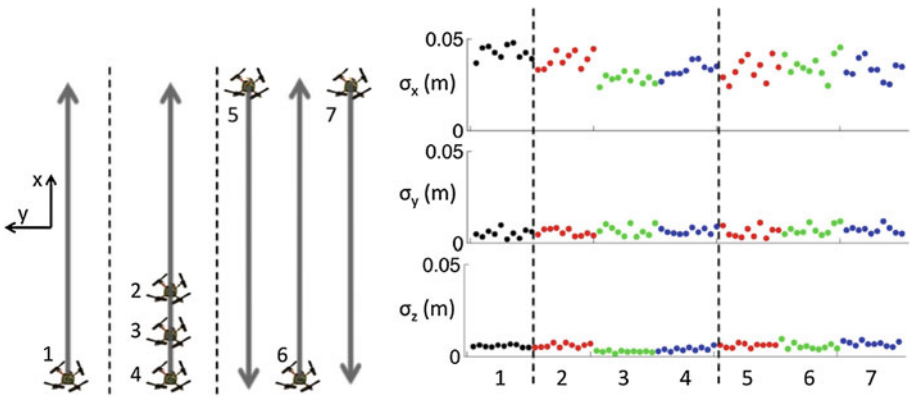
In Fig. 13b, we see the average deviations from the trajectory on a given run for each of the vehicles. Each configuration was run 10 times, with each run represented by a single dot. For all vehicles, the greatest deviations are seen in the direction of the x axis, due to lag in the trajectory. The tight control performance of the system is seen in the small deviations (less than 5 cm) in y and z directions. For all of the 10 runs, vehicles 3 and 4 (followers) exhibit smaller σ_x than either the leader (vehicle 2) or the control (vehicle 1). This



(a) Error Data

(b) Time-Separated Trajectory Following

Fig. 12 **a** The average standard deviation for x, y, and z (shown in *red*, *green* and *blue* respectively) for 20 quadrotors in a grid formation. In **b** we show 16 quadrotors following a figure eight pattern. See the video at http://youtu.be/50Fdi7712KQ (Color figure online)



(a) Trajectory Illustration

(b) Average Deviations from Trajectory

Fig. 13 **a** The trajectories followed to test proximity effects. In **b** we show average standard deviation for x, y, and z for each of the quadrotors, labeled by number shown in (**a**)

may be an indication of some drafting effect seen in many aerodynamic situations.

6.4 Time-separated trajectory following

In Fig. 12b we show a team of 16 vehicles following a cyclic *figure eight* pattern. The time to complete the entire cycle is t_c and the vehicles are equally spaced in time along the trajectory at time increments of $\frac{t_c}{16}$. In order to guarantee collision-free trajectories at the intersection, vehicles spend $\frac{15}{32}t_c$ in one loop of the trajectory and $\frac{17}{32}t_c$ in the other. A trajectory that satisfies these timing constraints and has some specified velocity at the intersection point (with zero acceleration and jerk) is generated using the optimization-based method for a single vehicle described in Mellinger and Kumar (2011).

6.5 MIQP trajectories

In this paper, we use a branch and bound solver (CPL 2009) to solve the MIQP trajectory generation problem. The solving time for the MIQP is an exponential function of the number of binary constraints and also the geometric complexity of the environment. The first solution is often delivered within seconds but finding the true optimal solution and a certificate of optimality can take as long as 20 min on a 3.4 GHz Core-i7 Quad-Core desktop machine for the examples presented here.

6.5.1 Planning for groups within a team

In Fig. 14 we show snapshots from an experiment for four groups of four quadrotors transitioning from one side of a gap to the other. Note that in this example the optimal goal assignment is performed at the group-level.

6.5.2 Planning for sub-regions

In Fig. 15 we show snapshots from an experiments with 16 vehicles transitioning from a planar grid to a three-dimensional helix and pyramid. Directly using the MIQP approach to generate trajectories for 16 vehicles is not practical. Therefore, in both experiments the space is divided into

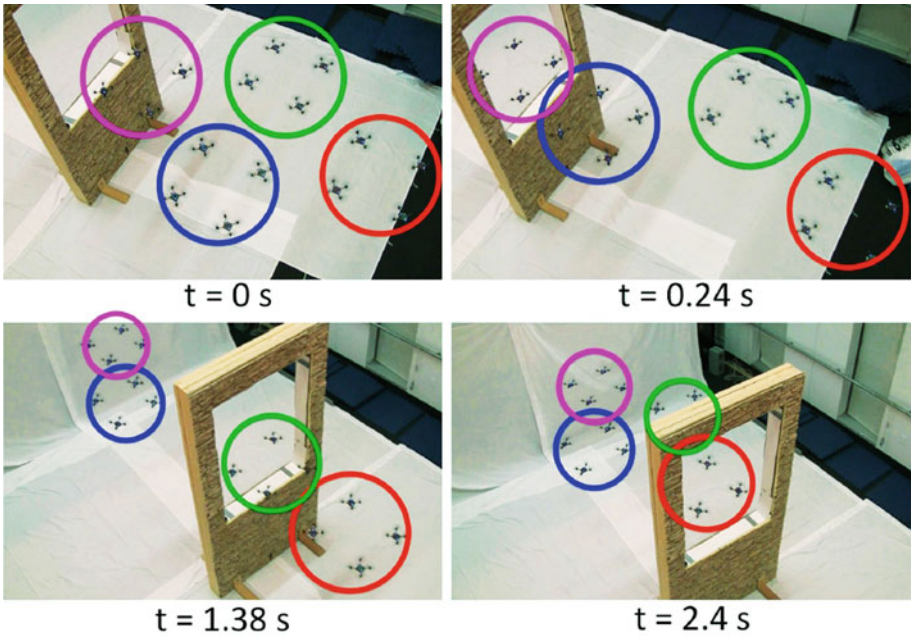


Fig. 14 Four groups of four quadrotors flying through a window

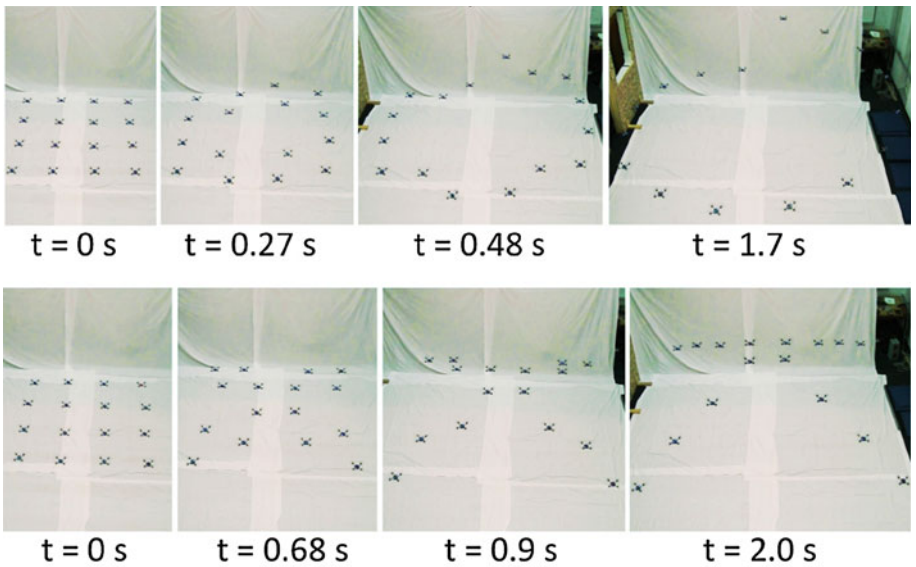


Fig. 15 A team of 16 vehicles transitioning from a planar grid to a three-dimensional helix (*top*) and pyramid (*bottom*)

two regions and separate MIQPs with eight vehicles each are used to generate trajectories for vehicles on the left and right sides of the formation. Note that, in general, the formations do not have to be symmetric but here we exploit the symmetry and only solve a single MIQP for eight vehicles for these examples. Optimal goal assignment is used so that the vehicles collectively choose their goals to minimize the total cost.

7 Conclusion

In this paper we describe the design, modeling and control of a micro quadrotor that has a mass of 75 g and is 21 cm in diameter, and the architecture and software for coordinating a team of micro quadrotors with experimental results. While our quadrotors rely on an external localization system for position estimation and therefore cannot be truly decentralized at this stage, these results represent the first step toward the development of a swarm of micro quadrotors. The small size is shown to increase agility and the ability to fly in close proximity with less than one body length separation without significant aerodynamic effects. Mixed integer quadratic programming techniques are used to coordinate twenty micro quadrotors in known three-dimensional environments with obstacles. MIQP techniques suffer from increased run times as the number of variables and constraints

increase. We mitigate this problem by dividing large teams of robots into groups, each with a small number of robots in a rigid formation, which trades off flexibility in trajectories for reduced planning time. Planning time can also be reduced by using feasible solutions instead of finding optimal solutions. For applications depending heavily on planning time, other planning techniques should be considered. The videos of all experiments are available at http://youtu.be/50Fdi7712KQ

Acknowledgments We would like to thank Dr. Bruce Kothmann for his help and insight into helicopter aerodynamics.

References

Aeroenvironment. (2011). Aeroenvironment nano hummingbird. http://www.avinc.com/nano. Accessed Oct 2012.

Anderson, C., & Franks, N. R. (2001). Teams in animal societies. *Behavioral Ecology*, *12*(5), 534540.

Ascending Technologies, GmbH. (2012). http://www.asctec.de. Accessed Oct 2012.

Beard, R. W., Lawton, J., & Hadaegh, F. Y. (2001). A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology*, *9*(6), 777–790.

Bermes, C (2010). Design and dynamic modeling of autonomous coaxial micro helicopters. PhD thesis, ETH Zurich, Switzerland.

Bouabdallah, S (2007). Design and control of quadrotors with applications to autonomous flying. PhD thesis, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland.

Boost C++ Libraries. (2012). http://www.boost.org. Accessed Oct 2012.

Bullo, F., Cortés, J., & Martínez, S. (2009). *Distributed control of robotic networks: A mathematical approach to motion coordination algorithms*. Princeton, NJ: Princeton University Press. (Applied Mathematics Series).

Desai, J. P., Ostrowski, J. P., & Kumar, V. (2001). Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics*, *17*(6), 905–908.

Egerstedt, M., & Hu, X. (2001). Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, *17*(6), 947–951.

Floreano, D., Zufferey, J. C., Klaptocz, A., Germann, J. M., & Kovac, M. (2011). Aerial locomotion in cluttered environments. In *Proceedings of the 15th international symposium on robotics research*. Flagstaff, AZ.

Huang, H., Hoffmann, G., Waslander, S., & Tomlin, C. (2009). Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In *ICRA '09 IEEE international conference on robotics and automation* (pp. 3277–3282). doi:10.1109/ROBOT.2009.5152561.

International Business Machines Corporation. (2009). *IBM ILOG CPLEX V12.1: Users manual for CPLEX*. New York: IBM.

Inter Process Communication. (2012). http://www.cs.cmu.edu/ipc/. Accessed Oct 2012.

Johnson, W. (1980). *Helicopter theory*. Princeton, NJ: Princeton University Press.

Kim, H.J., Kim, S., Lee, D., & Awan, A. (2012). Adaptive control for a VTOL UAV operating near a wall. In *AIAA Guidance, Navigation, and Control Conference, Guidance, Navigation, and Control and co-located conferences*. Boston: American Institute of Aeronautics and Astronautics. doi:10.2514/6.2012-4835.

Kumar, V., & Michael, N. (2011). Opportunities and challenges with autonomous micro aerial vehicles. In *International symposium on robotics research*. Flagstaff, AZ.

Kumar, V., & Michael, N. (2012). Opportunities and challenges with autonomous micro aerial vehicles. *International Journal of Robotics Research*, *31*(11), 1279–1291.

Kushleyev, A., Mellinger, D., & Kumar, V. (2012). Towards a swarm of agile micro quadrotors. In *Proceedings of robotics: Science and systems*. Sydney, Australia.

Leishman, J. G. (2000). *Principles of helicopter aerodynamics* (Chap. 10). New York: Cambridge University Press.

Lupashin, S., Schollig, A., Sherback, M., & D'Andrea, R. (2010). A simple learning strategy for high-speed quadrocopter multi-flips. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 1642–1648). Anchorage, AK.

Mellinger, D., & Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 2520–2525). Shanghai, China.

Mellinger, D., Michael, N., & Kumar, V. (2010). Trajectory generation and control for precise aggressive maneuvers. In *International symposium on experimental robotics*. Boca Raton, FL: CRC Press.

Mellinger, D., Kushleyev, A., & Kumar, V. (2012). Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In *IEEE international conference on robotics and automation* (pp. 477–483). Saint Paul, MN.

Michael, N., & Kumar, V. (2011). Control of ensembles of aerial robots. *Proceedings of the IEEE*, *99*(9), 1587–1602.

Mulgaonkar, Y. (2012). Automated recharging for persistence missions with multiple micro aerial vehicles. Master's thesis, University of Pennsylvania.

Parrish, J., & Hamner, W. (1997). *Animal groups in three dimensions*. New York: Cambridge University Press.

Pines, D., & Bohorquez, F. (2006). Challenges facing future micro air vehicle development. *AIAA Journal of Aircraft*, *43*(2), 290–305.

Powers, C., Mellinger, D., Kushleyev, A., Kothmann, B., & Kumar, V. (2012). Influence of aerodynamics and proximity effects in quadrotor flight. In *Proceedings of the international symposium on experimental robotics*. Singapore.

Schouwenaars, T., DeMoor, B., Feron, E., & How, J. (2001). Mixed integer programming for multi-vehicle path planning. In *European control conference* (pp 2603–2608). Porto, Portugal.

Schouwenaars, T., Stubbs, A., Paduano, J., & Feron, E. (2006). Multi-vehicle path planning for non-line of sight communication. In American Control Conference.

Shen, S., Michael, N., & Kumar, V. (2011). Autonomous multi-floor indoor navigation with a computationally constrained MAV. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 20–25). Shanghai, China.

Tanner, H., Jadbabaie, A., & Pappas, G. J. (2007). Flocking in fixed and switching networks. *IEEE Transactions on Automatic Control*, *52*(5), 863–868.

Valenti, M., Bethke, B., Fiore, G., How, J., & Feron, E. (2006). Indoor multi-vehicle flight testbed for fault detection, isolation, and recovery. In *AIAA guidance, navigation, and control conference and exhibit, American Institute of Aeronautics and Astronautics, Guidance, Navigation, and Control and co-located conferences*. doi:10.2514/6.2006-6200.

van den Berg, J., Guy, S. J., Lin, M., & Manocha, D. (2011). Reciprocal n-body collision avoidance. In C. Pradalier, R. Siegwart, & G. Hirzinger (Eds.), *Robotics research* (pp. 3–19). Springer.

Vicon Motion Systems, Inc. (2012). http://www.vicon.com. Accessed Oct 2012.

Wolowicz, C. H., Bowman, J. S., & Gilbert, W. P. (1979). *Similitude requirements and scaling relationships as applied to model testing*. Technical report. Washington, DC: NASA.

Alex Kushleyev received his BS degree in Electrical Engineering from the University of Pennsylvania is 2007 and his MS in Electrical Engineering from the University of Pennsylvania in 2009. He is a co-founder of KMel Robotics LLC and a researcher at the University of Pennsylvania.

Daniel Mellinger received his BS degree in mechanical engineering from North Carolina State University in 2007 and his Ph.D. from the University of Pennsylvania in 2012. His research focus is problems concering the dynamics and control of robotic systems, in particular quadrotors.

Caitlin Powers received her BS degree in mechanical engineering from Northwestern University in 2010. She is currently a PhD student at the University of Pennsylvania.

Vijay Kumar received his PhD degree in mechanical engineering from Ohio State University in 1987. He is currently the UPS foundation professor and the deputy dean in the School of Engineering and Applied Science at the University of Pennsylvania, focusing on mechanical engineering and applied mechanics as well as computer and information science. He is a Fellow of the IEEE and American Society of Mechanical Engineers.