




A Distributed Control Approach to Formation Balancing and Maneuvering of Multiple Multirotor UAVs

Yuyi Liu , Jan Maximilian Montenbruck, Daniel Zelazo , Marcin Odelga, Sujit Rajappa, Heinrich H. Bühlhoff, Frank Allgöwer , and Andreas Zell

Abstract—In this paper, we propose and experimentally verify a distributed formation control algorithm for a group of multirotor unmanned aerial vehicles (UAVs). The algorithm brings the whole group of UAVs simultaneously to a prescribed submanifold that determines the formation shape in an asymptotically stable fashion in two- and three-dimensional environments. The complete distributed control framework is implemented with the combination of a fast model predictive control method executed at 50 Hz on low-power computers onboard multirotor UAVs and validated via a series of hardware-in-the-loop simulations and real-robot experiments. The experiments are configured to study the control performance in various formation cases of arbitrary time-varying (e.g., expanding, shrinking, or moving) shapes. In the actual experiments, up to four multirotors have been implemented to form arbitrary triangular, rectangular, and circular shapes drawn by the operator via a human–robot interaction device. We also carry out hardware-in-the-loop simulations using up to six onboard computers to achieve spherical formations and a formation moving through obstacles.

Index Terms—Aerial robotics, distributed formation control, human–swarm interaction, multiagent systems.

I. INTRODUCTION

MULTIAGENT systems have become one of the most active research topics within the robot control community. A fundamental concern in multiagent systems is the *formation control* problem. In this problem, a team of agents is tasked with arranging a prespecified spatial configuration [1]. Often, formations are specified by certain relative state information that can be sensed between agents; these include position-based strategies [2], distance-based strategies [3], [4], and bearing-based strategies [5], [6]. The sensing capabilities of the vehicles will dictate which formation control strategy is most appropriate. More recently, a new approach to formation control was proposed in [7], where the formation is specified by prescribing a *shape* (i.e., a circle, triangle, rectangle, etc.). Each agent then implements a decentralized controller that asymptotically stabilizes the agents to the desired shape while, simultaneously, a distributed controller balances their configuration on that shape.

As the theory of formation control has developed, so have the practical implementation of these strategies. Teams of unmanned aerial vehicles (UAVs), for example, have the potential to perform versatile tasks, such as aerial transportation [8] and building constructions [9]. They are also popularly utilized for entertainment via swarming above audience [10], [11].

Much research on formation control of UAVs has been carried out in both aeronautics and robotics communities. A collision-free control method based on the modified Grossberg neural network for a group of UAVs in square formation has been proposed in [12]. In [13], an integrated optimal formation method has been presented, which employs an inverse optimal control approach to achieve the formation of multiple UAVs with obstacle avoidance. A nonlinear model predictive control (MPC) method incorporating obstacle-avoiding conditions using Karush–Kuhn–Tucker conditions has been proposed in [14] for formation flight. In [15], a decentralized formation control algorithm has been presented and tested on a group of quadrotors, which enables the robots to safely change the formation shape by following a specified group trajectory. A bearing formation control method based on relative angles has been proposed and tested on multirotor UAVs in real experiments in [16]. A formation controller based on the virtual rigid body in SE(3) has been proposed and experimentally validated in [17]. The latter method allows the quadrotors to simultaneously execute collision-free agile

Manuscript received May 31, 2017; revised February 14, 2018; accepted April 23, 2018. Date of publication August 8, 2018; date of current version August 15, 2018. This paper was recommended for publication by Associate Editors S.-J. Chung and E. Shaojie Shen upon evaluation of the reviewers' comments. (Corresponding author: Yuyi Liu.)

Y. Liu is with the Chair of Cognitive Systems, Department of Computer Science, University of Tübingen, Tübingen 72074, Germany, and also with the Autonomous Robotics and Human-Machine Systems Group, Max Planck Institute for Biological Cybernetics, Tübingen 72076, Germany (e-mail: yuyi.liu@uni-tuebingen.de).

J. M. Montenbruck and F. Allgöwer are with the Institute for Systems Theory and Automatic Control, University of Stuttgart, Stuttgart 70174, Germany (e-mail: montenbruck@ist.uni-stuttgart.de; frank.allgower@ist.uni-stuttgart.de).

D. Zelazo is with the Faculty of Aerospace Engineering, Israel Institute of Technology, Haifa 3200003, Israel (e-mail: dzelazo@technion.ac.il).

S. Rajappa and A. Zell are with the Chair of Cognitive Systems, Department of Computer Science, University of Tübingen, Tübingen 72074, Germany (e-mail: sujitraj369@gmail.com; andreas.zell@uni-tuebingen.de).

M. Odelga and H. H. Bühlhoff are with the Autonomous Robotics and Human-Machine Systems Group, Max Planck Institute for Biological Cybernetics, Tübingen 72076, Germany (e-mail: m.odelga@gmail.com; heinrich.buehlhoff@tuebingen.mpg.de).

This paper has supplementary downloadable material available at <http://ieeexplore.ieee.org>.

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

Digital Object Identifier 10.1109/TRO.2018.2853606

maneuvers as a group. In [18], the formation control with static and dynamic obstacles was considered as a constrained optimization solved via sequential convex programming. A partially distributed vision-based formation control method was proposed and validated in [19]. This method computes the motion of a group of UAVs based on the view overlaps of multiple onboard cameras. A nonsmooth artificial potential field was used for formation control in [20]. A vision-based leader–follower method of robots without communication was proposed in [21], where the camera on the follower was controlled to observe the leader to guarantee the formation maintenance. In [22], a consensus-based approach was proposed for the predefined time-varying formations and was outdoor experimentally validated on a group of quadrotors.

In this paper, we propose a distributed and decentralized formation control algorithm based on our earlier results in [7], which brings a group of systems to a balanced configuration on a specified shape in a stable fashion. The agents in the group exchange their position information only with their neighbors specified by a static and given information exchange network, and eventually arrange their positions on the specified shape.

Apart from the advantage of distributed fashion, the property of collision avoidance, and the capability of trajectory following, which have been included in some other formation approaches, the key novelties of our proposed approach are as follows.

- 1) This method is driven by a complete target shape, which is different from many existing formation approaches that lead the agents to the (absolute or relative) target positions of the target polygon vertices. More specifically, we define a desired formation by simply prescribing a shape in either \mathbb{R}^2 , e.g., a circle, triangle, or rectangle, or a shape in \mathbb{R}^3 , e.g., a sphere, etc., plus the center and size of the desired formation. This is suitable for certain practical applications such as target capture, since we can simply define a moving target (not necessary an agent in the group of robots) as the center of a target shape.
- 2) The proposed method enables the agents to simultaneously reach to a balanced and converged condition. This allows the agents to start from most of the initial conditions, e.g., a line array, etc. The strong convergence property also ensures the group of agents to switch among various target formation shapes smoothly.

The main contribution of this paper is an integration of the retraction balancing control with an onboard MPC-based control framework on a group of multirotor UAVs. By further combining a disturbance estimation method and a human–robot interaction way through the finger tracking via a virtual reality device, this integration enables an operator to steer a group of multirotor UAVs simultaneously toward a desired formation by drawing a shape configuration. The desired formation can be time varying (e.g., expanding, shrinking, or moving) in 2-D and 3-D environments. The performance of the complete control framework for a group of robots has been validated via both a series of hardware-in-the-loop (HIL) simulations and real-world indoor experiments. In order to make simulations more practical and convincing, all control parameters are fit identical in both sets of experiments. In general, this paper is a nontrivial

application and extension of the fundamentally theoretical result of a distributed and decentralized formation control method.

The outline of this paper is as follows. In Section II, we introduce the formation control algorithm that enables a group of agents to simultaneously reach balanced retractions. More specifically, we describe how to implement the distributed algorithm on multiple agents through case studies with various shapes. We then present a practical application of our proposed algorithm onto a group of multirotor UAVs in Section III, where the complete control system of each multirotor UAV based on an MPC method is demonstrated in detail. Section IV introduces the experimental validation of the complete distributed control framework and discussions on both HIL simulation and real-world experimental stages. Section V concludes the paper.

II. FORMATION CONTROL AND THE BALANCING PROBLEM

We begin this section with a brief overview of the retraction balancing problem, originally proposed in [7]. We then generalize the results from [7] to include target shapes described by arbitrary convex polygons (e.g., triangles, rectangles, etc.). Finally, we present another extension to this paper, whereby we consider coordinated maneuvering strategies for the formation of a multiagent system, i.e., the desired formation shape and position can be time varying.

We consider n dynamical systems with positions x_i in \mathbb{R}^2 (or \mathbb{R}^3). Our goal throughout the paper is to eventually bring these positions x_i toward an evenly spaced, “balanced” (equidistant) configuration on a given shape $M \subset \mathbb{R}^2$ (or $M \subset \mathbb{R}^3$). Thereby, we assume that M is a smoothly embedded submanifold. In particular, M could be a circle, leading to a circular formation, a triangle (with smoothed out corners), leading to a triangular formation, or a portion of a line, leading to a collinear formation, etc.

We hereby introduce the scalar field

$$\phi(x) := \sum_{j>i} W_{ij} \ln(d(x_i, x_j)) \quad (1)$$

whereby x denotes the tuple of all positions x_i and $d(x_i, x_j)$ is the length of the shortest curve on M joining x_i with x_j , in resemblance to the potential whose maximizers are the so-called Fekete points known from mathematics [23]. Therein, the weights W_{ij} are determined by a weighted connected undirected graph, with W_{ij} denoting the weight of the edge joining i and j , and the convention that $W_{ij} = 0$ whenever there is no such edge. For most of the paper, we will take our graph to be the (undirected) cycle graph with $W_{ij} = W_{ji} = 1$ for $j = (i + 1) \bmod n$ and $W_{ij} = W_{ji} = 0$ else. The scalar field (1) has the purpose of evenly spacing points on M through its gradient flow. If one, in addition, ensures that the submanifold M is asymptotically stable [24], then one will eventually attain an evenly spaced configuration on M in a stable fashion. To this end, we studied the convergence properties of the formation control algorithm

$$\dot{x} = r(x) - x + \text{grad } \phi(r(x)). \quad (2)$$

Therein, r is the smooth retraction onto M^n . Thus, the control action $r(x) - x$ asymptotically brings our positions x_i toward

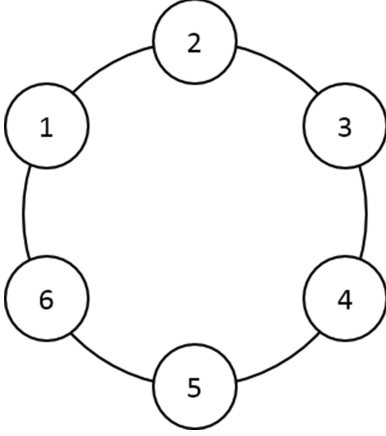


Fig. 1. Cycle graph C_6 for a group of six agents.

M . At the same time $\text{grad } \phi(r(x))$, the gradient vector field of ϕ evaluated at the retraction of x onto M^n has the purpose of evenly spacing the points x_i on M . Together, these two controls, thus, bring the points x_i toward an evenly spaced configuration on M , as desired.

The differential equation (2) has quite strong convergence properties. In particular, all solutions initialized in the preimage of any regular superlevel set of the potential ϕ under the retraction converge to the desired formation in a stable fashion.

According to the derivation in our previous work [7], $\text{grad } \phi(r(x))$ is computed via

$$\text{grad } \phi(r(x)) = \sum_{j=1}^n \frac{W_{ij}}{d(r(x_i), r(x_j))} V_{ij} \quad (3)$$

where $(x_i, x_j) \mapsto V_{ij}$ is the velocity vector of the unit speed geodesic (distance on submanifold) joining $r(x_i)$ and $r(x_j)$.

Therefore, the differential equation to regulate the motion of each agent in the group is given by

$$\dot{x}_i = r(x_i) - x_i + \sum_{j=1}^n \frac{W_{ij}}{d(r(x_i), r(x_j))} V_{ij} \quad (4)$$

In the following sections, we introduce two examples of prescribed shapes in 2-D environment as well as an example of a formation shape in 3-D environment.

A. \mathbb{R}^2 Formation: The Circle

In 2-D formation cases, the agents in the group communicate through the unweighted cycle graph (e.g., an example of six agents is shown in Fig. 1). Let the formation M be an arbitrary circle embedded in \mathbb{R}^2 , centered at c with the radius r_c , i.e.,

$$M = \{x_i \in \mathbb{R}^2 \mid \|x_i - c\| = r_c\}. \quad (5)$$

The (smooth) retraction of a certain agent x_i from the tubular neighborhood of the circle, thus, is equal to the length r_c vector

$$r(x_i) = \frac{r_c}{\|x_i - c\|} (x_i - c) + c. \quad (6)$$

Algorithm 1: Compute Retraction Point $r(x_i)$ on Polygon.

- 1) Locate an edge on a polygon with n vertices, which is closest to the position x_i ;
 - 2) Calculate the shortest distance, between x_i and any arbitrary point on the closest edge found;
 - 3) Set the point with the shortest distance to x_i as the retraction $r(x_i)$.
-

For arbitrary two retractions $r(x_i)$ and $r(x_j)$ on the circular formation, employing the canonical (Lie) group isomorphism

$$\begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix} \mapsto \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \quad (7)$$

between the circle and the rotation group $\text{SO}(2)$, the scaled velocity vector of the geodesic joining $r(x_i)$ and $r(x_j)$ can be computed by

$$\begin{aligned} & -r_c d(x_i, x_j) V_{ij} \\ &= \log \left(\frac{1}{\|x_i - c\| \|x_j - c\|} \begin{bmatrix} x_i \cdot x_j & x_i \cdot \Omega x_j \\ x_j \cdot \Omega x_i & x_i \cdot x_j \end{bmatrix} \right) \frac{(x_i - c)}{\|x_i - c\|} \end{aligned} \quad (8)$$

where Ω denotes the infinitesimal rotation $\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$.

Therefore, to form a prescribed circle shape, the control (4) is finally given by

$$\begin{aligned} \dot{x}_i &= \left(\frac{r_c - \|x_i - c\|}{\|x_i - c\|} \right) (x_i - c) + \sum_{j=1}^n \frac{W_{ij}}{r_c \|x_i - c\|} \cdot \left(\log \right. \\ & \left. \left(\frac{1}{\|x_i - c\| \|x_j - c\|} \begin{bmatrix} x_i \cdot x_j & x_i \cdot \Omega x_j \\ x_j \cdot \Omega x_i & x_i \cdot x_j \end{bmatrix} \right) \right)^{-1} (x_i - c). \end{aligned} \quad (9)$$

The computed \dot{x}_i is employed as the terminal velocity reference in the MPC method that controls the multirotor UAV (we will introduce the details in latter sections) at each time step.

B. \mathbb{R}^2 Formation: The Convex Polygon

Apart from the arbitrary circle shape case from the foregoing section, our algorithm fits for rather arbitrary polygonal formations, e.g., triangle, rectangle, pentagon, etc. Unlike a circular shape whose retraction can be represented explicitly, we compute the retraction point $r(x_i)$ on a target polygonal shape by the following Algorithm 1.

Considering that the geodesic between two agents $d(x_i, x_j)$ and their relative velocity vectors V_{ij} depends on their retractions, namely $r(x_i)$ and $r(x_j)$, then $d(x_i, x_j)$ and V_{ij} can be computed via the general formulation displayed in Algorithm 2.

Finally, the differential equation of motion can be computed by substituting the calculated $d(x_i, x_j)$ and V_{ij} into (4). The output of (4), namely the computed velocity vector \dot{x}_i of an agent in the group, will be employed as the terminal velocity reference in an MPC method to control the multirotor UAVs.

Algorithm 2: Compute $d(x_i, x_j)$ and V_{ij} on a Polygon.

For a target polygon with n vertices ($n \geq 3$)
if $r(x_i)$ and $r(x_j)$ are located on the same edge of polygon
then
 $d(x_i, x_j) = \|r(x_j) - r(x_i)\|$
 $V_{ij} = -\frac{r(x_j) - r(x_i)}{\|r(x_j) - r(x_i)\|}$
else
if $r(x_i)$ and $r(x_j)$ are located on two neighboring edges
that share a vertex V_s **then**
 $d(x_i, x_j) = \|r(x_j) - V_s\| + \|V_s - r(x_i)\|$
 $V_{ij} = -\frac{V_s - r(x_i)}{\|V_s - r(x_i)\|}$
end if
else
For $n \geq 4$
if $r(x_i)$ and $r(x_j)$ are located on two edges with no
shared vertex but linked by the edge whose vertices are
 $V_{l,i}$ and $V_{l,j}$ **then**
 $d(x_i, x_j) = \|r(x_j) - V_{l,j}\| + \|V_{l,j} - V_{l,i}\|$
 $+ \|V_{l,i} - r(x_i)\|$
 $V_{ij} = -\frac{V_{l,i} - r(x_i)}{\|V_{l,i} - r(x_i)\|}$
end if
else
For $n \geq 5$
if $r(x_i)$ and $r(x_j)$ are located on two edges with no
shared vertex, and the bridge edges among these two
edges are with the vertices $V_{l,i}, V_{l,j}$ and
 $V_{l,k_1}, V_{l,k_2}, \dots, V_{l,k_n}$, where $(1 \leq k_n \leq n-4)$ **then**
 $d(x_i, x_j) = \|r(x_j) - V_{l,j}\| + \|V_{l,j} - V_{l,k_n}\|$
 $+ \|V_{l,k_n} - V_{l,k_n-1}\| + \dots + \|V_{l,k_2} - V_{l,k_1}\|$
 $+ \|V_{l,k_1} - V_{l,i}\| + \|V_{l,i} - r(x_i)\|$
 $V_{ij} = -\frac{V_{l,i} - r(x_i)}{\|V_{l,i} - r(x_i)\|}$
end if
end if

C. \mathbb{R}^3 Formation: The Sphere

Our distributed control algorithm is not only restricted to formations in \mathbb{R}^2 . A group of multirotor UAVs is, e.g., able to form a spherical shape centered at arbitrary points in \mathbb{R}^3 from arbitrary initial positions. The communication among the group of agents in \mathbb{R}^3 is based on the complete graph.

Most considerations from the circular formation in \mathbb{R}^2 remain correct. Similarly as in (7), we employ the (Lie) group isomorphism

$$\begin{bmatrix} \sin(\beta) \cos(\alpha) \\ \sin(\beta) \sin(\alpha) \\ \cos(\beta) \end{bmatrix} \mapsto \begin{bmatrix} \cos(\beta) \cos(\alpha) & -\sin(\alpha) & \sin(\beta) \cos(\alpha) \\ \cos(\beta) \sin(\alpha) & \cos(\alpha) & \sin(\beta) \sin(\alpha) \\ -\sin(\beta) & 0 & \cos(\alpha) \end{bmatrix} \quad (10)$$

from the sphere onto $\text{SO}(3)$. We denote the representation (10) of a certain retracted x_i as a member of $\text{SO}(3)$ by R_i . The retrac-

tion of a certain agent $x_i \in \mathbb{R}^3$ from the tubular neighborhood of the sphere, whose center locates at an arbitrary point c in \mathbb{R}^3 with the radius r_c , is still given by

$$r(x_i) = \frac{r_c}{\|x_i - c\|} (x_i - c) + c. \quad (11)$$

Thus, we can apply the logarithmic map $\log : \text{SO}(3) \rightarrow \mathfrak{so}(3)$ to $R_j^\top R_i$, in order to find the velocity vector of the geodesic joining R_i and R_j . Since in $\text{SO}(3)$, the tangent space is no longer 1D, we use the identity $2d(x_i, x_j)^2 = -\text{tr}(\log(R_j^\top R_i)^2)$ and can finally obtain the differential equation of the velocity vector subject to an arbitrary sphere in \mathbb{R}^3

$$\begin{aligned} \dot{x}_i = & \left(\frac{r_c - \|x_i - c\|}{\|x_i - c\|} \right) (x_i - c) \\ & + \sum_{j=1}^n \frac{2W_{ij}}{r_c \|x_i - c\| \|\text{tr}(\log(R_j^\top R_i)^2)\|} \log(R_j^\top R_i) (x_i - c). \end{aligned} \quad (12)$$

Similarly, as in all formation cases in \mathbb{R}^2 , we employ the velocity vector computed from (12) as the 3-D terminal velocity reference in the MPC-based controller for the multirotor UAVs.

III. MPC OF MULTIROTORS

For the onboard implementation of the formation control algorithm onto a multirotor UAV platform, we propose a framework based on an MPC method to control the position and attitude of the multirotors.

Following the formation approach we proposed in the last section, we develop a distributed framework as shown in the right part of Fig. 2. For the purpose of robustness and fast onboard implementation, a linear MPC method plus a nonlinear geometric approach are used hierarchically to regulate the translational motion and rotational motion, respectively. A nonlinear wrench observer is further implemented to estimate the unknown disturbances as external force and torque terms, and pass the estimates to the controller for compensation. The overall control system is optimized to fit the real-time computations into low-power onboard computers. We will show the experimental verifications later in Section IV.

A. Dynamic Model

We consider a multirotor UAV as a rigid body [25], the dynamic equations of which are given by

$$\dot{x} = v \quad (13a)$$

$$m\dot{v} = -mge_3 + RF + b_F \quad (13b)$$

$$\dot{R} = RQ(\omega) \quad (13c)$$

$$J\dot{\omega} = -\omega \times J\omega + \tau + b_\tau \quad (13d)$$

where m is the mass of a multirotor, x denotes the translational position, g is the scalar value of the gravitational acceleration, and $e_3 = (0, 0, 1)^\top$. The nonconservative forces and moments in the body frame generated by the rotor propellers on the mul-

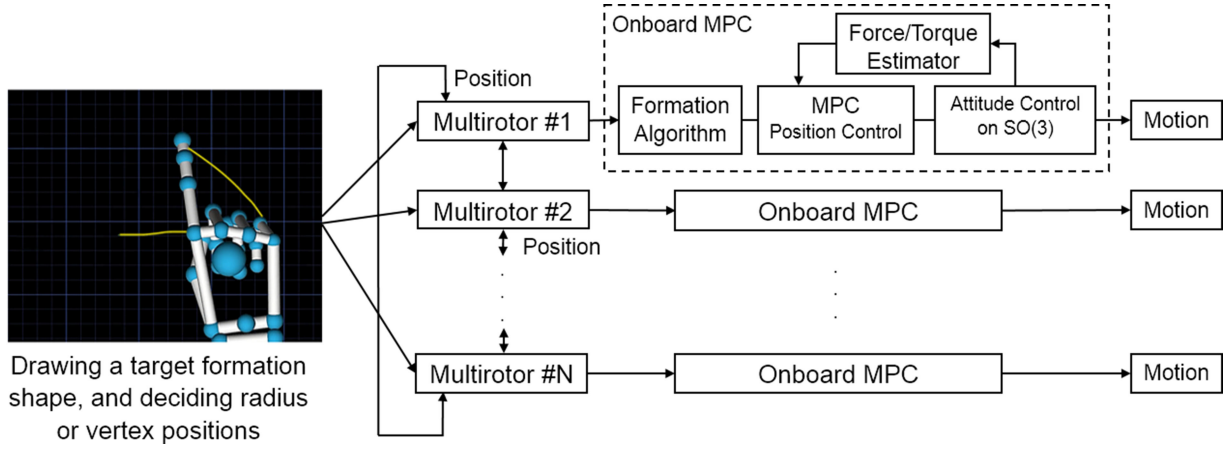


Fig. 2. Block diagram of a distributed predictive control framework for the balancing formation of a group of multirotors.

tirotors are represented by $F \in \mathbb{R}^3$ and $\tau = (\tau_x, \tau_y, \tau_z)^\top$, while b_F , considered in the form of an external force in the inertial frame, denotes the additional forces due to external disturbances and unmodeled dynamics, and b_τ is considered as an external torque in the body frame. The inertia matrix J in the body frame is computed from a CAD model of the multirotor UAV. The skew-symmetric matrix form of the angular velocity $\omega = (\omega_1, \omega_2, \omega_3)^\top$ in the body frame cross product $Q(\omega)$ is given as

$$Q(\omega) = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}. \quad (14)$$

The rotation matrix R , which is an element of the special orthogonal group $SO(3) = \{R \in \mathbb{R}^{3 \times 3} | R^{-1} = R^\top, \det R = 1\}$, represents the attitude of the quadrotor with respect to the inertial frame.

We assume that the direction of the thrust T on a multirotor with n propellers is along the z -axis of the body frame, the force vector F has only its third entry nonzero that equals the total thrust of the n rotors, which is given by $F = (0, 0, T)^\top$ for non-negative $T \triangleq \sum_{i=1}^n f_i$, where the thrust of a single propeller is modeled as $f_i \triangleq c_T \omega_{i,i}^2, \forall i = 1, \dots, n$. The constant c_T is the thrust coefficient determined by static thrust tests. The final mapping from the rotor speeds into the forces and moments on a multirotor UAV, e.g., a quadrotor or a hexarotor, varies depending on the number of rotors. The desired angular velocity of each motor, without considering the error from the motor controller, can be computed via solving the inverse of the mapping.

B. Position Control

We hereby introduce a linear model predictive controller for the position control of a multirotor UAV. The discrete-time model of the translational motion of a multirotor, under no disturbance assumption, sets the acceleration in 3-D as the inputs for an optimal control problem (OCP) describing the multirotor motion. The state $\bar{\xi}$ consists of the position and velocity along three axes, namely $(x, \dot{x}, y, \dot{y}, z, \dot{z})^\top$.

In summary, the discrete-time state $\bar{\xi}$ and control input \bar{u} at the k th time step with a sampling time Δt are given by

$$\bar{\xi}[k] = \begin{bmatrix} x(k\Delta t) \\ \dot{x}(k\Delta t) \\ y(k\Delta t) \\ \dot{y}(k\Delta t) \\ z(k\Delta t) \\ \dot{z}(k\Delta t) \end{bmatrix}, \bar{u}[k] = \begin{bmatrix} \ddot{x}(k\Delta t) \\ \ddot{y}(k\Delta t) \\ \ddot{z}(k\Delta t) \end{bmatrix} \quad (15a)$$

$$\bar{\xi}[k+1] = A\bar{\xi}[k] + B(\bar{u}[k] - g_{e3}) \quad (15b)$$

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (15c)$$

$$B = \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 & 0 \\ \Delta t & 0 & 0 \\ 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & \Delta t \end{bmatrix}. \quad (15d)$$

The OCP can be described with a quadratic cost function

$$\min_{\bar{\xi}, \bar{u}} J = \sum_{k=0}^N (\bar{u}[k]^\top P \bar{u}[k] + d_{\bar{\xi}}[k]^\top L_s d_{\bar{\xi}}[k]) + d_{\bar{\xi}}[N+1]^\top L_t d_{\bar{\xi}}[N+1] \quad (16)$$

subject to the dynamics explained above and corresponding boundary conditions

$$a_{\min} \leq \bar{u}[k] \leq a_{\max} \quad (17a)$$

$$\xi_{\min} \leq \bar{\xi}[k] \leq \xi_{\max} \quad (17b)$$

where N represents the length of the receding horizon, $d_{\bar{\xi}}[k]$ denotes the error between the state and reference, and P , L_s , and L_t are the weights of the input cost, stage state cost, and terminal cost, respectively. The described convex OCP is solved

via CVXGen [26] using the interior point method. The vectors $a_{\min}, a_{\max} \in \mathbb{R}^3$ represent the lower and upper constraints of multirotor accelerations along three axes. Similarly, $\xi_{\min}, \xi_{\max} \in \mathbb{R}^6$ denote the position and velocity boundaries. The computed first step control input $\bar{u}[0]$ is used in the control law at each time step.

The OCP is designed with adaptive cost weight settings, so that the controller can predict a locally optimal trajectory and generate a 3-D vector as the control input for a given path or simply a waypoint. For a waypoint navigation task, the weight of the terminal cost is automatically set dominant, while the weight of all stage state costs is set to zero. In contrast, for a path following task, the weight of the terminal cost is set to zero.

For the formation tasks of multiple multirotor UAVs, we keep the weight of the stage cost at zero. Different from a waypoint task, we consider the output of (4) \dot{x} as the terminal velocity reference in the OCP. Therefore, the translational control based on MPC of a multirotor becomes a “velocity control” method instead of a “position control” method.

Assuming that R in (13b) equals R_{ref} , the control law for the multirotor position is given by

$$R_{\text{ref}} F \triangleq m \bar{u}[0] := F'. \quad (18)$$

Considering that R_{ref} in (18) is orthonormal and only the third entry of F , i.e., the thrust T , is nonzero, one must choose $T = \|F'\|$ in order to suffice (18). Therefore, the third column of $R_{\text{ref}} = [r_{\text{ref}}^1 \ r_{\text{ref}}^2 \ r_{\text{ref}}^3]$ can be solved from

$$r_{\text{ref}}^3 = \frac{F'}{\|F'\|} \quad (19)$$

and the other two columns of R_{ref} can be filled orthonormally, for instance by a Gram–Schmidt process with candidates r_{ref}^3 from the previous equation and r^1, r^2 from the actual rotation $R = [r^1 \ r^2 \ r^3]$, i.e.,

$$r_{\text{ref}}^{2'} = r^2 - \frac{r^2 \cdot r_{\text{ref}}^3}{r_{\text{ref}}^3 \cdot r_{\text{ref}}^3} r_{\text{ref}}^3 \quad (20a)$$

$$r_{\text{ref}}^{2'} = \frac{r_{\text{ref}}^{2'}}{\|r_{\text{ref}}^{2'}\|} \quad (20b)$$

$$r_{\text{ref}}^{1'} = r^1 - \frac{r^1 \cdot r_{\text{ref}}^3}{r_{\text{ref}}^3 \cdot r_{\text{ref}}^3} r_{\text{ref}}^3 - \frac{r^1 \cdot r_{\text{ref}}^{2'}}{r_{\text{ref}}^{2'} \cdot r_{\text{ref}}^{2'}} r_{\text{ref}}^{2'} \quad (20c)$$

$$r_{\text{ref}}^{1'} = \frac{r_{\text{ref}}^{1'}}{\|r_{\text{ref}}^{1'}\|} \quad (20d)$$

such that we obtain both the desired total thrust T and the desired attitude R_{ref} for the attitude control.

By building a constrained MPC method, we are able to directly employ the resulting velocity vectors computed from the formation algorithm, or waypoint if the robot is executing the navigation task. There is no further requirement of tuning weight parameter or additional building of a local trajectory planner to avoid overshooting.

C. Attitude Control

In this section, we introduce a backstepping-like control method for the tracking of the attitude R of a multirotor with strong convergence properties. This controller was first introduced in [27]. The specific approach is based on the solution of a class of output regulation problems, which contains the rotational motion for a rigid body. The tracking error is given by $E = R R_{\text{ref}}^\top$, where R corresponds to the rotation matrix describing the current attitude, and R_{ref} , computed via the approach introduced in Section III-B, represents the desired attitude. The concept is similar with the geometric tracking method proposed in [28], but with different selection of the cost function. The error between the current and a desired angular velocity in body frame are

$$e_\omega = Q(\omega) - Q(\omega_d). \quad (21)$$

A Lyapunov function candidate

$$V_{\text{att}}(E, e_\omega) = \frac{1}{2} Q^{-1}(e_\omega)^\top Q^{-1}(e_\omega) + P_2(E) \quad (22)$$

is built. While a cost function $P_2(E) = \frac{1}{2} \text{tr}((E - I)^\top (E - I)) = n - \text{tr}(E)$, for which

$$\text{grad } P_2(E) = \frac{1}{2} (E - E^\top) E \quad (23)$$

is chosen, where $\text{grad } P_2$ is the projection of ∇P_2 onto tangent spaces of $\text{SO}(3)$. In this case, we choose $P_2(E)$ in the form of

$$P_2(E) = 3 - \text{tr}(E). \quad (24)$$

For the sake of brevity, the full convergence proof, which can be found in our previous work [29], is not reported in this paper.

The control law for the multirotor UAV's attitude is given by

$$\tau_d = J Q^{-1} \left(-k_\omega e_\omega - \frac{k_{\text{rot}}}{2} \left[\dot{R}_{\text{ref}}^\top R + R_{\text{ref}}^\top \dot{R} - \dot{R}^\top R_{\text{ref}} - R^\top \dot{R}_{\text{ref}} \right] - \dot{R}_{\text{ref}}^\top R_{\text{ref}} - \dot{R}_{\text{ref}}^\top \dot{R}_{\text{ref}} - 2 R_{\text{ref}}^\top \dot{R} \right) + \omega \times J \omega \quad (25)$$

where k_ω is a positive gain for the tracking error of the multirotor angular velocity. The derivative \dot{R} can be obtained by introducing the current attitude R into (13c), while R_{ref} inherits from the results of (19) and (20). We choose the value $\omega_{1,\text{ref}}, \omega_{2,\text{ref}} = 0$ and $\omega_{3,\text{ref}} = \psi_{\text{ref}} - \psi_{\text{current}}$ for the reference and current yaw attitude ψ , and \dot{R}_{ref} , hence, can be solved via (13c).

D. Disturbance Estimation

In this section, we extend the control approach discussed in the previous sections with the inclusion of a nonlinear force/torque external wrench observer using the fault detection and isolation (FDI) method [30]. The observer estimates all the system offsets, parameter uncertainties, and external disturbances, and passes the estimates to the controller for compensation. The detailed introduction and the experimental validation on a multirotor UAV can be found in our previous work [29].

Denoting the external forces acting on the multirotor in the inertial frame with $b_F \in \mathbb{R}^3$ and the external torque acting on the multirotor in the body frame with $b_\tau \in \mathbb{R}^3$, the external

disturbance Λ_{ext} can be defined as

$$\Lambda_{\text{ext}} = \begin{bmatrix} b_F \\ b_\tau \end{bmatrix}. \quad (26)$$

The external disturbance may be variable (e.g., wind gusts) or permanent (e.g., due to uncertainty in parameters). Therefore, the FDI technique needs to generate an asymptotically stable residual vector signal, to make sure that disturbance recovery is occurring and that each scalar value of the residual is decoupled from each other. The FDI design is based on the simple but powerful idea of the generalized momenta $Q_w = M\zeta$. In fact, one can write the following first-order dynamic equation for the momentum as

$$\dot{Q}_w = \Lambda + \Lambda_{\text{ext}} + C^\top(\zeta)\zeta - G \quad (27)$$

where $\Lambda = [RF^\top \ \tau^\top]^\top \in \mathbb{R}^6$, $G = [0 \ 0 \ mg \ 0 \ 0 \ 0]^\top$ and the coriolis term $C^\top(\zeta)$ involves the partial differentiation of the mass matrix M w.r.t. to the system state

$$\zeta \triangleq [\dot{x} \ \dot{y} \ \dot{z} \ \omega_1 \ \omega_2 \ \omega_3]^\top. \quad (28)$$

We define the residual vector $\mathbf{r} \in \mathbb{R}^6$ for the disturbance estimation of the multirotor as

$$\mathbf{r}(t) = K_I \left(Q_w - \int_0^t (\Lambda + C^\top(\zeta)\zeta - G + \mathbf{r}) ds \right) \quad (29)$$

where $Q_w = M\zeta$ is the momentum of the multirotor and $K_I > 0$ is a diagonal gain matrix. The evolution of the residual \mathbf{r} satisfies

$$\dot{\mathbf{r}} = K_I (\Lambda_{\text{ext}} - \mathbf{r}), \quad \text{when } \mathbf{r}(0) = 0 \quad (30)$$

which has an exponentially stable equilibrium at $\mathbf{r} = \Lambda_{\text{ext}}$. For sufficiently large gains, the residual in (30) becomes

$$\mathbf{r} \simeq \Lambda_{\text{ext}}. \quad (31)$$

This approach provides a model-based estimate of the external disturbances Λ_{ext} resulting from the forces/torques acting on the multirotor. If a particular component of the external disturbance is zero, then the scalar residual value corresponding to that component converges to zero. Hence, we can write the estimated external wrench as

$$\widehat{\Lambda}_{\text{ext}} = \begin{bmatrix} \widehat{b}_F \\ \widehat{b}_\tau \end{bmatrix} = \mathbf{r} \quad (32)$$

where $\widehat{\cdot}$ indicates the estimated value of the variable.

With the inclusion of the external wrench estimate, the position control law in (18) is given by

$$RF \triangleq -\widehat{b}_F + F' = -\widehat{b}_F + m\bar{u}[0] \quad (33)$$

while the attitude control law in (25) becomes

$$\begin{aligned} \tau = & -\widehat{b}_\tau + JQ^{-1} \left(-K_\omega e_\omega - \frac{K_{\text{rot}}}{2} \left[\dot{R}_{\text{ref}}^\top R \right. \right. \\ & + R_{\text{ref}}^\top \dot{R} - \dot{R}^\top R_{\text{ref}} - R^\top \dot{R}_{\text{ref}} \left. \right] - \ddot{R}_{\text{ref}}^\top R_{\text{ref}} \\ & \left. - \dot{R}_{\text{ref}}^\top \dot{R}_{\text{ref}} - 2R_{\text{ref}}^\top \dot{R} \right) + \omega \times J\omega. \end{aligned} \quad (34)$$

The equations (33) and (34), plus the final mapping from desired thrust/torque to motor speeds, provide the multirotors with a robust MPC method for high-speed waypoint navigation and formation tasks.

Although it is not straightforward to prove the asymptotic stability of the MPC-observation frame, the proposed framework experimentally performs well (we will show it later) in the case that unknown disturbances exist due to model uncertainties, open-loop fashion of hardware, and external wind, etc.

Our flight control approach is “robust” with respect to “accuracy,” which is highly required by the formation algorithm since it updates online based on the position of the robots and their neighbors. High accuracy of flight control warrants the convergence of a whole group of robots to a large extent.

IV. VERIFICATION

We carried out two stages of experiments, with a connection of human–swarm interaction via the finger tracking device, to validate the performance of the proposed distributed control approach, including simulations and real-world experiments.

At the simulation stage, we validated the presented formation algorithm combined with our proposed MPC method via ROS/Gazebo physical simulations of multiple (up to 6) multirotor UAV models with artificial parametric mismatch and disturbances. Since the computational cost of MPC is quite critical for the multirotor UAVs with limited computational capability in actual experiments, we have set up HIL simulations using the low-power onboard computational units, LiPo battery for power, and WiFi communication, which ensure that our proposed control approach can be executed properly at 50 Hz within real-world experiments.

At the real-world experiment stage, we implemented the complete formation control framework onto up to four quadrotor UAVs with onboard computational units (same as those used in HIL simulations) and carried out a series of experiments on formation of various shapes.

We tested our proposed approach via circular, triangular, and rectangular formations in a 2-D environment, as well as via a spherical formation in a 3-D environment. In this section, we will introduce the detailed configuration of our experimental and HIL simulation platform, and then demonstrate the results for various shapes, respectively. For the sake of brevity, we hereby mainly report the results from real-world experiments if we have tested one formation in both experimental stages. We additionally report several selected results of the HIL simulation, where more than four multirotor UAVs are required.

The reader may refer to the supplementary multimedia material of this paper for the highlights of the experimental validation.

A. Human–Swarm Interaction

Since our proposed formation algorithm relies on a prescribed shape instead of relative distance of targets, it enables a human operator to decide a target formation by simply “drawing” a geometrical shape. To realize the interaction between human and multiple aerial robots, a Leap Motion 3-D gesture device

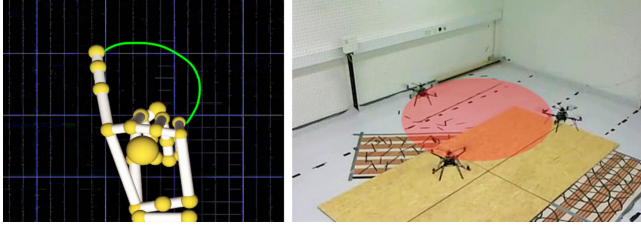


Fig. 3. Screenshot from the progress of prescribing a formation shape via human–robot interaction. Left: The user draws a circular trajectory via the leap motion device. Right: The trajectory of finger motions is then fitted and matched as a circular shape and mapped as the target formation for the group of multirotor UAVs.

has been implemented into our formation control framework, so that the formation shapes for the multirotor UAVs can be generated via finger/palm motions (shown in Fig. 3). In order to command the UAVs to generate the required formation in a safe and practical way, an end-user (human) needs to draw the formation shape, e.g., circle, triangle, rectangle, sphere, etc., and also provide auxiliary information, i.e., the geometric center and the radius (when giving a circle shape) or the position of vertices (when providing a polygonal shape).

More specifically, the human–swarm interaction between the operator and a group of UAVs can be achieved by the following procedures.

- 1) The “drawing” from the operator through the figure tracking device is first collected as a set of trajectory.
- 2) The trajectory is passed into a customized curve-fitting program, where the drawing from leap motion will be filtered and described in the form of one second-order polynomial, or a cluster of several first-order polynomials.
- 3) The polynomials are paired with the geometric shapes: we naively map a second-order polynomial into a circle, a group of first-order polynomials including one with near-zero slope as triangle, and a group of four first-order polynomials including two with near-zero slopes as rectangle, etc.
- 4) The fitted shape is mapped to the experimental area with proper 3-D positions and implemented onto the group of multirotor UAVs for the target formation.

The reader may note that in the real-robot experiments, the center position of formation shape and size is prescribed in a database due to the limited experimental area (also to avoid that a drawn shape is too sharp or too far from robots), while the shape detection via LeapMotion is always online. In our case, there is only one waypoint that commands each robot to hover at its initial horizontal position and 1-m height given offline.

Once the shape drawn by the operator is detected, the formation position and size information is then passed together with the shape information to each UAV continuously. Therefore, it is possible that UAVs are commanded to switch, for example, from a polygon to a circle during the flight.

B. Experimental Configuration

Up to four quadrotor UAVs with 10 in propellers are set as the platform for a series of the real-world experiments. Each



Fig. 4. Screenshot on the real-world formation experiment. A group of four multirotor UAVs were commanded to generate a prescribed rectangular formation.

quadrotor is equipped with an Odroid-XU4 board with Cortex-A15 CPUs. Thus, all the computations are carried out onboard during the experiments. We also have carried out several HIL simulations using the Odroid boards in the same model as implemented on the quadrotors. The complete control approach we proposed in previous sections is implemented within a ROS-based framework, which continuously passes the required motor speed messages at each time step in the feedback system.

In the cases of real-world, indoor experiments (e.g., in Fig. 4), the position data of a multirotor are passed via the external motion capture system (e.g., Vicon) at 100 Hz, while the attitude data are filtered using an extended Kalman filter from the raw angular velocity data collected by a low-cost onboard inertial measurement unit (IMU). The linear acceleration data are also collected by the IMU and then filtered into linear velocity data at each time step. The motor speeds are computed and passed to the open-loop brushless motor controllers at the frequency of the proposed attitude controller and finally to the embedded motors.

In addition, each robot keeps passing their own position data to their neighbors at each time step for the usage of formation algorithm. The number of neighbors is defined beforehand based on the communication graph. All communication is within a local WiFi environment.

Meanwhile, in HIL simulations, the motor speed messages obtained from the proposed controller are passed into the built multirotor models in Gazebo. Differently from the real experiments, we employ hexarotor models in the virtual tests; thus, the desired thrust/torque computed from (33) and (34) are mapped into six motor speed values at each time step. The position data of each multirotor are generated with no noise, since in real experiments we regard the position data from the motion capture system as “ground truth.” The IMU data are generated with artificial Gaussian noises, drift, and initial bias to simulate a practical onboard IMU. In the simulations of multiple hexarotor UAVs, each UAV is able to obtain its own position, (raw) linear acceleration, and (raw) angular velocity data, as well as the position data of its neighbors at 100 Hz.

The average computational cost of our robust MPC approach is approximate 0.0149 s. Therefore, each Odroid board is set to execute the MPC position controller at 50 Hz, with a receding horizon of 15 steps. The attitude controller computes the motor speed commands at 100 Hz with the usage of collected and

filtered data. The decision on the present formation is also made at a frequency of 50 Hz.

By simply switching the channel between real-flight and simulation in ROS environment, the two kinds of experiments can be carried out without reconfiguration. The major benefit of our HIL simulation configuration is that we only tune the control parameters (within our proposed control framework) during the simulation but do not further optimize the relevant parameters before each real-robot experiments. Therefore, we guarantee that our HIL simulation is highly practical. Although the parameters that work perfectly in the virtual environment may not be the most optimal choice in the real world, we ensure that those parameters still work well during real-robot experiments.

In most cases, the HIL simulation and real-robot experiments are carried out step by step, with the control parameters on each multirotors kept identical and fixed. The hovering position error of each UAV in simulation is within a range of 2 cm, while in actual experiments the hovering position error of a quadrotor after a waypoint navigation task is within a range of 5 cm. However, if the delay of WiFi communication is large, i.e., over 5 ms, the hovering error would increase by about 1–2 cm. In addition, we project the target centers fixed to the center of the experimental area during the actual formation experiments.

In the real-world experiments, the disturbances due to the hardware, the delay on transfer of control commands and the wireless communication among the UAVs, and the external disturbances, e.g., winds generated from the group of multirotors and from the ventilation system, highly affect the control performance. The resulting trajectories, therefore, would not be as perfect as the results in simulations. However, the convergence of our proposed approach has still been experimentally validated.

The linear proportional-integral-derivative (PID) approach, instead of our proposed MPC method for a position controller, has also been tested in real-robot experiments. However, since we have no advanced system identification approach employed onboard to fully optimize the control parameters for all robots during each test, significant overshooting and undershooting occurred during the tests and sometimes led to crashes. For the sake of brevity, we do not demonstrate the failures during validation. The reader who has interests in implementing the PID controller in similar systems with unknown disturbance may combine some robust PID approach, e.g., the one effectively validated in [31], with the disturbance observer proposed in this paper for further robustness.

C. Triangular Formation

In the first case, several quadrotor UAVs were commanded to take off from random initial positions and generate a 2-D triangular formation at a height of 1 m. We employed three quadrotors in the experiments for an intuitive visualization. The switch of experimental tasks on each UAV was operated together via a joystick. Once the formation task was launched, the UAVs started to decide upon their motions online and

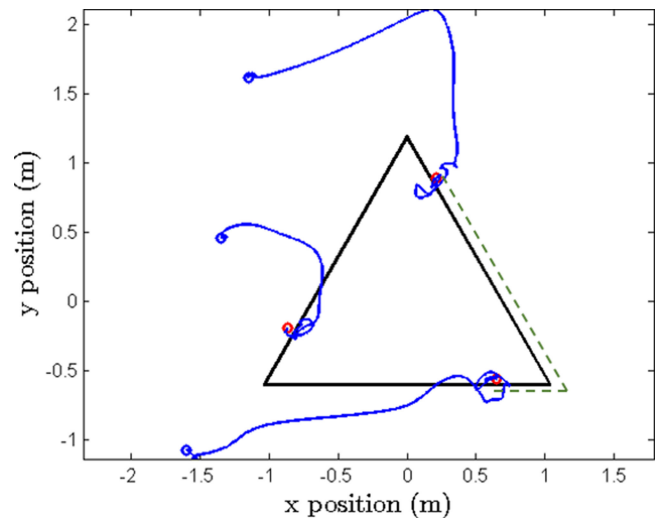


Fig. 5. Real-robot-experiment result: multirotor trajectories of a triangular formation. The assembly of solid black lines represent the prescribed triangle shape. The blue bubbles show the initial position where the multirotors hovered before the start of the formation task. The solid blue lines represent the flight trajectories of each multirotor. The red bubbles are the stable positions of UAVs in the target triangular formation. The green dashed line visualizes the geodesic between two multirotors.

subsequently moved using the distributed MPC approach that we have presented in previous sections.

The shape of the UAV formation was first decided before the formation task started. In this case, the end-user provided a triangular shape through the finger movement detected via the VR device and then defined the position of vertices. During the experiment, a triangle with vertices of $(0, 1.2, 1)^T$, $(\frac{3\sqrt{3}}{5}, -0.6, 1)^T$, and $(-\frac{3\sqrt{3}}{5}, -0.6, 1)^T$, w.r.t. the prescribed target center, was generated. Despite that our formation control algorithm is theoretically convergent for an arbitrary triangle, we set the formation shape not overly blunt or sharp so that the collisions among multirotors can be avoided.

We carried out ten trials on the triangular formation, where the initial conditions of multirotors were randomly set. The formation approach successfully converged in all the trials during the experiments. For the sake of brevity, we hereby only illustrate one trial of the triangular formation in Fig. 5. The multirotors were commanded to start the formation task from the initial condition of one line array. The UAVs converged to three equilibrium positions (the red bubbles in Fig. 5) on the edges of the prescribed triangle whose horizontal projections are the calculated retraction points on the edge of the triangle shape. The velocity reference of each multirotor passed to the MPC was computed negligible at the equilibrium position, thus each multirotor kept hovering.

In the real-world experiments, the UAVs sometimes oscillated due to the external disturbance and the loss of measurements due to the delay of wireless communication. There were also slight drifts during the hovering (within an error ranging in ± 5 cm). Largely, the three multirotors are seen to retract into the target triangle and converge to their equilibrium position. Compared to the auxiliary triangle (solid black lines), the three multirotors

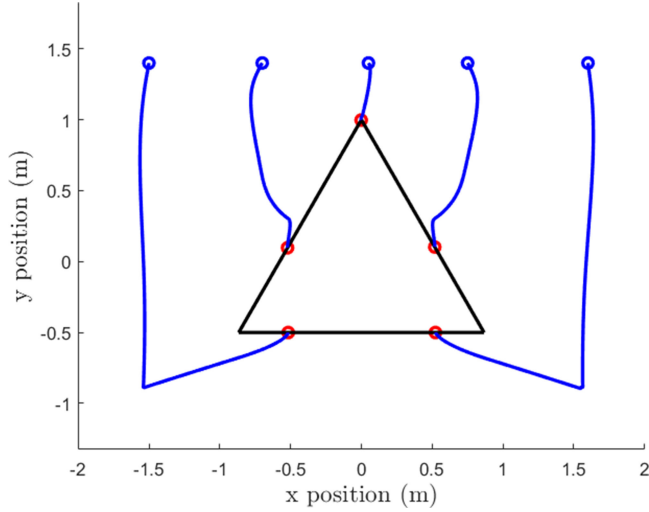


Fig. 6. Simulation result: multirotor trajectories of a triangular formation with five UAVs. The assembly of solid black lines represent the prescribed triangle shape. The blue bubbles show the initial position where the multirotors hovered before the start of the formation task. The red bubbles are the stable positions of UAVs in the target triangular formation. The solid blue lines represent the flight trajectories of each multirotor.

are seen to retract into a tilted triangle; since in our formation approach, the multirotors are commanded to the configuration with equal geodesics on the submanifold rather than to reach the vertices.

In addition, our proposed approach has no limits that the number of a group of robots should be equal to the number of edges on the target polygon to form. An intuitive instance can be seen in Fig. 6. In this case, five multirotors were commanded to form a triangle from a line initial condition. One of the multirotors converged to the vertex of the target triangle while other four UAVs at the end converged to somewhere at the edges, where each agent kept the equal-geodesic condition.

D. Rectangular Formation

In the second case, we commanded four quadrotors to achieve a rectangular formation. Still, ten trials were carried out. The resulting trajectories of UAVs in one of the trials can be found in Fig. 7.

The shape was decided as a rectangle with vertices of $(-1, -1, 1)^T$, $(-1, 1, 1)^T$, $(1, 1, 1)^T$, and $(1, -1, 1)^T$ w.r.t. the prescribed target center. In this trial, the UAVs started the formation task from three hovering points on one side of the target rectangle, and the rest from the opposite side. Each multirotor converged to an equilibrium position whose horizontal projection is a calculated retraction point on one edge of the rectangle and hovered with equal distances (on the submanifold) to both of its neighbor UAVs.

Without the auxiliary rectangle (solid black lines), the four UAVs are seen to converge to a tilted and smaller rectangular shape, since the Euclidian distances between the neighboring multirotors are shorter than the geodesics on the submanifold.

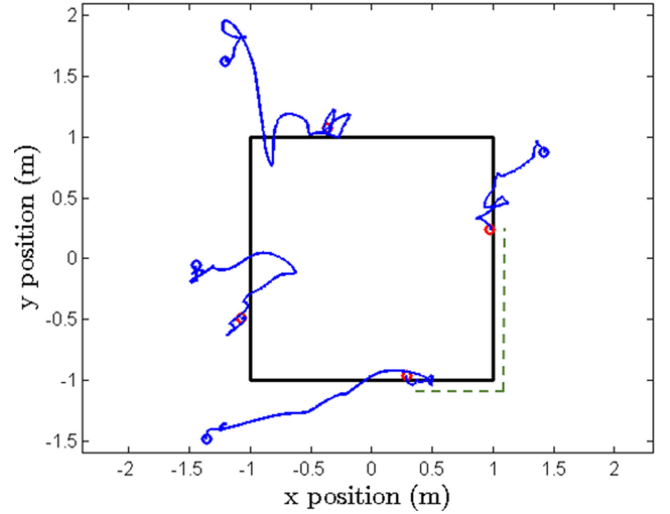


Fig. 7. Real-robot-experiment result: multirotor trajectories of a rectangular formation. The assembly of solid black lines represent the prescribed rectangle shape. The blue bubbles show the initial position where the multirotors hovered before the start of the formation task. The solid blue lines represent the flight trajectories of each multirotor. The red bubbles are the stable positions of UAVs in the target rectangular formation. The green dashed line visualizes the geodesic between two multirotors.

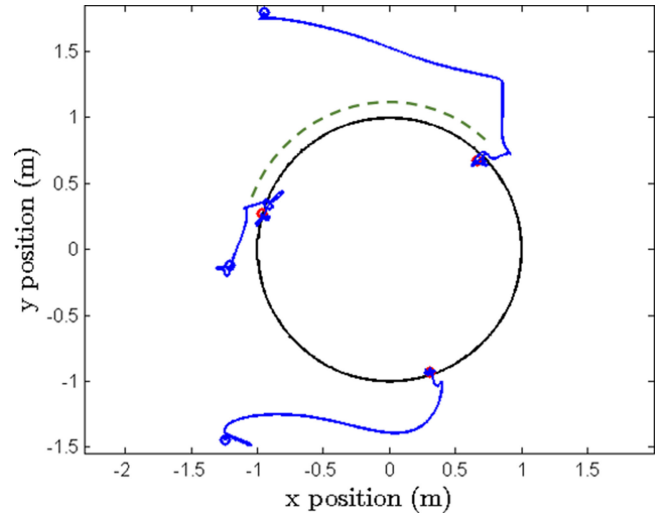


Fig. 8. Real-robot-experiment result: multirotor trajectories of a circular formation. The assembly of solid black lines represent the prescribed circular shape. The blue bubbles show the initial position where the multirotors hovered before the start of the formation task. The solid blue lines represent the flight trajectories of each multirotor. The red bubbles are the stable positions of UAVs in the target circular formation. The green dashed line visualizes the geodesic between two multirotors.

E. Circular Formation

In the third case, three quadrotors were employed to form a circle. Similarly, as in the cases we presented in the previous sections, the end-user drew a circle and set up the circular center for the group of UAVs. Ten successful trials have been carried out. The resulting trajectories in one circular formation case are displayed in Fig. 8.

It can be inferred from the results that the three quadrotors enabled to form the prescribed circle in convergence. Thus, the

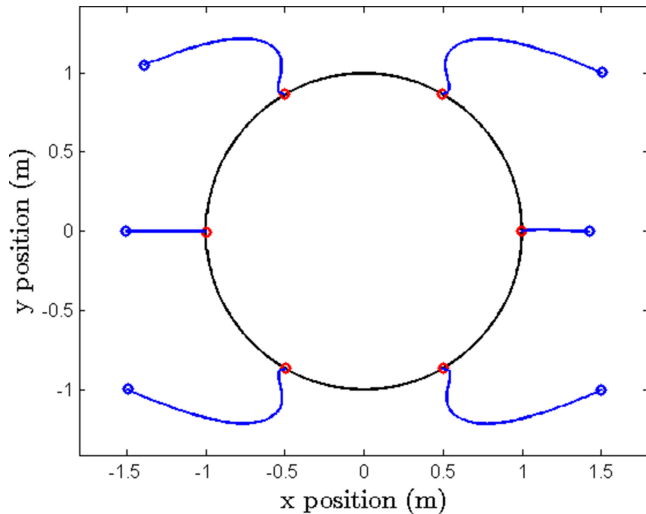


Fig. 9. Simulation result: multirotor trajectories of a circular formation. The assembly of solid black lines represent the prescribed circle shape. The initial position (blue bubbles) of multirotors are $(-1.5, -1, 1)^T$, $(-1.5, 0, 1)^T$, $(-1.5, 1, 1)^T$, $(1.5, 1, 1)^T$, $(1.5, 0, 1)^T$, and $(1.5, -1, 1)^T$, respectively. The solid blue lines represent the flight trajectories. The red bubbles are the positions of UAVs when the formation task ends.

quadrotors kept hovering at the arc on the circle in a balanced configuration (equidistant when measured with geodesic arc length).

We hereby study two more simulation cases on the circular formation. More specifically, we employed up to six onboard computers (with the same configurations as in the real-world experiments) to simulate up to six multirotor UAVs and commanded them to fly through obstacles in a time-varying circular formation.

In the first simulation trial, a group of six UAVs were commanded to generate a circular formation. Similarly as in the real experiments, the communication between neighboring UAVs is based on the circle graph. The trajectories of multirotors are displayed in Fig. 9.

The multirotors succeeded in converging to a balanced circular shape from arbitrary initial positions.

In addition, by utilizing the characteristic of the time-varying formation shape, we prescribed a trajectory of the center and the radius of the shape (or vertices for polygonal shapes). In this case, the three multirotors were first commanded to retract into a circle of a radius of 1.3 m and then were required to fly (in a balanced, circular configuration) through a gap of 2 m-wide between two gateposts. Since the circle of a radius of 1.3 m is too large for the gatepost obstacles, we prescribed a time-varying (virtual) target that shrinks into a circle of a radius of 0.5 m when the obstacles are approaching and expands back to a circle of a radius of 1.2 m after the multirotors fly through the obstacles (as shown in Fig. 10).

The reader may note that in our proposed approach, we cannot expect the robots to track a specific formation shape if it is too far away from their initial condition, since UAVs might flap with big tilted angle in the horizontal movement. In order to avoid the divergence, we always consider the velocity vector computed in the formation algorithm as a terminal state in the

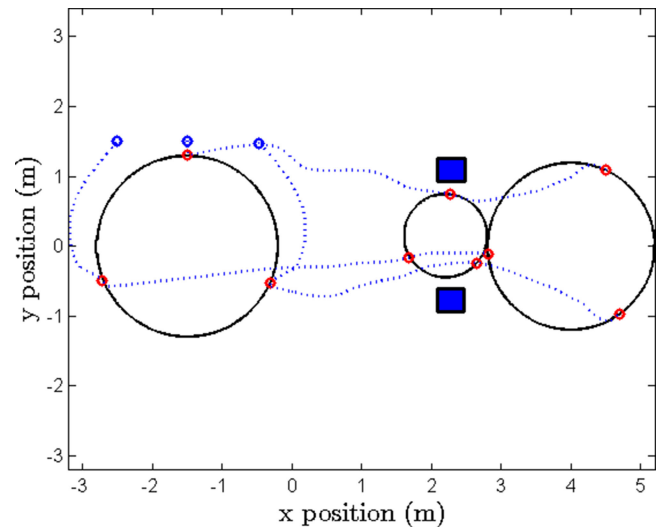


Fig. 10. Simulation result: the horizontal projection of the resulting trajectories of three multirotor UAVs in the “Moving formation through obstacles” experiment. The blue bubbles represent the initial positions of UAVs. The dashed blue lines represent the trajectories of multirotors following the moving circular formation maneuvers. The solid black lines and the red bubbles display the target circle shape submanifold and the positions of three multirotors at 10 s, 14 s, and 18 s, respectively. The solid blue boxes represent the two gatepost obstacles.

bounded OCP, instead of directly using it as reference to track in the position/velocity controller at each time step. Through this kind of solution, we practically improve the convergence of our proposed approach in the real-world implementation.

F. Spherical Formation

Apart from the experiments on the cases of the formation in \mathbb{R}^2 , we extended our control algorithm and adapted it to formation maneuvers in a 3-D environment. We hereby demonstrate a case study on a spherical formation in \mathbb{R}^3 via a simulation of five hexarotor UAVs. Differently from the \mathbb{R}^2 cases, the communication between the multirotors in \mathbb{R}^3 is based on the complete graph, as we have briefly introduced in Section II-C. The resulting trajectories of the group of five UAVs are illustrated in Fig. 11.

In the experiment, the multirotors all took off and hovered at a height of 1 m. They were then commanded to form a sphere in a 3-D environment. Although the geometry and the radius of the formation shape is usually decided offline, it is not necessary to keep the target shape static. In this case, the center and the radius of the target sphere was set changing.

In the first stage, the target sphere [the color meshed sphere in Fig. 11(a)] was centering at $(0, 1, 2)^T$ with a radius of 1 m. The UAVs succeeded in converging to the target sphere from their initial positions (the blue bubbles). In the coming second stage, the center of the target sphere was kept but we expanded the radius to 1.3 m. In Fig. 11(b), we see that the UAVs reached balanced positions (the red bubbles) and converged to the surface of the expanded sphere. The multirotors were finally commanded to form a 1 m-radius sphere centered at $(0, -0.2, 1.6)^T$ [the color meshed sphere in Fig. 11(c)] in the last stage.

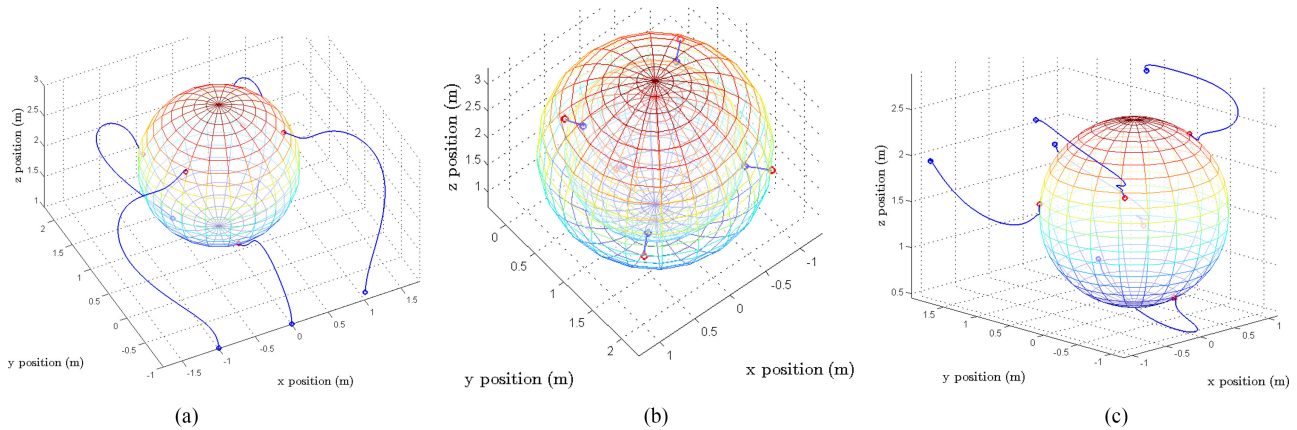


Fig. 11. Simulation result: multirotor trajectories in spherical formation in \mathbb{R}^3 . The solid blue lines represent the trajectories of multirotors following the moving spherical formation maneuvers. The blue bubbles are the initial positions of UAVs. The red bubbles are the positions of UAVs when they converged to the target sphere. (a) Stage 1. (b) Stage 2. (c) Stage 3.

V. CONCLUSION

In this paper, we proposed a distributed formation control approach for a group of multirotor UAVs. The approach enables them to simultaneously achieve a balanced configuration on a prescribed shape either in 2-D or 3-D. We combined our formation algorithm with a flight control structure based on an on-line MPC method plus a disturbance observer and implemented the complete framework on low-power onboard computational units. Both HIL simulations and real-world experiments have validated that the distributed formation control approach is feasible for arbitrary, time-varying circular, triangular, rectangular or spherical formations, etc. Future work could include an extension of the formation algorithm in the 3-D scenarios with multiple obstacles. Another potential improvement is to extend the formation control approach with a human operator directing the swarm by “drawing” the desired (time-varying) formation shape and trajectory online.

REFERENCES

- [1] K.-K. Oh, M.-C. Park, and H.-S. Ahn, “A survey of multi-agent formation control,” *Automatica*, vol. 53, pp. 424–440, 2015.
- [2] W. R. W. Ren, “Consensus based formation control strategies for multi-vehicle systems,” in *Proc. 2006 Amer. Control Conf.*, 2006, pp. 4237–4242.
- [3] B. D. O. Anderson, C. Yu, B. Fidan, and J. M. Hendrickx, “Rigid graph control architectures for autonomous formations,” *IEEE Control Syst. Mag.*, vol. 28, no. 6, pp. 48–63, Dec. 2008.
- [4] L. Krick, L. Broucke, and B. Francis, “Stabilization of infinitesimally rigid formations of multi-robot networks,” *Int. J. Control*, vol. 82, no. 3, pp. 423–439, 2009.
- [5] D. Zelazo, P. R. Giordano, and A. Franchi, “Bearing-only formation control using an SE(2) rigidity theory,” in *Proc. IEEE Conf. Decision Control*, 2015, pp. 6121–6126.
- [6] S. Zhao and D. Zelazo, “Bearing rigidity and almost global bearing-only formation stabilization,” *IEEE Trans. Autom. Control*, vol. 61, no. 5, pp. 1255–1268, May 2016.
- [7] J. M. Montenbruck, D. Zelazo, and F. Allgöwer, “Fekete points, formation control, and the balancing problem,” *IEEE Trans. Autom. Control*, vol. 62, no. 10, pp. 5069–5081, Oct. 2017.
- [8] N. Michael, J. Fink, and V. Kumar, “Cooperative manipulation and transportation with aerial robots,” *Auton. Robots*, vol. 30, no. 1, pp. 73–86, 2011.
- [9] F. Augugliaro *et al.*, “The flight assembled architecture installation: Cooperative construction with flying machines,” *IEEE Control Syst. Mag.*, vol. 34, no. 4, pp. 46–64, Aug. 2014.
- [10] R. D’Andrea, “Meet the dazzling flying machines of the future,” *TED Talks*. [Podcast]. Feb. 19, 2016. [Online]. Available: https://www.ted.com/talks/raffaello_d_andrea_meet_the_dazzling_flying_machines_of_the_future. Accessed on: Aug. 30, 2016.
- [11] “Pepsi Zero Super Bowl LI halftime show,” Intel. [Online Video]. Feb. 5, 2017. [Online]. Available: <https://www.youtube.com/watch?v=txXwg712zw4>. Accessed on: Feb. 25, 2017.
- [12] X. Wang, V. Yadav, and S. N. Balakrishnan, “Cooperative UAV formation flying with obstacle/collision avoidance,” *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 4, pp. 672–679, Jul. 2007.
- [13] J. Wang and M. Xin, “Integrated optimal formation control of multiple unmanned aerial vehicles,” *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 5, pp. 1731–1744, Sep. 2013.
- [14] J. Shin and H. J. Kim, “Nonlinear model predictive formation flight,” *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 5, pp. 1116–1125, Sep. 2009.
- [15] M. Turpin, N. Michael, and V. Kumar, “Decentralized formation control with variable shapes for aerial robots,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 23–30.
- [16] A. Franchi, C. Masone, V. Grabe, M. Ryll, H. H. Bühlhoff, and P. R. Giordano, “Modeling and control of UAV bearing formations with bilateral high-level steering,” *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1504–1525, 2012.
- [17] D. Zhou and M. Schwager, “Virtual rigid bodies for coordinated agile maneuvering of teams of micro aerial vehicles,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 1737–1742.
- [18] J. Alonso-Mora, S. Baker, and D. Rus, “Multi-robot navigation in formation via sequential convex programming,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 4634–4641.
- [19] M. Aranda, G. Lopez-Nicolas, C. Sagues, and Y. Mezouar, “Formation control of mobile robots using multiple aerial cameras,” *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 1064–1071, Aug. 2015.
- [20] L. A. V. Reyes and H. Tanner, “Flocking, formation control, and path following for a group of mobile robots,” *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 4, pp. 1268–1282, Jul. 2014.
- [21] X. Chen and Y. Jia, “Adaptive leader-follower formation control of non-holonomic mobile robots using active vision,” *IET Control Theory Appl.*, vol. 9, no. 8, pp. 1302–1311, 2015.
- [22] X. Dong, B. Yu, Z. Shi, and Y. Zhong, “Time-varying formation control for unmanned aerial vehicles: Theories and applications,” *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 340–348, Jan. 2015.
- [23] M. Fekete, “über die verteilung der wurzeln bei gewissen algebraischen gleichungen mit ganzzahligen koeffizienten,” *Mathematische Zeitschrift*, vol. 17, no. 1, pp. 228–249, 1923.
- [24] N. Bhatia and G. Szegö, *Dynamical Systems: Stability Theory and Applications*. Berlin, Germany: Springer, 1967.
- [25] S. Bouabdallah and R. Siegwart, “Full control of a quadrotor,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Diego, CA, USA, 2007, pp. 153–158.
- [26] J. Mattingley, Y. Wang, and S. Boyd, “Automatic generation of high-speed solvers,” *IEEE Control Syst. Mag.*, vol. 31, no. 3, pp. 52–65, Jun. 2011.

- [27] Y. Liu, J. M. Montenbruck, P. Stegagno, F. Allgöwer, and A. Zell, "A robust nonlinear controller for nontrivial quadrotor maneuvers: Approach and verification," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 600–606.
- [28] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. 49th IEEE Conf. Decision Control*, Atlanta, GA, USA, Dec. 2010, pp. 5420–5425.
- [29] Y. Liu *et al.*, "Robust nonlinear control approach to nontrivial maneuvers and obstacle avoidance for quadrotor UAV under disturbances," *Robot. Auton. Syst.*, vol. 98, pp. 317–332, 2017.
- [30] S. Takakura, T. Murakami, and K. Ohnishi, "An approach to collision detection and recovery motion in industrial robot," in *Proc. IEEE 15th Annu. Conf. Ind. Electron. Soc.*, Nov. 1989, vol. 2, pp. 421–426.
- [31] D. Morgan, G. Subramanian, S. Chung, and F. Hadaegh, "Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming," *Int. J. Robot. Res.*, vol. 35, no. 10, pp. 1261–1285, 2016.



Yuyi Liu received bachelor's degrees from Fudan University, Shanghai, China and the University of Birmingham, Birmingham, U.K., in 2011, and the master's degree in aeronautical engineering from Imperial College London, London, U.K., in 2012.

He had a research stay with Katholieke Universiteit Leuven, Leuven, Belgium, in 2013. He is currently a Research Assistant with the University of Tübingen, Tübingen, Germany, and the Max Planck Institute for Biological Cybernetics, Tübingen. His research is focused on nonlinear optimal control, and

formation and human–robot interaction algorithms for aerial and mobile robots.



Jan Maximilian Montenbruck received the B.Sc. in mechanical engineering from the University of Duisburg-Essen, Duisburg, Germany, and Pennsylvania State University, State College, PA, USA, in 2011, the M.Sc. degree in mechanical engineering from the University of Duisburg-Essen, and Harvard University, Cambridge, MA, USA, in 2012, and the doctoral degree in systems theory and automatic control from the University of Stuttgart, Stuttgart, Germany, in 2016.

He is currently a Postdoctoral Researcher and Lecturer with the University of Stuttgart. His research interests are in the area of systems and control theory.



Daniel Zelazo received the B.Sc. and M.Eng. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 1999, and 2001, respectively, and the Ph.D. degree in aeronautics and astronautics from the University of Washington, Tacoma, WA, USA, in 2009.

He is an Assistant Professor of aerospace engineering with the Technion–Israel Institute of Technology, Haifa, Israel. From 2010–2012, he served as a Postdoctoral Research Associate and Lecturer with the Institute for Systems Theory and Automatic Control,

University of Stuttgart. His research interests include topics related to multiagent systems, optimization, and graph theory.



Marcin Odelga received the B.Eng. degree in robotics and automation from Wasaw University of Technology in 2009, the M.Sc. degree in automatic control and robotics from École Centrale de Nantes, Nantes, France, in 2011. He is working toward the Ph.D. degree in autonomous robotics and human–machine systems from the Max Planck Institute for Biological Cybernetics, Tübingen, Germany.

His research mainly focuses on the development of haptic teleoperation, vision-based navigation, and obstacle avoidance algorithms for aerial robots. He is

also interested in the novel mechanism design of flying robots.



Sujit Rajappa received the B.Eng. degree in electronics and instrumentation from Anna University, MIT Campus, Chennai, India, in 2008, the M.Sc. degree in automatic control and robotics from Ecole Centrale de Nantes, France, in 2013, and the Ph.D. degree in aerial robotics from Max Planck Institute for Biological Cybernetics, in 2017.

From 2008–2011, he was the Control Engineer with Honeywell Automation, Pearl GTL Site, Ras Laffan Industrial City, Qatar. He is currently a Postdoctoral Researcher with the University of Tübingen, Tübingen, Germany. His research interests also includes multiagent aerial perception, unmanned aerial vehicle modeling, and dynamics and control.



Heinrich H. Bühlhoff received the Ph.D. degree in the natural sciences from the Eberhard Karls Universität, Tübingen, Germany, in 1980.

From 1980–1988, he was a Research Scientist with the Max Planck Institute for Biological Cybernetics, Tübingen, Germany, and the Massachusetts Institute of Technology, Cambridge, MA, USA. He was Assistant, Associate, and Full Professor of cognitive science with Brown University, Providence, RI, USA, from 1988–1993. He is scientific member with the Max Planck Society and the Director and Head of the

Department of Perception, Cognition, and Action with the Max Planck Institute for Biological Cybernetics. He is a Honorary Professor with the Eberhard Karls Universität and with the Korea University, Seoul, South Korea. He is an editor of several international journals and has been involved in many international collaborations as well as a member of many national and international university boards.



Frank Allgöwer received the B.S. degree in engineering cybernetics from the University of Stuttgart, Stuttgart, Germany, the M.S. degree in applied mathematics from the University of California Los Angeles, Los Angeles, CA, USA, in 1987, and the Ph.D. degree from the University of Stuttgart, in 1996.

He is the Director of the Institute for Systems Theory and Automatic Control and an Executive Director of the Stuttgart Research Centre Systems Biology, University of Stuttgart. His research interests include

cooperative control, predictive control, and nonlinear control with application to a wide range of fields, including systems biology. He is currently the President Elect of the International Federation of Automatic Control IFAC and the Vice President of the German Research Foundation, Bonn, Germany.



Andreas Zell received the Diploma degree in computer science from the University of Kaiserslautern, Kaiserslautern, Germany, in 1986, the M.S. degree in symbolic and heuristic computation from Stanford University, Stanford, CA, USA, in 1987, and the Ph.D. degree in computer science from the University of Stuttgart, Stuttgart, Germany, in 1989.

From 1990–1994, he was a Lecturer and later an Assistant Professor with the University of Stuttgart. Since 1995, he has been a Full Professor with the Eberhard Karls Universität Tübingen, Tübingen, Germany, and is currently the Chair of Cognitive Systems. His research interests include mobile robots, robot vision, navigation and SLAM, artificial neural networks, evolutionary algorithms, and machine learning in general. As the founding Director of the Centre for Bioinformatics, Tübingen, he used machine learning for many bioinformatics applications, but is now concentrating on cognitive robotics and deep learning for technical applications.