2012 IEEE International Conference on Robotics and Automation
RiverCentre, Saint Paul, Minnesota, USA
May 14-18, 2012

Decentralized Formation Control with Variable Shapes for Aerial Robots

Matthew Turpin, Nathan Michael, and Vijay Kumar

Abstract— **We address formation control for a team of quadrotors in which the robots follow a specified group trajectory while safely changing the shape of the formation according to specifications. The formation is prescribed by shape vectors which dictate the relative separations and bearings between the robots, while the group trajectory is specified as the desired trajectory of a leader or a virtual robot in the group. Each robot plans its trajectory independently based on its local information of neighboring robots which includes both the neighbor's planned trajectory and an estimate of its state. We show that the decentralized trajectory planners (a) result in consensus on the planned trajectory for predefined shapes and (b) achieve safe reconfiguration when changing shapes.**

I. Introduction

There are several applications in which teams of aerial robots must precisely control their formation while navigating to a destination. Examples include cooperative manipulation of large payloads either via grippers or cables, aerial reconnaissance where imagery from different sensors needs to be registered and stitched together, and persistent surveillance where robots maintain coverage of a three-dimensional, dynamically changing environment. In these applications, a group of micro-aerial vehicles must be flown in a specified but possibly changing formation while following 3-D, fast trajectories as a group. There are two elements to the specification in such tasks. First, a gross trajectory is specified for the group [1], leading to the specification of a time-parameterized trajectory of the leader or a virtual robot in the team. Second, the shape of the formation is specified using relative position vectors and bearing information. This shape can change as a specified function of either the gross position of the group or of time. Most generally, the team must follow the specified group trajectory while achieving or tracking the desired shape.

There is extensive literature using a leader-follower formation control approach in the area of multi-robot systems. The problem of controlling a formation of robots to follow a group motion while maintaining a reference shape or structure using only local information and the formulation using control graphs is analyzed in [1]. A distributed controller for trajectory tracking by a team of robots maintaining a rigid virtual structure is discussed in [2]. In [3], an architecture is proposed for precision spacecraft formation control applicable to space-based interferometry for imaging stars. In this approach, the authors detail the use of coordination variables determined by the system state as compared to a desired reference state. The authors of [4] develop stabilizing controllers for driving a formation of robots to rotate, translate, expand, or contract based upon a virtual structure representation. Control of aerial vehicles following the leader-follower framework is also discussed in [5], where experimental results are analyzed to consider performance of the controllers applied to unmanned aerial vehicles.

There is also a rich literature in networked control systems where the communication, sensing, and control are formulated as a weighted graph. In addition to considering stability and convergence properties, it is possible to derive conditions that ensure feasible formation structures based on individual robot capabilities [6]. Input-to-state stability of formations is discussed in [7], by considering the construction of graphs from primitives which provide known stability properties. The authors of [8, 9] consider information flow in feedback-based controllers and the consequential effects on the stability of the system to converge to the desired formation. The introduction of exogenous second-order inputs as a virtual leader is explored in [10] with analysis of system stability to these inputs. Also relevant to our work is the fact that dynamic formation graphs pose interesting problems when dealing with the switching topology of the graph. In [11], the authors consider the controllability of state-dependent dynamic graphs. Consensus problems on formation graph structures with switching topologies and time-delays are discussed in [12].

Because of the relatively short time scales associated with aerial robot dynamics, and the dynamic interactions between different robots, it is not sufficient to simply rely on feedback policies. In our work, we develop trajectory planners for each robot that rely only on information on the current state and planned trajectory of each neighbor, in addition to addressing feedback controllers for following the planned trajectory. Thus the work on model predictive or receding horizon control control [13], and the extensions to decentralized model predictive control [14, 15] is also relevant to our work.

In our own previous work [16], we investigated the coupling between planning, control and communication in a small team of quadrotors, and the influence of network time-delays and robot failures on overall system performance. We defined a framework in which neighboring robots exchange limited information about their state and the planned trajectories and the planning by each robot occurs over finite time horizons. We also derived control laws that enable the system to follow the planned trajectories while maintaining a desired, *constant* formation shape. Our experiments showed

M. Turpin, N. Michael, and V. Kumar are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA.
Email: {mturpin, nmichael, kumar}@seas.upenn.edu

acceptable performance even as the group trajectory becomes increasingly aggressive and communication rates between the robots degrade.

In this paper, we first show that decentralized trajectory planners based on information from neighbors result in consensus on the planned trajectory for fixed shapes under some mild assumptions on the rates of communication and connectivity of the communication graph[1]. Second, we relax the assumption of constant shapes from our previous work [16] and show decentralized planners (and controllers) that yield safe trajectories, enabling reconfiguration while following a desired group trajectory. Third, we also relax the implicit assumption of "labelled" robots. When identical robots are commanded to follow a target shape, there need be no *a priori* assignment of robots to designated positions in the formation. Instead the target assignment problem for the robots is solved on the fly. Finally, our experimental studies demonstrate the effects of the connectivity graph. If the definition of the neighborhood of the robot is defined by a sensing range, changing this range (described by a radius) influences the performance of the team. For myopic robots, this radius is small and the resulting plans are suboptimal requiring frequent re-planning, while robots with long-range sensors have more information and can develop more optimal plans.

II. Preliminaries

In this section we provide a brief summary of the modeling approach from [17] and reproduce the main results for completeness.

A. Modeling, Control and Planning for a Single Quadrotor

Much of the presentation in this section follows our previous work [18, 19]. It is well-known that the quadrotor is underactuated and differentially flat [20]. The four rotor inputs allow us to specify the force along the body-fixed z-axis and the three moments in the body-fixed frame. We accordingly choose four output variables as three spatial components and the world yaw angle to specify the desired trajectory in the time interval $[t_0, t_f]$:

$$\mathbf{x}_{\mathrm{d}}(t) : [t_0,\, t_f] \rightarrow \mathbb{R}^3 \times SO(2) \tag{1}$$

The controllers achieve attitude stabilization in $SO(3)$ with a basin of attraction that covers almost all of the rotation group [21, 22]. The specification of the trajectory restricted to the subgroup $\mathbb{R}^3 \times SO(2)$ allows each robot to choose its roll and pitch angle to track the specified trajectory. This specification is also practical from the standpoint of commanding the robot — it is quite natural to specify the desired position trajectory and the heading (yaw) along the robot trajectory.

Following our previous work [16], we propose an optimization-based methodology for determining smooth trajectories for (1) that minimize an integral cost function consisting of time derivatives up to the l^{th} derivative of the trajectory. In [19], piecewise-smooth polynomial functions of order p were used to obtain trajectories that minimize the integral of the snap of the trajectories while satisfying constraints on state and input variables. Because of the differential flatness property this trajectory generation algorithm can run in real-time [17, 19]. In this work, we will assume that any segment of the planned trajectory takes the form of a polynomial function of order p:

$$\mathbf{x}_{\mathrm{d}}(t) = \sum_{k=0}^{p} \boldsymbol{\beta}^k t^k \tag{2}$$

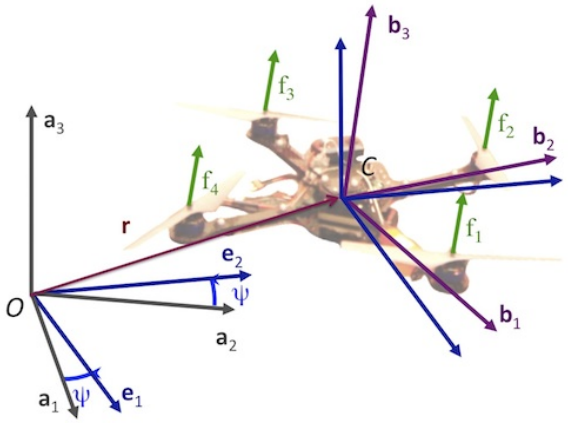


Fig. 1. Reference frames on the quadrotor

B. Modeling the Robot Formation

The configuration of a team of N robots can be represented as the Cartesian product of the configurations of each robot. As discussed above, it is convenient to only partially specify individual configurations, $\mathbf{x}_{i,\mathrm{d}} = [x_i, y_i, z_i, \psi_i]^T \in \mathbb{R}^3 \times SO(2)$, so that $4N$ variables are required to specify the team configuration. Instead of this, we specify the time-varying *group position and orientation*, $g(t) \in \mathbb{R}^3 \times SO(2)$, and a collection of time-varying *shape vectors*. The group element g can either represent the position and yaw angle of the lead robot or that of a virtual robot. The shape vectors are 4×1 vectors relating positions of pairs of robots. For agents i and j:

$$\mathbf{s}_{i,j}(t) = \mathbf{x}_j(t) - \mathbf{x}_i(t) = \begin{bmatrix} x_j(t) - x_i(t) \\ y_j(t) - y_i(t) \\ z_j(t) - z_i(t) \\ \psi_j(t) - \psi_i(t) \end{bmatrix}$$

These will be used to describe the desired relative configuration (up to roll and pitch) for each pair of robots (see Fig. 2).

Shape vectors must satisfy the following properties:

$$\begin{aligned} \mathbf{s}_{i,k} &= \mathbf{s}_{i,j} + \mathbf{s}_{j,k}, \quad \forall i,\, j,\, k \in 1, \ldots, N \\ \mathbf{s}_{i,i} &= [0,\, 0,\, 0]^{\mathrm{T}} \\ \mathbf{s}_{i,j} &= -\mathbf{s}_{j,i} \end{aligned} \tag{3}$$

In the rest of the paper, we will assume one of the robots (generally robot 1) is specified as the leader for the team

[1]Part of this discussion appears in a submission to a special issue of *Autonomous Robot* but has not appeared in a conference publication. This submission is currently under review.

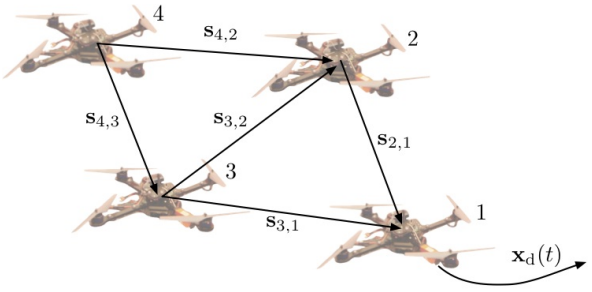


Fig. 2. The formation of quadrotors follow the leader (robot 1) which tracks a desired trajectory $\mathbf{x}_{1(t)}$. Each robot controls to maintain a desired shape defined by shape vectors and denoted by $\mathbf{ss}_{i,j}$.

and the group motion, $g(t) = \mathbf{x}_1$. The overall shape can be prescribed by a $(N-1) \times (N-1)$ vector-valued, skew-symmetric matrix.

Unlike other lead-follower schemes proposed in the literature which use hierarchical structures (see for example [1]), a follower robot uses state information from *all* other robots that it can "see" or communicate with in its neighborhood. Let $\mathcal{N}_i$ be the communication neighborhood of robot i which includes at most $(N-1)$ other robots. Thus robot i has the ability to use up to $(N-1)$ shape vectors, $\mathbf{s}_{i,j}$, $j \in \mathcal{N}_i$. However, since some of these robots are further away than other robots and the quality of communication or sensing can vary across pairs of robots, we allow each robot to choose a convex combination of weights $c_{i,j}$, which reflects the i^{th} robot's confidence in the relative estimate of the state of the j^{th} robot such that

$$\sum_{j \in \mathcal{N}_i} c_{i,j} = 1 \quad \text{and} \quad c_{i,j} \geq 0$$

These components are then used to construct the $N \times N$ *confidence matrix*, $\mathbf{C}$, whose entries are non-negative and the row sums are all 1. Note that each robot does not need to know the state of all other robots or even how many robots are in the system, but only informations about its neighbors. Finally, if one of the robots is designated as the lead robot, its row has a one in the diagonal and zeros elsewhere. If this is robot 1, its coefficients are $c_{1,1} = 1$ and $c_{1,j} = 0$, $j \neq 1$. The leader can also be a virtual robot with a reference trajectory for the formation, thus eliminating a single point of failure. Robots other than the leader will not be allowed to have dependence on their own trajectory, or $c_{i,i} = 0$ for $i \neq 1$.

C. Planning and Control for the Team

We would like the desired location of robot i to be defined by

$$\mathbf{x}_{i,\mathrm{d}} = \sum_{j \in \mathcal{N}_i} c_{i,j} \left(\mathbf{x}_i + \mathbf{s}_{i,j} \right) \tag{4}$$

Assuming that the shape vectors are selected such that they respect the required separation distance between robots to avoid collisions or aerodynamic interactions (see [23] for a discussion of these interactions), we now discuss the computation of individual robot trajectories to drive the robots to the desired formation.

The lead robot (which again can be replaced by a virtual robot or reference trajectory) computes its desired trajectory according to [19] and will follow the trajectory using the methods described in Sect. II-A. For the remaining robots, the error between the current system position and the desired position as defined by $\mathbf{s}_{i,j}$ for all i, j is:

$$\mathbf{e}_i(t) = \sum_{j \in \mathcal{N}_i} c_{i,j} (\mathbf{x}_i(t) - \mathbf{x}_j(t) - \mathbf{s}_{i,j}(t)) \tag{5}$$

We propose a receding horizon control approach which addresses network delays and other sources of information latency (see [17] for more detail). We assume that all robots operate with a synchronized system clock. We define t_c as the current time and t_h as the time horizon which a follower uses for trajectory generation. We can now determine the controls that minimize the integral of the error from (5) squared over the interval $[t_c, t_c + t_h]$. However, since the rotor speeds show up in the 4^{th} derivative of the flat outputs, we choose to add error terms incorporating higher order derivatives to the cost functional:

$$\begin{aligned}
\underset{\mathbf{x_i}(t)}{\text{minimize}} \quad & \int_{t_c}^{t_c+t_h} \left[\sum_{k=0}^{l} \kappa_k \mathbf{e}_i^{(k)}(t) \right]^2 dt \\
\text{subject to} \quad & \mathbf{x}_i(t_c) = \text{robot } i \text{ current state} \\
& \dot{\mathbf{x}}_i(t_c) = \text{robot } i \text{ current velocity} \\
& \vdots \\
& \mathbf{x}_i^{(l-1)}(t_c) \\
& \mathbf{x}_i(t) \text{ within environment} \\
& \dot{\mathbf{x}}_i(t),\, \ddot{\mathbf{x}}_i(t) \text{ within actuator limits}
\end{aligned} \tag{6}$$

where κ_k is the weight applied to the k^{th} derivative and κ_1 is greater than zero to ensure error convergence to zero.

Similarly to (2), we define the trajectory over the finite horizon, $t_c \leq t \leq (t_c + t_h)$ to be a polynomial of order $p, \mathbf{x}_i(t) = \sum_{k=0}^{p} \boldsymbol{\beta}_i^k t^k$, where $\boldsymbol{\beta}_i^k \in \mathbb{R}^4$ is the i^{th} robot's k^{th} polynomial coefficients. Similarly define all of the shape vectors over the interval of the lead robots trajectory $t_0 < t < t_f$ as: $\mathbf{s}_{i,j}(t) = \sum_{k=0}^{p} \boldsymbol{\sigma}_{i,j}^k t^k$, where $\boldsymbol{\sigma}_{i,j}^k \in \mathbb{R}^4$ is the k^{th} polynomial coefficients of the shape between robots i and j.

We then substitute these polynomial representations into the error of robot i in (5) as:

$$\mathbf{e}_i(t) = \sum_{j \in \mathcal{N}_i} c_{i,j} \left(\sum_{k=0}^{p} \left(\boldsymbol{\beta}_i^k - \boldsymbol{\beta}_j^k - \boldsymbol{\sigma}_{i,j}^k \right) t^k \right)$$

We solve (6) for robot i as a QP in real time to generate a reference trajectory which minimizes the weighted sum of the error and the derivatives of the error.

Thus the algorithm running on each robot is given below in Algorithm 1.

III. Decentralized Trajectory Planning with Specified Shapes

A. Assumptions

We first summarize the key assumptions in the paper.

Algorithm 1 Trajectory Planning Algorithm

1: Compute desired trajectory using the methods outlined in Sect. II-A.
2: Broadcast a message containing the polynomial coefficients ($\boldsymbol{\beta}_1^k$) and time interval ($[t_c, t_h]$) that fully specify its trajectory to all robots connected to it in the communication graph.
3: Receive messages from neighboring robots about their states and their planned trajectories.
4: Recompute desired shape vectors.

(A1) Homogenous Robots: We will assume all robots are homogeneous, interchangeable, and use exactly the same optimization criteria, and therefore employ the same optimization coefficients κ in (6).
(A2) Synchronous Clocks: We will assume that all of the robot clocks are synchronized and the solution to (6) can be calculated very fast. If this calculation is computed in less than δ s, we can model all robots as having synchronous network updates with all neighbors every δ s.
(A3) Neighborhoods of Robots: A pair of robots can communicate and exchange plans or state estimates with each other only if they are separated by a distance of less than the communications range h. In other words, $i \in \mathcal{N}_i$ if and only if $\|\mathbf{r}_i - \mathbf{r}_j\| \leq h$.
(A4) Robot Geometry: Robots can be treated as spheres with diameter d_r.
(A5) We will not consider inequality constraints on state variables or inputs that may be induced by saturations of actuators or sensors, or by physical constraints. In such a setting, the trajectories satisfying (6) are decoupled polynomial functions of time. Hence, the solution for each component of $\mathbf{x}_i$ is independent of other components allowing us to decouple the solution to (6) into four one-dimensional subproblems.

In this section, we will also assume some restrictions on shape specifications. Specifically, we assume labelled robots, and smooth and *safe* specifications of shape changes that explicitly specify the changes using polynomial functions of time with constant coefficients (note that constant coefficients does not mean static shape). This safety assumption will be removed in the next section.

B. Analysis

Using (A5) above, we can solve the minimization for each component of $\mathbf{x}_i \in \mathbb{R}^3 \times SO(2)$ for robot i and write the solution as follows:

$$x_i(t) = \sum_{k=0}^{p} \alpha_i^k t^k = \boldsymbol{\alpha}_i^{\mathrm{T}} \mathbf{t} \qquad t_c < t < (t_c + t_h)$$

and

$$s_{i,j}(t) = \sum_{k=0}^{p} \gamma_{i,j}^k t^k = \boldsymbol{\gamma}_{i,j}^{\mathrm{T}} \mathbf{t} \qquad t_0 < t < t_f$$

where $\boldsymbol{\alpha}_i = [\alpha_i^0, \alpha_i^1, \ldots, \alpha_i^p]^{\mathrm{T}}$, $\boldsymbol{\gamma}_{i,j} = [\gamma_{i,j}^0, \gamma_{i,j}^1, \ldots, \gamma_{i,j}^p]^{\mathrm{T}}$, and $\mathbf{t} = [t^0, t^1, \ldots, t^p]^{\mathrm{T}}$. Note that the coefficients $\gamma_{i,j}^k$ are constant as explained in Section III-A. Further, we assume that this specification is safe. In other words, robots faithfully tracking this shape specification will never collide.

Additionally define $\mathbf{d}_i = [d_i^0, d_i^1, \ldots, d_i^{(l-1)}]^{\mathrm{T}}$ such that the equality continuity constraints can then be vectorized as $\mathbf{E}_i \boldsymbol{\alpha}_i = \mathbf{d}_i$. The integral in the term to be minimized in (6) can be evaluated and then reformulated as:

$$(\boldsymbol{\alpha}_i - \boldsymbol{\tau}_i)^{\mathrm{T}} \mathbf{H}_i (\boldsymbol{\alpha}_i - \boldsymbol{\tau}_i)$$

where $\boldsymbol{\alpha}_i$ and $\boldsymbol{\tau}_i$ have dimension $(p+1) \times 1$, $\mathbf{H}_i$ is $(p+1) \times (p+1)$, and:

$$\boldsymbol{\tau}_i = \sum_{j \in \mathcal{N}_i} (c_{i,j}(\boldsymbol{\alpha}_j + \boldsymbol{\gamma}_{i,j}))$$

resulting in the quadratic program:

$$\begin{aligned} \underset{\boldsymbol{\alpha}_i}{\text{minimize}} \quad & \frac{1}{2}\boldsymbol{\alpha}_i \mathbf{H}_i \boldsymbol{\alpha}_i - \boldsymbol{\tau}_i^{\mathrm{T}} \mathbf{H}_i \boldsymbol{\alpha}_i \\ \text{subject to} \quad & \mathbf{E}_i \boldsymbol{\alpha}_i = \mathbf{d}_i \end{aligned}$$

where $\mathbf{E}_i$ is $l \times (p+1)$. This QP can be rewritten as an unconstrained optimization using an $l \times 1$ vector of Lagrange multipliers, $\boldsymbol{\lambda}_i$. The solution to this equality constrained QP is given by the solution to:

$$\begin{bmatrix} \mathbf{H}_i & \mathbf{E}_i^{\mathrm{T}} \\ \mathbf{E}_i & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_i \\ \boldsymbol{\lambda}_i \end{bmatrix} = \begin{bmatrix} \mathbf{H}_i \boldsymbol{\tau}_i \\ \mathbf{d}_i \end{bmatrix}$$

Using the Schur compliment, the closed form solution for $\boldsymbol{\alpha}_i$ is

$$\boldsymbol{\alpha}_i = (\mathbf{I} - \mathbf{Z}_i \mathbf{E}_i)\, \boldsymbol{\tau}_i + \mathbf{Z}_i \mathbf{d}_i$$

where

$$\mathbf{Z}_i = \mathbf{H}_i^{-1} \mathbf{E}_i^{\mathrm{T}} \left(\mathbf{E}_i \mathbf{H}_i^{-1} \mathbf{E}_i^{\mathrm{T}} \right)^{-1}$$

Now let $\mathbf{A}$ be a $N(p+1) \times 1$ vector of $\boldsymbol{\alpha}_j$, $\mathbf{G}$ a $N(p+1) \times 1$ vector of $\boldsymbol{\gamma}_{1,j}$, and $\mathbf{D}$ a $Nl \times 1$ vector of $\mathbf{d}_j$ as follows:

$$\mathbf{A} = [\boldsymbol{\alpha}_1^{\mathrm{T}}, \boldsymbol{\alpha}_2^{\mathrm{T}}, \ldots, \boldsymbol{\alpha}_N^{\mathrm{T}}]^{\mathrm{T}}$$

$$\mathbf{G} = [\boldsymbol{\gamma}_{1,1}^{\mathrm{T}}, \boldsymbol{\gamma}_{1,2}^{\mathrm{T}}, \ldots, \boldsymbol{\gamma}_{1,N}^{\mathrm{T}}]^{\mathrm{T}}$$

$$\mathbf{D} = [\mathbf{d}_1^{\mathrm{T}}, \mathbf{d}_2^{\mathrm{T}}, \ldots, \mathbf{d}_N^{\mathrm{T}}]^{\mathrm{T}}$$

From (A1), all robots have the same κ weights. Thus, we see that $\mathbf{H}_i = \mathbf{H}$, $\mathbf{E}_i = \mathbf{E}$, and $\mathbf{Z}_i = \mathbf{Z}$.

Under assumption (A2), each robot will have to recalculate its controls every δ s. The desired shape trajectory, $s_{i,j}(t)$, is now reparameterized in time to get:

$$s_{i,j}(\bar{t}) = \bar{\boldsymbol{\gamma}}_{i,j}^T \bar{\mathbf{t}}$$

where $\bar{t} = t + \delta$ and $\bar{\mathbf{t}} = [(t+\delta)^0, (t+\delta)^1, \ldots, (t+\delta)^p]^{\mathrm{T}}$. We will define the $(p+1) \times (p+1)$ matrix $\mathbf{Y}$, which gives us the new coefficients for this δ-shifted trajectory:

$$\bar{\boldsymbol{\gamma}}_{i,j} = \mathbf{Y} \boldsymbol{\gamma}_{i,j}$$

and correspondingly:

$$\mathbf{T} = \mathbf{I} \otimes \mathbf{Y}$$

where $\otimes$ is the Kronecker product. This results in the desired shape at time $t_c + \delta$ satisfying:

$$\mathbf{G}(t_c + \delta) = \mathbf{T}\mathbf{G}(t_c)$$

According to (A2), all robots follow their trajectory based on the coefficients $\boldsymbol{\alpha}_i$ for δ s. The robots then exchange information with their neighbors and calculate their new coefficients independently. The new coefficients for all robots will be given by the equation:

$$\mathbf{A}(k+1) = \mathbf{M}\mathbf{A}(k) + \mathbf{L}\mathbf{T}\mathbf{G}(k)$$

where $\mathbf{A}(k)$ and $\mathbf{G}(k)$ are the coefficients at time $k\delta$ and

$$\begin{aligned} \mathbf{M} &= (\mathbf{C} \otimes (\mathbf{I} - \mathbf{Z}\mathbf{E}) + \mathbf{I} \otimes \mathbf{Z}\mathbf{E})\,\mathbf{T} \\ \mathbf{L} &= (\mathbf{C} - \mathbf{I}) \otimes (\mathbf{I} - \mathbf{Z}\mathbf{E}) \end{aligned}$$

From this we can show

$$\mathbf{A}(k) = \mathbf{M}^k\mathbf{A}(0) + \sum_{j=0}^{k-1} \mathbf{M}^j\mathbf{L}\mathbf{T}^{(k-j)}\mathbf{G}(0) \qquad (7)$$

The question of interest is if the trajectories of the individual robots, given by the coefficients in $\mathbf{A}$ reach a constant $\mathbf{A}_c$. This is given by the following theorem.

Theorem 3.1: Under Assumptions (A1), (A2) and (A5), as the time between updates, δ, tends to zero, $\lim_{k\to\infty} \mathbf{A}(k) \to \mathbf{A}_c$.

Proof:

As δ approaches zero we can ignore the reparameterization in shape so that:

$$\lim_{\delta \to 0} \mathbf{Y} = \mathbf{I}$$

which in turn means $\mathbf{T} = \mathbf{I} \otimes \mathbf{I}$. Taking these limits, (7) can be converted to:

$$\mathbf{A}(k) = \mathbf{Q}^k\mathbf{A}(0) + \sum_{j=0}^{k-1} \mathbf{Q}^j\mathbf{R}$$

where

$$\mathbf{Q} = \mathbf{C} \otimes (\mathbf{I} - \mathbf{Z}\mathbf{E})$$

and

$$\mathbf{R} = ((\mathbf{C} - \mathbf{I}) \otimes (\mathbf{I} - \mathbf{Z}\mathbf{E}))\mathbf{G} + (\mathbf{I} \otimes \mathbf{Z}\mathbf{E})\mathbf{A}(0)$$

Note that for $k > 0$:

$$\mathbf{Q}^k = \mathbf{C}^k \otimes (\mathbf{I} - \mathbf{Z}\mathbf{E}).$$

Because $\mathbf{C}$ is a right stochastic matrix, from the Perron-Frobenius theorem the largest absolute value of an eigenvalue is always 1 [24]. We know this Perron-Frobenius eigenvalue corresponds to the eigenvector $\mathbf{1}$. All other eigenvalues are guaranteed to be less than 1 in magnitude and

$$\lim_{k\to\infty} \mathbf{C}^k = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 0 \end{bmatrix}$$

Thus $\mathbf{Q}^k$ converges to a constant and this in turn implies $\lim_{k\to\infty} \mathbf{A}(k) \to \mathbf{A}_c$, a constant. In other words, the robots reach consensus on their trajectories. ■

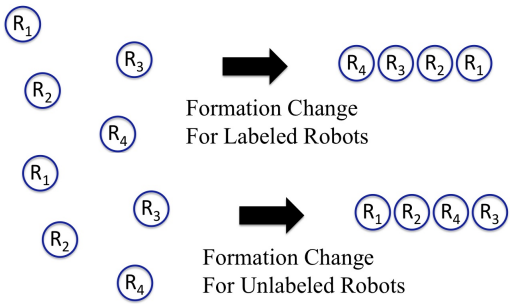


Fig. 3. This figure demonstrates the effect of labeling. For unlabeled robots, the assignment choices will determine the plans by the individual robots.

IV. Decentralized Robot Assignment For Collision Avoidance

In the previous section we assumed the shapes were specified and restricted the changes in shape to be limited to constant coefficients polynomials. In other words, the coefficients $\gamma_{i,j}^k$ are known constants. In this section we relax the assumption and address the concept of changing from one specified formation shape to another desired formation shape, developing plans for shape changes and trajectories between the shapes. We will require all the assumptions in Section III-A.

One change in the overall approach is that we now allow the robots to switch labels or assignments in the problem. As shown in Fig. 3, the robots must (a) assign targets to robots; (b) plan trajectories to ensure the shape is changed safely; and (c) follow trajectories so that the group trajectory specification is satisfied. The collision free assignment problem in (a, b) can also be formulated as a graph similarity assignment problem, which is addressed in great detail in [25]. Because the complexity of this problem grows as $\mathcal{O}(N!)$, we propose a decentralized but suboptimal approach to these problems.

Because the robots are spherical (A4), we can ignore the rotation (including the yaw angle) in the robot description. Rather than add more notation, we will let the same variable, $\mathbf{x}_i$, denote the 3×1 position vector of the robot in this section. Similarly $\mathbf{s}_{i,j}$ are also 3×1 vectors. Let the initial shape (at the time of planning) be given by vectors denoted by $\mathbf{u}_{i,j} = \mathbf{s}_{i,j}(t_{s1})$ and let the current shape at the time of computation be $\mathbf{v}_{i,j} = \mathbf{s}_{i,j}(t_c)$ where $t_c \in [t_{s1}, t_{s2}]$. Finally, $\mathbf{w}_{i,j} = \mathbf{s}_{i,j}(t_{s2})$ is the final desired shape for a given transition. For collision avoidance, we will require the specified initial and target shape to satisfy $||\mathbf{u}_{i,j}||, ||\mathbf{w}_{i,j}|| \geq \Delta\ \forall i \neq j$, where Δ is a constant that will be defined shortly.

To transition from configuration 1 at time t_{s1} to configuration 2 at time t_{s2}, we propose the simplest transition: a linear interpolation between shapes. We define the scalar $b(t) \in [0, 1]$ which increases monotonically, $b(t_{s1}) = 0$, and $b(t_{s2}) = 1$. With this notation, the initial linearly interpolated spatial shape vector between robots can be represented as

$$\mathbf{s}_{i,j}(t) = (1 - b(t))\,\mathbf{u}_{i,j} + b(t)\mathbf{w}_{i,j}\ \forall t \in [t_{s1}, t_{s2}]$$

Since $\mathbf{v}_{i,j} = \mathbf{s}_{i,j}(t_c)$, we can rewrite the expression for the

shape vectors in the interval $[t_c, t_{s2}]$ as follows:

$$\mathbf{s}_{i,j}(t) = \frac{1-b(t)}{1-b(t_c)}\mathbf{v}_{i,j} + \frac{b(t)-b(t_c)}{1-b(t_c)}\mathbf{w}_{i,j} \ \forall t \in [t_c, t_{s2}]. \tag{8}$$

Now consider the shape changes for a pair of robots. We want to determine the conditions under which the shape change through linear interpolation will lead to safety violations. Whenever necessary we want the pair of robots to switch assigned positions if it will decrease the possibility for collisions. From 3-D geometry, it is clear that it if

$$\mathbf{v}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j} \geq 0$$

there is no incentive to swap the assignment between robots i and j since this will only decrease the closest distance between the robots over $[t_c, t_{s2}]$. However, if

$$\mathbf{v}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j} < 0 \tag{9}$$

it is beneficial for robots i and j to trade their future assigned positions in the final configuration so that $\mathbf{s}_{i,j}(t_{s2})$ and $\mathbf{s}_{j,i}(t_{s2})$ have opposite signs to what they had before the trade. Alternatively, we can consider this trade to be a switch in labels. This algorithm proceeds as follows.

Algorithm 2 Reassignment Algorithm

1: Plan shape change using linear interpolation at t_{s1}
2: **while** $t < t_{s2}$ **do**
3: $\forall j \in \mathcal{N}_i$, if (9) is satisfied, trade assignments
4: Linearly interpolate for δ seconds along the desired shape using (8)

Before we show that Algorithm 2 ensures safety, we first consider the worst case scenario when a pair of robots are closest during the trajectory. At this point, $\mathbf{v}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j} = 0$. If all the robots use the same function $b(t)$, the closest robots i and j will ever be is when

$$b^* = \frac{\mathbf{v}_{i,j}{}^{\mathrm{T}}(\mathbf{v}_{i,j} - \mathbf{w}_{i,j})}{(\mathbf{v}_{i,j} - \mathbf{w}_{i,j})^{\mathrm{T}}(\mathbf{v}_{i,j} - \mathbf{w}_{i,j})}$$

This can be verified by writing the distance between the robots as a function of time and differentiating to find the minimum of this function. If b^* is outside the range $[0, 1]$, the robots are at a minimum at either the start or end of the shape change and the robots will not collide by design. If however, $b^* \in [0, 1]$, we the minimum distance achieved is

$$d_{\min} = \sqrt{\mathbf{v}_{i,j}{}^{\mathrm{T}}\mathbf{v}_{i,j} - \frac{(\mathbf{v}_{i,j}{}^{\mathrm{T}}(\mathbf{v}_{i,j} - \mathbf{w}_{i,j}))^2}{(\mathbf{v}_{i,j} - \mathbf{w}_{i,j})^{\mathrm{T}}(\mathbf{v}_{i,j} - \mathbf{w}_{i,j})}}$$

Thus in the worst case, $\mathbf{u}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j} = 0$. Further if $||\mathbf{u}_{i,j}||$ and $||\mathbf{w}_{i,j}||$ are each allowed to be as small as possible (call that value Δ), the minimum distance encountered is

$$d_{\min} = \frac{\Delta}{\sqrt{2}}$$

Since we want $d_{\min}$ to be at least equal to d_r, the diameter of the robot,

$$\Delta \geq \sqrt{2} d_r \tag{10}$$

In other words, we want the robot sensors to be such that they can detect proximity of other robots as they get to within Δ. This means that the sensor range h must be at least greater than Δ. Thus if the initial shape and the final shape are such that $||\mathbf{u}_{i,j}|| > \Delta$ and $||\mathbf{w}_{i,j}|| > \Delta$ with Δ given by (10), the minimum distance encountered by following Algorithm 2 during the trajectory is greater than d_r.

Because every robot is running Algorithm 2, there is the natural question if the process will converge. We can show that the quantity $||\mathbf{v}_{i,j} - \mathbf{w}_{i,j}||$ will always decrease after a trading of assignments since:

$$||\mathbf{v}_{i,j} - \mathbf{w}_{i,j}||^2 = \mathbf{v}_{i,j}{}^{\mathrm{T}}\mathbf{v}_{i,j} + \mathbf{w}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j} - 2\mathbf{v}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j}$$

and $\mathbf{v}_{i,j}{}^{\mathrm{T}}\mathbf{w}_{i,j}$ will change from a negative value to a positive value after a switch. Therefore for a finite set of robots, there is a finite number of possible assignments and the maximum number of iterations is $\frac{N(N-1)}{2}$. Thus, the quantity $\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}||\mathbf{v}_{i,j} - \mathbf{w}_{i,j}||^2$ will decrease with every switch and eventually converge to zero as the shape approaches the specified shape.

V. Experimental Results

The robots used are sold commercially [26] and follow a standard four-propeller design (Fig. 1). The team of four robots is shown controlling to two distinct shapes in Fig. 4. The pose of the quadrotor is observed using a VICON motion capture system at 150 Hz [27]. The pose is numerically differentiated to compute the linear and angular velocities of the robot. These values are available to MATLAB via ROS [28] and a ROS-MATLAB bridge [29]. All formation control commands are computed at 40 Hz in MATLAB using the latest state estimate at the rate of the VICON. The commands in MATLAB are bridged to ROS where they are interpreted by a finite-state machine (FSM) wich aids in the experimental process [23]. The FSM manages the individual robots, places the robots at the appropriate initial conditions, then relinquishes control to MATLAB where the desired trajectory controls are computed based on the methods outlined in the previous sections. The FSM computes the required inputs (Sect. II-A) specified by the MATLAB trajectory commands and transmits these values to the robot via ZIGBEE at a fixed rate of 50 Hz. This fixed rate is due to the limited bandwidth of ZIGBEE (57.6 kbps). These commands are interpreted by the attitude and body-fixed thrust controllers operating on each robot's programmable embedded microprocessor and applied at a 1 kHz update rate.

The scalar $b(t)$ can be any monotonically increasing function of time such that $b(t_{s1}) = 0$ and $b(t_{s2}) = 1$. One candidate function for $b(t)$ has zero velocity and acceleration at t_{s1} and t_{s2} and constant snap over the entire segment. Choosing $b(t)$ in this manner will ensure $\mathbb{C}^2$ continuity for each robot's desired position if there is no shape assignment switching. This is an ideal $b(t)$ for the quadrotor if the sensing range is large enough to ensure a fully connected graph and communications take place fast enough such that assignment convergence time is negligible. However, when robots trade future shape locations at $t_c \in [t_{s1}, t_{s2}]$, the

(a)

(b)

Fig. 4. Experimental trials of distinct shapes. Figure 4(a) shows a square with edge length of 1m and Fig. 4(b) shows a line with unit separation between neighboring robots. Video of experimental trials can be found at: http://mrsl.grasp.upenn.edu/mturpin/ICRA2012.mov.

robot's goal vector will change discontinuously. For the quadrotor with relative degree 4, this means the system will exhibit significant error which will increase with velocity at the time of the switch. In our experimental trials, we choose

$$b(t) = \frac{t - t_{s1}}{t_{s2} - t_{s1}}$$

as this minimizes the maximum time derivative of $b(t)$ over the interval $t \in [t_{s1}, t_{s2}]$.

Due to a confined operating environment, we need the robots to move as close to each other as is possible to allow for any reasonable formation motion. Therefore Δ is selected at its minimum of $\Delta = \sqrt{2}d_r$. Since the guarantees set forth for collision avoidance in Sect. IV are only valid for kinematic agents ($\dot{x} = u$), we need to account for nonzero reaction time of the dynamic system. Therefore, in experimental trials we necessarily choose h larger than the minimum value of Δ to avoid collisions. This is especially important for small interval time $t_{s2} - t_{s1}$ or large shape transitions.

We use a virtual leader in these experiments as we demonstrated the use of real robot leaders in our previous work [16]. To faithfully simulate communications with variable range, we emulate decentralized communications where each agent is programmed as its own object with access to the local information it would know and the only information exchanged between robots is its trajectory, $\boldsymbol{\beta}$, current shape, $\mathbf{v}_{i,j}$, and goal shape, $\mathbf{w}_{i,j}$. Communications are simulated between each pair of robots at $\delta = 40$Hz.

The effective diameter of the quadrotor in the $x-y$ plane is 0.7 m and the quadrotor is modeled as an ellipsoid to account for the aerodynamic effects of one quadrotor over

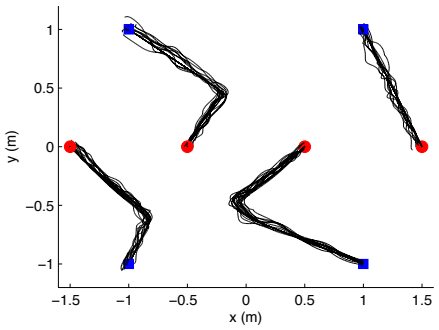


Fig. 5. A projection onto the x-y axis of 10 trials of robots transitioning from the square formation (designated by blue squares) to a line (designated by red circles). Notice how the robots always detect other agents at approximately the same location and then switch goal locations to end up at the same destination.

another [23]. Other relevant parameters include: $\kappa_1 = 1$, $\kappa_2 = 2/3$, $\kappa_3 = 1/9$, $t_h = 1s$ as well as all polynomials used are $p = 10^{th}$ order

To experimentally validate the methods derived in this paper, we carried out two investigations. The first study concerns reliability and the second study investigates the effects of discontinuous desired velocities on the performance of the quadrotor.

A. Study 1

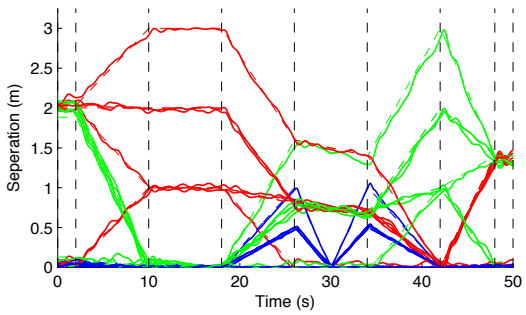
To test the effectiveness of our decentralized assignment and tracking approach to reliably resolve collision avoidance and converge to the desired shape, we repeatedly changed between the simple shapes of a square and a line. The quadrotors were randomly assigned a corner of a square and given instruction to morph to a line. Sensing range is set to its minimum bound of $h = \Delta$ in order to provide distinguished interactions between quadrotors. Figure 5 shows the paths of the robots over four trials from initial to the final configurations. The limited sensing radius greatly degrades tracking performance resulting in noticeable error.

B. Study 2

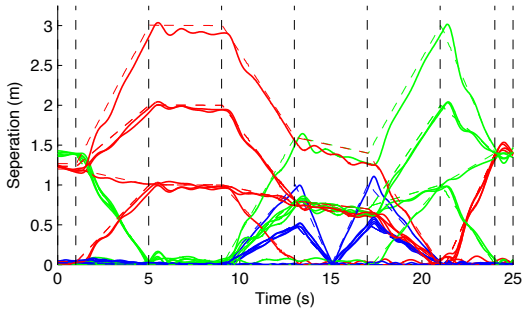
The second study is designed to test the limits of our method applied to the quadrotor. The kinematic assumptions made in Sect. IV will result in tracking errors at higher speeds. Sensing range is set to $h = 3$ m, which for the environment means that every robots can almost always see all other robots. This value is chosen to ensure fast assignment convergence as well as smaller changes in derivatives for the quadrotor. Several trials are run for the same leader path followed and shape specification over a variety of speeds. Two trials are shown in Fig. 6 with clearly superior shape error for the trial conducted at half the speed as the other. The faster experimental trials pushed the limits of the quadrotor while maintaining an acceptable separation distance between robots at all times.

VI. Conclusion and Future Work

In this work, we consider two problems relevant to formation flight of micro-aerial vehicles tracking an aggressive



(a)



(b)

Fig. 6. Relative shape tracked for changing formation shapes. Red, green, and blue are x,y, and z separations, respectively. Dotted lines indicate the desired shape and solid lines are the experimentally tracked shape. Figure 6(a) shows the relative shape for an expressive trajectory at moderate speed. Figure 6(b) shows the same experiment carried out at twice the speed with noticeable deviation from the desired shape. Black dotted lines indicate shape transition times. Note that only the top half of this plot is shown as it will be symmetric about the abscissa.

trajectory. We prove convergence of all robots to a centralized trajectory as opposed to convergence to only a desired state. We also develop a methodology with simple rules to solve the assignment problem for homogeneous robots to achieve decentralized collision avoidance. These methods result in robust performance with no collisions in over repeated trials while pushing the quadrotors to their actuator limits in a confined environment.

In the future, we plan to investigate decentralized methods for selecting shapes. This pursuit will however be very task dependent and may require a better formulation of possible missions. We are also actively working to move this research out of the laboratory and toward formations controlling along trajectories in the real world.

We gratefully acknowledge the support of ARO Grant W911NF-05-1-0219, ONR Grants N00014-07-1-0829, N00014-08-1-0696 and N00014-09-1-1051, and ARL Grant W911NF-08-2-0004. Matthew Turpin was supported by a NSF Graduate Fellowship.

REFERENCES

[1] J. P. Desai, J. P. Ostrowski, and V. Kumar, "Modeling and control of formations of nonholonomic mobile robots," *IEEE Trans. Robot.*, vol. 17, no. 6, pp. 905–908, Dec. 2001.

[2] M. Egerstedt and X. Hu, "Formation constrained multi-agent control," *IEEE Trans. Robot. Autom.*, vol. 17, no. 6, pp. 947–951, Dec. 2001.

[3] R. W. Beard, J. Lawton, and F. Y. Hadaegh, "A coordination architecture for spacecraft formation control," *IEEE Trans. Control Syst. Technology*, vol. 9, no. 6, pp. 777–790, Nov. 2001.

[4] P. Ogren, E. Fiorelli, and N. Leonard, "Formations with a mission: stable coordination of vehicle group maneuvers," in *Proc. of Intl. Sym. on Mathematical Theory Networks and Syst.*, Notre Dame, IN, Aug. 2002.

[5] Y. Gu, B. Seanor, G. Campa, M. R. Napolitano, L. Rowe, S. Gururajan, and S. Wan, "Design and flight testing evaluation of formation control laws," *IEEE Trans. Control Syst. Technol.*, vol. 14, no. 6, pp. 1105–1112, Nov. 2006.

[6] P. Tabuada, G. J. Pappas, and P. Lima, "Feasible formations of multi-agent systems," in *Proc. of the Amer. Control Conf.*, Arlington, VA, June 2001, pp. 56–61.

[7] H. Tanner, G. J. Pappas, and V. Kumar, "Input-to-state stability on formation graphs," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Las Vegas, NV, Dec. 2002, pp. 2439–2444.

[8] R. Olfati-Saber and R. M. Murray, "Distributed cooperative control of multiple vehicle formations using structural potential functions," in *Proc. of the IFAC World Congress*, Barcelona, Spain, July 2002.

[9] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1465–1476, Sept. 2004.

[10] A. Franchi, P. Giordano, C. Secchi, H. Son, and H. Bülthoff, "A passivity-based decentralized approach for the bilateral teleoperation of a group of uavs with switching topology," in *Proc. IEEE Intl Conf. on Robotics & Automation*, 2011.

[11] M. Mesbahi, "On state-dependent dynamic graphs and their controllability properties," *IEEE Trans. Autom. Control*, vol. 50, no. 3, pp. 387–392, Mar. 2005.

[12] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sept. 2004.

[13] A. Jadbabaie, J. Yu, and J. Hauser, "Unconstrained receding-horizon control of nonlinear systems," *Automatic Control, IEEE Transactions on*, vol. 46, no. 5, pp. 776–783, May 2001.

[14] T. Keviczky and K. Johansson, "A study on distributed model predictive consensus," *Arxiv preprint arXiv:0802.4450*, 2008.

[15] D. Shim, H. Kim, and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 4. IEEE, 2003, pp. 3621–3626.

[16] M. Turpin, N. Michael, and V. Kumar, "Trajectory design and control for aggressive formation flight with quadrotors," in *Proc. of the Intl. Sym. of Robotics Research*, Flagstaff, AZ, August 2011.

[17] ——, "Trajectory design and control for aggressive formation flight with quadrotors," *Autonomous Robotics*, 2011, Under Review.

[18] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Proc. of the Intl. Sym. on Exp. Robot.*, Delhi, India, Dec. 2010.

[19] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Shanghai, China, May 2011.

[20] M. J. V. Nieuwstadt and R. M. Murray, "Real-time trajectory generation for differentially flat systems," *Intl. J. Robust and Nonlinear Control*, vol. 8, no. 11, pp. 995–1020, Dec. 1998.

[21] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proc. of the IEEE Conf. on Decision and Control*, Atlanta, GA, Dec. 2010.

[22] T. Lee, "Geometric tracking control of the attitude dynamics of a rigid body on SO(3)," in *Proc. of the Amer. Control Conf.*, San Francisco, CA, Apr. 2011.

[23] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro UAV testbed," *IEEE Robot. Autom. Mag.*, vol. 17, no. 3, pp. 56–65, Sept. 2010.

[24] G. Latouche and V. Ramaswami, "Introduction to matrix analytic methods in stochastic modeling," *ASA-SIAM, Philadelphia*, 1999.

[25] T. Kämpke, "Distance patterns in structural similarity," *The Journal of Machine Learning Research*, vol. 7, pp. 2065–2086, 2006.

[26] "Ascending Technologies, GmbH," http://www.asctec.de.

[27] "Vicon Motion Systems, Inc." http://www.vicon.com.

[28] "Robot Operating System (ROS)," http://www.ros.org.

[29] "ROS-Matlab Bridge," http://github.com/nmichael/ipc-bridge.