Automatica 49 (2013) 424–433

Contents lists available at SciVerse ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Brief paper

Hybrid three-dimensional formation control for unmanned helicopters☆

Ali Karimoddini [a], Hai Lin [b,1], Ben M. Chen [a], Tong Heng Lee [a]

[a] National University of Singapore, Graduate School for Integrative Sciences and Engineering (NGS), and Department of Electrical and Computer Engineering (ECE), Singapore

[b] Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN 46556, United States

ARTICLE INFO

Article history:
Received 2 April 2011
Received in revised form
9 June 2012
Accepted 16 August 2012
Available online 8 December 2012

Keywords:
Formation control
Unmanned aerial vehicles
Hybrid supervisory control

ABSTRACT

This paper presents a hybrid supervisory control framework for the three-dimensional leader–follower formation control of unmanned helicopters. In particular, a spherical abstraction of the state space is proposed. Utilizing the properties of multi-affine functions over the partitioned space, a finite state model is obtained, which is shown to be bisimilar to the original continuous-variable dynamical system. Then, in the discrete domain, a logic supervisor is modularly designed for the abstracted model, which can be recaptured as a hybrid controller for the original continuous-variable dynamics. The designed hybrid supervisor is able to bring the unmanned helicopters to the desired formation, starting from any initial point inside the control horizon, and then maintain the formation. Moreover, a collision avoidance mechanism is embedded in the designed supervisor. The algorithm is verified through hardware-in-the-loop simulations.

1. Introduction

Formation control of unmanned aerial vehicles (UAVs) is a typical cooperative control scenario, which can leverage limited capabilities of individual UAVs to collectively perform complicated tasks that are impossible or difficult for individual agents (Anderson, Fidan, Yu, & Walle, 2008). In general, a formation scenario consists of several subtasks. First, the UAVs should be controlled to form the desired formation. After reaching the formation, the UAVs should maintain the achieved formation while tracking a desired path as a team. Meanwhile, the control algorithm is required to take care of collision between vehicles.

To address these problems, many studies have been conducted in the literature. For reaching the formation, there are several existing methods such as optimal control techniques, navigation function, and potential field (De Gennaro & Jadbabaie, 2006; How, King, & Kuwata, 2004; Paul, Krogstad, & Gravdahl, 2008). Maintaining the formation can be seen as a standard control problem in which the system's actual position has slightly deviated from the desired position, for which many control approaches have been developed, such as feedback control, rigid graph, and virtual structure (Hassan, Yahya, & ul Haq, 2006; Linorman & Liu, 2008; Shames, Fidan, & Anderson, 2009). Finally, in Cetin, Bikdash, and Hadaegh (2007), Jansson and Gustafsson (2008), and Schlanbusch, Kristiansen, and Nicklasson (2011), different mechanisms for collision avoidance have been introduced using probabilistic methods, MILP programming, and behavioral control.

However, there is still a lack of a unified solution to address the whole process, starting from reaching the formation, to maintaining the formation while avoiding collision. A usual practice is to separately design controllers for each of these tasks in a decoupled way, and then a decision making unit is needed to orchestrate the switching between these subcontrollers according to different scenarios. Although the negligence of the interactions between the continuous dynamics and the discrete logic decisions simplifies the design, this may degrade the reliability of the overall system and may cause unexpected failures. Therefore, a more comprehensive analysis requires an in-depth understanding of the interplay between the continuous dynamics and the discrete supervisory logic of the system. Hence, we are motivated to propose a unified control mechanism for the formation control of UAVs based on hybrid modeling and control theory (Koutsoukos, Antsaklis, Stiver, & Lemmon, 2000; Tabuada, 2009) that can provide a solution for all parts of the formation process and can capture both the discrete and the continuous dynamics of the system.

In this paper, we propose a three-dimensional (3D) hybrid supervisory control architecture for the path planner layer of two UAV helicopters that are involved in a leader–follower formation mission. First, a new method of abstraction based on spherical

☆ The authors gratefully acknowledge financial support from TDSI (TDSI/08-004/1A) and TL@NUS (TL/CG/2009/1). The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Andrey V. Savkin under the direction of Editor Ian R. Petersen.

E-mail addresses: akarimoddini@gmail.com (A. Karimoddini), hlin1@nd.edu (H. Lin), bmchen@ieee.org (B.M. Chen), eleleeth@nus.edu.sg (T.H. Lee).

[1] Tel.: +1 574 6313177; fax: +1 574 631 4393.

0005-1098/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2012.10.008

partitioning of the state space is introduced, by which the original continuous system with infinite states is bisimilarly abstracted to a finite state machine. Then, for the resulting abstracted system, we can take advantage of the well-developed theory of supervisory control of discrete event systems (DESs) (Ramadge & Wonham, 1989), and modularly design the discrete supervisors for reaching the formation, keeping the formation, and avoiding collisions.

This work represents an extension of our previous work (Karimoddini, Lin, Chen, & Lee, 2011), which focused on a two-dimensional (2D) formation algorithm by assuming that the UAVs' altitude remain unchanged and irrelevant. Compared with Karimoddini, Lin et al. (2011), the main contributions of this paper are that, first, the results are extended to the 3D space and the spherical partitioning of the state space is provided. The extension of the polar partitioning in 2D space to the spherical abstraction is not trivial, since the structures of the sectors in polar and spherical coordinate systems are different, so their resulting DES models are also not the same. Second, to show that the designed controller for the abstract model works for the original plant, the bisimulation of the partitioned model of the plant and its abstracted DES model is proved, and the bisimulation relation is explicitly described. Third, the proposed algorithm has been verified through a hardware-in-the-loop simulation platform which was explained in Cai, Chen, Lee, and Dong (2009).

To construct a bisimulation-based abstraction, we have utilized the properties of multi-affine functions over a spherically partitioned space. Multi-affine functions describe a large class of dynamics that are decidable under triangularization or rectangularization of the state space. In Belta and Habets (2006), a class of nonlinear systems has been abstracted using rectangular partitioning. In Broucke (2009), it has been shown that an affine feedback over a simplex can be designed to steer the system's trajectory to its exit facets, and in Habets, Collins, and van Schuppen (2006) the method is extended to the reachability problem over a partitioned system whose elements are simplices. Despite the existing theoretical developments, so far, the use of these methods for practical robotic applications is still in its infancy, and in particular, these methods have not been used in UAV path planning and formation control applications. Furthermore, the proposed spherical partitioning of the state space, locating the target at the center of the spherically partitioned space, makes it possible to generate a direct path towards the desired point, which facilitates the implementation and the design of the formation algorithm.

The rest of this paper is organized as follows. First, the problem of formation control is formulated in Section 2, and then Section 3 describes the principles of the spherical partitioning of the state space and utilizes the properties of multi-affine functions over the partitioned state space. Using this method, the partitioned model can be abstracted into a finite state machine. In Section 4, first, the DES model of the system has been developed, and then a discrete supervisor has been designed modularly. It has been also shown that how the discrete supervisor can be applied to the plant and how the closed-loop system works. Section 5 verifies the proposed algorithm through hardware-in-the-loop simulation results. Finally, the paper is concluded in Section 6. The bisimulation relation between the partitioned system and the abstracted model is proven in the Appendix. Some of the proofs and more details are presented in the extended version of this paper (Karimoddini, Lin, Chen, & Lee, 2011) as a technical report.

2. Problem formulation

The modeling and low-level control structure of the National University of Singapore (NUS) UAV helicopters are explained in Karimoddini, Cai, Chen, Lin, and Lee (2011) and Peng et al. (2009). These UAVs are Raptor-90 helicopters with 1410 mm full length and 190 mm full width of the fuselage (Cai, Feng, Chen, & Lee, 2008). For each of these helicopters we have used a multi-layer control structure whose inner-loop controller stabilizes the system using H_∞ control design techniques, and its outer loop is used to drive the UAV towards the desired position. The inner loop is fast enough to track the given references Karimoddini, Cai et al. (2011), so the outer loop dynamics can be approximately described as follows:

$$\dot{x} = u \qquad x \in \mathbb{R}^3, \quad u \in U \subseteq \mathbb{R}^3, \tag{1}$$

where x is the position of the UAV, u is the UAV's velocity reference generated by the formation algorithm, and U is the velocity constraint set, which is a convex set.

Now, consider the follower's velocity in the following form:

$$V_{follower} = V_{leader} + V_{rel}. \tag{2}$$

It is assumed that the follower always has the velocity and position information of the leader. The follower UAV should reach and keep the formation by tuning the relative velocity, V_{rel}. Alternatively, one can consider a relatively fixed frame, in which the follower moves with velocity V_{rel} and the leader has a relatively fixed position (Fig. 1). The problem is to design a supervisor in the outer loop to bring the follower UAV to the desired position with respect to the leader. Apparently, the desired position moves when the leader changes its position. Here, the control horizon is a sphere, S_{R_m}, with radius R_m, that is centered at the desired position. Therefore, the formation problem can be stated as follows.

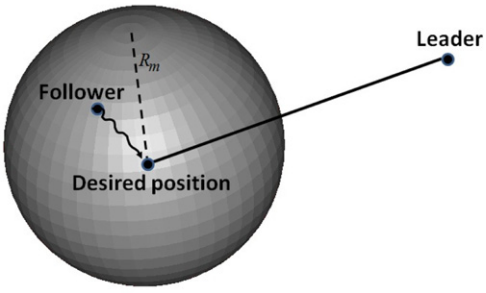

Fig. 1. In the relative frame, the follower should reach the desired position with respect to the leader.

Problem 1. Given the dynamics of the follower UAV as (1) and its velocity in the form of (2), design the formation controller to generate the relative velocity of the follower, V_{rel}, such that, starting from any initial state inside the control horizon, it eventually reaches the desired position, while avoiding a collision between the leader and the follower. Moreover, after reaching the formation, the follower UAV should remain at the desired position.

To solve this problem, we further assume that $|V_{leader_{\max}}| < |V_{follower_{\max}}|$ to make the follower be able to track the leader. On the other hand, based on Eq. (2), we should have $|V_{follower}| = |V_{leader} + V_{rel}| < |V_{follower_{\max}}|$. To satisfy this requirement, it is sufficient to have $|V_{rel}| < |V_{follower_{\max}}| - |V_{leader_{\max}}|$. This is the velocity constraint that should be considered when we design the controller to adjust V_{rel}.

To address this problem, first, we will spherically partition the control horizon, S_{R_m}, and then, using the properties of multi-affine functions, for each partitioning element, several controllers are designed which can make a partitioning element an invariant region or can derive the trajectory of the follower UAV to exit from one of its facets. Such a system can be abstracted to a finite state machine for which we will design a supervisor that can decide about the regions that should be traversed by the follower UAV to accomplish the formation task. The next section describes our proposed method of abstraction based on spherical partitioning of the control horizon.

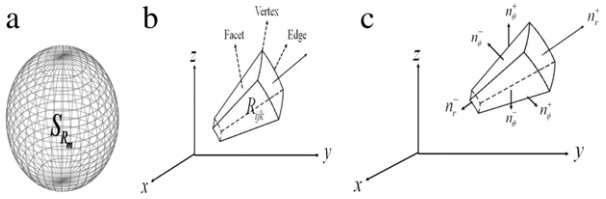

Fig. 2. (a) The partitioned sphere, (b) vertices, edges, and facets of the element $R_{i,j,k}$, and (c) outer normal vectors of the element $R_{i,j,k}$.

3. Spherical abstraction of the state space

3.1. Spherical partitioning

Consider a plant with continuous dynamics $\dot{x} = g(x) = h(x, u(x))$, defined over the sphere S_{R_m}, with radius R_m, where $u(x)$ is the control value computed based on the feedback position of the system. The sphere S_{R_m} can be partitioned in the spherical coordinate system with $r \geq 0$, $0 \leq \theta < 2\pi$, and $0 \leq \phi \leq \pi$ (Fig. 2(a)). The curves $\{r = r_i \mid 0 \leq r_i \leq R_m, \text{for } i < j : r_i < r_j, i, j = 1, \ldots, n_r, r_1 = 0, r_{n_r} = R_m\}$, $\{\theta = \theta_i \mid 0 \leq \theta_i \leq 2\pi, \text{for } i < j : \theta_i < \theta_j, i, j = 1, \ldots, n_\theta, \theta_1 = 0, \theta_{n_\theta} = 2\pi\}$, and $\{\phi = \phi_i \mid 0 \leq \phi_i \leq \pi, \text{for } i < j : \phi_i < \phi_j, i, j = 1, \ldots, n_\phi, \phi_1 = 0, \phi_{n_\phi} = \pi\}$, with $n_r, n_\theta, n_\phi \geq 2$, partition the control horizon S_{R_m}. Equivalently partitioning, we will use $\{r_i = \frac{R_m}{n_r - 1}(i-1), i = 1, \ldots, n_r\}$, $\{\theta_j = \frac{2\pi}{n_\theta - 1}(j-1), j = 1, \ldots, n_\theta\}$, and $\{\phi_k = \frac{\pi}{n_\phi - 1}(k-1), k = 1, \ldots, n_\phi\}$ as the partitioning curves. Clearly, choosing large partitions reduces the maneuverability of the UAVs in the partitioned space. Therefore, it is desired to have smaller partitions. Theoretically we can choose very small partitions for the system with a mass point model, but this increases the computation cost due to the increase in the number of discrete states, and also it may cause collision between two UAVs that are in adjacent regions. Therefore, the size of partitions should be bigger than the size of the helicopters.

In the partitioned space, the region $\bar{R}_{i,j,k} = \{x = (r, \theta, \phi) | r_i \leq r \leq r_{i+1}, \theta_j \leq \theta \leq \theta_{j+1}, \phi_k \leq \phi \leq \phi_{k+1}\}$ is a subset of S_{R_m} surrounded by the above curves. We use the term $R_{i,j,k}$ to denote the interior of the region $\bar{R}_{i,j,k}$. The intersection between the region $\bar{R}_{i,j,k}$ and the partitioning curves is called a *face*, and it could be 0-dimensional, 1-dimensional, or 2-dimensional, which are named as *vertex*, *edge*, and *facet*, respectively (Fig. 2(b)). Each region $R_{i,j,k}$ has eight vertices, v_m, $m = (m_\phi\, m_\theta\, m_r)_2$, where m_ϕ, m_θ, and m_r are the binary indices refer to the partitioning curves that have generated the vertex. For example, if $m_r = 1$, this shows that the vertex v_m of the region $R_{i,j,k}$ touches the curve r_{i+1}, and if $m_r = 0$, it touches the curve r_i. The set $V(*)$ stands for the vertices that belong to $*$ ($*$ can be a facet, a region $R_{i,j,k}$, or the sphere S_{R_m}), and $F(v_m)$ is the set of facets that the vertex v_m belongs to. Furthermore, the element $R_{i,j,k}$ has six facets $\{F_r^+, F_r^-, F_\theta^+, F_\theta^-, F_\phi^+, F_\phi^-\}$ and, correspondingly, six outer normal vectors $\{n_r^+, n_r^-, n_\theta^+, n_\theta^-, n_\phi^+, n_\phi^-\}$, as shown in Fig. 2(c). The exception is when the region $R_{i,j,k}$ touches the origin or the z axis. In this case, some of the vertices are coincident.

In the sphere S_{R_m}, let us define $\bar{S}$ as the sphere surface, and E as the set of all edges and vertices. Also, consider the *detection element* $d([i,j,k],[i',j',k']) = \bar{R}_{i,j,k} \cap \bar{R}_{i',j',k'} - E$, which is defined for two regions $R_{i,j,k}$ and $R_{i',j',k'}$ that are adjacent in a common facet. Indeed, the detection elements are the facets in which the edges and the vertices are excluded. With this procedure, the sphere S_{R_m} has been partitioned into $E \cup R_{i,j,k} \cup d([i,j,k],[i',j',k']) \cup S$, where $1 \leq i, i' \leq n_r - 1$, $1 \leq j, j' \leq n_\theta - 1$, $1 \leq k, k' \leq n_\phi - 1$ and $S = \bar{S} - E$. Correspondingly, consider $\tilde{E}, \tilde{R}_{i,j,k}, \tilde{d}([i,j,k],[i',j',k'])$, and $\tilde{S}$ as the labels for these partitioning elements, where $\Im(\tilde{r}) = r$ relates the label $\tilde{r}$ to the set r. This partitioned space can be captured by the equivalence relation $Q = \{(x_1, x_2) | \exists \tilde{r} \in \{\tilde{E}, \tilde{R}_{i,j,k}, \tilde{d}([i,j,k],[i',j',k']), \tilde{S}\}\, s.t.\, x_1, x_2 \in \Im(\tilde{r})\, and\, 1 \leq i, i' \leq n_r - 1, 1 \leq j, j' \leq n_\theta - 1, 1 \leq k, k' \leq n_\phi - 1\}$. Correspondingly, the projection map $\pi_Q(x)$ shows the partitioning element that x belongs to it: $\pi_Q(x) = \tilde{r} \in \{\tilde{E}, \tilde{R}_{i,j,k}, \tilde{d}([i,j,k],[i',j',k']), \tilde{S}\}\, s.t.\, x \in r and \Im(\tilde{r}) = r$, where $1 \leq i, i' \leq n_r - 1$, $1 \leq j, j' \leq n_\theta - 1$, $1 \leq k, k' \leq n_\phi - 1$.

In the next section, we will utilize the properties of multi-affine functions over the above partitioned space.

3.2. Multi-affine vector fields over spherically partitioned space

Multi-affine functions are a class of functions defined as follows.

Definition 1 (*Multi-Affine Function (Tabuada, 2009)*)**.** A function $g = (g_1, g_2, \ldots, g_m) : \mathbb{R}^n \to \mathbb{R}^m$ is said to be multi-affine if, for all $g_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, m$, and for every a_1, a_2 satisfying $a_1 + a_2 = 1$, the following equality holds: $g_i(x_1, \ldots, (a_1 x_{k_1} + a_2 x_{k_2}), \ldots, x_n) = a_1 g_i(x_1, \ldots, x_{k_1}, \ldots, x_n) + a_2 g_i(x_1, \ldots, x_{k_2}, \ldots, x_n)$.

3.2.1. Properties of multi-affine functions over the spherically partitioned space

Theorem 1. *For a multi-affine function $g(x) : S_{R_m} \to \mathbb{R}^3$, the following property holds:*

$$\forall x = (r, \theta, \phi) \in \bar{R}_{i,j,k} : \quad g(x) = \sum_m \lambda_m g(v_m), \quad m = 0, 1, 2, \ldots, 7, \tag{3}$$

where $v_m \in V(R_{i,j,k})$ are the vertices of the element $R_{i,j,k}$ and λ_m can be obtained uniquely as follows:

$$\lambda_m = \lambda_r^{m_r}(1-\lambda_r)^{1-m_r}\lambda_\theta^{m_\theta}(1-\lambda_\theta)^{1-m_\theta}\lambda_\phi^{m_\phi}(1-\lambda_\phi)^{1-m_\phi}, \tag{4}$$

where m_r, m_θ, m_ϕ are the corresponding binary digits of the index m, and

$$\lambda_r = \frac{r - r_i}{r_{i+1} - r_i} \qquad \lambda_\theta = \frac{\theta - \theta_j}{\theta_{j+1} - \theta_j} \qquad \lambda_\phi = \frac{\phi - \phi_k}{\phi_{k+1} - \phi_k}.$$

Proof. The proof is given in Karimoddini et al. (2011). □

Remark 1. It can be verified that the resulting coefficients $\lambda_m, m = 0, 1, \ldots, 7$, have the property that $\lambda_m \geq 0$ and $\sum_m \lambda_m = 1$.

As a special case, Theorem 1 also holds true for the points on the facets, as described in the following corollary.

Corollary 1 (*Karimoddini et al., 2011*)**.** *For a multi-affine function defined over $\bar{R}_{i,j,k}$, for all of the facets F_q^s of $R_{i,j,k}$, $q \in \{r, \theta, \phi\}$ and $s \in \{+, -\}$, the following property holds:*

$$\forall x = (r, \theta, \phi) \in F_q^s : g(x) = \sum_{v_m} \lambda_m g(v_m), \quad v_m \in V(F_q^s). \tag{5}$$

3.2.2. Utilizing multi-affine functions over the spherically partitioned space

For a multi-affine vector field defined over the region $R_{i,j,k}$, two important control features can be defined: *the invariant region* and *the exit facet*. For an invariant region, starting from any point inside it, the trajectories do not leave the region; and for a region with an exit facet, starting from any point inside the region, the trajectories leave the region through the exit facet within a finite time. Similar definitions have been used for the simplices in Broucke (2009). These control features later will be used to construct the abstracted model. The following theorems provide sufficient conditions that make a region an invariant region or make one of its facets an exit facet.

Theorem 2 (*Sufficient Condition for $R_{i,j,k}$ to be an Invariant Region*). *For a multi-affine vector field $\dot{x} = g(x), g : S_{R_m} \to \mathbb{R}^3$, $R_{i,j,k}$ is an invariant region if, for each facet F_q^s and its corresponding outer normal n_q^s, $q \in \{r, \theta, \phi\}$ and $s \in \{+, -\}$, the following inequality holds:*

$$n_q^s(y)^T \cdot g(v_m) < 0, \quad \forall v_m \in V(F_q^s), \forall y \in F_q^s. \tag{6}$$

Proof. We should show that $n_q^s(y)^T \cdot g(y) < 0$ holds true for any $y \in F_q^s$. First, according to Corollary 1, we know that $\forall y \in F_q^s : g(y) = \sum_{v_m} \lambda_m g(v_m)$, $v_m \in V(F_q^s)$. Using this value of $g(y)$, we can write

$$\begin{aligned} n_q^s(y)^T \cdot g(y) &= n_q^s(y)^T \cdot \sum_{v_m} \lambda_m g(v_m) \\ &= \sum_{v_m} \lambda_m \, n_q^s(y)^T \cdot g(v_m). \end{aligned} \tag{7}$$

Now, according to the assumption described in (6), we know that $n_q^s(y)^T \cdot g(v_m) < 0$ for all $v_m \in V(F_q^s)$ and $y \in F_q^s$. On the other hand, according to Remark 1, we have $\lambda_m \geq 0$ and $\sum_m \lambda_m = 1$. Hence, from (7), it can be concluded that $n_q^s(y)^T \cdot g(y) < 0$.

This means that the trajectories of the system cannot leave $R_{i,j,k}$ through the facet F_q^s. Since this is true for all of the facets, the trajectories of the system will not leave the region $R_{i,j,k}$ and will remain inside it forever. □

Theorem 3 (*Sufficient Condition for an Exit Facet*). *For a multi-affine vector field $\dot{x} = g(x), g : S_{R_m} \to \mathbb{R}^3$, the facet F_q^s with the outer normal n_q^s, $q \in \{r, \theta, \phi\}$ and $s \in \{+, -\}$, is an exit facet if*

1. $n_{q'}^{s'}(y)^T \cdot g(v_m) < 0 \, \forall v_m \in V(F_{q'}^{s'}), \forall y \in F_{q'}^{s'}, q' \neq q$, *or* $s' \neq s$
2. $n_q^s(y)^T \cdot g(v_m) > 0 \, \forall v_m \in V(R_{i,j,k})$, *for all* $y \in F_q^s$.

Proof. The first requirement guarantees that the trajectories of the system do not leave $R_{i,j,k}$ through the non-exit facets $F_{q'}^{s'} \neq F_q^s$. This has already been proven in Theorem 2. The second requirement is to drive the trajectory of the system out through the facet F_q^s. Based on the assumption, for all $y \in F_q^s$ and for all $v_m \in V(R_{i,j,k})$, we have $n_q^s(y)^T \cdot g(v_m) > 0$. According to Theorem 1, for the multi-affine function g, there exist $\lambda_m$such that $\forall x \in \bar{R}_{i,j,k} : g(x) = \sum_m \lambda_m g(v_m)$, $m = 0, 1, \ldots, 7$. Since $\lambda_m \geq 0$ and $\sum_m \lambda_m = 1$, $n_q^s(y)^T \cdot \lambda_m g(v_m) > 0$ for all v_m. This will lead to having $n_q^s(y)^T \cdot g(x) > 0$ for all $x \in \bar{R}_{i,j,k}$, which means that the trajectories of the system have a strictly positive velocity in the direction of $n_q^s(y)$ steering them to exit from $R_{i,j,k}$ through the facet F_q^s. □

Remark 2. Respecting the second condition of Theorem 3 for the points on the exit facet F_q^s, we will have $n_q^s(y)^T \cdot g(y) > 0, \forall y \in F_q^s$. This strictly positive inequality guarantees the following.

1. The trajectories that leave the region do not return back any more.
2. The points on the exit facet are not reachable from other points on the facet.
3. A trajectory that has reached the exit facet leaves it immediately.

3.2.3. Control over the spherically partitioned space

By proper selection of the control value $u(x)$, it is possible to tune the multi-affine vector field, $\dot{x} = g(x) = h(x, u(x))$, at the vertices, so that the region $R_{i,j,k}$ becomes an invariant region or one of its facets becomes an exit facet. Indeed, to make the region $R_{i,j,k}$ an invariant region, the value of the control signal at the vertices, $u(v_m)$, should be chosen such that $g(v_m) = h(v_m, u(v_m))$ falls in the set $U_m(\mathrm{Inv}(R_{i,j,k})) = \mathrm{Inv}_m(R_{i,j,k}) \cap U$, for $m = 0, \ldots, 7$, where $\mathrm{Inv}_m(R_{i,j,k})$ is the eligible set for the vertex v_m so that $g(v_m)$ satisfies the conditions of Theorem 2, and U is the velocity bound, which comes from the practical limitations. If $U_m(\mathrm{Inv}(R_{i,j,k})) \neq \emptyset$, for all $m = 0, \ldots, 7$, then making the region $R_{i,j,k}$ an invariant region is feasible. Based on Theorem 1, having the value of the control function u at the vertices of the region, it is possible to construct a multi-affine controller $u(x)$ for all $x \in \bar{R}_{i,j,k}$. We will use the notation C_0 to label this controller.

To make the facet F_q^s an exit facet, similar to the invariant controller, it is sufficient to choose the values of $u(v_m)$ such that $g(v_m) = h(v_m, u(v_m))$ falls in the set $U_m(Ex(F_q^s(R_{i,j,k}))) = Ex_m(F_q^s(R_{i,j,k})) \cap U$, where $Ex_m(F_q^s(R_{i,j,k}))$ is the eligible set for the vertex v_m that satisfies the exit facet condition for $F_q^s(R_{i,j,k})$, as explained in Theorem 3, and U is the velocity constraint. Therefore, for the region $R_{i,j,k}$, if all of $U_m(Ex(F_q^s(R_{i,j,k}))) \neq \emptyset$, $m = 0, \ldots, 7$, then corresponding to each of its facets, F_r^+, F_r^-, F_θ^+, F_θ^-, F_ϕ^+, F_ϕ^-, there are controllers that can make them exit facets. We label these controllers as $C_r^+, C_r^-, C_\theta^+, C_\theta^-, C_\phi^+, C_\phi^-$, respectively. In Karimoddini, Lin et al. (2011), a geometric way is introduced to solve the inequalities in Theorems 2 and 3, and to construct the eligible sets.

3.3. Abstraction of the state space

Now, consider the original system which is defined over the partitioned space. The equivalence relation Q, defined in Section 3.1, describes this partitioned space. The system over the partitioned space can be captured by a transition system $T_Q = (X_Q, X_{Q_0}, U_Q, \to_Q, Y_Q, H_Q)$, where

- $X_Q = E \cup R_{i,j,k} \cup d([i, j, k], [i', j', k']) \cup S$ is the set of system states, where $1 \leq i, i' \leq n_r - 1$, $1 \leq j, j' \leq n_\theta - 1$, $1 \leq k, k' \leq n_\phi - 1$.
- X_{Q_0} is the set of initial states. Assuming that the system initially starts from inside one of the regions $R_{i,j,k}$, $X_{Q_0} = \bigcup R_{i,j,k}$, where $1 \leq i, i' \leq n_r - 1$, $1 \leq j, j' \leq n_\theta - 1$, $1 \leq k, k' \leq n_\phi - 1$.
- $U_Q = U_a \cup U_d$, where
  - $U_a = \{C_r^+, C_r^-, C_\theta^+, C_\theta^-, C_\phi^+, C_\phi^-, C_0\}$ is the set of labels corresponding to the controllers that can make the region $R_{i,j,k}$ an invariant region or can make one of its facets an exit facet. For these control labels, as discussed in Section 3.2.3, the sets of control actions that can be activated in this region are $r(C_q^s) = \{u(x) | u(x) = \sum_m \lambda_m u(v_m), m = 0, 1, \ldots, 7, v_m \in V(R_{i,j,k}), u(v_m) \in U_m(Ex(F_q^s))\}$, and $r(C_0) = \{u(x) | u(x) = \sum \lambda_m u(v_m), v_m \in V(R_{i,j,k}), u(v_m) \in U_m(\mathrm{Inv}(R_{i,j,k}))\}$, where λ_m can be obtained by (4).
  - $U_d = U_c \cup U_e$ is the set of the detection events, where $U_c = \{\hat{d}([i, j, k], [i', j', k']) | \, 1 \leq i, i' \leq n_r - 1, 1 \leq j, j' \leq n_\theta - 1, 1 \leq k, k' \leq n_\phi - 1\}$. Here, $\hat{d}([i, j, k], [i', j', k'])$ is an event that shows that the detection element $d([i, j, k], [i', j', k'])$ has been crossed and U_e is the set of external events such as entering into an alarm zone of collision.
- $(x, x', v) \in \to_Q$, denoted by $x \xrightarrow{v}_Q x'$, if and only if one of the following conditions holds true.
  1. Actuation.
     - Exit facet: $v \in \{C_q^s | \, q \in \{r, \theta, \phi\}, s \in \{+, -\}\}$; $\pi_Q(x) \neq \pi_Q(x')$; $\exists \tilde{R}_{i,j,k}$ *and* $\tilde{d}([i, j, k], [i', j', k'])$ such that $\pi_Q(x) = \tilde{R}_{i,j,k}$ and $\pi_Q(x') = \tilde{d}([i, j, k], [i', j', k'])$; $\exists \tau$ (*finite*) *and* $\varepsilon > 0$ such that $\psi(t) : [0, \tau + \varepsilon] \to \mathbb{R}^3$ is the solution of $\dot{x} = h(x, r(v))$, $\psi(0) = x$; $\psi(\tau) = x'$, $\pi_Q(\psi(t)) = \pi_Q(x)$ for $t \in [0, \tau)$, and $\pi_Q(\psi(t)) \neq \pi_Q(x)$ for $t \in [\tau, \tau + \varepsilon]$. Here, $r(v)$ is the continuous controller corresponding to the control label v, which can be constructed as discussed above.

- Invariant region: $v = C_0$; $\exists \tilde{R}_{i,j,k}$ such that $\pi_Q(x) = \pi_Q(x') = \tilde{R}_{i,j,k}$; $\psi(t) : \mathbb{R}^+ \to \mathbb{R}^3$ is the solution of $\dot{x} = h(x, r(v))$, $\psi(0) = x$, $\psi(\tau) = x'$, and $\pi_Q(\psi(t)) = \pi_Q(x)$ for all $t \geq 0$.

2. Detection.
   - Crossing a detection element: $v \in U_c$; $\pi_Q(x) \neq \pi_Q(x')$; $\exists \tilde{R}_{i,j,k}, \tilde{R}_{i',j',k'}, \tilde{d}([i,j,k],[i',j',k'])$ such that $\pi_Q(x) = \tilde{d}([i,j,k],[i',j',k'])$ and $\pi_Q(x') = \tilde{R}_{i',j',k'}$; $\exists 0 < \varepsilon < \tau$ and $\exists w \in \{C_q^s | q \in \{r,\theta,\phi\}, s \in \{+,-\}\}$ such that $\psi(t) : [0,\tau] \to \mathbb{R}^3$ is the solution of $\dot{x} = h(x, r(w))$, $\psi(\varepsilon) = x$; $\psi(\tau) = x'$, $\pi_Q(\psi(t)) = \tilde{R}_{i,j,k}$ for $t \in (0,\varepsilon)$, and $\pi_Q(\psi(t)) = \tilde{R}_{i',j',k'}$ for $t \in (\varepsilon,\tau]$.
   - External events: $v \in U_e$, and $x = x'$. In this case, x is the value of the system state at the time instant at which event v appears. The external events do not affect the system dynamics.

- $Y_Q = X_Q$ is the output space.
- $H_Q : X \to Y_Q$ is the output map. Here, we have chosen $H_Q(x) = \pi_Q(x)$.

Although T_Q contains only important transitions that either cross the boundaries or remain inside the regions, it still has infinite states, and the analysis of such a system might be difficult. Abstraction (Alur, Henzinger, Lafferriere, & Pappas, 2000) is a technique that can reduce the complexity and can lead to a finite state machine for which the DES supervisory control tools can be used for the system analysis and control synthesis. To do so, each partitioning element can be considered one state in the abstracted model. Hence, the abstract model is a tuple $T_\xi = (X_\xi, X_{\xi_0}, U_\xi, \to_\xi, Y_\xi, H_\xi)$, where

- $X_\xi = \{\tilde{R}_{i,j,k} | 1 \leq i \leq n_r - 1, 1 \leq j \leq n_\theta - 1, 1 \leq k \leq n_\phi - 1\} \bigcup \{\tilde{d}([i,j,k],[i',j',k']) | 1 \leq i, i' \leq n_r - 1, 1 \leq j, j' \leq n_\theta - 1, 1 \leq k, k' \leq n_\phi - 1\}$, where $\tilde{R}_{i,j,k}$ and $\tilde{d}([i,j,k],[i',j',k'])$ are the labels for the regions $R_{i,j,k}$ and $d([i,j,k],[i',j',k'])$, respectively. Note that, since the system starts from a point inside the regions $R_{i,j,k}$ and never crosses the edges or the vertices (see Lemma 2 in the Appendix), the set E does not need to be considered in the abstracted system. Moreover, as the sphere S_{R_m} is the control horizon, its surface, S, should not be crossed.
- $X_{\xi_0} = \{\tilde{R}_{i,j,k} | 1 \leq i, i' \leq n_r - 1, 1 \leq j, j' \leq n_\theta - 1, 1 \leq k, k' \leq n_\phi - 1\}$.
- $U_\xi = U_a \cup U_d$ is like what we defined in T_Q.
- $(r, r', v) \in \to_\xi$, denoted by $r \xrightarrow{v}_\xi r'$, if $\exists v \in U_\xi$, $x \in \Im(r)$, $x' \in \Im(r')$ such that $x \xrightarrow{v}_Q x'$.
- $Y_\xi = X_\xi$.
- $H_\xi(r) = r$ is the output map, which is selected as an identity map.

In general, the abstract model contains all of the behaviors of the partitioned system; however, the converse might not be always true. If the converse is also true, we say that the original partitioned system and the abstract model are bisimilar. A bisimulation relation between two transition systems can be formally defined as follows.

Definition 2 (*Alur et al., 2000*)**.** Given $T_i = (Q_i, Q_i^0, U_i, \to_i, Y_i, H_i)$, $(i = 1, 2)$, R is a bisimulation relation between T_1 and T_2, denoted by $T_1 \approx_R T_2$, iff the following hold.

1. $\forall q_1 \in Q_1^0$ then $\exists q_2 \in Q_2^0$ such that $(q_1, q_2) \in R$. Also, $\forall q_2 \in Q_2^0$ then $\exists q_1 \in Q_1^0$ such that $(q_1, q_2) \in R$.
2. $\forall q_1 \to_1 q_1'$, and $(q_1, q_2) \in R$ then $\exists q_2' \in Q_2$ such that $q_2 \to_2 q_2'$ and $(q_1', q_2') \in R$. Also, $\forall q_2 \to_2 q_2'$, and $(q_1, q_2) \in R$ then $\exists q_1' \in Q_1$ such that $q_1 \to_1 q_1'$ and $(q_1', q_2') \in R$.

Theorem 4. *The original partitioned system, T_Q, and the abstract model, T_ξ, are bisimilar.*

Proof. See the Appendix for the proof. □

This bisimulation relation ensures that the abstract model, T_ξ, and the original partitioned system, T_Q, behave exactly the same, so, for the control synthesis, we can use the abstract model with finite states instead of the original partitioned system with infinite states, as we will do in the following sections.

4. Hybrid supervisory control of the plant

4.1. DES model of the plant

The finite state machine T_ξ can be formally presented by an automaton $G = (X, \Sigma, \alpha, X_0, X_m)$, where $X = X_\xi$ is the set of states; $X_0 = X_{\xi_0} \subseteq X$ is the set of initial states; $X_m = \{\tilde{R}_{1,j,k} | 1 \leq j \leq n_\theta - 1, 1 \leq k \leq n_\phi - 1\}$ is the set of final (marked) states. Σ is the (finite) set of events. The sequence of these events forms a string. We use ε to denote an empty string, while Σ^* is the set of all possible strings over the set Σ including ε. The language of the automaton G, denoted by $L(G)$, is the set of all strings that can be generated by G, starting from the initial states. The marked language, $L_m(G)$, is the set of strings that belong to $L(G)$ and end with the marked states. $L(G(x_0))$ is the set of strings that belong to $L(G)$ and start from the initial state x_0. $\bar{L}$ is the set of all prefixes to the strings that belong to the language L.

Here, the event set Σ consists of the actuation events $U_a = \{C_q^s | q \in \{r,\theta,\phi\}, s \in \{+,-\}\} \cup \{C_0\}$, the crossing detection events $U_c = \{\hat{d}([i,j,k],[i',j',k']) | 1 \leq i, i' \leq n_r - 1, 1 \leq j, j' \leq n_\theta - 1, 1 \leq k, k' \leq n_\phi - 1\}$, and the external events U_e. The set $U_e = \{C^+, C^-\}$ contains the events that are used for the collision avoidance in Section 4.2.2. The event set Σ consists of the controllable event set $\Sigma_c = U_a$ and uncontrollable event set $\Sigma_{uc} = U_d = U_c \cup U_e$. The uncontrollable events are those that cannot be affected by the supervisor. In automaton G, $\alpha : X \times \Sigma \to X$ is the transition function, which is a partial function and determines the possible transitions in the system caused by an event. This function is corresponding to $\to_\xi$ in T_ξ, so that for any $r \xrightarrow{v}_\xi r'$ we have $\alpha(r, v) = r'$. Based on the definition of T_ξ and the constructed controllers $C_0, C_r^+, C_r^-, C_\theta^+, C_\theta^+, C_\phi^+, C_\phi^-$, we have

$$\alpha(\tilde{R}_{i,j,k}, \sigma) = \begin{cases} \tilde{R}_{i,j,k} & \text{for } \sigma = C_0 \\ \tilde{R}_{i,j,k} & \text{for } \sigma \in U_e \text{ for } i \neq 1 \\ \tilde{d}([i,j,k],[i+1,j,k]) & \text{for } \sigma = C_r^+ \text{ for } i \neq n_r - 1 \\ \tilde{d}([i,j,k],[i-1,j,k]) & \text{for } \sigma = C_r^- \text{ for } i \neq 1 \\ \tilde{d}([i,j,k],[i,j+1,k]) & \text{for } \sigma = C_\theta^+ \text{ for } j \neq n_\theta - 1 \\ \tilde{d}([i,j,k],[i,1,k]) & \text{for } \sigma = C_\theta^+ \text{ for } j = n_\theta - 1 \\ \tilde{d}([i,j,k],[i,j-1,k]) & \text{for } \sigma = C_\theta^- \text{ for } j \neq 1 \\ \tilde{d}([i,j,k],[i,n_\theta - 1,k]) & \text{for } \sigma = C_\theta^- \text{ for } j = 1 \\ \tilde{d}([i,j,k],[i,j,k+1]) & \text{for } \sigma = C_\phi^+ \text{ for } k \neq n_\phi - 1 \\ \tilde{d}([i,j,k],[i,j,k-1]) & \text{for } \sigma = C_\phi^- \text{ for } k \neq 1 \end{cases}$$

$$\alpha(\tilde{d}([i,j,k],[i',j',k']), \sigma) = \tilde{R}_{i',j',k'} \quad \text{for } \sigma = \hat{d}([i,j,k],[i',j',k']).$$

Some parts of the graph representation of the system automaton are shown in Fig. 3. In this automaton, the arrows starting from one state and ending to another state represent the transitions, labeled by the events belonging to Σ. The states with an entering arrow stand for the initial states. As is shown in Fig. 3, the system could start from any of the states $\tilde{R}_{i,j,k}$.

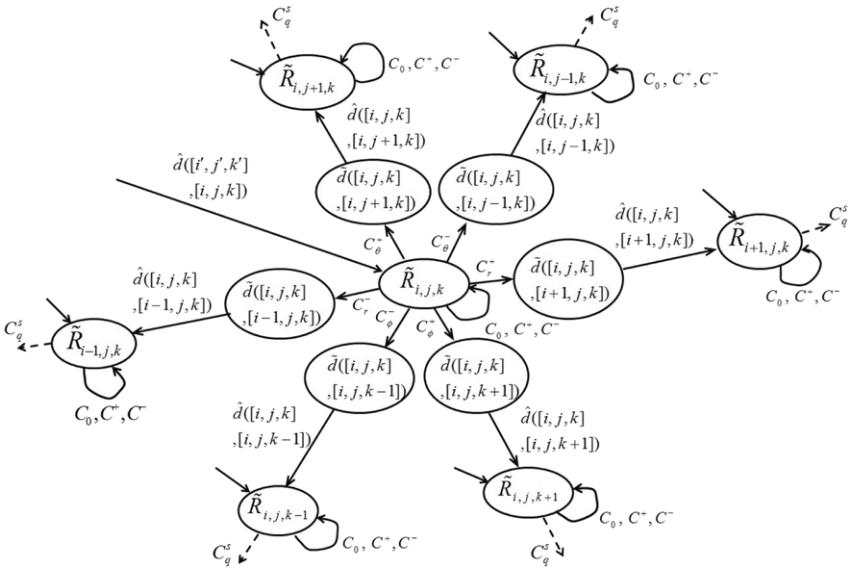

Fig. 3. DES model of the plant.

4.2. Design of the discrete supervisor

The logical behavior of the system can be modified by a discrete supervisor to achieve a desired order of events. Indeed, the supervisor, S, observes the executed events of the plant G and disables the undesirable controllable strings. Here, we assume that all of the events are observable. The language and marked language of the closed-loop system, $L(S/G)$ and $L_m(S/G)$, can be constructed as follows:

(1) $\varepsilon \in L(S/G)$,
(2) $[(s \in L(S/G))\ and\ (s\sigma \in L(G))\ and\ (\sigma \in L(S))] \Leftrightarrow (s\sigma \in L(S/G))$,
(3) $L_m(S/G) = L(S/G) \cap L_m(G)$,

where s is the string that has been generated so far, and σ is an event which the supervisor should decide whether to keep active or not.

Within this framework, we can use parallel composition to facilitate the control synthesis. Parallel composition is a binary operation between two automata. Here, parallel composition is used to combine the plant discrete model and the supervisor.

Definition 3 (*Parallel Composition (Kumar & Garg, 1995)*). Given $G = (X_G, \Sigma_G, \alpha_G, x_{0_G}, X_{m_G})$ and $S = (X_S, \Sigma_S, \alpha_S, x_{0_S}, X_{m_S})$, $G_{cl} = G \parallel S = (X_{cl}, \Sigma_{cl}, \alpha_{cl}, x_{0_{cl}}, X_{m_{cl}})$ is said to be the parallel composition of G and S with $X_{cl} = X_G \times X_S$, $\Sigma_{cl} = \Sigma_G \cup \Sigma_S$, $x_{0_{cl}} = (x_{0_G}, x_{0_S})$, $X_{m_{cl}} = X_{m_G} \times X_{m_S}$, and $\forall x = (x_1, x_2) \in X_{cl}$, $\sigma \in \Sigma_{cl}$, *then*

$$\alpha_{cl}(x,\sigma) = \begin{cases} \bullet(\alpha_G(x_1,\sigma), \alpha_S(x_2,\sigma)) \\ \quad if\ \alpha_G(x_1,\sigma)!\ and\ \alpha_S(x_2,\sigma)!\ and\ \sigma \in \Sigma_G \cap \Sigma_S \\ \bullet(\alpha_G(x_1,\sigma), x_2) \\ \quad if\ \alpha_G(x_1,\sigma)!\ and\ \sigma \in \Sigma_G - \Sigma_S \\ \bullet(x_1, \alpha_S(x_2,\sigma)) \\ \quad if\ \alpha_S(x_2,\sigma)!\ and\ \sigma \in \Sigma_S - \Sigma_G \\ \bullet undefined \quad otherwise \end{cases}$$

where $\alpha_*(x,\sigma)!$ shows the existence of a transition from state x by event σ in system $*$. In this definition, the initial conditions of these automata were assumed to be the states x_{0_G} and x_{0_S}. Extending this definition to the case that the automata G and S have the initial state sets X_{0_G} and X_{0_S}, the initial state set of the composed system will be $X_{0_{cl}} = \Upsilon(X_{0_G}, X_{0_S}) \subseteq X_{0_G} \times X_{0_S}$, where the relation Υ describes the initial states in G and S that are coupled to synchronously generate a string in the composed system.

In fact, parallel composition synchronizes operand systems on their common events; however, their private events can transit independently. The following lemma uses the parallel composition of the plant and the supervisor to obtain the closed-loop system.

Lemma 1 (*Karimoddini et al., 2011*). *Let $G = (X, \Sigma, \alpha, X_0, X_m)$ be the plant with the initial state set $X_0 = \{x_0^1, x_0^2, \ldots\}$ and $K = \bigcup K_i \subseteq \Sigma^*$ be a desired language, where K_i is the desired language that should be generated starting from x_0^i. If $\emptyset \neq K_i = \bar{K}_i \subseteq L(G(x_0^i))$ and K_i is controllable for all $i = 1, 2, \ldots, |X_0|$, then there exists a nonblocking supervisor S such that $L(S/G) = L(S \parallel G) = K$. In this case, S could be any automaton that has the initial state set $S_0 = \{s_0^1, s_0^1, \ldots, s_0^m\}$, $m \leq |X_0|$, and for any x_0^i there exists an s_0^j, $(x_0^i, s_0^j) \in \Upsilon$, which satisfies $L_m(S(s_0^j)) = L(S(s_0^j)) = K_i$, where Υ is the coupling relation between the initial states of supervisor S and plant G.*

In the next section, we will design the supervisor for reaching the formation, keeping the formation, and collision avoidance, modularly.

4.2.1. Design of the supervisor for reaching and keeping the formation

For reaching the formation, it is sufficient to drive the follower UAV directly towards one of the regions $R_{1,j,k}$, $1 \leq j \leq n_\theta - 1$, $1 \leq k \leq n_\phi - 1$, located in the first sphere. After reaching $R_{1,j,k}$, the UAV should remain inside it, for ever. This specification, K_F, is realized in Fig. 4. When the UAV is not in the first sphere, the command C_r^- will be generated to push the UAV towards the origin. Entering into a new state, the event $d([i,j,k],[i',j',k'])$ will appear to show the current state of the system. This will continue until the event $d([i,j,k],[1,j',k'])$ is generated, which shows that the formation is reached. In this case, event C_0 will be activated, which keeps the system trajectory in the first region. Since there is another module to handle the collision avoidance, the formation supervisor does not change the generable language after events C^+ and C^-, and it lets the collision avoidance supervisor disable undesirable events, as will be explained in the next section. It can be seen that K_F is controllable, as it does not disable any uncontrollable event.

Based on Lemma 1, there exists a supervisor that can control the plant to achieve this specification. The supervisor is the realization of the above specification in which all states are marked. Marking all states of the supervisor allows the closed-loop marked states to be solely determined by the plant marked states. The supervisor for reaching the formation and keeping the formation is denoted

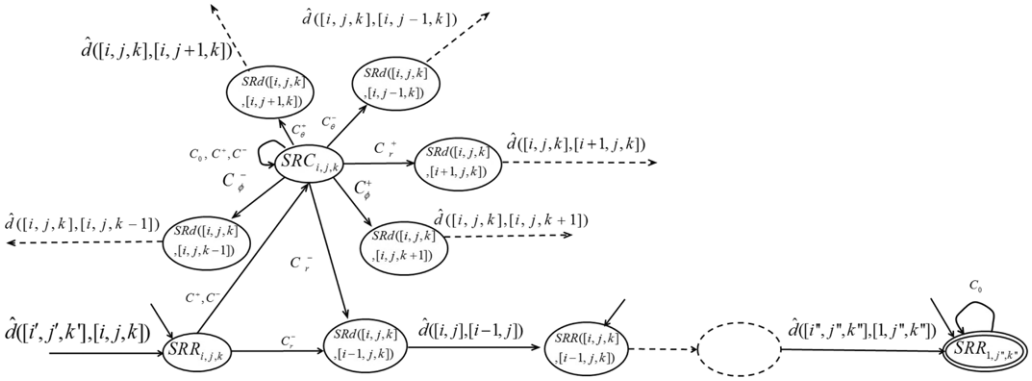

Fig. 4. The realization of reaching and keeping the formation specification.

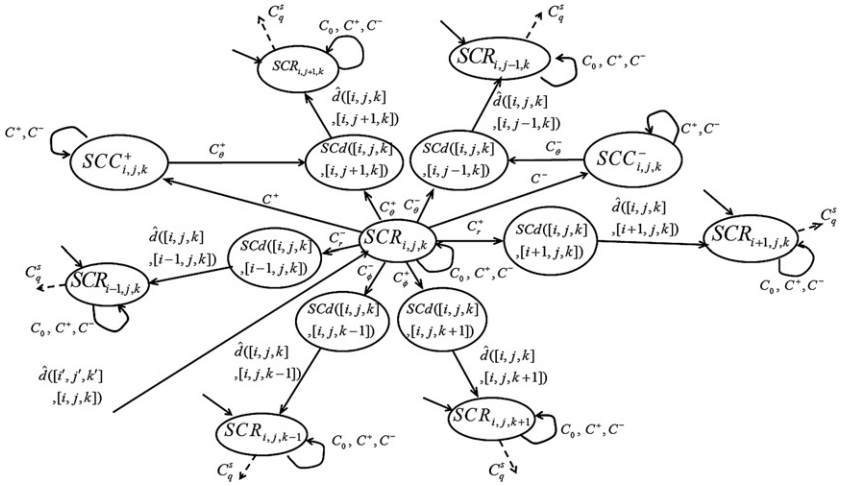

Fig. 5. Collision avoidance specification, K_C.

by S_F. The closed-loop system can be obtained using the parallel composition $G_{cl} = S_F/G = S_F \parallel G$. Here, the coupling relation is $\Upsilon = \{(R_{i,j,k}, SRR_{i,j,k}) | 1 \leq i, i' \leq n_r - 1, 1 \leq j, j' \leq n_\theta - 1, 1 \leq k, k' \leq n_\phi - 1\}$. All of the events are common between the plant and the supervisor. Moreover, it can be seen that $L(S_F) \subseteq L(G)$, which leads to $L(S_F/G) = L(G \parallel S_F) = L(G) \cap L(S_F) = L(S_F) = K_F$.

Remark 3. The tracking error in the reaching formation mode depends on the size of the partitioning elements in the first sphere, $R_{1,j,k}$, and the value of the vector field at its vertices. Indeed, there are two ways to reduce the tracking error. One way is to reduce the size of $R_{1,j,k}$, which is restricted by the size of the helicopter. But a more suitable way is to choose the value of the vector field at the vertices of $R_{1,j,k}$ so that the equilibrium point for the invariant region is pushed towards the origin. For this purpose, it is sufficient to choose bigger values for the vector field at the vertices 1, 3, 5, 7 of region $R_{1,j,k}$.

4.2.2. Design of the supervisor for collision avoidance

When the follower UAV is going to reach the desired position, in some situations, the follower may collide with the leader. If the leader is located in the way of the follower towards the desired position and the follower enters into the alarm zone, a collision alarm will be generated. More precisely, assume that the follower is in region $R_{i,j,k}$, and that the leader is in region $R_{i',j',k'}$. If $i' < i$, $|i - i'| \leq 2$, $|j - j'| < 2$, and $|k - k'| < 2$, then the follower has entered into the alarm zone and may collide with the leader. If we look at this problem from the relative frame point of view, the leader UAV has a fixed position in this frame, and therefore, for collision avoidance, it suffices that the follower turns to change its azimuth angle, θ, to exit from the alarm zone, and then it can resume the task of reaching the formation. The turning direction depends on the values of j and j'. If $j' \leq j$, then the collision alarm C^+ will be generated, which requires the follower to change the azimuth angle anticlockwise by activating the command C_θ^+, and if $j < j'$, then the collision alarm C^- will be generated, which requires the follower to change the azimuth angle clockwise by activating the command C_θ^-. This procedure will continue till the follower exits from the alarm zone. In this case, the collision alarm will be removed and the collision avoidance supervisor lets the formation supervisor resume reaching the formation. To do so, the collision avoidance supervisor only changes the generable language after the occurrence of events C^+ and C^- and lets the rest be treated by the formation supervisor. The collision avoidance specification K_C is shown in Fig. 5. Here, the coupling relation is $\Upsilon = \{(R_{i,j,k}, SCR_{i,j,k}) | 1 \leq i, i' \leq n_r - 1, 1 \leq j, j' \leq n_\theta - 1, 1 \leq k, k' \leq n_\phi - 1\}$. Again, all of the events are common between the plant and the supervisor, S_C. Hence, $L(S_C) \subseteq L(G)$, which leads to $L(S_C/G) = L(G \parallel S_C) = L(G) \cap L(S_C) = L(S_C) = K_C$, where K_C is the collision avoidance specification.

4.2.3. The closed-loop system

For prefix closed languages K_F and K_C, we can apply modular synthesis (Kumar & Garg, 1995), by the composition of the plant, the reaching and keeping the formation supervisor, and the collision avoidance supervisor: $G_{cl} = G \| S_F \| S_C$. Hence, the closed-loop system's language can be achieved as $L(G \| S_F \| S_C) = L(G) \cap L(S_F) \cap L(S_C) = L(S_F) \cap L(S_C) = K_F \cap K_C$. The refined closed-loop automaton, G_{cl}, is shown in Fig. 6.

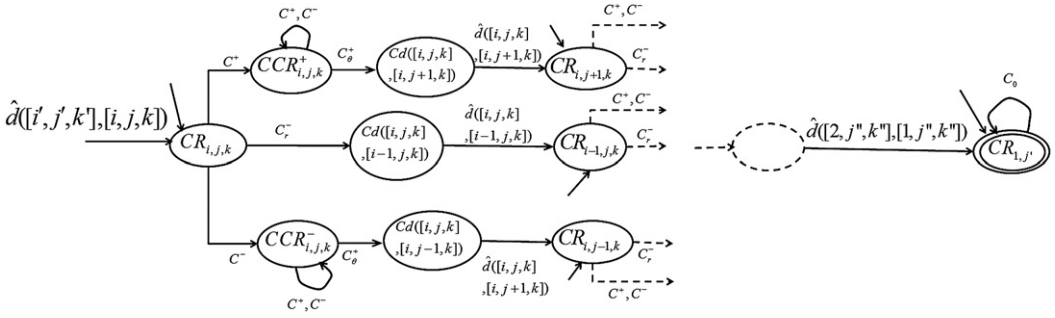

Fig. 6. The closed-loop system.

4.3. Construction of the hybrid supervisor

To describe the partitioned system and its relation with the discrete supervisor, analogous with (Koutsoukos et al., 2000) and referring to the definition of T_Q, we can define an interface layer, which connects the supervisor to the plant. The interface layer has two main blocks: the *detector* and the *actuator*.

The detector converts the continuous time signals to a sequence of symbols. Upon crossing the detection elements, a plant symbol, $\hat{d}([i, j, k], [i', j', k'])$, is generated, which informs the supervisor about the current situation of the plant. Based on the observed plant symbols, the supervisor decides which control signal should be given to the plant to satisfy the desired specification. This command has a discrete nature, but the control commands to be given to the plant need to be continuous.

The actuator translates the discrete commands to continuous ones: $r(v) = u(x) = \Sigma_m \lambda_m(x) u(v_m)$, where v is the discrete command and $u(v_m)$ should be chosen from the sets $U_m(\text{Inv}(R_{i,j,k}))$ if $v = C_0$. Otherwise, it should be selected from the set $U_m(Ex(F^s_q(R_{i,j,k})))$ for $v = C^s_q$. The coefficients $\lambda_m(x)$, $m = 0, \ldots, 7$, can be obtained from (4). The whole structure, including the interface layer and the supervisor, is implemented on the follower UAV. The following theorem shows that this hybrid structure can behave as the same as the closed-loop system of the abstracted DES model.

Theorem 5. *With the aid of the interface layer, the discrete supervisor $S = S_C \parallel S_F$ can be applied to the original partitioned system, T_Q, so that the closed-loop system satisfies the required specification, $K_F \cap K_C$.*

Proof. As constructed in Section 3.3, the plant and the interface layer elements, together with the actuator and the detector, form the transition system T_Q. Theorem 4 shows that this transition system can be bisimilarly abstracted to the finite state machine T_ξ for which we designed the discrete supervisor. Due to the bisimilarity of T_Q and T_ξ, the designed supervisor for T_ξ can work for T_Q, so their closed-loop system behaviors are the same. □

5. Verifying the algorithm

To verify the proposed algorithm, we have used a hardware-in-the-loop simulation platform (Cai et al., 2009) developed for NUS UAV helicopters (Peng et al., 2009). In this platform, the nonlinear dynamics of the UAVs have been replaced with their nonlinear model, and all software and hardware components that are involved in a real flight test remain active during the simulation so that the simulation results achieved from this simulator are very close to the actual flight tests. This multi-UAV simulator test bed is used to simulate the algorithm for the following cases.

5.1. Simulation of reaching the formation and collision avoidance

First, to monitor the reaching the formation and the collision avoidance behavior of the UAVs, assume that the leader has a fixed position and that the follower should reach the desired position with respect to the leader. The controller, $u(x)$, drives the UAV inside the spherical partitioned space. This control signal is generated using the control mechanism described in Section 3.2.3. The control horizon is a sphere of diameter 50 m. The partitioning parameters are selected as $n_r = 15$, $n_\theta = 20$, and $n_\phi = 10$. To construct the controllers C_0, C^+_r, C^-_r, C^+_θ, C^-_θ, C^+_ϕ, and C^-_ϕ, we can apply Theorems 2 and 3. Now, assume that the relative distance between the follower and the desired position is $(dx, dy, dz) = (-17, -18, -8)$. Hence, the initial state of the system is $R_{8,13,5}$. Also, assume that the leader is in $R_{4,13,5}$. Since the leader is located on the path of follower towards the desired position, when the follower reaches region $R_{6,13,5}$, a collision avoidance alarm will be generated to activate the collision avoidance mechanism and to push the follower UAV away from the alarm zone. The collision avoidance behavior of the system is shown in Fig. 7. The projection of the relative distance between the follower and its desired position onto the x–y plane is shown in Fig. 8. After observing the collision alarm, the follower first has moved towards the region $R_{6,15,5}$ to avoid the collision, and then it has resumed reaching the formation to complete the mission.

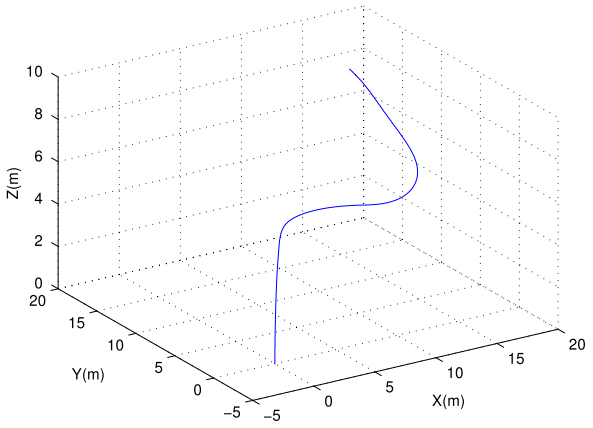

Fig. 7. The position of the UAV for the collision avoidance mechanism.

5.2. Simulation of keeping the formation

To monitor reaching and keeping the formation, let the leader track a circle with the diameter of 20 m and choose the partitioning parameters as in the previous mentioned scenario. The follower is initially located at $(dx, dy, dz) = (-20, -20, -20)$ with respect to the leader. It is expected that, after a while, the follower reaches the relative distance of $(dx, dy, dz) = (5, 5, 5)$ with respect to the leader. The behavior of the follower UAV is shown in Fig. 9. As can be seen, the follower has finally reached the desired formation and has successfully kept it.

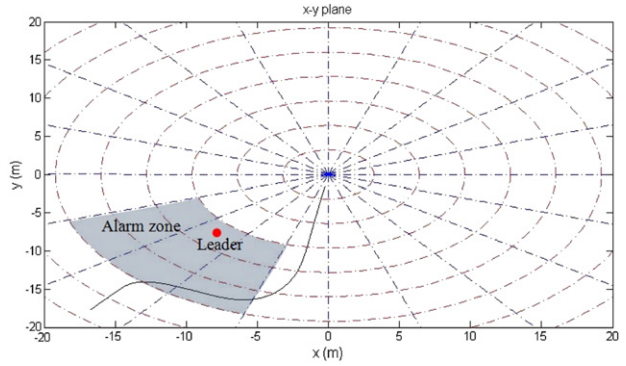

Fig. 8. The relative distance between the follower and the desired position for the collision avoidance mechanism projected onto the x–y plane.

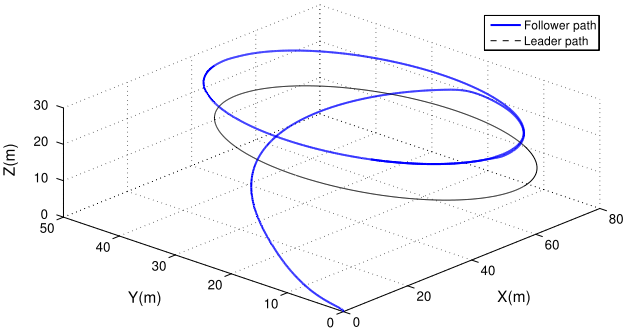

Fig. 9. The position of the UAVs in a circle formation mission.

6. Conclusion

In this paper, a hybrid supervisory control scheme was proposed for the three-dimensional leader–follower formation control of unmanned helicopters. The approach was based on spherical abstraction of the state space and utilizing the properties of multi-affine functions over the partitioned space. The designed supervisor is able to form the formation, maintain the achieved formation, and take care of collisions between the agents. The effectiveness of the algorithm was verified through hardware-in-the-loop simulations. Currently, velocity bounds are applied to the algorithm. We will try to capture some other practical limitations such as acceleration constraints through the design. Furthermore, exploring more features of the proposed method, we will focus on the extension of the results to a multi-follower case as the future direction of this research.

Acknowledgments

The authors gratefully acknowledge financial support from TDSI (TDSI/08-004/1A) and TL@NUS (TL/CG/2009/1). Also, the authors would like to thank the anonymous reviewers, associate editor, and editor for their valuable and constructive comments and suggestions.

Appendix

A.1. Proof of Theorem 4

The following two lemmas will be used for the proof of Theorem 4.

Lemma 2. *For a multi-affine vector field $\dot{x} = g(x), g : S_{R_m} \to \mathbb{R}^3$, in a region $R_{i,j,k}$ with the exit facet F^s_q constructed by Theorem 3, the trajectories that leave the region can only pass through the detection elements.*

Proof. As we saw in the proof of Theorem 2, for all points on the non-exit facets $F^{s'}_{q'}$, we have $n^{s'}_{q'}(y)^T \cdot g(y) < 0$. This strictly negative inequality shows that the trajectories of the system cannot pass through the non-exit facets, including the edges and the vertices that belong to them. In particular, the trajectories cannot cross the edges and the vertices that are common between the non-exit facets and the exit facet. On the other hand, Theorem 3 shows that the trajectories of the system cannot remain inside the region. Hence, the only way is that the trajectories pass through the internal area of the exit facet, which we have called the detection element. □

Lemma 3. *For a multi-affine vector field $\dot{x} = g(x), g : S_{R_m} \to \mathbb{R}^3$, in a region $R_{i,j,k}$ with the exit facet F^s_q constructed by Theorem 3, all $y \in F^s_q \setminus E$ are reachable from a point inside region $R_{i,j,k}$.*

Proof. Since any $y \in F^s_q$ is not reachable form an adjacent region (Part 1, Remark 2) or from another point on F^s_q (Part 2, Remark 2), then, considering $n^s_q(y)^T \cdot g(y) > 0$, by continuity of g, there is a point inside region $R_{i,j,k}$ on the neighborhood of y from which y is reachable. □

Now, to prove Theorem 4, consider the relation $R = \{(q_Q, q_\xi) | q_Q \in X_Q, q_\xi \in X_\xi, \text{ and } q_Q \in \Im(q_\xi)\}$. We will show that this relation is a bisimulation relation between T_Q and T_ξ.

Let us start with the first condition of the bisimulation relation, defined in Definition 2. For any $q_Q \in X_{Q_0}$ there exists a region $R_{i,j,k}$ such that $q_Q \in R_{i,j,k}$. For this region, there exists a label, $\tilde{R}_{i,j,k}$, such that $R_{i,j,k} = \Im(\tilde{R}_{i,j,k})$ and $\tilde{R}_{i,j,k} \in X_{\xi_0}$. Hence, $(q_Q, \tilde{R}_{i,j,k}) \in R$. Conversely, it can be similarly shown that, for any $q_\xi \in X_{\xi_0}$, there exists a $q_Q \in X_{Q_0}$ such that $(q_\xi, q_Q) \in R$.

For the second condition of the bisimulation relation, following from the definition of T_ξ, for any $(q_Q, q_\xi) \in R$ and $q_Q \xrightarrow{u}_Q q'_Q$, there exists a transition $q_\xi \xrightarrow{u}_\xi q'_\xi$, where $q'_Q \in \Im(q'_\xi)$ or equivalently $(q'_Q, q'_\xi) \in R$. For the converse case, assume that $q_\xi \xrightarrow{u}_\xi q'_\xi$. According to the definition of R, all $x \in \Im(q_\xi)$ are related to q_ξ. Hence, to prove the second condition of the bisimulation relation, we should investigate it for all $x \in \Im(q_\xi)$. Based on the control construction procedure, the labels u, q_ξ, and q'_ξ can be one of the following cases.

1. $u = C_0$ and $q_\xi = q'_\xi$. In this case, since the controller C_0 makes the region an invariant region (Theorem 2), all of the trajectories starting from any $q_Q \in \Im(q_\xi)$ will remain inside the region $\Im(q_\xi)$. Therefore, for any $q_Q \in \Im(q_\xi)$, there exists a $q'_Q \in \Im(q_\xi)$ such that $q_Q \xrightarrow{u}_Q q'_Q$ and $q'_Q = \Im(q'_\xi)$.
2. $u \in C^s_q$, $q_\xi \in \{\tilde{R}_{i,j,k} | \, 1 \le i \le n_r - 1, 1 \le j \le n_\theta - 1, 1 \le k \le n_\phi - 1\}$, and $q'_\xi \in \{\tilde{d}([i,j,k],[i',j',k'])| \, 1 \le i, i' \le n_r - 1, 1 \le j, j' \le n_\theta - 1, 1 \le k, k' \le n_\phi - 1\}$. In this case, based on Theorem 3 and Lemma 2, starting from any $q_Q \in \Im(q_\xi)$, the controller C^s_q drives the system trajectory towards the detection element $\Im(q'_\xi)$. Therefore, for any $q_Q \in \Im(q_\xi)$, there exists a $q'_Q \in \Im(q'_\xi)$ such that $q_Q \xrightarrow{u}_Q q'_Q$ and $q'_Q \in \Im(q'_\xi)$.
3. $u \in U_c = \{\hat{d}([i,j,k],[i',j',k'])| 1 \le i, i' \le n_r - 1, 1 \le j, j' \le n_\theta - 1, 1 \le k, k' \le n_\phi - 1\}$ and $q'_\xi \in \{\tilde{R}_{i',j',k'} | \, 1 \le i' \le n_r - 1, 1 \le j' \le n_\theta - 1, 1 \le k' \le n_\phi - 1\}$, and $q_\xi \in \{\tilde{d}([i,j,k],[i',j',k'])| \, 1 \le i, i' \le n_r - 1, 1 \le j, j' \le n_\theta - 1, 1 \le k, k' \le n_\phi - 1\}$. In this case, based on Lemma 3, for any $q_Q \in \Im(q_\xi)$ there exists a controller $v \in C^s_q$ that has led the trajectory of the system from the region $R_{i,j,k}$ to the point q_Q on the detection element $d([i,j,k],[i',j',k'])$. Since $R_{i',j',k'}$ is the unique adjacent region of the element $R_{i,j,k}$, common in the detection element $d([i,j,k],[i',j',k'])$, based on the definition

of the controller for the exit facet and Theorem 3, the controller v leads the trajectory of the system to a point inside the region $R_{i',j',k'}$ so that the detection event $u = \hat{d}([i, j, k], [i', j', k'])$ is generated. Therefore, for any $q_Q \in \Im(q_\xi)$, there exists a $q'_Q \in \Im(q'_\xi)$ such that $q_Q \xrightarrow{u}_Q q'_Q$ and $q'_Q \in \Im(q'_\xi)$.

4. $u \in U_e$ is the external event. In this case, the state of the system does not change, meaning that $q_Q = q'_Q$ and $q_\xi \in q'_\xi$. Therefore, trivially for any $q_Q \in \Im(q_\xi)$ and $q_\xi \xrightarrow{u} q'_\xi$, we have $q_Q \xrightarrow{u}_Q q'_Q$, where $q'_Q \in \Im(q'_\xi)$.

In all of the above-mentioned cases, the second condition of the bisimulation relation for the converse case holds true. Hence, T_ξ and T_Q are bisimilar. □

References

Alur, R., Henzinger, T., Lafferriere, G., & Pappas, G. (2000). Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, *88*(7), 971–984.

Anderson, B., Fidan, B., Yu, C., & Walle, D. (2008). UAV formation control: theory and application. In V. Blondel, S. Boyd, & H. Kimura (Eds.), *Lecture notes in control and information sciences*, *Recent advances in learning and control*. Berlin/Heidelberg: Springer.

Belta, C., & Habets, L. (2006). Controlling a class of nonlinear systems on rectangles. *IEEE Transactions on Automatic Control*, *51*(11), 1749–1759.

Broucke, M. (2009). Reach control on simplices by continuous state feedback. In *American control conference* (pp. 1154–1159). IEEE.

Cai, G., Chen, B. M., Lee, T. H., & Dong, M. (2009). Design and implementation of a hardware-in-the-loop simulation system for small-scale UAV helicopters. *Mechatronics*, *19*(7), 1057–1066.

Cai, G., Feng, L., Chen, B., & Lee, T. (2008). Systematic design methodology and construction of UAV helicopters. *Mechatronics*, *18*(10), 545–558.

Cetin, B., Bikdash, M., & Hadaegh, F. (2007). Hybrid mixed-logical linear programming algorithm for collision-free optimal path planning. *Control Theory Applications, IET*, *1*(2), 522–531.

De Gennaro, M., & Jadbabaie, A. (2006). Formation control for a cooperative multi-agent system using decentralized navigation functions. In *American control conference*. IEEE.

Habets, L., Collins, P., & van Schuppen, J. (2006). Reachability and control synthesis for piecewise-affine hybrid systems on simplices. *IEEE Transactions on Automatic Control*, *51*(6), 938–948.

Hassan, G., Yahya, K., & ul Haq, I. (2006). Leader–follower approach using full-state linearization via dynamic feedback. In *International conference on emerging technologies*, ICET'06, (pp. 297–305).

How, J., King, E., & Kuwata, Y. (2004). Flight demonstrations of cooperative control for UAV teams. In *AIAA 3rd unmanned unlimited technical conference, workshop and exhibit*.

Jansson, J., & Gustafsson, F. (2008). A framework and automotive application of collision avoidance decision making. *Automatica*, *44*(9), 2347–2351.

Karimoddini, A., Cai, G., Chen, B.M., Lin, H., & Lee, T.H. (2011). Hierarchical control design of a UAV helicopter. In *Advances in flight control systems*. INTECH. Vienna, Austria.

Karimoddini, A., Lin, H., Chen, B. M., & Lee, T. H. (2011). Hybrid formation control of the unmanned aerial vehicles. *Mechatronics*, *21*(5), 886–898.

Karimoddini, A., Lin, H., Chen, B.M., & Lee, T.H. (2011). Hybrid 3-d formation control for unmanned helicopters. Technical Report: NUS-ACT-11-005-Ver.1. Advanced Control Technology Laboratory. Available: http://arxiv.org/abs/1108.3405.

Koutsoukos, X., Antsaklis, P., Stiver, J., & Lemmon, M. (2000). Supervisory control of hybrid systems. *Proceedings of the IEEE*, *88*(7), 1026–1049.

Kumar, R., & Garg, V. K. (1995). *The Springer international series in engineering and computer science*: *Vol. 300*. *Modeling and control of logical discrete event systems*. Springer.

Linorman, N., & Liu, H. (2008). Formation UAV flight control using virtual structure and motion synchronization. In *American control conference* (pp. 1782–1787). IEEE.

Paul, T., Krogstad, T., & Gravdahl, J. (2008). Modelling of UAV formation flight using 3d potential field. *Simulation Modelling Practice and Theory*, *16*(9), 1453–1462.

Peng, K., Cai, G., Chen, B. M., Dong, M., Lum, K. Y., & Lee, T. H. (2009). Design and implementation of an autonomous flight control law for a UAV helicopter. *Automatica*, *45*(10), 2333–2338.

Ramadge, P., & Wonham, W. (1989). The control of discrete event systems. *Proceedings of the IEEE*, *77*(1), 8–98.

Schlanbusch, R., Kristiansen, R., & Nicklasson, P. J. (2011). Spacecraft formation reconfiguration with collision avoidance. *Automatica*, *47*(7), 1443–1449.

Shames, I., Fidan, B., & Anderson, B. D. (2009). Minimization of the effect of noisy measurements on localization of multi-agent autonomous formations. *Automatica*, *45*(4), 1058–1065.

Tabuada, P. (2009). *Verification and control of hybrid systems: a symbolic approach*. Springer-Verlag New York Inc.

Ali Karimoddini received his Bachelor of Electrical and Electronics Engineering from the Polytechnic University, Tehran, Iran, in 2003. He then joined the Petroleum University of Technology and received his Master of Science in Instrumentation and Automation Engineering in 2007. In 2008, he joined the National University of Singapore, Graduate School for Integrative Sciences and Engineering (NGS), where he has worked toward his Ph.D. degree. His research interests include hybrid modeling and control, discrete event systems, cooperative control, and formation control.

Hai Lin is currently a faculty member in the University of Notre Dame. Before joining Notre Dame, worked in the National University of Singapore as an Assistant Professor between 2006 and 2011. He received his B.S. degree from University of Science and Technology, Beijing, China, in 1997, his M.Eng. degree from the Chinese Academy of Science in 2000, and his Ph.D. degree from the University of Notre Dame, USA, in 2005. He served as the chair of the IEEE SMC Singapore Chapter 2009 and 2010, and has served on several editorial boards and conference organizing committees. His research interests are in the multidisciplinary study of the problems at the intersection of control, communication, computation, and life sciences. His current research thrust is on hybrid control systems, multi-robot coordination, and systems biology.

Ben M. Chen is a professor in Department of Electrical and Computer Engineering, National University of Singapore. His current research interests are in systems and control, unmanned aerial systems, and financial market modeling. Dr. Chen is an IEEE Fellow. He is the author/co-author of 9 research monographs including Loop Transfer Recovery: Analysis and Design (Springer, London, 1993), H2 Optimal Control (Prentice Hall, London, 1995), Robust and H∞ Control (Springer, New York, 2000), Hard Disk Drive Servo Systems (Springer, New York, 1st Edition, 2002; 2nd Edition, 2006), Linear Systems Theory: A Structural Decomposition Approach (Birkhäuser, Boston, 2004), Unmanned Rotorcraft Systems (Springer, New York, 2011), and Stock Market Modeling and Forecasting: A System Adaptation Approach (Springer, in press). He currently serves as an editor-in-chief of a newly established journal, Unmanned Systems, published by World Scientific and Imperial College Press. He had also served on the editorial boards of a number of journals, including IEEE Transactions on Automatic Control, Systems & Control Letters, and Automatica. He was the recipient of the Best Poster Paper Award, 2nd Asian Control Conference, Seoul, Korea (1997); IES Prestigious Engineering Achievement Award, Institution of Engineers, Singapore (2001); Temasek Young Investigator Award, Defence Science & Technology Agency, Singapore (2003); Best Industrial Control Application Prize, 5th Asian Control Conference, Melbourne, Australia (2004); Best Application Paper Award, 7th Asian Control Conference, Hong Kong (2009), and Best Application Paper Award, 8th World Congress on Intelligent Control and Automation, Jinan, China (2010).

Tong Heng Lee received his B.A. degree with First Class Honours in the Engineering Tripos from Cambridge University, England, in 1980, and his Ph.D. degree from Yale University in 1987. He is a Professor in the Department of Electrical and Computer Engineering at the National University of Singapore (NUS), and also a Professor in the NUS Graduate School, NUS NGS. He was a Past Vice-President (Research) of NUS. Dr. Lee's research interests are in the areas of adaptive systems, knowledge-based control, intelligent mechatronics, and computational intelligence. He currently holds Associate Editor appointments for IEEE Transactions in Systems, Man and Cybernetics; IEEE Transactions in Industrial Electronics; Control Engineering Practice (an IFAC journal); and the International Journal of Systems Science (Taylor and Francis, London). In addition, he is the Deputy Editor-in-Chief of the IFAC Mechatronics journal. Dr. Lee was a recipient of the Cambridge University Charles Baker Prize in Engineering; the 2004 ASCC (Melbourne) Best Industrial Control Application Paper Prize; the 2009 IEEE ICMA Best Paper in Automation Prize; and the 2009 ASCC Best Application Paper Prize. He has also co-authored five research monographs (books), and holds four patents (two of which are in the technology area of adaptive systems; the other two are in the area of intelligent mechatronics). He has published more than 300 international journal papers. Dr. Lee was an Invited Panelist at the World Automation Congress, WAC2000 Maui USA; an Invited Keynote Speaker for IEEE International Symposium on Intelligent Control, IEEE ISIC 2003 Houston USA; an Invited Keynote Speaker for LSMS 2007, Shanghai China; an Invited Expert Panelist for IEEE AIM2009; an Invited Plenary Speaker for IASTED RTA 2009, Beijing China; an Invited Keynote Speaker for LSMS 2010, Shanghai China; an Invited Keynote Speaker for IASTED CA 2010, Banff Canada; an Invited Keynote Speaker for IFTOMM ICDMA 2010, Changsha China; and an Invited Keynote Speaker for ICUAS 2011, Denver USA. Additionally, Dr. Lee was a recipient of a Singapore President's Scholarship; an NUS Overseas Graduate Scholarship; and a Yale University Graduate Fellowship. Dr. Lee is also a multiple winner of the National University of Singapore FoE Engineering Educator Award, and is now on the Engineering Educator Honour Roll.